A Heuristic Selection Strategy for Lexicographic Gröbner Bases ?

S. R. Czapor

Department of Mathematics, Statistics and Computing Science
Dalhousie University
Halifax, Nova Scotia, Canada B3H 3J5

ABSTRACT

It is well known that the computation of lexicographic Gröbner bases using Buchberger's algorithm is more difficult than the computation of Gröbner bases with respect to total degree orderings. The lexicographic algorithm is particularly susceptible to the problem of intermediate expression swell; that is, intermediate polynomials may be far larger than those which make up the final basis. To some extent, this is a function of "selection strategy", i.e. the order in which S-polynomials are used to extend a partial basis. We argue and provide empirical evidence that *for the lexicographic ordering* (in direct contrast to the case of degree orderings), a simple *heuristic* strategy will in practice control intermediate growth more effectively than the normal strategy based on the lexicographic term ordering alone. The result is usually a much more efficient computation, even for ideals of nonzero dimension (where lexicographic bases are uniquely well suited to solving the corresponding algebraic systems of equations).

1. Introduction

The use of Gröbner basis techniques [3] for solving systems of algebraic equations has become increasingly popular over the past several years. This is due both to an increased awareness of Gröbner bases in the mathematical community, and to various important advances in their computation (e.g. [2]). Of course, the actual process for obtaining solutions from a Gröbner basis depends on the term ordering used. Lexicographic Gröbner bases are particularly useful in that they give (in a sense) the closest possible analogue to the triangulation of a linear system of equations (see [3]).

However, it is also well known that lexicographic bases are generally much more difficult to compute than those with respect to degree-based orderings (e.g. the "total degree" ordering of [3]). For example, the former computation is doubly exponential in the degree of the input while the latter is singly exponential (see [9]). In addition, since the lexicographic process is an elimination, it should be carried out with respect to a permutation of variables in which this elimination is relatively easy. Such a selection is often difficult to find even when one (or more) exists.

For the above reasons, the lexicographic method has often been avoided in favour of a less straightforward procedure [3] which computes the solutions from a basis with respect to an arbitrary ordering. Recently, an important advance in this approach has been obtained by Faugère et al. [9], who give an algorithm for the conversion of Gröbner bases with respect to an arbitrary admissible order into lexicographic

This research was supported by the Natural Sciences and Engineering Research Council of Canada under Grant A8967, and by the Killam Trust through the award of a Postdoctoral Fellowship.

bases. In this way, the explicit representation offered by the lexicographic basis is still achieved. It is crucial to note, however, that these methods are only applicable to systems with *finitely many* solutions. Unfortunately, there do arise systems of equations with infinitely many solutions for which the direct approach is necessary.

Beyond the above consideration, it should also be remembered that there do exist systems of equations for which the lexicographic Gröbner basis computation is actually easier in practice than the corresponding degree basis computation. Namely, if there is a permutation of variables in which a system is "very close" to the lexicographic basis (in that permutation), the Gröbner basis is relatively easy to compute. (See, e.g., Problems 4-6 in the Appendix.) So, while computational complexities should never be ignored, other considerations are often important as well. It therefore seems clear that there is still some practical use for the direct computation of lexicographic Gröbner bases, in spite of the difficulty.

There are a number of ways in which the practical behaviour of the Gröbner basis algorithm may be further improved. For example, by reformulating the criteria of Buchberger [2], it is possible to avoid a greater amount of unnecessary computation (see, e.g. [11]). For the lexicographic order especially, the efficiency of the reduction sub-algorithm may be improved by careful formulation of the arithmetic used for certain coefficient fields ([5, 6]). Recent progress in Hensel-type methods for Gröbner bases (e.g. [15]) is also encouraging. A particular problem with the lexicographic order, though, is the rapid growth of intermediate results produced by the algorithm. In [8] we demonstrated that by decomposing intermediate results, this growth may often be reduced; hence the algorithm may be much more efficient. In [7] we found that the order of S-polynomial selection / reduction plays a key role in this regard; a heuristic for this selection was briefly examined. Different schemes for S-polynomial selection (along with many other aspects of the Gröbner basis algorithm) have also been considered in [14]. However, in the present paper we concentrate solely on motivating and testing our selection / reduction heuristic which seems to be effective on a wide range of problems; this will be compared directly to the "normal" strategy and to one of the heuristics mentioned in [14].

