پایه گربنر و ارایه الگوریتم جدید

بهروز صادقی[1]
گروه ریاضی، دانشکده علوم، دانشگاه آزاد اسلامی واحد مرند، مرند ، ایران

چکیده
پایه های گُربنر ابزار اصلی جبرجابجایی - محاسباتی و عامل عمده این مبحث از ریاضیات است . پایه های گُربنر ، محاسبات روی چند جمله ای ها را به محاسبه روی یک جمله ای ها تبدیل می کند . اعمال روی یک جمله ای ها ماهیت الگوریتمی و ترکیبیاتی دارند و بطور طبیعی استفاده از کامپیوتر و نرم افزارهای محاسباتی در این زمینه را ضروری می سازد .پایه های گُربنر و الگوریتم بوخبرگر به مفاهیمی بنیادی در جبر و محاسبات جبری تبدیل شده اند و نشان داده شده که در علوم بهینه سازی ، کد گذاری ، کنترل ، رباتیک ، آمار و ... کاربردهای فراوانی دارند ، امروزه با پیشرفت مبانی نظریه پایه های گُربنر و با توجه به نقش اساسی چند جمله ای ها در علوم ریاضی ، کاربردهای گسترده ای در شاخه های مختلف از ریاضیات نظیر رمزنگاری و رمز گشایی پیدا کرده است .
در این مقاله با محاسبه پایه های گرُبنر بوسیله نرم افزار های جبر جابجایی از نظر زمان و مراحل محاسباتی در الگوریتم های $F۴$ و $F۵$ بررسی می شوند.

واژه‌های کلیدی: الگوریتم بوخبرگر ، الگوریتم $F۴, F۵$
[2010]: 14Q20,1404,13P10

۱ مقدمه

پایه گرُبنر ابزار اصلی جبر جابجایی - محاسباتی است . به بیان ساده ، هر پایه گرُبنر محاسبات روی چندجمله ایها را به محاسبات روی یک جمله ایها تبدیل می کند . عملیات روی یک جمله ایها ماهیت الگوریتمی - محاسباتی دارند و ورود کامپیوتر به این عرصه از ریاضیات تاثیر شگرفی روی برهان های ریاضی ایجا نموده است .

در اوایل دههٔ ۱۹۶۰ ، زمانیکه ولفگانگ گُربنر [2] استاد ریاضیات دانشگاه اینسبورک اتریش مسئله ای در رابطه با حلقه خارج قسمتی یک ایده آل صفر بعدی مطرح نمود ، احتمالاً تصور نمی کرد که ارائه یک الگوریتم برای حل این مسئله توسط یکی از دانشجویان وی بنام برونو بوخبرگر [3] شاخه ای جدید در ریاضیات با نام پایه های گُربنر ایجاد نماید و گُربنر در زمره افرادی باشد که نامش در طبقه بندی موضوعی ریاضیات (MSC)[4] ظاهر شود .

گُربنر در سال ۱۹۳۹ [۱۳] مقاله ای منتشر نمود که حاوی کاربردهای متنوعی از روش های ''استفاده از تک جمله ایها '' بود . او در این مقاله قضیه مکالی را در حالت ایدآل تک جمله ای اثبات نمود . او در مقاله اش نوشته بود '' من این روش ها را در ۱۷ سال آزموده و به کاربرده ام اکنون به این باور رسیده ام که این روشها ابزاری قوی در حل مسائل نظریه ایدآل ها تبدیل شود . '' گُربنر منتظر کسی بود که با ارائه یک الگوریتم ، این ابزار قوی را معرفی نماید که در سال ۱۹۶۳ آنرا بعنوان رساله دکتری بوخبرگر جوان در نظر گرفت . بوخبرگر پس از دوسال تلاش و تعمق روی مسئله گُربنر

[1] سخنران
[2] Wolfgang Gruobner
[3] Bruno Buchberger
[4] 13P10

نشان داد که مولد متناهی برای ایدآل I موجود است که جملات پیشرو آن ، ایدآل تولید شده توسط همه جملات پیشرو اعضای I را تولید می کنند . در این مورد به [۴] مراجعه شود . بوخبرگر خود را مدیون گُربنر دانست که با وسعت فکر و آزادی عملی که به او بخشیده بود و با ارائه یک مسئله عالی موفقیتی بزرگ به استادش هدیه کرد : بوخبرگر برای ادای دین خود ، مولد ویژه ای برای ایدآلها بدست آورد ، پایه گُربنر نام نهاد و آنرا بصورت رسمی در سمیناراروپائی " محاسبات نمادین و جبری " در سال ۱۹۷۹ ، دو سال قبل از فوت گُربنر ، اعلام نمود .

