Modifying Faugère's F5 Algorithm to ensure termination

Christian Eder[1], Justin Gash[2], and John Perry[3]
[1]Department of Mathematics, TU Kaiserslautern, P.O. Box 3049
67653 Kaiserslautern, Germany
[2]Department of Mathematics, Franklin College
101 Branigin Blv., Franklin IN 46131 USA
[3]University of Southern Mississippi, Box 5045
Hattiesburg MS 39406 USA

Abstract

The structure of the F5 algorithm to compute Gröbner bases makes it very efficient. However, while it is believed to terminate for so-called "regular sequences", it is not clear whether it terminates for all inputs.

This paper has two major parts. In the first part, we describe in detail the apparent obstacles to termination. In the second part, we explore three variants that ensure termination. Two of these have appeared previously in dissertations, and ensure termination by checking for a Gröbner basis using traditional criteria. The third variant, F5+, identifies a degree bound using a distinction between "necessary" and "redundant" critical pairs that follows from the analysis in the first part. Experimental evidence suggests this third approach is the most efficient of the three.

1 Introduction

The computation of a Gröbner basis is a central step in the solution of many problems of computational algebra. First described in 1965 by Bruno Buchberger [7], researchers have proposed a number of important reformulations of his initial idea [5,6,8,9,15,18,21]. Faugère's F5 Algorithm, published in 2002 [16], is in many cases the fastest, most efficient of these reformulations. Due to its powerful criteria, the algorithm computes very few zero-reductions, and if the input is a so-called "regular sequence", it never reduces a polynomial to zero (see Section 2 for basic definitions). In general, reduction to zero is *the* primary bottleneck in the computation of a Gröbner basis; moreover, many of the most interesting polynomial ideals are regular sequences. It is thus no surprise that F5 has succeeded at computing many Gröbner bases that were previously intractable [14,16].

An open question surrounding the F5 algorithm regards termination. In a traditional algorithm to compute a Gröbner basis, the proof of termination follows from the algorithm's ability to exploit the Noetherian property of polynomial rings: each polynomial added to the basis G expands the ideal generated by the leading monomials of G, and this can happen only a finite number of times. In F5, however, the same criteria that detect reduction to zero also lead the algorithm to add to G polynomials which do not expand the ideal of leading terms. We call these polynomials *redundant*. Thus, although the general belief is that F5 terminates at least for regular sequences, no proof of termination has yet appeared, *not even if the inputs are a regular sequence* (see Remark 20). On the

other hand, at least two systems of polynomials have been proposed as examples of non-termination (one in the source code accompanying [22]), but in our experience, these systems fail only on an incorrect implementation of F5; a correct implementation terminates even for these.

Is it possible to modify F5 so as to ensure termination? Since the problem of an infinite loop is due to the appearance of redundant polynomials, one might be tempted simply to discard them. Unfortunately, as we show in Section 3, this breaks the algorithm's correctness. Another approach is to supply, or compute, a degree bound, and to terminate once this degree is reached. Tight degree bounds are known for regular and "semi-regular" sequences [2,20], but not in general, so for an arbitrary input it is more prudent to calculate a bound based on the data. To that end,

- [17] tests for zero-reductions of these redundant polynomials (Section 4.1); whereas
- [1] applies Buchberger's lcm criterion (or "chain" criterion) on critical pairs (Section 4.2).

These approaches rely exclusively on traditional criteria that are extrinsic to the F5 algorithm, so they must interrupt the flow of the basic algorithm to perform a non-trivial computation, incurring an observable penalty to both time and memory.

This paper shows that it is possible to guarantee termination by relying primarily on the criteria that are intrinsic to the F5 algorithm. After a review of the ideas and the terminology in Section 2, we show precisely in Theorem 23 of Section 3 why one cannot merely discard the redundant polynomials *in medio res*: many of these "redundant" polynomials are "necessary" for the algorithm's correctness. Section 4.3 uses this analysis to describe a new approach that distinguishes between two types of critical pairs: those that generate polynomials necessary for the Gröbner basis, and those that generate polynomials "only" needed for the correctness of F5. This distinction allows one to detect the point where all necessary data for the Gröbner basis has been computed. We then show how to implement this approach in a manner that incurs virtually no penalty to performance (Section 4.4). Section 4.5 shows that this new variant, which we call F5+,

- computes a reasonably accurate degree bound for a general input,
- relies primarily (and, in most observed cases, only) on criteria intrinsic to F5, and
- minimizes the penalty of computing a degree bound.

Section 5 leaves the reader with a conjecture that, if true, could compute the degree bound even more precisely.

We assume the reader to be familiar with [16], as the modifications are described using the pseudo code and the notations stated there.

2 Basics

Sections 2.1–2.2 give a short review of notations and basics of polynomials and Gröbner bases; Section 2.3 reviews the basic ideas of F5.

For a more detailed introduction on non-F5 basics we refer the reader to [19]. Readers familiar with these topics may want to skim this section for notation and terminology.

2.1 Polynomial basics

Let $\mathcal{K}$ be a field, $\mathcal{P} := \mathcal{K}[\underline{x}]$ the polynomial ring over $\mathcal{K}$ in the variables $\underline{x} := (x_1, \ldots, x_n)$. Let T denote the set of terms $\{x^\alpha\} \subset \mathcal{P}$, where $x^\alpha := \prod_{i=1}^n x_i^{\alpha_i}$ and $\alpha_i \in \mathbb{N}$.

A *polynomial* p over $\mathcal{K}$ is a finite $\mathcal{K}$-linear combination of terms, i.e. $p = \sum_\alpha a_\alpha x^\alpha, a_\alpha \in R$. The *degree of* p is the integer $\deg(p) = \max\{\alpha_1 + \cdots + \alpha_n \mid a_\alpha \neq 0\}$ for $p \neq 0$ and $\deg(p) = -1$ for $p = 0$.

In this paper $>$ denotes a fixed admissible ordering on the terms T. W.r.t. $>$ we can write any nonzero p in a unique way as

$$p = a_\alpha x^\alpha + a_\beta x^\beta + \ldots + a_\gamma x^\gamma, \quad x^\alpha > x^\beta > \cdots > x^\gamma$$

where $a_\alpha, a_\beta, \ldots, a_\gamma \in \mathcal{K}\backslash\{0\}$. We define the *head term of* p $\mathrm{HT}(p) = x^\alpha$ and the *head coefficient of* p $\mathrm{HC}(p) = a_\alpha$.

2.2 Gröbner basics

We work with homogeneous ideals I in $\mathcal{P}$. For any $S \subset \mathcal{P}$ let $HT(S) := \langle \mathrm{HT}(p) \mid p \in S\backslash\{0\}\rangle$. A finite set G is called a *Gröbner basis* of an ideal I if $G \subset I$ and $HT(I) = HT(G)$. Let $p \in \mathcal{P}$. If $p = 0$ or there exist $\lambda_i \in \mathcal{P}, q_i \in G$ such that $p = \sum_{i=1}^k \lambda_i q_i$ and $\mathrm{HT}(p) \geq \mathrm{HT}(\lambda_i q_i)$ for all nonzero q_i, then we say that there exists a *standard representation of* p *w.r.t.* G, or that p *has a standard representation w.r.t* G. We generally omit the phrase "w.r.t. G" when it is clear from the context.