2. Background and Notation

We will denote a multivariate polynomial ring over a field K by $K[x_1, \ldots, x_n]$, or simply $K[\mathbf{x}]$ when the ordered set of indeterminates $\mathbf{x} = \{x_1, \ldots, x_n\}$ is understood. We then define the set of *terms* in $\mathbf{x}$ by

$$T_{\mathbf{x}} = \{ x_1^{i_1} \cdots x_n^{i_n} : i_j \in \mathbf{N} \}$$

where $\mathbf{N}$ are the non-negative integers, and write $M(p)$, $HT(p)$, $HC(p)$ to denote the *leading* ("*head*") *monomial*, *leading term*, and *leading coefficient*, respectively, of $p \in K[\mathbf{x}]$ with respect to an *admissible ordering* (see [3]) of $T_{\mathbf{x}}$. (We adopt the convention that $HT(0)=1$, $HC(0)=M(0)=0$.) We will be concerned solely with the *lexicographic ordering* induced by the precedence in $\mathbf{x}$. Such an ordering of terms induces the following reduction relation on $K[\mathbf{x}]-\{0\}$: *p reduces with respect to q* iff

$$p = \sum_{i=1}^{m} \alpha_i t_i, \ \alpha_i \in K-\{0\}, \ t_i \in T_{\mathbf{x}},$$

$$\exists j, \ 1 \le j \le m \ \text{such that } HT(q) \mid t_j,$$

i.e., a multiple of q (by a monomial) may be subtracted from p to cancel t_j. One may then construct an algorithm to compute the reflexive, transitive closure of the above (Noetherian) rela-

tion for a given p and fixed set of nonzero reducers $Q = \{q_1, \ldots, q_m\}$. Since we consider 0 irreducible, we define $R_{0,Q} = \emptyset$; then for $p \neq 0$ define

$$R_{p,Q} = \{q \in Q \text{ such that } \mathrm{HT}(q) \mid \mathrm{HT}(p)\}.$$

Using this notation, the following computes the full reduction of p modulo Q.

Algorithm 1. Full Reduction of p Modulo Q

procedure Reduce(p, Q)
  $r \leftarrow p$; $s \leftarrow 0$
  while $r \neq 0$ **do** {
    while $R_{r,Q} \neq \emptyset$ **do** {
      $q \leftarrow$ Select•poly($R_{r,Q}$)
      $r \leftarrow r - \frac{\mathrm{M}(r)\,q}{\mathrm{M}(q)}$ }
    $s \leftarrow s + \mathrm{M}(r)$; $r \leftarrow r - \mathrm{M}(r)$ }
  return(s)
end

The procedure Select•poly in Algorithm 1 simply selects a reducer from Q for the next reduction. Since this selection is of no theoretical importance, we assume for now that the first reducer found in $R_{r,Q}$ (according to the order of these elements within Q) is used.

We may also define the *S-polynomial* of polynomials $p_1, p_2 \in \mathrm{K}[\mathbf{x}]$ by

$$\mathrm{Spoly}(p_1, p_2) = \mathrm{LCM}(\mathrm{HT}(p_1), \mathrm{HT}(p_2))\left[\frac{p_1}{\mathrm{M}(p_1)} - \frac{p_2}{\mathrm{M}(p_2)}\right],$$

which is the essential component of Buchberger's algorithm [2, 3] for computing Gröbner bases (Algorithm 2).

The procedure Inter•reduce in Algorithm 2 transforms an ideal basis so that each element is reduced modulo the others. Such a procedure

Algorithm 2. Buchberger's Algorithm

procedure Gbasis(Q)
  $G \leftarrow$ Inter•reduce(Q) ; $k \leftarrow$ Length(G)
  $B \leftarrow \{[i,j] : 1 \le i < j \le k \text{ and } \mathrm{Crit1}(G_i, G_j)\}$
  while $B \neq \emptyset$ **do** {
    $[i,j] \leftarrow$ Select•pair(B, G)
    $B \leftarrow B - \{[i,j]\}$
    if Crit2($[i,j]$, B, G) **then** {
      $h \leftarrow$ Reduce(Spoly(G_i, G_j), G)
      if $h \neq 0$ **then** {
        $G \leftarrow G \cup \{h / \mathrm{HC}(h)\}$; $k \leftarrow k+1$
        $B \leftarrow B \cup \{[i,k] : 1 \le i < k$ and
                    $\mathrm{Crit1}(G_i, G_k)\}$ } } }
  redundant $\leftarrow \{h \in G : R_{h,G} \neq \{h\}\}$
  return(Inter•reduce(G – *redundant*))
end

(described in [3]) is usually invoked at the start of the above algorithm to remove dependencies, and again at the end to produce a reduced (or unique, minimal) basis.