اکنون پایه های گُربنر و الگوریتم بوخبرگر برای محاسبه آن ، در تمام نرم افزارهای ریاضی نظیر CoCoA و یا SINGULAR گنجانده شده و میلیونها کاربر در سرتاسر جهان دارد . در مجموع ، می توان پذیرفت که پایه های گُربنر و الگوریتم بوخبرگر به مفاهیمی بنیادی در جبر و محاسبات جبری تبدیل شده اند و نشان داده شده که در علوم بهینه سازی ، کد گذاری ، کنترل ، رباتیک ، آمار و ... کاربردهای فراوانی دارند که در این زمینه ها می توان به [۱۸] و [۱۹] مراجعه نمود .

تعریف ۱.۱. فر ض کنید K میدان و $K[x_۱,\cdots,x_n]$ حلقه چند جمله ایها یش باشد I. ، ایدآل تولید شده توسط $x_۱^{\alpha_۱}\cdots x_n^{\alpha_n}$ را یک ایدآل تک جمله ای نامیده می شود .

قضیه ۲.۱. ([۷]) فر ض کنید K میدان و $K[x_۱,\cdots,x_n]$ حلقه چند جمله ایها یش باشد . اگر $f=\sum c_i m_i$ که $c_i\in K-\{۰\}$ و m_i ها مجموعه ای از تک جمله ای ها باشد .اگر $f_i\in I$ آنگاه برای هر i ، $m_i\in I$.

لم ۳.۱ (لم دیکسون). ([۷]) هر ایدآل در $K[x_۱,\cdots,x_n]$ با تولید متناهی است .

تعریف ۴.۱ (پایه گُربنر). فرض کنید $\mathfrak{a}$ ایدآلی در $K[x_۱,\cdots,x_n]$ باشد . مجموعه $\{g_۱,\cdots,g_s\}\in\mathfrak{a}$ را پایه گُربنر برای $\mathfrak{a}$ گویند بطوریکه

$$<LM(g_۱),...,LM(g_s)>=LT(\mathfrak{a})$$

و در اینصورت $G=<g_۱,...,g_s>=\mathfrak{a}$ خواهد بود .

مثلاً اگر $I=<xy^۲+۱,۲x^۲+y>$ و رابطه $\prec$ همان رابطه $\prec_{lex}$ آنگاه

$$LT(I)=<۲x,y^۵>\Longrightarrow G=\{۲+y^۵,۲x-y^۳\}.$$

برای یافتن یک پایه گُربنر برای یک ایدآل ، بوخبرگر ، یکی ازدانشجویان گُربنر ، الگوریتمی را طراحی نمود که بنام الگوریتم بوخبرگر معروف است :

الگوریتم ۱ الگوریتم بوخبرگر.

ورودی: مجموعه متناهی $F=\{f_۱,..,f_m\}$ از مولدهای ایدآل $\mathfrak{a}$ با ترتیب الفبایی.
خروجی: محاسبه پایه گُربنر از $\mathfrak{a}$ برای $<F>$

۱: قرار دهید $G:=F$؛
۲: قرار دهید $\mathbb{B}=\{(g_۱,g_۲)|g_۱,g_۲\in G\}$؛
۳: تا زمانیکه $\mathbb{B}=\emptyset$ مرحله زیر انجام شود ؛
۴: مجموعه $\{g_۱,g_۲\}\in\mathbb{B}$ انتخاب شود؛
۵: قرار دهید $\mathbb{B}=\mathbb{B}\setminus\{g_۱,g_۲\}$ ؛
۶: $h:=Spoly(g_۱,g_۲)$ محاسبه کن ؛
۷: اگر $h=\emptyset$ به مرحله ۳ برو ؛
۸: اگر $h\neq\emptyset$ آنگاه $G:=G\cup\{h\}$ ؛
۹: قرار دهید $\mathbb{B}:=\mathbb{B}\cup\{(g,h)|g\in G\}$ ؛
۱۰: به مرحله ۳ برو ؛
۱۱: پایان