Let $p_i, p_j \in \mathcal{P}$. We define the *s-polynomial of the critical pair* (p_i, p_j) to be

$$p_{ij} := \mathrm{HC}(p_j)\frac{\gamma_{ij}}{\mathrm{HT}(p_i)}p_i - \mathrm{HC}(p_i)\frac{\gamma_{ij}}{\mathrm{HT}(p_j)}p_j$$

where $\gamma_{ij} := \mathrm{lcm}\,(\mathrm{HT}(p_i), \mathrm{HT}(p_j))$.

Theorem 1. *Let I be an ideal in $\mathcal{P}$ and $G \subset I$ finite. G is a Gröbner basis of I iff for all $p_i, p_j \in G$ p_{ij} has a standard representation.*

Proof. See Theorem 5.64 and Corollary 5.65 in [3, pp. 219–221]. □

In addition to inventing the first algorithm to compute Gröbner bases, Buchberger discovered two relatively efficient criteria that imply when one can skip an s-polynomial reduction [7, 9]. We will refer occasionally to the second of these criteria.

Theorem 2 (Buchberger's lcm criterion)**.** *Let $G \subset \mathcal{P}$ be finite, and $p_i, p_j, p_k \in \mathcal{P}$. If*

(A) $\mathrm{HT}(p_k) \mid lcm(\mathrm{HT}(p_i), \mathrm{HT}(p_j))$*, and*

(B) p_{ik} and p_{jk} have standard representations w.r.t. G,

then p_{ij} also has a standard representation w.r.t. G.

In the homogeneous case one can define a *d-Gröbner basis* G_d of an ideal I: This is a Gröbner basis of I for which all s-polynomials up to degree d have standard representations (cf. Definition 10.40 in [3, p. 473]).

The following definition is crucial for understanding the problem of termination of F5.

Definition 3. Let G be a finite set of polynomials in $\mathcal{P}$. We say that $p \in G$ is *redundant* if there exists an element $p' \in G$ such that $\mathrm{HT}(p') \mid \mathrm{HT}(p)$.

Remark 4. While computing a Gröbner basis, a Buchberger-style algorithm does *not* add polynomials that are redundant *at the moment they are added to the basis*, although the addition of other polynomials to the basis later on may render them redundant. This ensures termination, as it expands the ideal of leading monomials, and $\mathcal{P}$ is Noetherian. However, F5 adds many elements that are redundant *even when they are added to the basis*; see Section 3.

It is easy and effective to interreduce the elements of the initial ideal before F5 starts, so that the input contains only non-redundant polynomials; in all that follows, we assume that this is the case. However, even this does not prevent F5 from generating redundant polynomials.

Finally, we denote by $\varphi(p, G)$ the normal form of p with respect to the Gröbner basis G.

2.3 F5 basics

It is beyond the scope of this paper to delve into all the details of F5; for a more detailed discussion we refer the reader to [16], [12], and [13]. In particular, we do not consider the details of correctness for F5, which are addressed from two different perspectives in [16] and [13]. We assume that *if* the algorithm terminates, then the output is correct. This paper is concerned with showing that the algorithm can be modified so that it terminates, *and* that the modification does not disrupt the correctness of the algorithm.

We now recall some basic definitions and notation of [16] that we use in the following. Let $\mathbf{F}_i$ be the i-th canonical generator of $\mathcal{P}^m$ and define $\prec$, the extension of $<$ to $\mathcal{P}^m$, by $\sum_{k=i}^{m} g_k \mathbf{F}_k \prec \sum_{k=j}^{m} h_k \mathbf{F}_k$ iff

1. $i > j$ and $h_j \neq 0$, or
2. $i = j$ and $\mathrm{HT}(g_i) < \mathrm{HT}(h_i)$.

Moreover, denote $\mathbf{T} = \cup_{i=1}^{m} \mathbf{T}_i$ where $\mathbf{T}_i = \{t\mathbf{F}_i \mid t \in T\}$ and $R = \mathbf{T} \times \mathcal{P}$.

Definition 5. Borrowing from [22], we call the element

$$r = (t\mathbf{F}_i, p) \in R$$

of [16] a *labeled polynomial*. (It is referred to as the representation of a polynomial in [16].) We also denote

1. the *polynomial part of* r $\mathrm{poly}(r) = p$,
2. the *signature of* r $\mathcal{S}(r) = t\mathbf{F}_i$, and
3. the *signature term of* r $\mathrm{ST}(r) = t$, and
4. the *index of* r $\mathrm{index}(r) = i$.

Following [16], we extend the following operators to R:

1. $\mathrm{HT}(r) = \mathrm{HT}(p)$.
2. $\mathrm{HC}(r) = \mathrm{HC}(p)$.

3. $\deg(r) = \deg(p)$.

Let $0 \neq c \in \mathcal{K}$, $\lambda \in T$, $r = (t\mathbf{F}_i, p) \in R$. Then we define the following operations on R resp. $\mathbf{T}$:

1. $cr = (t\mathbf{F}_i, cp)$,
2. $\lambda r = (\lambda t\mathbf{F}_i, \lambda p)$,
3. $\lambda(t\mathbf{F}_i) = (\lambda t)\mathbf{F}_i$.

Proposition 6. *Let the list $F = (f_1, \ldots, f_m) \in \mathcal{P}^m$ be the input of F5. For any labeled polynomial $r = (t\mathbf{F}_i, p)$, $t \in T$, $1 \leq i \leq m$, computed by the algorithm, there exist $h_1, \ldots, h_m \in \mathcal{P}$ such that*

1. $p = h_1 f_1 + \ldots + h_m f_m$,
2. $h_1 = \ldots = h_{i-1} = 0$, *and*
3. $\mathrm{ST}(r) = \mathrm{HT}(h_i) = t$.

Let $G = \{r_1, \ldots, r_{n_G}\} \subset \mathcal{P}$. We denote $\mathrm{poly}(G) = \{\mathrm{poly}(r_1), \ldots, \mathrm{poly}(r_{n_G})\}$.

Definition 7. Let $r, r_1, \ldots, r_{n_G} \in R$, $G = \{r_1, \ldots, r_{n_G}\}$. Assume $\mathrm{poly}(r) \neq 0$. We say that r has a *standard representation w.r.t. G* if there exist $\lambda_1, \ldots, \lambda_{n_G} \in \mathcal{P}$ such that

$$\mathrm{poly}(r) = \sum_{i=1}^{n_G} \lambda_i \mathrm{poly}(r_i),$$

$\mathrm{HT}(r) \geq \mathrm{HT}(\lambda_i)\mathrm{HT}(r_i)$ for all i, and $\mathcal{S}(r) \succ \mathrm{HT}(\lambda_i)\mathcal{S}(r_i)$ for all i except possibly one, say i_0, where $\mathcal{S}(r) = \mathcal{S}(r_{i_0})$ and $\lambda_{i_0} = 1$. We generally omit the phrase "w.r.t. G" when it is clear from the context.

Remark 8. The standard representation of a labeled polynomial r has two properties:

1. The polynomial part of r has a standard representation as defined in Section 2.2, and
2. the signatures of the multiples of the r_i are not greater than the signature of r.

This second property makes the standard representation of a labeled polynomial more restrictive than that of a polynomial.

Definition 9. Let $r_i = (t_i\mathbf{F}_k, p_i), r_j = (t_j\mathbf{F}_\ell, p_j) \in R$. We define the *s-polynomial of r_i and r_j* by $r_{ij} := (m', p_{ij})$ where

$$m' = \max_{\prec} \left\{ \frac{\gamma_{ij}}{\mathrm{HT}(r_i)} t_i \mathbf{F}_k, \frac{\gamma_{ij}}{\mathrm{HT}(r_j)} t_j \mathbf{F}_\ell \right\}$$

and $\gamma_{ij} = \mathrm{lcm}\,(\mathrm{HT}(r_i), \mathrm{HT}(r_j))$.