The procedures Crit1, Crit2 are criteria of Buchberger [2, 3] to detect *0-reductions* (i.e. S-polynomials which yield only 0). The former rejects a pair if $\mathrm{LCM}(\mathrm{HT}(G_i), \mathrm{HT}(G_j)) = \mathrm{HT}(G_i)\cdot\mathrm{HT}(G_j)$; the latter rejects $[i, j]$ if two "overlapping" pairs, i.e. $[i, u]$, $[u, j]$, $i \neq u \neq j$ such that

$$\mathrm{HT}(G_u) \mid \mathrm{LCM}(\mathrm{HT}(G_i), \mathrm{HT}(G_j)),$$

have already been considered. Such criteria are essential to a practical implementation.

Finally, the procedure Select•pair (analogous to Select•poly) chooses a pair $[i, j]$ from B corresponding to the next S-polynomial to be considered. Although the algorithm is correct irrespective of this choice, it turns out that a good choice is of crucial practical importance.

3. The Normal Selection Strategy

In [2,3] it is suggested that the procedure Select•pair in Algorithm 2 should always select from B the pair $[i, j]$ such that

$$\text{LCM}(\text{HT}(G_i), \text{HT}(G_j)) =$$
$$\min \{ \text{LCM}(\text{HT}(G_u), \text{HT}(G_v)) : [u, v] \in B \},$$

where the minimum is with respect to the ordering of terms. This strategy (hereafter referred to as the *normal selection strategy*) offers a number of advantages. First, it clearly increases the likelihood that Crit2 may be applied (and hence that more 0-reductions will be omitted). Second, it ensures that pairs rejected by this criterion *always* reduce to 0, i.e.

$$\neg \text{Crit2}([i, j], B, G)$$
$$\Rightarrow \text{Reduce}(\text{Spoly}(G_i, G_j), G) = 0$$

regardless of the formulation of the reduction algorithm. (Otherwise there exists some doubt as to whether a useful, though not crucial, nonzero result might be obtained.) Moreover, Buchberger & Winkler [1] further show that S-polynomials so selected always reduce to a *unique* result. Hence the order of reductions used in Reduce may be chosen to lessen the cost of reduction ([6]). Finally, if a degree-based ordering (such as the total degree ordering of Buchberger) is used, the above selection is also the "simplest" available S-polynomial.

When the pure lexicographic ordering is used, however, the final comment above does not necessarily apply. In order to see this more clearly, it is useful to consider a simple example.

Example 1: Consider the three polynomials

$$f_1 = x^2 + 4x + 2y^2 + 11y + 17z^2 + 10z + 2,$$
$$f_2 = xy + 17x + 5y^2 + yz + 4y + 10z^2 + 6z + 4,$$
$$f_3 = xz + 18x + 18y^2 + yz + 2y + 18z^2 + 18z + 14,$$

and impose the lexicographic ordering on $T_{\mathbf{x}}$ for $\mathbf{x} = \{x, y, z\}$. When Algorithm 2 is applied over $\mathbf{Z}_{19}[\mathbf{x}]$ using the normal selection strategy, we obtain the following (after re-scaling by the final head coefficient):

$$G_4 = \text{Reduce}(\text{Spoly}(G_2, G_3), G)$$
$$= y^3 + 4y^2z + 10y^2 + 2yz^2 + 6yz + 5y$$
$$+ 10z^3 + 13z^2 + 15z + 5; \quad (3.1)$$

$$\text{Reduce}(\text{Spoly}(G_2, G_4), G) = 0; \quad (3.2)$$

$$G_5 = \text{Reduce}(\text{Spoly}(G_1, G_3), G)$$
$$= x + 13y^2z + 2y^2 + 18yz + 8y$$
$$+ 8z^3 + 10z^2 + 16z + 1; \quad (3.3)$$

$$G_6 = \text{Reduce}(\text{Spoly}(G_3, G_5), G)$$
$$= y^2z^2 + 5y^2z + 16y^2 + 16yz^2 + 5yz + 8y$$
$$+ 5z^4 + 6z^3 + 2z^2 + 15z + 12; \quad (3.4)$$