برای امتحان الگوریتم بوخبرگر ، فرض کنید $\mathfrak{a}=<f_۱=x^۲+۲xy^۲,f_۲=xy+۲y^۳-۱>$ روی $K[x,y]$ باشد . همچنین $\prec$ ترتیب $\prec_{lex}$ روی $K[x,y]$ باشد . با ادامهٔ روند الگوریتم ، خواهیم داشت :

$$G=\{x^۲+۲xy^۲,xy+۲y^۳-۱,x,۲y^۳-۱\}$$

این الگوریتم گرچه بسیار ساده است اما به لحاظ تئوری کاستی هایی نیز دارد . در واقع در حالت کلی زمان اجرا ی الگوریتم دارای حد و مرز مشخصی نیست . اما با مطالعاتی که روی این الگوریتم انجام شده تاحدودی مسئله زمان مرتفع گردیده و بهبود یافته است که به صورت زیر است :

الگوریتم ۲ الگوریتم بهبودیافته بوخبرگر.

ورودی: مجموعه متناهی $F = \{f_1, .., f_m\}$ از مولدهای ایدآل $\mathfrak{a}$ با ترتیب الفبایی.
خروجی: محاسبه پایه گُربنر از $\mathfrak{a}$ برای $< F >$؛
۱: $G := \emptyset$ و $\mathbb{B} := \emptyset$؛
۲: تازمانیکه $F = \emptyset$ دو مرحله زیر انجام شود ؛
۳: $f \in F$ انتخاب شود و $F := F \setminus \{f\}$؛
۴: $(G, \mathbb{B}) := (G, \mathbb{B}, f)$؛
۵: تازمانیکه $B :\neq \emptyset$ سه مرحله زیر انجام شود ؛
۶: $(g_1, g_2) \in \mathbb{B}$ ؛
۷: قرار دهید $\mathbb{B} := \mathbb{B} \setminus \{(g_1, g_2)\}$؛
۸: $h := Spoly(g_1, g_2)$
۹: اگر $h =\neq 0$ آنگاه
۱۰: $(G, \mathbb{B}) := (G, \mathbb{B}, h)$؛
۱۱: به مرحله ۳ برو ؛
۱۲: پایان

در زمان اجرای الگوریتم بوخبرگر چندین انتخاب وجود دارد که عبارتند از :
۱- انتخاب زوج نقطه از بین نقاط بحرانی که بصورت زوج مرتب هستند ،
۲- انتخاب چند جمله ای تقلیل یافته از بین مراحل الگوریتم که کاهشی است . در [۳] ثابت شده است که این انتخابها برای دقت الگوریتم مناسب نیستند ، لیکن در [۱۱] استراتژی جدیدی بکار گرفته شده که بشرح زیر است :
حالتی را در نظر بگیرید که تمامی جملات ورودی جمله پیشرو یکسانی داشته باشند . در این حالت تمامی زوج نقاط در الگوریتم بوخبرگر مساوی شده و ادامه روند غیر ممکن خواهد بود . یک روش ساده و شگفت آور این استکه انتخابی برای ما امکان نداشته باشد . یعنی بجای اتخاب یک زوج نقطه ، یک زیر مجموعه ای از نقاط در همان زمان انتخاب نماییم . لذا در مرحله دوم الگوریتم ما یک مجموعه از نقاط مورد نظر انتخاب نموده ایم که همان قسمت جبر خطی الگوریتم است . بنابراین می توان تعریف زیر را مطرح نمود:

$$Pair[f_i, f_j] = lcm_{ij} = HT(t_i f_i) = HT(t_j f_j) = [HT(f_i), HT(f_j)]$$

که $(f_i, f_j) \in T^2 \times \mathbb{R}[\underline{X}] \times T \times \mathbb{R}[\underline{X}]$ بنابر این آماده هستیم تا با مطالب گفته شده الگوریتم $F4$ را ارائه دهیم :