All polynomials are kept monic in F5; thus we always assume in the following that $\mathrm{HC}(p_i) = \mathrm{HC}(p_j) = 1$ for $p_i \neq 0 \neq p_j$. Moreover we always assume γ_{ij} to denote the least common multiple of the head terms of the two considered polynomial parts used to compute r_{ij}.

Next we review the two criteria used in F5 to reject critical pairs which are not needed for further computations.

Definition 10. Let $G = \{r_1, \ldots, r_{n_G}\}$ be a set of labeled polynomials. The critical pair (r_i, r_j) *is detected by Faugère's Criterion* if for any $k \in \{i, j\}$ and $u_k = \frac{\gamma_{ij}}{\mathrm{HT}(r_k)}$ there exists $r \in G$ such that

1. $\mathrm{index}(r) > \mathrm{index}(r_k)$ and
2. $\mathrm{HT}(r) \mid u_k \mathrm{ST}(r_k)$.

Definition 11. Let $G = \{r_1, \ldots, r_{n_G}\}$ be a set of labeled polynomials. The critical pair (r_i, r_j) *is detected by the Rewritten Criterion* if for any $k \in \{i, j\}$ and $u_k = \frac{\gamma_{ij}}{\mathrm{HT}(r_k)} \in T$ there exist $r_a, r_b \in G$ such that

1. $\mathrm{index}(r_{ab}) = \mathrm{index}(r_k)$,
2. r_{ab} is computed after r_k, and
3. $\mathrm{ST}(r_{ab}) \mid u_k \mathrm{ST}(r_k)$.

Next we can give the main theorem for the idea of F5.

Theorem 12. *Let $I = \langle f_1, \ldots, f_m \rangle$ be an ideal in $\mathcal{P}$, and $G = \{r_1, \ldots, r_{n_G}\}$ a set of labeled polynomials such that $f_i \in \mathrm{poly}(G)$ for $1 \leq i \leq m$. Let $d \in \mathbb{N}$. If one of the following holds for all pairs (r_i, r_j) such that $\deg r_{ij} \leq d$:*

1. *(r_i, r_j) is detected by Faugère's Criterion,*
2. *(r_i, r_j) is detected by the Rewritten Criterion, or*
3. *r_{ij} has a standard representation,*

then $\mathrm{poly}(G)$ *is a d-Gröbner basis of I.*

Proof. See Theorem 1 in [16], Theorem 3.4.2 in [17] and Theorem 21 in [13]. □

Remark 13.

1. Requiring a standard representation of a labeled polynomial is stricter than the criterion of Theorem 1, but when used carefully, any computational penalty imposed by this stronger condition is negligible when compared to the benefit from the two criteria it enables.
2. It is possible that r_{ij} does not have a standard representation (cf. Proposition 17 in [13]) at the time either Criterion rejects (r_i, r_j). Since F5 computes the elements degree-by-degree, computations of the current degree add new elements such that r_{ij} has a standard representation w.r.t. the current Gröbner basis $\mathrm{poly}(G)$ before the next degree step is computed. Thus, at the end of each such step, we have computed a d-Gröbner basis of I.

Next we give a small example which shows how the criteria work during the computation of a Gröbner basis in F5.

Example 14. Let $>$ be the degree reverse lexicographical ordering with $x > y > z$ on $\mathbb{Q}[x, y, z]$. Let I be the ideal generated by the following three polynomials:

$$p_1 = xyz - y^2 z,$$
$$p_2 = x^2 - yz,$$
$$p_3 = y^2 - xz.$$

Let the corresponding labeled polynomials be $r_i = (\mathbf{F}_i, p_i)$. For the input $F = (p_1, p_2, p_3)$, F5 computes a Gröbner basis of $\langle p_2, p_3 \rangle$ as a first step: Since $\mathrm{ST}(r_{2,3}) = y^2 = \mathrm{HT}(r_3)$, $r_{2,3}$ is discarded by Faugère's Criterion. Thus $\{p_2, p_3\}$ is already a Gröbner basis of $\langle p_2, p_3 \rangle$.

Next the Gröbner basis of I is computed, i.e. r_1 enters the algorithm: Computing $r_{1,3}$ we get a new element: $r_4 = (y\mathbf{F}_1, xz^3 - yz^3)$. $r_{1,2}$ is not discarded by any criterion, but reduces to zero. Nevertheless its signature is recorded,[1] thus we still have $\mathcal{S}(r_{1,2}) = x\mathbf{F}_1$ stored in the list of rules to check subsequent elements.

Next check all s-polynomials with r_4 sorted by increasing signature:

1. Since $\mathcal{S}(r_{4,1}) = y^2\mathbf{F}_1$, $r_{4,1}$ is discarded by Faugère's Criterion using $\mathrm{HT}(r_3) = y^2$.
2. Since $\mathcal{S}(r_{4,2}) = xy\mathbf{F}_1$, $r_{4,2}$ is discarded by the Rewritten Criterion due to $\mathcal{S}(r_{1,2}) = x\mathbf{F}_1$, $r_{1,2}$ being computed after r_4.
3. Since $\mathcal{S}(r_{4,3}) = y^3\mathbf{F}_1$, $r_{4,3}$ is discarded by Faugère's Criterion using $\mathrm{HT}(r_3) = y^2$.

The algorithm now concludes with $G = \{r_1, r_2, r_3, r_4\}$ where $\mathrm{poly}(G)$ is a Gröbner basis of I.

3 Analysis of the problem

The root of the problem lies in the algorithm's reduction subalgorithms, so Section 3.1 reviews these in detail. In Section 3.2, we show how the criteria force the reduction algorithms not only to add redundant polynomials to the basis, but to do so in a way that does not expand the ideal of leading monomials (Example 16)! One might try to modify the algorithm by simply discarding redundant polynomials, but Section 3.3 shows that this breaks the algorithm's correctness. This analysis will subsequently provide insights on how to solve the problem.

Throughout this section, let the set of labeled polynomials computed by F5 at a given moment be denoted $G = \{r_1, \ldots, r_{n_G}\}$.

3.1 F5's reduction algorithm

For convenience, let us summarize the reduction subalgorithms in some detail here. Let i be the current iteration index of F5. All newly computed labeled polynomials r satisfy $\mathrm{index}(r) = i$. Let G_{i+1} denote the set of elements of G with index $> i$. We are interested in `Reduction`, `TopReduction` and `IsReducible`. F5 sorts s-polynomials by degree, and supplies to `Reduction` a set F of s-polynomials of minimal degree d. Let $r \in F$.

1. First, `Reduction` replaces the polynomial part of r with its normal form with respect to G_{i+1}. This clearly does not affect the property $\mathcal{S}(r) = \mathrm{ST}(r)\mathbf{F}_i$. `Reduction` then invokes `TopReduction` on r.
2. `TopReduction` reduces $\mathrm{poly}(r)$ w.r.t. G_i, but invokes `IsReducible` to identify reducers. `TopReduction` terminates whenever $\mathrm{poly}(r) = 0$ or `IsReducible` finds no suitable reducers.
3. `IsReducible` checks all elements $r_{\mathrm{red}} \in G$ such that $\mathrm{index}(r_{\mathrm{red}}) = i$.
   (a) If there exists $u_{\mathrm{red}} \in T$ such that $u_{\mathrm{red}}\mathrm{HT}(r_{\mathrm{red}}) = \mathrm{HT}(r)$ then $u_{\mathrm{red}}\mathcal{S}(r)$ is checked by both Faugère's Criterion and the Rewritten Criterion.