$$G_7 = \text{Reduce}(\text{Spoly}(G_4, G_6), G)$$
$$= y^2z + 18y^2 + 3yz^4 + 16yz^3 + 11yz^2 + 10yz$$
$$+ 17y + 10z^5 + 7z^4 + 17z^3 + 7z^2 + 16z; \quad (3.5)$$

$$G_8 = \text{Reduce}(\text{Spoly}(G_6, G_7), G)$$
$$= y^2 + 18yz^5 + 14yz^4 + 15yz^3 + 18yz^2$$
$$+ 14yz + 13y + 3z^6 + 3z^5$$
$$+ z^4 + 10z^3 + 13z^2 + 11z + 4; \quad (3.6)$$

$$G_9 = \text{Reduce}(\text{Spoly}(G_7, G_8), G)$$
$$= yz^6 + 4yz^5 + 2yz^4 + 13yz^3 + 15yz^2$$
$$+ 11yz + 11y + 16z^7 + 12z^5 + 17z^4$$
$$+ 14z^3 + 9z^2 + 4z + 4. \quad (3.7)$$

At this point the normal strategy may choose any of the pairs [6, 9], [7, 9], [8, 9], which all yield

$$G_{10} = yz^5 + 4yz^4 + 14yz^3 + 16yz^2 + 3y + 3z^8$$
$$+ 15z^7 + 15z^6 + 2z^5 + 13z^4 + 9z^2 + 5z + 14.$$

(The first pair is marginally better than the others since fewer reductions will be needed.) Eventually, the algorithm produces a univariate polynomial in z of degree 12. □

Still, we observe that the *final* Gröbner basis for Example 1 contains a univariate polynomial of degree only 8. That is, the use of normal selection apparently restricts the practical behaviour of the algorithm in the sense of reflecting (to some extent) its worst-case behaviour.

4. A Heuristic Selection Strategy

The effect of the intermediate growth exhibited in Example 1 becomes quite pronounced on even slightly larger examples over the fields of rational numbers or functions. In view of the above observations, it is natural to wonder if there are non-normal strategies which will result in less unnecessary growth. To see that this is indeed possible, let us return to our previous example.

Example 2: Suppose we proceed as in Example 1 up to the end of (3.3). The pair-set B now consists of $\{[1,2], [1,5], [2,5], [3,5]\}$, from which the normal selection is [3, 5]. It turns out that the second and third selections above reduce to results similar to G_6 in (3.4). However, the S-polynomial corresponding to [1, 2] would instead yield

$$G_6 = y^2z + 6y^2 + 10yz^2 + 7yz + 17y + 13z^3 + 6z^2 + z + 7. \quad (4.1)$$

This has not only a lower-order headterm than before, but lower-order tail terms as well. Even if the normal strategy were followed to the end of (3.6), the selection [1, 2] would yield

$$G_9 = yz^5 + 10yz^4 + 18yz^3 + 9yz^2 + 10yz + 6y + 16z^6 + z^5 + 17z^4 + 3z^3 + 14z^2 + 14z + 16 \quad (4.2)$$

which is again superior to the result (3.7) of the normal selection. □

It is important to note that if one could achieve similar gains throughout the course of the algorithm, the effect on its overall behaviour would be a substantial improvement. We are therefore led to consider how to judge "good" S-polynomial selections.

Two distinctions of the selection [1, 2] are apparent: the polynomials G_1, G_2 are themselves of low degree; and, relatively small "correction" terms are used in forming their S-polynomial. Hence the (non-reduced) S-polynomial is "simple" in the sense of overall degree. Example 2, along with other examples, seems to suggest that S-polynomials which are of low order (not only in the head, but tail terms as well) may tend to reduce to polynomials of relatively low order. A simple heuristic which exploits this possibility is to choose $[i, j]$ such that

$$\text{Degree}(\text{Spoly}(G_i, G_j)) = \min \{ \text{Degree}(\text{Spoly}(G_u, G_v)) : [u, v] \in B \}, \quad (4.3)$$

where ties are broken using normal selection. (The S-polynomials themselves need not be formed.)

However, before proceeding to test this approach it is prudent to consider two potential difficulties which may arise as a result. The first problem is that with the above selection we are no longer *guaranteed* a unique reduced result. Therefore, even if a result of low degree is *possible*, we may not obtain it in practice.