الگوریتم ۳ الگوریتم $F4$

ورودی: مجموعه متناهی مانند $F \subset R[\underline{x}]$ با تابعی مانند $\varphi : List(Pair) \to List(Pair)$ بطوریکه $\varphi(l) \neq \emptyset$ برای $l \neq \emptyset$؛
خروجی: یک زیرمجموعه متناهی از $R[\underline{x}]$؛
۱: قرار دهید $G := F$ و $\tilde{F}_0^+ := F$ و $d := 0$؛
۲: قرار دهید $P := \{Pair(f, g) | f, g \in G, f \neq g\}$؛
۳: تا زمانیکه $P \neq \emptyset$ مراحل زیر انجام شود ؛
۴: $d := d + 1$ و $Pd := \varphi(P)$؛
۵: قرار دهید $P := P \setminus Pd$ و $Ld := Left(Pd) \cup Right(Pd)$؛
۶: قرار دهید $\tilde{F}_d^1 := Reduction(Ld, G)$؛
۷: برای $h \in \tilde{F}_d^+$ به مرحله بعد برو ؛
۸: $P := P \cup \{Pair(h, g) | g \in G\}$؛
۹: $G := G \cup \{h\}$؛
۱۰: G را محاسبه نماید ؛
۱۱: پایان

بطوریکه در الگوریتم فوق ، تقلیل یافته چند جمله ای ، $Reduction(Ld, G)$ ، را بصورت زیر داریم :

الگوریتم ۴ Reduction

ورودی: $L \subseteq T \times R[\underline{x}]$ و مجموعه متناهی $G \subseteq R[\underline{x}]$ ؛
خروجی: یک زیرمجموعه متناهی از $R[\underline{x}]$؛
۱: قرار دهید $F := SymbolicPreprocessing(L, G)$؛
۲: $\tilde{F}$ را سطری پلکانی از F قرار دهد ؛
۳: $\tilde{F}^{+} := \{f \in \tilde{F} | HT(f) \notin HT(F)\}$؛
۴: $\tilde{F}^{-}$ را محاسبه نماید ؛
۵: پایان .

قضیه ۵.۱. ([۵].) الگوریتم $F۴$ یک پایه گُربنر G در $R[\underline{x}]$ محاسبه می کند بطوریکه $F \subseteq G$. بعبارت دیگر الگوریتم پایان پذیر است .

با توجه به الگوریتم بوخبرگر و $F۴$ می توان چنین استنباط نمود که گرچه الگوریتم $F۴$ در مقایسه با الگوریتم بوخبرگر زمان کمتری برای محاسبه پایه گروبنر یک چند جمله ای نیاز دارد اما بدلیل پیچیدگی ذاتی و خروجی طولانی ، زیاد موفق نیست . بعبارت دیگر خروجی این الگوریتم ، بسیار بزرگ و از لحاظ زمانی طولانی است ، زیرا ضرایب چند جمله ای بسیار بزرگ و محاسبات شناور و غیر ثابت ، تعریفی برای این پیشرفت الگوریتم نیست .
بنابراین چنین نتیجه می شود که الگوریتم $F۴$ از لحاظ بزرگی و شدت محاسباتی در مقایسه با الگوریتم قبلی سریعتر عمل می کند . با این حال می توان الگوریتمی سریعتر از این نیز طرح نمود که از نظر زمانی و محاسباتی کمتر از الگوریتم مذکور در این بخش گردد . در فصل بعدی به این الگوریتم خواهیم پرداخت . برای مثالهای بیشتر و همچنین محاسبات پایه گُربنر این مثالها به [۱۲] مراجعه کنید.

مثال ۶.۱. اگر در ترتیب $\prec_{lex}$ با $z < y < x$ قرار دهیم :

$$F = [f_۱ = xy^۲ + ۱, f_۲ = xz^۲ + ۱, f_۳ = y^۳ + y^۲]$$

و φ تابع همانی باشد ، $\tilde{F}_۱^{+} = \{y^۲ - z^۲, y + ۱\}$ بطوریکه $HT(\tilde{F}_۱^{+}) = \{y\}$ که در الگوریتم بوخبرگر چنین نتیجه ای به سادگی حاصل نمی شود .