[1] Failing to record the signature of a polynomial reduced to zero is an implementation error that can lead to an infinite loop.

(α) If neither criterion holds, the reduction takes place, but a further check is necessary to preserve $\mathcal{S}(r) = \mathrm{ST}(r)\mathbf{F}_i$. If $\mathcal{S}(r) \succ u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}})$, then it rewrites $\mathrm{poly}(r)$:

$$r = \big(\mathcal{S}(r), \mathrm{poly}(r) - u_{\mathrm{red}}\mathrm{poly}(r_{\mathrm{red}})\big).$$

If $\mathcal{S}(r) \prec u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}})$, then r is not changed, but a new labeled polynomial is computed and added to F for further reductions,

$$r' = \big(u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}}), u_{\mathrm{red}}\mathrm{poly}(r_{\mathrm{red}}) - \mathrm{poly}(r)\big).$$

The algorithm adds $\mathcal{S}(r')$ to the list of rules and continues with r.

(β) If $u_{\mathrm{red}}r_{\mathrm{red}}$ is detected by one of the criteria, then the reduction does *not* take place, and the search for a reducer continues.

(b) If there is no possible reducer left to be checked then r is added to G if $\mathrm{poly}(r) \neq 0$.

Note that if $\mathcal{S}(r) = u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}})$ then $u_{\mathrm{red}}r_{\mathrm{red}}$ is rewritable by r, thus Case (3)(a)(β) avoids this situation.

3.2 What is the problem with termination?

The difficulty with termination arises from Case (3)(a)(β) above.

Situation 15. *Assume that*

1. *the number of reducers whose head terms divide* $\mathrm{HT}(r)$ *is not zero, and*
2. *all such reducers are rejected by one of the two criteria.*

Then F5 adds r *to* G *even though* $\mathrm{poly}(r)$ *is redundant in* $\mathrm{poly}(G)$*. Moreover, no new polynomial is added to* $\mathrm{poly}(G)$ *which is not already generated by* $HT(\mathrm{poly}(G))$*.*

Example 16. Situation 15 is not a mere hypothetical: as described in Section 3.5 of [17], an example appears in Section 8 of [16], which computes a Gröbner basis of $(yz^3 - x^2t^2, xz^2 - y^2t, x^2y - z^2t)$. Without repeating the details, at degree 7, F5 adds r_8 to G, with $\mathrm{HT}(r_8) = y^5t^2$. At degree 8, however, `Reduction` returns $R_8 = \{r_{10}\}$, with $\mathrm{HT}(r_{10}) = y^6t^2$. This is due to the fact that the reduction of r_{10} by yr_8 is rejected by the algorithm's criteria, and the reduction does *not* take place. In other words, r_{10} is added to G even though $\mathrm{poly}(r_{10})$ is redundant in $\mathrm{poly}(G)$.

Definition 17. A labeled polynomial r computed in F5 is called *redundant* if, when `Reduction` returns r, we have $\mathrm{poly}(r)$ redundant w.r.t. $\mathrm{poly}(G)$.

Lemma 18. *In Situation 15, at least one of the rejected reducers of* r *is a non-redundant element for* G*.*

Proof. If a reducer r_j of r is redundant, then there has to exist another element r_k such that $\mathrm{HT}(r_k) \mid \mathrm{HT}(r_j)$ and thus $\mathrm{HT}(r_k) \mid \mathrm{HT}(r)$. Follow this chain of divisibility down to the minimal degree; we need to show that there do not exist two polynomials r_j, r_k such that $\mathrm{HT}(r_j) = \mathrm{HT}(r_k)$. Assume to the contrary that r_k is computed before r_j and $\mathrm{HT}(r_j) = \mathrm{HT}(r_k)$, because the reduction of r_j by r_k in `IsReducible` was forbidden. There are three possibilities:

1. If $\mathrm{index}(r_k) > \mathrm{index}(r_j)$, to the contrary, `IsReducible` cannot interfere with this reduction, because such reductions are always carried out by the normal form computation in `Reduction`.

2. If r_k is rejected by the Rewritten Criterion, then there exists r' such that $\mathrm{ST}(r') \mid \mathrm{ST}(r_k)$. Thus r' must have been computed after r_k. As F5 computes incrementally on the degree and $\mathrm{ST}(r') \mid \mathrm{ST}(r_k)$, it follows that $\deg(r') = \deg(r_k)$. Hence $\mathrm{ST}(r') = \mathrm{ST}(r_k)$. Thus the Rewritten Criterion would have rejected the computation of r', again a contradiction.

3. If $\mathcal{S}(r_k)$ is rejected by Faugère's Criterion, to the contrary, r_k should not have been computed in the first place.

Thus $\mathrm{HT}(r_j) \neq \mathrm{HT}(r_k)$. It follows that we arrive at a non-redundant reducer after finitely many steps. □

Lemma 19. *Denote by R_d the result of* `Reduction` *at degree d. There exists an input $F = (f_1, \ldots, f_m)$ and a degree d such that if* $\mathrm{poly}(G)$ *is a $(d-1)$-Gröbner basis of $\langle f_1, \ldots, f_m \rangle$, then*

(A) $R_d \neq \emptyset$, and

(B) $HT(\mathrm{poly}(G \cup R_d)) = HT(\mathrm{poly}(G))$.

Proof. Such an input F is given in Example 16: once reduction concludes for $d = 8$, $\mathrm{HT}(r_8) \mid \mathrm{HT}(r_{10})$, so $\mathrm{HT}(\mathrm{poly}(G)) = \mathrm{HT}(\mathrm{poly}(G \cup R_8))$. □

Remark 20. In [16, Corollary 2], it is argued that termination of F5 follows from the (unproved) assertion that for any d, if no polynomial is reduced to zero, then $\mathrm{HT}(\mathrm{poly}(G)) \neq \mathrm{HT}(\mathrm{poly}(G \cup R_d))$. But in Example 16, $\mathrm{HT}(\mathrm{poly}(G)) = \mathrm{HT}(\mathrm{poly}(G \cup R_8))$, *even though there was no reduction to zero!* Thus, Corollary 2 of [16] is incorrect: termination of F5 is unproved, *even for regular sequences*, as there could be infinitely many steps where new redundant polynomials are added to G. By contrast, a Buchberger-style algorithm *always* expands the monomial ideal when a polynomial does not reduce to zero; this ensures its termination.

Having shown that there *is* a problem with termination, we can now turn our attention to devising a solution.

3.3 To sort the wheat from the chaff . . . isn't that easy!

The failure of F5 to expand the ideal of leading monomials raises the possibility of an infinite loop of redundant labeled polynomials. However, we cannot ignore them.

Example 21. Suppose we modify the algorithm to discard critical pairs with at least one redundant labeled polynomial. Consider a polynomial ring in a field of characteristic 7583.

1. For Katsura-5, the algorithm no longer terminates, but computes an increasing list of polynomials with head terms $x_2^k x_4$ with signatures $x_2 x_3^k x_5 x_6$ for $k \geq 1$.

2. For Cyclic-8, the algorithm terminates, but its output is not a Gröbner basis!

How can critical pairs involving "redundant" polynomials can be necessary?

Definition 22. A critical pair (r_i, r_j) is a *GB-critical pair* if neither r_i nor r_j is redundant. If a critical pair is not a GB-critical pair, then we call it an *F5-critical pair*.

We now come to the main theoretical result of this paper.