Example 3: Consider the first reduction of $\text{Spoly}(G_1, G_2)$ in Example 2. If reductions are performed with respect to the entire basis $G = \{G_1, \ldots, G_5\}$ in its natural order, we obtain (4.1). If however we reduce using only

$\{G_4, G_5\}$, we obtain

$$G_6 = y^2z^4 + 3y^2z^3 + 8y^2z^2 + 11y^2z + 7y^2 + 11yz^5 + 9yz^4 + 15yz^3 + 12yz^2 + 11yz + 15y + 13z^6 + 12z^5 + 6z^4 + 5z^3 + 13z^2 + 8z + 4,$$

which is far worse than the result (3.4) from the normal selection. □

Note that if we attempt to inter-reduce the partial basis whenever possible, we may not only violate the normal selection strategy, but may also remove the best reducers from the basis. Hence the above effect is a likely one. (Evidence of this phenomenon was provided in [6, 7]. This shows how one variant of the algorithm may perform poorly relative to another, even if fewer "unnecessary reductions" occur.) On the other hand, for our formulation of Algorithm 2, it seems that the *natural* order of the intermediate basis G is a very reasonable choice. This is so since polynomials which appear later in the course of the algorithm have lower order headterms *and* higher order tail terms. (That is, it is usually better to use the "early" reducers when possible.)

It should be clear, though, that the natural order of the partial basis will not *always* yield the best reducer for a given polynomial. It may therefore be advantageous to modify the procedure Select•poly which chooses a reducing polynomial. Since single headterm reductions are actually S-polynomials, it seems reasonable to choose reducers in accordance with the above heuristic for S-polynomials (i.e. so that the degree - and then headterm - of the reduced result is minimal over all possible reductions from a given $R_{r,G}$). As with (4.3), this does not require all these reductions to be carried out.

A second problem due to leaving the normal strategy stems from the use of Buchberger's criteria. Namely, one would expect the incidence of 0-reductions to be higher since Crit2 cannot be applied as often. The overall effect of this is difficult to predict, since *if* the heuristic results in less complex calculations the algorithm may be faster overall. In addition, one cannot be certain that an S-polynomial will *necessarily* reduce to 0 if Crit2 rejects it. It is noted in [1] that in some cases it may actually be better to obtain such a nonzero result than to skip the S-polynomial entirely (and perhaps obtain a more complicated result later on). Still, even if this occurs occasionally the overall performance of the algorithm may be better than when using normal selection.

What follows in Table 1 is a comparison of the computation of (reduced) lexicographic Gröbner bases using a variety of selection and reduction strategies in the "main loop" of Algorithm 2. The approaches considered are:

Normal: use normal selection of S-polynomials, but "optimize" the reduction order as in [6];

Heuristic 1: use the heuristic selection strategy (4.3), and reduce with the full intermediate basis in natural order;

Heuristic 2: use heuristic selection of S-polynomials *and* reducers;

Heuristic 3: as above, but do not apply Crit2 to the first S-polynomial selected with a given minimal [degree, headterm]; [1]

Other: select $[i, j]$ such that

$$\text{Degree}(\text{LCM}(\text{HT}(G_i), \text{HT}(G_j))) =$$

$$\min\{\text{Degree}(\text{LCM}(\text{HT}(G_u), \text{HT}(G_v))) : [u, v] \in B\},$$

with ties broken by normal selection. (Select reducers by the heuristic selection.)

[1] This tactic may easily be stopped at an advanced stage of the elimination, e.g. when a univariate polynomial appears.

The code "Heuristic 3" is meant to illustrate the effect of "occasionally" ignoring the criterion Crit2. The object of the "Other" strategy is to choose S-polynomials which use small cross-multiplications, but in a way in which the power of Crit2 may not be as seriously diluted. Since no one strategy will be best for *all* problems, this scheme (which is just one of many similar schemes we tried) gives some idea of the relative performance of other non-normal strategies. We compare: (i) the total time (in seconds) and space (in bytes) required; (ii) the number of necessary / 0-reductions performed in the main loop of Algorithm 2; (iii) the maximum degree of intermediate results obtained during the course of the algorithm. All timings (except those marked by a "*")[2] were made using Maple 4.3 on a MIPS M/2000 processor. All codes use identical inter-reduced input polynomials, and common sub-algorithm codes where appropriate. We mention also that both reduction arithmetic and application of Buchberger's criteria were implemented as in [5], rather than exactly as stated in section 2. The test problems are described in the Appendix.