۲ الگوریتم جدید

در این بخش الگوریتم موثر دیگری برای محاسبه پایه گُربنر معرفی می کنیم و آنرا با معیارهای بوخبر گر جایگزین می کنیم . این الگوریتم یک راه حل جدیدی برای پایه های گُربنر ارائه می دهد . حل دستگاه چند جمله ای ها یکی از قسمتهای مهم جبر کامپیوتری است ؛ چرا که در بیشتر علوم نظیر رباتیک ، کارتوگرافی و کد نویسی و... با این الگوریتم قابل حل هستند .
فرض کنید $(f_۱, \cdots, f_m)$ چند جمله ای m تایی از مدول آزاد $\mathcal{P}^m$ و I ایده آل تولید شده توسط $(f_۱, \cdots, f_m)$ با شد . همچنین $\mathbf{F_i}$ ، $i-$اُمین بردار واحد از $\mathcal{P}^m$ باشد . تابع ارزیاب v :

$$v \begin{pmatrix} \mathcal{P}^m \longrightarrow \mathcal{P} \\ \mathbf{g} = (g_۱, \cdots, g_m) \longmapsto \sum_{i=۱}^{m} f_i g_i \end{pmatrix} \qquad (۱)$$

را معرفی می کنیم که $v(\mathbf{F_i}) = f_i$ و $\mathbf{g} = \sum_{i=۱}^{m} g_i \mathbf{F_i}$. m تایی $\mathbf{g} = (g_۱, \cdots, g_m)$ یک سی زی جی نامیده می شود اگر $v(\mathbf{g}) = ۰$. و $\mathbf{s_{i,j}} = f_j \mathbf{F_i} - f_i \mathbf{F_j}$ را سی زی جی های اساسی نامیم . مجموعه تمام سی زی جی ها را که یک مدول است با Syz نشان می دهیم . برای اطلاعات تکمیلی به [۳] و یا [۱۵] مراجعه نمایید .

با توجه به ترتیب $\prec_{lex}$، ترتیب مذکور را برای $\mathcal{P}^m$ چنین تعریف می کنیم :

$$\sum_{k=۱}^{m} g_k\mathbf{F}_k \prec \sum_{k=j}^{m} h_k\mathbf{F}_k \Leftrightarrow \begin{cases} i > j, \quad h_j \neq ۰ \\ or \\ i = j, \quad HT(g_i) < HT(h_i) \end{cases}$$

بویژه $\mathbf{F}_۱ \prec \cdots \prec \mathbf{F}_{m-۱} \prec \mathbf{F}_m$. همچنین $\mathbf{g} = \sum_{i=۱}^{m} g_i\mathbf{F}_i$درجه آنرا با

$$deg(\mathbf{g}) = \{deg(g_i) + deg(f_i) | ۱ \leq i \leq m\}.$$

نشان می‌دهیم.

قضیه ۰.۱.۲ ([۷]) فرض کنید

$$w \begin{pmatrix} T \longrightarrow \mathcal{P}^m \\ t \longmapsto min_{\prec} W(t) \end{pmatrix} \tag{۲}$$

و اگر $(t_۱, t_۲) \in T(I)^۲$ آنگاه $HT(W(t_۱)) \neq HT(W(t_۲))$

نتیجه ۰.۲.۲ ([۶]) برای تمام چند جمله ای های $p \in I$ تعریف می کنیم : $v_۱(p) = HT(W(HT(p)))$. اگر $p_۲, p_۱$ دوچند جمله ای در ایدآل I باشند که $HT(p_۱) \neq HT(p_۲)$ آنگاه داریم : $v_۱(p_۱) \neq v_۱(p_۲)$

با این مقدمات اکنون آماده ایم تا الگوریتم تقسیم جدید ، $F۵$ ، را مطرح نماییم . قبل از آن ، ابتدا الگوریتم Reset Specification rules را برای اعضای $T \times \mathbb{N}$ می نویسیم :

Reset Specification rules

Input:m number of polynomials

for $i := 1, cdots, m$ **do**
$Rule[i] := \emptyset$
Add Rule
$(r_k = (t\mathbf{F}_i, p) \in \mathbf{T} \times \mathcal{P})$
$Rule[i] := ([t, k], Rule[i-1])$
$L := Rule[i] = [[t_1, k_1], ..., [t_r, k_r]]$
for $i = 1, ..., r$**do**
if $t_i | (u \times t)$**then**
return$(\frac{u \times t}{t_i}, r_{k_i})$
return(u, r_k)

مثلاً اگر $r_۴ = (\mathbf{F}_۲, f_۴)$ و $r_۶ = (x\mathbf{F}_۲, f_۶)$ از مثال قبلی باشند ، آنگاه $AddRule(r_۴)$ و $AddRule(r_۶)$ دو قاعده جدید $xF_۲ \to f_۶$ و $\mathbf{F}_۲ \to f_۴$ اضافه می گردد . نتیجتاً $(xy, r_۴)$ به $(y, r_۶)$ برخواهد گشت و الگوریتم جواب $true$ خواهد داد .