Theorem 23. *If (r_i, r_j) is an F5-critical pair, then one of the following statements holds at the moment of creation of r_{ij}:*

(A) $\mathrm{poly}(r_{ij})$ *already has a standard representation.*

(B) There exists a GB-critical pair (r_k, r_ℓ), a set $W \subset \{1, \ldots, n_G\}$, and terms λ_w (for all $w \in W$) such that

$$\mathrm{poly}(r_{ij}) = \mathrm{poly}(r_{k\ell}) + \sum_w \lambda_w \mathrm{poly}(r_w), \tag{3.1}$$

$\gamma_{ij} = \gamma_{k\ell}$ and $\gamma_{k\ell} > \lambda_w \mathrm{HT}(r_w)$ for all w.

Lemma 23 implies that an F5-critical pair *might not* generate a redundant polynomial: it might rewrite a GB-critical pair which is *not* computed. In terms of the Macaulay matrix [16, 20], we can think of an F5-critical pair as a pair of rows where one row, generated by a redundant element of the basis, is preferred over another row with the same signature, generated by a non-redundant element of the basis. Due to this choice, the notions of "redundant" and "necessary" critical pairs are somewhat ambiguous in F5: necessary for a Gröbner basis, or for a correct reduction? On the other hand, the notions of F5- and GB-critical pairs are absolute.

To prove Theorem 23, we need the following observation:

Lemma 24. *Let $r_i, r_j \in G$ computed by F5, and assume that $\mathrm{HT}(r_j) \mid \mathrm{HT}(r_i)$. Then* `Spol` *does not generate an s-polynomial for (r_i, r_j).*

Proof. We have assumed that the input is interreduced, so $\mathrm{poly}(r_i)$ is not in the input. Since $\mathrm{HT}(r_j) \mid \mathrm{HT}(r_i)$ there exists $u \in T$ such that $u\mathrm{HT}(r_j) = \mathrm{HT}(r_i)$. Since the reduction of $\mathrm{poly}(r_i)$ by $u\mathrm{poly}(r_j)$ was rejected, $u\mathcal{S}(r_j)$ was detected by one of the criteria. It will be detected again in `CritPair` or `Spol`. Thus `Spol` will not generate r_{ij}. □

Proof of Theorem 23. Assume that r_i and r_j are both redundant; the case where only r_i (resp. r_j) is redundant is similar. By Lemma 18 there exists for r_i (resp. r_j) at least one non-redundant reducer r_k (resp. r_ℓ). By Lemma 24, we may assume that r_i and r_j are of degree smaller than r_{ij}. Using the fact that $\mathrm{poly}(G)$ is a d-Gröbner basis for $d = \max(\deg r_i, \deg r_j)$, we can write

$$\begin{aligned}\mathrm{poly}(r_i) &= \lambda_{ik}\mathrm{poly}(r_k) + \sum_{u\in U} \lambda_u \mathrm{poly}(r_u)\\ \mathrm{poly}(r_j) &= \lambda_{j\ell}\mathrm{poly}(r_\ell) + \sum_{v\in V} \lambda_v \mathrm{poly}(r_v),\end{aligned}$$

such that

$$\begin{aligned}\mathrm{HT}(r_i) &= \lambda_{ik}\mathrm{HT}(r_k) > \lambda_u \mathrm{HT}(r_u) \text{ and}\\ \mathrm{HT}(r_j) &= \lambda_{j\ell}\mathrm{HT}(r_\ell) > \lambda_v \mathrm{HT}(r_v)\end{aligned}$$

where $U, V \subset \{1, \ldots, n_G\}$. As $\gamma_{k\ell} \mid \gamma_{ij}$, the representations of $\mathrm{poly}(r_i)$ and $\mathrm{poly}(r_j)$ above imply that there exists $\lambda \in T$ such that

$$\begin{aligned}\mathrm{poly}(r_{ij}) &= \frac{\gamma_{ij}}{\mathrm{HT}(r_i)}\mathrm{poly}(r_i) - \frac{\gamma_{ij}}{\mathrm{HT}(r_j)}\mathrm{poly}(r_j)\\ &= \lambda \mathrm{poly}(r_{k\ell}) + \sum_{w\in W} \lambda_w \mathrm{poly}(r_w)\end{aligned} \tag{3.2}$$

where $W = U \cup V$ and $\lambda_w = \lambda\lambda_u$ for $w \in U \backslash V$, $\lambda_w = \lambda\lambda_v$ for $w \in V \backslash U$, and $\lambda_w = \lambda(\lambda_u - \lambda_v)$ for $w \in U \cap V$. In Equation (3.2) we have to distinguish two cases:

1. If $\lambda > 1$ then $\deg(r_{k\ell}) < \deg(r_{ij})$, thus $r_{k\ell}$ is already computed (or rewritten) using a lower degree computation, which has already finished. It follows that there exists a standard representation of $\text{poly}(r_{k\ell})$ and thus a standard representation of $\text{poly}(r_{ij})$.
2. If $\lambda = 1$ then (A) holds if $\text{poly}(r_{kl})$ is already computed by F5; otherwise (B) holds.

□

We can now explain why discarding redundant polynomials wreaks havoc in the algorithm.

Situation 25. *Let (r_i, r_j) be an F5-critical pair. Suppose that all GB-critical pairs (r_k, r_ℓ) corresponding to case (B) of Lemma 23 are rejected by one of F5's criteria, but lack a standard representation.*

Situation 25 is possible if, for example, the Rewritten Criterion rejects all the (r_k, r_ℓ).

Corollary 26. *In Situation 25 it is necessary for the correctness of F5 to compute a standard representation of r_{ij}.*

Proof. Since $\text{poly}(r_{k\ell})$ lacks a standard representation, and the algorithm's criteria have rejected the pair (r_k, r_ℓ), then it is necessary to compute a standard representation of r_{ij}. Once the algorithm does so, we can rewrite (3.1) to obtain a standard representation of $\text{poly}(r_{k\ell})$. □

In other words, "redundant" polynomials are necessary in F5.

4 Variants that ensure termination

Since we cannot rely on an expanding monomial ideal, a different approach to ensure termination could be to set or compute a degree bound. Since a Gröbner basis is finite, its elements have a maximal degree. Correspondingly, there exists a maximal possible degree d_{GB} of a critical pair that generates a necessary polynomial. Once we complete degree d_{GB}, no new, non-redundant data for the Gröbner basis would be computed from the remaining pairs, so we can terminate the algorithm. The problem lies with identifying d_{GB}, which is rarely known beforehand, if ever.

Before describing the new variant that follows from these ideas above, we should review two known approaches, along with some drawbacks of each.

4.1 F5t: Reduction to zero

In [17], Gash suggests the following approach, which re-introduces a limited amount of reduction to zero. Once the degree of the polynomials exceeds $2M$, where M is the Macaulay bound for regular sequences [2, 20], start storing redundant polynomials in a set D. Whenever subalgorithm `Reduction` returns a nonempty set R_d that does not expand the ideal of leading monomials, reduce all elements of R_d completely w.r.t. $G \cup D$ and store any non-zero results in D instead of adding them to G. Since complete reduction can destroy the relationship between a polynomial and its signature, the rewrite rules that correspond to them are also deleted. Subsequently, s-polynomials built using an element of D are reduced without regard to criterion, and those that do not reduce to zero are also added to D, generating new critical pairs. Gash called the resulting variant F5t.