5. Conclusions

We observe from Table 1 that the normal strategy exhibits a limited applicability. While it sometimes performs well for "easy" problems (i.e. "nearly triangular" input in a suitable permutation of variables), it is unsuitable for more difficult systems with more homogeneous structure (e.g. dense systems). This character would remain in any other variant using normal selection, regardless of the number of 0-reductions performed. Note also that an easy problem may become much more difficult after very small changes (cf. Problem 6). The sensitivity to permutations of the variable order is exhibited in Problems 1, 2.

On the other hand, all of the codes using the heuristic (4.3) seem to perform fairly well on most problems. While the proportion of 0-reductions is high, the progress of the algorithm is more strongly governed by the number and degree of *nonzero* results. Note that with this selection strategy the degrees of intermediate results are often no larger than the degree of the final basis; hence the algorithm also seems more stable with respect to permutations of variables. (It is of course still true that the intrinsic difficulty of a given problem depends on the permutation chosen.) We might also have included some measure of the coefficient growth; however, this is already reflected in the total time / space requirements. Moreover, computations with relatively large degree growth typically involve relatively large coefficients since more reductions are required.

There does seem to be some variation in performance according to the manner in which reducers are chosen. This reflects the fact that certain objects (i.e. S-polynomials) may reduce to 0 in one reduction scheme, but not the other. However, it is not clear whether the "heuristic selection" of reducers (cf. Heuristic 2) constitutes a general improvement to the natural selection or not. It does bear mention that the performance of other reduction schemes (e.g. lexicographic sorting of the basis) will often be much poorer due to the type of expression growth shown in Example 3.

In view of the above remarks, it is not really surprizing that the algorithm may actually *benefit* from occasionally ignoring the criterion Crit2 (cf. Heuristic 3); while more 0-reductions follow, simpler nonzero results are sometimes obtained. Hence the weakening of Crit2 due to the use of a non-normal strategy does not seem

[2] These timings were made using Maple V on a Sun4/490 processor.

Prob.		Code				
		Normal	Heur. 1	Heur. 2	Heur. 3	Other
1a	time/space reductions degree	7/565K 11/2 12	8/655K 8/7 10	8/672K 8/7 10	8/672K 8/9 10	8/606K 10/1 11
1b	time/space reductions degree	178/1.7M 42/15 21	29/901K 16/9 10	29/910K 16/9 10	28/877K 15/14 10	8753/5.3M 35/1 21
1c	time/space reductions degree	772/2.3M 61/40 24	63/1.1M 19/10 10	282/1.4M 19/9 10	82/1.2M 18/16 10	**>50000**/≥11M - ≥15
2a	time/space reductions degree	81/1.2M 39/44 20	32/1.1M 22/30 13	33/1.1M 21/31 13	34/1.1M 21/37 13	55/1.1M 23/8 13
2b	time/space reductions degree	193/1.4M 53/63 23	48/1.2M 23/50 13	54/1.2M 23/50 13	56/1.2M 23/56 13	29/1.0M 23/16 13
2c	time/space reductions degree	1577/2.6M 140/110 42	99/1.2M 28/29 13	96/1.3M 27/30 13	98/1.2M 27/40 13	**>50000**/≥10M - ≥13
3	time/space reductions degree	>17467/**>16M** - ≥78	829/2.0M 33/23 16	851/2.1M 33/23 16	886/2.1M 33/36 16	>37091/**>16M** - ≥16
4	time/space reductions degree	43640/10.6M 235/347 16	295/1.7M 56/75 6	283/1.6M 54/81 6	270/1.5M 54/81 6	955/2.7M 56/53 10
5	time/space reductions degree	45/1.0M 36/35 9	173/1.6M 48/68 8	165/1.6M 48/65 8	233/1.7M 51/92 10	>12100/**>16M** - ≥19
6a	time/space reductions degree	435/1.7M 53/104 7	985/3.0M 51/76 12	580/2.1M 49/76 9	432/1.9M 44/95 7	>6603/**>16M** - ≥13
6b	time/space reductions degree	50527/10.3M 235/347 16	290/1.7M 56/74 6	227/1.5M 55/79 6	225/1.5M 55/79 6	>23670/**>16M** - ≥12
7	time/space reductions degree	>16757/**>16M** - ≥246	111/1.4M 41/92 18	126/1.2M 39/92 15	130/1.2M 39/105 15	335/1.5M 49/62 16
8	time/space reductions degree	1560/3.9M 211/18 71	528/1.9M 56/31 20	846/2.1M 56/31 20	514/1.8M 56/54 20	>7500/**>16M** - ≥20
9	time/space reductions degree	5481/7.0M 273/183 108	263/1.6M 63/114 20	120/1.4M 57/105 20	121/1.4M 57/119 20	>9270/**>16M** - ≥54
10	time/space reductions degree	4105/3.6M 112/161 21	1190/2.7M 69/90 14	3135/2.5M 71/95 14	3160/2.4M 71/96 14	>22872/**>16M** - ≥26
11	time/space reductions degree	>6945/**>16M** - ≥45	5814/4.0M 91/63 32	3762/3.7M 87/63 32	3896/3.6M 87/95 32	>12726/**>16M** - ≥32
12	time/space reductions degree	>22830/**>16M** - ≥38	3981/5.0M 128/317 12	16014/5.0M 142/320 12	3134/3.4M 138/355 12	>45335/**>16M** - ≥34
13	time/space reductions degree	>10473/**>16M** - ≥179	8317/5.7M 206/156 61	10848/5.9M 211/158 61	7691/5.0M 194/255 61	>2904/**>16M** - ≥61
14*	time/space reductions degree	>1806/**>16M** - ≥203	13134/3.3M 159/524 48	51588/5.2M 174/569 48	49667/5.2M 174/645 48	76753/10.5M 243/408 48