در [۶] نشان داده شده که الگوریتم مذکور پایان پذیر است .

چون الگوریتم افزایشی است ، لذا حلقه اصلی الگوریتم برای تعدادی از چندجمله ای ها تکرار خواهد شد .

Algorithm Incremental

Input:$F = (f_1, ..., f_m)$ number of polynomials with $\prec$
$N := m$ number of polynomials
Reset Simplification rules (m)
$r_m := (\mathbf{F}_m, f_m) \in T \times \mathcal{F}$, $G_m := [r_m]$
for $i := 1, \cdots, m-1$ **do**

G_i:=**Algorithm F5**(i, f_i, G_{i-1})
return$poly(G) = [poly(r)|r \in G_{m-1}]$

که

New Algorithm : F5

Input:$i \in \mathbb{N}$ and f_i and G_{i+1} a finite subset of $T \times \mathcal{P}$
Output: G_i is Groubner Bases .
$r_i := (\mathbf{F}_i, f_i) \in T \times \mathcal{P}$
$\varphi_{i+1} = NF(., poly(G_{i+1})$
$G_i := G_{i+1} \cup r_i$
$P := Sort[CritPair(r_i, r, i, \varphi_{i+1})|r \in G_{i+1}]$
while $P \neq \emptyset$**do**
$d := deg(P)$
$P_d := \{p \in P|deg(p) =\}$
$P := P - P_d$
$F := Spol(P_d)$
$R_d := Reduction(F, G_i, i, \varphi_{i+1})$
for $r \in R_d$ **do**
$P := P \cup \{CritPair(r, p, \varphi_{i+1})|p \in G_i\}$
$G_i := G_i \cup \{r\}$
P:=Sort P for the degree
return G_i

که در آن

Algorithm CritPair(r_1, r_2, k, φ)

Input:$k \in \mathbb{N}$, $r_1, r_2 \in R = T \times \mathcal{P}, \varphi : NormalForm$
Output$[t, u_1, r_1, u_2, r_2]$
$p_i := poly(r_i)$ for $i = 1, 2$
$t := [HT(p_1), HT(p_2)]$
$u_1 := \frac{1}{HT(p_i)}$ for $i = 1, 2$
if$u_1\varphi(r_1) \prec u_2\varphi(r_2)$ **then**
Swap r_1, r_2
$t_i\mathbf{F}_{k_i} := \varphi(r_i)$ for $i = 1, 2$
if$k_1 > k$ **then** **return**$\emptyset$
if$\varphi(u_1t_1) \neq u_1t_1$ **then** **return**$\emptyset$
return

و

Algorithm Spol

Input:$P = [p_1, ..., p_h]$ list of critical pairs
Output$F$
let $p_l := [t_l, u_l, r_{i_l}, v_l, r_{j_l}]$ for $l = 1, ..., h$
$F := \emptyset$
for l **from** 1 **to** h **do**
$N := N + 1$
Swap $r_N := (u_l\varphi(r_{i_l}), u_lpoly(r_{j_l}) - v_lpoly(r_{j_l}))$
Add Rule (r_N)
$F := F \cup \{r_N\}$
F:= Sort F by increasing φ
return F

در [۶] نشان داده شده که الگوریتم مذکور پایان پذیر است .
فرض کنید $\prec$ به صورت $t < z < y < x$ باشد و

$$f_3 = x^2y - z^2t$$

$$f_2 = xz^2 - y^2t$$

$$f_1 = yz^3 - x^2t^2$$

با توجه به الگوریتم $F5$ پایه گروبنر برابر است با :

$$G_3 = [r_3], G_2 = [r_3, r_2, r_4, r_5]$$

بطوریکه

$$r_3 = (\mathbf{F}_3, f_3), r_2 = (\mathbf{F}_2, f_2), r_4 = (xy\mathbf{F}_2, xy^3t - z^4t)$$

$$r_5 = (xyz^2\mathbf{F}_2, z^6t - y^5t^2)$$

$$\varphi_2 NF(., r_3, r_2, r_4, r_5) \quad r_1 = (\mathbf{F}_1, f_1)$$

$$G_1 = G_2 \cup \{r_1\} = [r_3, r_2, r_4, r_5, r_1]$$

لذا ۴ زوج بحرانی

$$p_7 = (xyz^3, x, r_1, yz, r_2) \quad p_8 = (x^2yz^3, x^2, r_1, z^3, r_3)$$

$$p_9 = (yz^6t, z^3t, r_1, y, r_5) \quad p_{10} = (xy^3z^3t, xy^2t, r_1, z^3, r_4)$$