One can identify four drawbacks of this approach:

1. The re-introduction of zero-reductions incurs a performance penalty. In Gash's experiments, this penalty was minimal, but these were performed on relatively small systems without many redundant polynomials. In some systems, such as Katsura-9, F5 works with hundreds of redundant polynomials.

2. It keeps track of two different lists for generating critical pairs and uses a completely new reduction process. An implementation must add a significant amount of complicated code beyond the original F5 algorithm.

3. It has to abandon some signatures due to the new, signature-corrupting reduction process. Thus, a large number of unnecessary critical pairs can be considered.

4. The use of $2M$ to control the size of D is an imprecise, ad-hoc patch. In some experiments from [17], F5t terminated on its own before polynomials reached degree $2M$; for other input systems, F5t yielded polynomials well beyond the $2M$ bound, and a higher bound would have been desirable.

4.2 F5B: Use Buchberger's lcm criterion

In [1], Ars suggests using Buchberger's lcm criterion to determine a degree bound.

- Initialize a global variable $d_B = 0$ storing a degree.
- Keep a second list of critical pairs, P^*, used *only* to determine a degree bound.
- When adding new elements to G, store a copy of each critical pair not detected by Buchberger's lcm criterion in P^*. Remove any previously-stored pairs that are detected by Buchberger's lcm criterion, and store the highest degree of an element of P^* in d_B.

If the degree of all critical pairs in P exceeds d_B, then a straightforward application of Buchberger's lcm criterion implies that the algorithm has computed a Gröbner basis, so it can terminate. We call this variant F5B.

It is important to maintain the distinction between the two lists of critical pairs. Otherwise, the correctness of the algorithm is no longer assured: Buchberger's criteria ignore the signatures, so P^* lacks elements needed on account of Situation 25.

While elegant, this approach has one clear drawback. *Every* critical pair is computed and checked twice: once for Buchberger's lcm criterion, and again for the F5 criteria. Although Faugère's Criterion also checks for divisibility, it checks only polynomials of smaller index, whereas Buchberger's criterion checks *all* polynomials, and in most systems the number of polynomials of equal index is much larger than the total of all polynomials having lower index. Indeed, we will see in Section 4.5 that this seemingly innocuous check can accumulate a significant time penalty. This would be acceptable if the algorithm routinely used d_B to terminate, but F5 generally terminates from its own internal mechanisms *before* $d = d_B$! Thus, except for pathological cases, the penalty for this short-circuiting mechanism is not compensated by a discernible benefit.

4.3 F5+: Use F5's criteria on non-redundant critical pairs

We now describe a variant that uses information from F5 itself, along with the theory developed in Section 3, to reduce, if not eliminate, the penalty necessary to force termination. We restate only those algorithms of [16] that differ from the original (and the differences are in fact minor).

The fundamental motivation of this approach stems from the fact that a polynomial is redundant if and only if `TopReduction` rejects a reductor on account of one of the F5 criteria. Understood correctly, this means that F5 "knows" at this point whether a polynomial is redundant. We would like to ensure that it does not "forget" this fact. As long as this information remains available to the algorithm, identifying GB- and F5-critical pairs will be trivial. Thus, our tasks are:

1. Modify the data structures to flag a labeled polynomial as redundant or non-redundant.
2. Use this flag to distinguish F5- and GB-critical pairs.
3. Use the GB-critical pairs to decide when to terminate.

We address each of these in turn.

To distinguish between redundant and non-redundant labeled polynomials, we add a third, boolean field to the structure of a labeled polynomial. We mark a redundant labeled polynomial with $b = 1$, and a non-redundant one with $b = 0$. Without loss of generality, the inputs are non-redundant, so the first line of subalgorithm `F5` can change to

$r_i := (\mathbf{F}_i, f_i, 0) \in R \times \{0, 1\}$

For all other labeled polynomials, the value of b is defined by the behaviour of the `Reduction` subalgorithm; see below.

The next step is to detect redundant polynomials; we do this in `IsReducible`. In an unmodified F5, the return value of `IsReducible` is either a labeled polynomial r_{i_j} (a polynomial that reduces r) or $\emptyset$. The return value $\emptyset$ can have two meanings:

1. There exists no reducer of the input.
2. There exist reducers of the input, but their reductions are rejected.

Algorithm 1, which replaces the original `IsReducible` subalgorithm, distinguishes these two possibilities by adding a boolean to the output: $b = 0$ in case (1) and $b = 1$ otherwise. We also need to modify subalgorithm `TopReduction` to use this new data; see Algorithm 2.

We now describe the main routine of the new variant, which fulfills the following conditions:

1. Compute as low a degree bound as possible.
2. Minimize any penalty to the algorithm's performance.

An easy way to estimate d_0 would be to compute the highest degree of a GB-critical pair. Although this would be correct, experience suggests that, in general, it is much higher than necessary (see Table 2 in Section 4.5). Instead, the new variant will use the criteria of the F5 algorithm to identify GB-critical pairs that *probably* reduce to zero. How can we identify such pairs? The following method seems intuitively correct: *when all GB-critical pairs are rejected by one of the F5 criteria.*

However, Situation 25 implies that this intuition may be *in*correct. Thus, *once the algorithm reaches that degree* (and not earlier), it uses Buchberger's lcm criterion to decide whether the remaining GB-critical pairs reduce to zero. If it can verify this, then the algorithm can terminate.

This differs from the approach of [1] in two important ways.

1. Rather than checking all pairs against the lcm criterion, it checks only GB-critical pairs that F5 also rejects as unnecessary. After all, it follows from Lemma 23 that F5-critical pairs can be necessary *only if they substitute for a GB-critical pair.*

Algorithm 1 `IsReducible`

Input: r_{i_0}, a labeled polynomial of R; $G = [r_{i_1}, \ldots, r_{i_r}]$; $k \in \mathbb{N}$; φ, a normal form

```
b := 0
for j from 1 to r do
  if (u := HT(r_{i_0}) / HT(r_{i_j}) ∈ T) then
    if (neither criterion detects (r_{i_0}, r_{i_j})) then
      return (r_{i_j}, 0)
    else
      b := 1
return (∅, b)
```

Algorithm 2 `TopReduction`

Input: r_{k_0}, a labeled polynomial of R; G, a list of elements of R; $k \in \mathbb{N}$; φ, a normal form

```
if poly(r_{k_0}) = 0 then
  return (∅, ∅)
(r', b) := IsReducible(r_{k_0}, G, k, φ)
if r' = ∅ then
  r_{k_0} := (S(r_{k_0}), 1/HC(r_{k_0}) poly(r_{k_0}), b)
  return (r_{k_0}, ∅)
else
  r_{k_1} = r'
  u := HT(r_{k_0}) / HT(r_{k_1})
  if uS(r_{k_1}) ≺ S(r_{k_0}) then
    r_{k_0} := (S(r_{k_0}), poly(r_{k_0}) − u poly(r_{k_1}), b)
    return (∅, {r_{k_0}})
  else
    N := N + 1
    r_N := (uS(r_{k_1}), u poly(r_{k_1}) − poly(r_{k_0}), b)
    Add Rule (r_N)
    return (∅, {r_N, r_{k_0}})
```

2. It checks the GB-critical pairs only once the F5 criteria suggest that it should terminate.

We call this variant F5+; see Algorithm 3.

Remark 27. An implementation of F5+ has to take care when checking Buchberger's lcm criterion, on account of the phenomenon of *Buchberger triples* [3, p. 229]. In [1], this is implemented similarly to the "`Update`" algorithm of [3, 18]. The current F5+ takes a more traditional route; it records all critical pairs that have generated s-polynomials. The burden on memory is minimal.