Table 1. A comparison of some lexicographic selection/reduction strategies

to be a major concern. We have not assessed the effect of similar treatment applied to the simpler Crit1. On one hand, in our formulation of the algorithm, failing pairs are *never* actually added to the pair-set at all; on the other, it seems extremely likely that failing pairs *will* reduce to 0 anyway. It might also be interesting to examine the effect of our strategy with a different formulation of the criteria (e.g. [11]). Still, it is clear from Table 1 that the overall success of the algorithm does not necessarily hinge upon the proportion of 0-reductions.

The "Other" strategy, while better than the normal strategy on some problems, is not a viable alternative. Other similar strategies for S-polynomial / reducer selection (i.e., based on headterms alone) were found to produce results much like those in the last column of Table 1.

It turns out that even such lexicographic bases as those of Problems 13, 14 can be computed directly (at least, over a small finite field). In fact, an earlier form of the heuristic strategy was used in [7] to compute lexicographic bases for the difficult (but zero-dimensional) problems of Katsura (Problem 11) and Rimey [13] over rational number and rational function fields respectively.[3] The basis conversion approach of [9] is a much better algorithm (and will usually be superior in practice) for such problems; hence a direct comparison with this scheme was not our primary interest. Still, the success of our simple heuristic on such a wide range of problems seems to warrant further investigation of strategies for S-polynomial / reducer selection. We also mention that the factorization / decomposition approach described in [8] would sometimes yield much more efficient computations than those of Table 1.[4]

[3] It has been pointed out that a similar heuristic also appears in the CoCoA [12] system.

[4] Actually, this approach made *direct* lexicographic solution of Rimey's problem [13] possible when basis conversion was not!

Appendix: Test Problems

Unless otherwise stated, the following problems are over the field of rational numbers. Variable orderings marked with a "$\dagger$" are heuristically optimal in the sense of [4].

Problem 1 : Trinks' system (see, e.g. [4]) of 6 equations in 6 unknowns, using the orderings:

(a) $\mathbf{x} = \{w, p, z, t, s, b\}^{\dagger}$; (b) $\mathbf{x} = \{w, b, p, z, s, t\}$;

(c) $\mathbf{x} = \{b, t, s, w, p, z\}$.

All systems have finitely many solutions, with final bases of 6 elements of maximum degree 10.

Problem 2 : Katsura's system (see [4]) of 5 equations in 5 unknowns, using the orderings:

(a) $\mathbf{x} = \{u_4, u_2, u_0, u_3, u_1\}$; (b) $\mathbf{x} = \{u_4, u_0, u_2, u_3, u_1\}$;

(c) $\mathbf{x} = \{u_4, u_0, u_3, u_2, u_1\}^{\dagger}$.

All systems have finitely many solutions; the final bases have 6, 8, 8, elements, resp., of maximum degree 13.