هستند و

$$\varphi(p_7) = x\mathbf{F}_1, \varphi(p_8) = x^2\mathbf{F}_1, \varphi(p_9) = z^3\mathbf{F}_1, \varphi(p_{10}) = xy^2\mathbf{F}_1$$

با $d = 5$ ، قرار دهید : $Spol(P_5)$ که $P_5 = [p_7]$ و $P = [p_8, p_9, p_{10}]$ ؛

$$r_6 = (x\mathbf{F}_1, y^3zt - x^3t^2), \quad F := [r_6], \quad x\mathbf{F}_1 \to r_6$$

با توجه به الگوریتم ، مقدار $R_5 = [r_6]$ حاصل می شود بنابراین $G_1 = [r_3, r_2, r_4, r_5, r_1, r_6]$ نتیجه می شود . اگر برای $d = 6, 7, 8, 9$ انجام دهیم ، به $F = \emptyset$ و $R_9 = \emptyset$ می رسیم . پس پایه گروبنر با این الگوریتم همان G_1 خواهد شد .
یادآوری می کنیم که با انجام این الگوریتم ، ۷ زوج بدون کاربرد و ۵ زوج مفید داریم .
فرض کنید

$$\begin{aligned} f_1 &= x_1x_3x_4x_5 + x_2x_3x_5 + x_3x_4x_5 \\ f_2 &= x_1x_2x_5 + x_1x_4x_5 \\ f_3 &= x_1x_2x_4 + x_1x_3x_4 + x_1x_3x_5 + x_2x_4x_5 \\ f_4 &= x_2x_4x_5 + x_3x_4 \end{aligned} \tag{3}$$

که جدول آن بصورت :

$Algoritm$	$Buchberger$	$F4$	$F5$
$Dimension$	11	11	10
$Monomial \quad Number$	30	31	27
$Computation \quad time$	0.433	0.322	0.264

است و

$$
\begin{aligned}
f_{۱} &= x_{۱}x_{۳}x_{۴}x_{۵} + x_{۲}x_{۳}x_{۵} + x_{۳}x_{۴}x_{۵} \\
f_{۲} &= x_{۱}x_{۲}x_{۴} + x_{۱}x_{۴}x_{۵} + x_{۲}x_{۴}x_{۵} + x_{۳}x_{۵} + x_{۴}x_{۵} \\
f_{۳} &= x_{۱}x_{۲} + x_{۱}x_{۳}x_{۴} + x_{۱}x_{۵} + x_{۲}x_{۴} \\
f_{۴} &= x_{۱}x_{۲}x_{۵} + x_{۲}x_{۴}x_{۵} + x_{۲}x_{۵} + x_{۳}x_{۴}x_{۵} + x_{۵} \\
f_{۵} &= x_{۱}x_{۲}x_{۳} + x_{۱}x_{۳}x_{۵} + x_{۲}x_{۳}x_{۵} + x_{۳}x_{۴}x_{۵} \\
f_{۶} &= x_{۱}x_{۲}x_{۳}x_{۴}x_{۵} + x_{۱}x_{۲}x_{۵} + x_{۱}x_{۳}x_{۴}x_{۵} \\
f_{۷} &= x_{۲}x_{۴} + x_{۳}x_{۴} + x_{۳}x_{۵} + x_{۴} \\
f_{۸} &= x_{۱}x_{۳}x_{۵} + x_{۲}x_{۳}x_{۵} + x_{۲}x_{۵}
\end{aligned}
\tag{۴}
$$

که جدول محاسباتی آن بصورت

$Algoritm$	$Buchberger$	$F4$	$F5$
$Dimension$	19	19	16
$Monomial \quad number$	67	60	60
$Computation \quad time$	2.875	0.940	0.633