Algorithm 3 `F5+`

Input: $\begin{cases} i \in \mathbb{N} \\ f_i \in \mathcal{K}[\underline{x}] \\ G_{i+1} \subset \mathcal{K}[\underline{x}], \text{ such that } \mathrm{poly}(G_{i+1}) \text{ is a Gröbner basis of } \mathrm{Id}(f_{i+1}, \ldots, f_m) \end{cases}$

$r_i := (\mathbf{F}_i, f_i, 0)$
$\varphi_{i+1} := \mathrm{NF}(., \mathrm{poly}(G_{i+1}))$
$G_i := G_{i+1} \cup \{r_i\}$
$\{P$ is the usual set of pairs; P^* is the set of GB-pairs detected by the F5 criterion$\}$
$P := \emptyset$
$P^* := \emptyset$
for $r_j \in G$ **do**
  $p := \mathtt{CritPair}(r_i, r_j, i, \varphi_{i+1})$
  if $p = \emptyset$ and r_j non-redundant **then**
    Add $(\mathrm{lcm}(\mathrm{HT}(\mathrm{poly}(r_i)), \mathrm{HT}(\mathrm{poly}(r_j))), r, r_j)$ to P^*
  else
    Add p to P
Sort P by degree
while $P \neq \emptyset$ **do**
  $d := \deg(\mathrm{first}(P))$
  Discard from P^* all pairs that are not of maximal degree
  if $d \leq \max\{\deg(p) : p \in P^*\}$ or $\exists p \in P^*$ that does not satisfy Buchberger's lcm criterion **then**
    $P_d := \{p \in P : \deg(p) = d\}$
    $P := P \backslash P_d$
    $F := \mathtt{Spol}(P_d)$
    $R_d := \mathtt{Reduction}(F, G_i, i, \varphi_{i+1})$
    for $r \in R_d$ **do**
      for $r_j \in G_i$ **do**
        $p := \mathtt{CritPair}(r, r_j, i, \varphi_{i+1})$
        if $p = \emptyset$ and r, r_j both non-redundant **then**
          Add $(\mathrm{lcm}(\mathrm{HT}(\mathrm{poly}(r)), \mathrm{HT}(\mathrm{poly}(r_j))), r, r_j)$ to P^*
        else
          Add p to P
      $G_i := G_i \cup \{r\}$
    Sort P by degree
  else
    $P := \emptyset$
return G_i

4.4 Correctness and termination of F5+

As a last step we have to show that F5+ terminates correctly.

Theorem 28. *If F5+ terminates, the result is a Gröbner basis of the input.*

Proof. This follows from Buchberger's lcm criterion. □

Theorem 29. *For a given homogeneous ideal I as input, F5+ terminates after finitely many steps.*

Proof. Assume that F5+ has already computed G such that $\mathrm{poly}(G)$ is a d_0-Gröbner basis in the i-th iteration step of F5+, and $\#P_d < \infty$ for $d > d_0$ at this point. Assume d to be the minimal degree of elements in P. First of all we must verify that at each degree there are only finitely many new elements computed. Three different situations can arise for an arbitrary r of degree d:

1. If $\varphi(\mathrm{poly}(r), \mathrm{poly}(G_{i+1})) = 0$, no new critical pair is generated.
2. If $\varphi(\mathrm{poly}(r), \mathrm{poly}(G_{i+1})) \neq 0$ then `IsReducible` checks for possible reducers:
   (a) If no non-rejected reducer is found, then r is added to G. All newly computed critical pairs generated by r have degree $> d$; their number is finite because G is currently finite.
   (b) If there exists a reducer r_{red} with multiplier $u_{\mathrm{red}} \in T$ such that $\mathcal{S}(r) \succ u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}})$ then $\mathrm{poly}(r) - u_{\mathrm{red}}\mathrm{poly}(r_{\mathrm{red}})$ is the new polynomial part with lower head term of r, and r is checked for further reductions.
   (c) If there exist r_{red} and u_{red} such that $u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}}) \succ \mathcal{S}(r)$ then r is not changed, but kept for further reduction checks. A new, non-redundant element $r' = (u_{\mathrm{red}}\mathcal{S}(r_{\mathrm{red}}), \mathrm{poly}(r) - u_{\mathrm{red}}\mathrm{poly}(r_{\mathrm{red}}))$ is generated, and its signature $\mathcal{S}(r')$ added to the list of rules.
   There are only finitely many different reducers which could lead to new elements r'. Since $\mathcal{S}(r')$ was added to the list of rules, r_{red} will not be chosen again as a reducer of r. As $\mathrm{HT}(r') < \mathrm{HT}(r)$, the process of initializing new elements of higher signature stops after finitely many steps due to $<$ being a fixed admissible ordering on T.
   Thus only finitely many new elements of degree d_0 can be generated.

It follows that in each degree step only finitely many new polynomials are computed, so only finitely many new critical pairs are generated.

To finish the proof we have to show that after finitely many steps, only F5-critical pairs are left in P. There can only be finitely many GB-critical pairs as their generating labeled polynomials have to be non-redundant. Since R is Noetherian, only finitely many non-redundant polynomials can be computed. The above discussion implies that F5+ cannot compute infinitely many redundant elements between two non-redundant ones.

Thus F5+ terminates after finitely many steps. □

4.5 Experimental results

We implemented these variants in the SINGULAR kernel to compare performance. (The F5 implementation in SINGULAR is still under development.) In Table 1 we compare timings for some examples. In Table 2 we compare degree bounds. All systems are homogeneous and computed over a field of characteristic 32003. This data was recorded from a workstation running Gentoo Linux on an Intel® Xeon® X5460 CPU at 3.16GHz with 64 GB RAM.

Table 1 shows that the tests for F5+ do not slow it down significantly. But this is expected, since the modifications add trivial overhead, and rely primarily on information that the algorithm already has available.

Table 2 bears some discussion. We have implemented F5+ in two different ways. Both are the same in that they estimate the maximum necessary degree by counting the maximal degree of a GB-critical pair not discarded by the `CritPair` subalgorithm. However, one can implement a slightly more efficient `CritPair` algorithm by discarding pairs that pass Faugère's Criterion, but are rewritable. (The basic F5 checks the Rewritten Criterion only in subalgorithm `Spol`.) Thus one might compute a different maximal degree of P^* in each case: when `CritPair` discards only those

Table 1: Timings (in seconds) of F5, F5B, and F5+

Examples	F5	F5B	F5+	F5/F5B	F5/F5+
`Katsura-9`	39.95	53.97	40.23	0.74	0.99
`Katsura-10`	1,145.47	1,407.92	1,136.43	0.80	1.00
`F-855`	9,831.81	11,364.47	9,793.17	0.86	1.00
`Eco-10`	47.26	57.97	46.67	0.82	1.01
`Eco-11`	1,117.13	1,368.44	1,072.47	0.82	1.04
`Cyclic-7`	6.24	9.18	6.21	0.67	1.00
`Cyclic-8`	3,791.54	4,897.61	3,772.66	0.77	1.00

Table 2: Degrees of F5, F5B, and F5+

Examples	d_{maxGB} [1]	d_{F5} [2]	$d_{\text{GB-pair}}$ [3]	d_B	d_F [4]	d_{FR} [5]
`Katsura-9`	13	16	21	13	16	16
`Katsura-10`	15	18	26	15	18	18
`F-855`	14	18	20	17	17	16
`Eco-10`	15	20	23	17	17	17
`Eco-11`	17	23	26	19	19	19
`Cyclic-7`	19	23	28	24	23	21
`Cyclic-8`	29	34	41	33	32	30

[1] maximal degree in GB
[2] observed degree of termination of F5
[3] maximal degree of a GB-critical pair
[4] maximal degree of all GB-critical pairs not detected by Faugère's Criterion
[5] maximal degree of all GB-critical pairs not detected by Faugère's Criterion *or* the Rewritten Criterion

pairs detected by Faugère's Criterion, we designate the maximal degree of P^* as d_F; when `CritPair` discards pairs detected by the Rewritten Criterion as well, we designate the maximal degree of P^* as d_{FR}. We denote the degree where the original F5 terminates by d_{F5}, and the maximal degree of a polynomial generated by d_{maxGB}.

Note that $d_{\text{maxGB}} < d_{\text{F5}}$ because F5 terminates after emptying the set P of critical pairs, and for the last few degrees, `Spol` usually rejects any pairs in P as rewritable. This need not be the case in general; for the trivial system $\{x^2+1, y^2+1\}$, no critical pairs are ever added to P, so F5 terminates immediately.

On the other hand, it is always the case that $d_F, d_{FR} \leq d_{\text{F5}}$; d_{F5} counts F5-critical pairs as well as GB-critical pairs, whereas d_F, d_{FR} count only GB-critical pairs that are not rejected by one or both of the F5 criteria. Thus F5+ always starts its manual check for termination no later than F5 would terminate, and sometimes terminates before F5. For example, the termination mechanisms activate for F-855, Eco-10 and -11, and Cyclic-8, so F5B and F5+ both terminate at lower degree than F5. With little to no penalty, F5+ terminates first, but F5B terminates *well after* F5 in spite of the lower degree!

In other examples, F5 terminates *before* reaching the minimal degree; in other words, $d_{\text{F5}} \leq$

$\min\{d_B, d_F, d_{FR}\}$. Hence, the termination mechanism does not kick in for these systems. Even though $d_{\text{maxGB}} = d_B < d_F = d_{FR} = d_{\text{F5}}$ for Katsura-n, the termination mechanism of F5+ incurs almost no penalty, so its timings are equivalent to those of F5, whereas F5B is slower.

5 Concluding remarks, and a conjecture

The new variant of F5 presented here is a straightforward solution to the problem of termination: it distinguishes F5- and GB-critical pairs and tracks the highest degree of a necessary GB-critical pair. Thus F5+ provides a self-generating, correct, and efficient termination mechanism in case F5 does not terminate for some systems. In practice, F5+ terminates before reaching the degree cutoff, but it is not possible to test all systems, nor practical to determine *a priori* the precise degree of each Gröbner basis. The question of whether F5, as presented in [16], terminates correctly on all systems, or even on all regular systems, remains an important open question.

The following conjecture arises from an examination of Table 2.

Conjecture 30. *The F5 algorithm can terminate once all GB-critical pairs are rejected by the F5 criteria. That is, it can terminate once $d = d_{FR}$.*

Conjecture 30 is *not* a Corollary of Theorem 12! There, correctness follows only if *all* critical pairs are rejected by the algorithm: GB- *and* F5-critical pairs. Similarly, a proof of Conjecture 30 would imply that we could drop altogether the check of Buchberger's criteria.

If one could show that $d_{\text{maxGB}} \leq d_{FR}$, Conjecture 30 would follow immediately. However, such a proof is non-trivial, and lies beyond the scope of this paper. The conjecture may well be false even if we replace d_{FR} by d_F, although we have yet to encounter a counterexample. The difficulty lies in the possibility that Situation 25 applies.

6 Acknowledgements

The authors wish to thank Martin Albrecht, Gerhard Pfister and Stefan Steidel for helpful discussions. Moreover, we would also like to thank the SINGULAR team at TU Kaiserslautern for their technical support.

References

[1] Gwénolé Ars. *Applications des bases de Gröbner à la cryptographie*. PhD thesis, Université de Rennes I, 2005.

[2] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. Asympotic expansion of the degree of regularity for semi-regular systems of equations. Manuscript downloaded from `www-calfor.lip6.fr/~jcf/Papers/BFS05.pdf`.

[3] Becker, T., Weispfenning, V., and Kredel, H. *Gröbner Bases*. Springer Verlag, 1993.

[4] Bosma, W., Cannon, J., and Playoust, C. The Magma algebra system. I. The user language. *Journal of Symbolic Computation*, 24(3-4):235–265, 1997. http://magma.maths.usyd.edu.au/magma/.

[5] Michael Brickenstein. Slimgb: Gröbner bases with slim polynomials. In *Proceedings of the Rhine Workshop on Computer Algebra (RWCA 06)*, pages 55–66, 2006.

[6] Brickenstein, M. and Dreyer, A. PolyBoRi: A framework for Gröbner basis computations with Boolean polynomials. *Journal of Symbolic Computation*, 44(9):1326–1345, September 2009.

[7] Buchberger, B. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, University of Innsbruck, 1965.

[8] Buchberger, B. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. *Aequ. Math.*, 4(3):374–383, 1970.

[9] Buchberger, B. A criterion for detecting unnecessary reductions in the construction of Gröbner bases. In *EUROSAM '79, An International Symposium on Symbolic and Algebraic Manipulation*, volume 72 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 1979.

[10] Decker, W., Greuel, G.-M., Pfister, G., and Schönemann, H. SINGULAR *3-1-1 — A computer algebra system for polynomial computations*, 2010. http://www.singular.uni-kl.de.

[11] Decker, W. and Lossen, C. *Computing in Algebraic Geometry - A Quick Start in* SINGULAR. ACM 16, Springer Verlag, 2006.

[12] Eder, C. On the criteria of the F5 Algorithm. *preprint math.AC/0804.2033*, 2008.

[13] Eder, C. and Perry, J. F5C: A Variant of Faugère's F5 Algorithm with reduced Gröbner bases. *Journal of Symbolic Computation*, to appear. `dx.doi.org/10.1016/j.jsc.2010.06.019`.

[14] Jean-Charles Faugère. Cryptochallenge 11 is broken or an efficient attack of the C* cryptosystem. Technical report, LIP6/Universitè Paris, 2005.

[15] Faugère, J.-C. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra (Elsevier Science)*, 139(1):61–88, June 1999.

[16] Faugère, J.-C. A new efficient algorithm for computing Gröbner bases without reduction to zero F5. In *ISSAC 2002, Villeneuve d'Ascq, France*, pages 75–82, July 2002. Revised version from `http://fgbrs.lip6.fr/ jcf/Publications/index.html`.

[17] Gash, J. M. *On efficient computation of Grobner bases.* PhD thesis, University of Indiana, 2008.

[18] Gebauer, R. and Möller, H. M. On an installation of Buchberger's algorithm. *Journal of Symbolic Computation*, 6(2-3):275–286, October/December 1988.

[19] Greuel, G.-M. and Pfister, G. *A* SINGULAR *Introduction to Commutative Algebra.* Springer Verlag, 2nd edition, 2007.

[20] Daniel Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In J. A. van Hulzen, editor, *EUROCAL'83, European Computer Algebra Conference*, volume 162 of *Springer LNCS*, pages 146–156, 1983.

[21] Möller, H.M., Traverso, C., and Mora, T. Gröbner bases computation using syzygies. In *ISSAC 92: Papers from the International Symposium on Symbolic and Algebraic Computation*, pages 320–328, 1992.

[22] Stegers, T. Faugère's F5 Algorithm revisited. Master's thesis, Technische Univerität Darmstadt, revised version 2007.