است. همچنان که از جدولها مشاهده می شود الگوریتم $F۵$ با زمان کمتر و بُعد بهتری این محاسبات را انجام می دهد .
با توجه به [۳] و [۸] و [۱۷] جدول زیر بدست آمده است :

$example$	[3]	[8]	[17]	$F4$	$F5$
Raksanyi	1	3	0	0	0
Hairer1	10	6	4	0	0
Rose	22	19	5	0	0
Trinks6	17	8	6	0	0
Trinks7	12	11	6	5	4
Katsura3	1	1	1	0	0
Katsura4	18	10	7	0	0
Katsura5	50	28	9	3	0
Katsura10	3936	2154	195	15	0
Binary 10	2147	1061	349	4	0
Noon 8	7886	3504	204	59	7
Eco8	315	27	11	6	0

جهت انجام چگونگی محاسبات به [۱۹] و سایت رسمی محاسبات جبری در دانشگاه جینوا ، [۹] ، رجوع شود . همچنین دقت شود که در برخی از مثال ها تقلیل ها به صفر نرسیده است .

مراجع

[1] W . Adams , P . Loustaunau , *An Introduction to Groubner Bases*, Vol . 3 , Graduate Texts in Mathematics , Berlin , 1994

[2] T . Becker , V . Weispfenning , *Groubner Bases: A computation approach to commutative algebra*, Graduate Texts in Mathematics , Springer , Berlin , 1993

[3] B . Buchberger, *Grobner bases: An algorithmic method in polynomial ideal theory*, N. K. Bose ed., Multidimensional systems theory pp. 184-232, 1985.

[4] B . Buchberger, *On finding a vector spasce basis of the rsidue class ring modulo a zero dimensional polynomial ideal* , PhD Thesis , Universitat Innsbrouk , Innsbrouk , 1965 .

[5] J . C . Faugere, *A new efficient algorithm for computing Groubner Bases*, Journal of Pure and Applied Algebra , 139, 1999,pp 61- 88.

[6] J . C . Faugere, *A new efficient algorithm for computing Groubner Bases Without reduction to Zero*, Journal of Pure and Applied Algebra , 142, 2002,pp 245 - 265.

[7] R . Froberg, *An Introduction to Groubner Bases*, 2nd Edition, Wiley, 1998.

[8] R . Gebauer , H . M . Mouller , *On an instalation of Boushberger's Algorith* , J . S . Comput . Vol . 6 , 275 - 286 , 1988.

[9] Genoa university , *http://cocoa.dima.unige.it*, John Abbott , Anna Bigatti , Massimo Caboara , 2007.

[10] V . P . Gerdt , Y . A . Blinkov , *Involutive Bases of Polynomial Ideals* , Computer In Simulation , Vol . 45 ,1999

[11] A . Giovini,T . Mora , G . Niesi, *Selection strategies in the buchberger alghorithm* , in :ISSAC , ACM. , New York, 1991.

[12] A . Goerge , J . W . Liu , *Computer Solution of Large Sparse Positive Systems* , Prentice-Hall , Englewood Cliffs, 1981.

[13] W . Groubner , *Uber die algebraischen Eigenschhaftan der Integrale von linearen Differentiagleichungen mit konstanten Koeffizienten* , Monatsh . Math . , vol.47 , 247 - 286 .

[14] M . Kreuzer , L . Robbiano ,*Computational Commutative Algebra* , springer , 2005,prop.3.7.1

[15] D . Lazard , *Gaussian Elimination and Resolution of System of aLgebric Equations.* , In Proc . Eurocal 83 , Vol . 162 , pp.146 - 157 , 1998 .

[16] J . I . Massey , *Shift-Register systim and CoCoA* , IEEE. Trans . , Vol . 15 , 122 - 271 , 1969

[17] T . Mora , H . M . Mouler , C . Traverso , *Groubner Bases Computation Using Syzygy* , ISSAC 92 , ACM Press , 320 - 329 , Julay 1992

[18] L . Robiano , A . Capani , G . Niesi , *CoCoA: A system for doing computations in commutative algebra* , http//cocoa.dima.unige.it

[19] G . Pfister , G . M . Greuel , *Singular 2.0* , Springer Verlag , 2010

پست الکترونیکی: behruz.sadeqi@gmail.com

پست الکترونیکی: b-sadeqi@marandiau.ac.ir