Problem 3: A system of 4 dense quadratic polynomials in 4 variables, with coefficients no larger than ± 10 (see Problem 4(b) in [6]). The system has finitely many solutions; its final basis has 4 elements of maximum degree 16.

Problem 4 : A system of Gear [10] of 8 equations in 10 unknowns, using the ordering of variables $\mathbf{x} = \{g_0, b_{32}, g_1, b_{22}, b_{33}, g_2, g_3, a_3, a_2, a_1\}^{\dagger}$. The system has infinitely many solutions; its final basis has 17 elements of maximum degree 6.

Problem 5 : Butcher's system [4] of 8 equations in 8 unknowns, using the ordering of variables $\mathbf{x} = \{b_1, a_{32}, b_2, b_3, a, c_3, c_2, b\}^{\dagger}$. The system has infinitely many solutions; its final basis has 13 elements of maximum degree 7.

Problem 6 : Hairer's system [4] of 11 equations in 13 unknowns, using the ordering of variables $\mathbf{x} = \{a_{21}, a_{31}, a_{41}, b_1, b_2, a_{42}, a_{32}, a_{43}, b_3, b_4, c_4, c_3, c_2\}^{\dagger}$.

(a) Set $c_4 \leftarrow 1$; (b) Use the full system.

Both have infinitely many solutions; the final bases have 19, 20 elements, resp., of maximum degree 6.

Problem 7: A system of Arnborg (due to Björk and Fröberg; see Problem I in [9]) of 5 equations in 5 unknowns, using $\mathbf{x} = \{a, b, c, d, e\}^{\dagger}$. The system has finitely many solutions; its final basis has 11 elements of maximum degree 15.

Problem 8: A modified system of Arnborg (Problem III in [9]) of 3 equations in 3 unknowns, using $\mathbf{x} = \{a, b, c\}$. The system has finitely many solutions; its final basis has 3 elements of maximum degree 20.

Problem 9: A system of Caprasse (Problem IV in [9]) of 4 equations in 4 unknowns, using $\mathbf{x} = \{x, y, z, t\}$. The system has finitely many solutions; its final basis has 7 elements of maximum degree 20.

Problem 10 : Fee's system [5] of 4 equations in 5 unknowns, using the ordering $\mathbf{x} = \{c, d, q, b, p\}$. The system has infinitely many solutions; its final basis has 14 elements of maximum degree 14.

Problem 11: Katsura's system [4] of 6 equations in 6 unknowns, using $\mathbf{x} = \{u_5, u_3, u_4, u_2, u_1, u_0\}$. The system has finitely many solutions; its final basis has 6 elements of maximum degree 32.

Problem 12: A system of Filliman [7] of 12 equations in 9 unknowns, using $\mathbf{x} = \{a_l, e, h, d, f, a, g, b, c\}^t$. The system has infinitely many solutions; its final basis has 17 elements of maximum degree 3.

Problem 13: A modified system of Arnborg (Problem V in [9]) of 5 equations in 5 unknowns over $\mathbf{Z}_{1831}$, using $\mathbf{x} = \{a, b, c, d, e\}^t$. The system has finitely many solutions; its final basis has 8 elements of maximum degree 61.

Problem 14: A system of Arnborg (Problem II in [9]) of 6 equations in 6 unknowns over $\mathbf{Z}_{1831}$, using $\mathbf{x} = \{a, b, c, d, e, f\}^t$. The system has finitely many solutions; its final basis has 17 elements of maximum degree 48.

References

1. B. Buchberger and F. Winkler, "Miscellaneous Results on the Construction of Gröbner Bases for Polynomial Ideals," Tech. Rep. 137, Univ. of Linz, Math. Inst., 1979.
2. B. Buchberger, "A Criterion for Detecting Unnecessary Reductions in the Construction of Gröbner Bases," in *Proc. EUROSAM '79, Lecture Notes in Computer Science* **72**, ed. W. Ng, pp. 3-21, Springer-Verlag, 1979.
3. B. Buchberger, "Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory," in *Progress, directions and open problems in multidimensional systems theory*, ed. N.K. Bose, pp. 184-232, D. Reidel, 1985.
4. W. Böge, R. Gebauer, and H. Kredel, "Some Examples for Solving Systems of Algebraic Equations by Calculating Gröbner Bases," *J. Symbolic Comp.*, vol. 2, no. 1, pp. 83-98, 1986.
5. S.R. Czapor and K.O. Geddes, "On Implementing Buchberger's Algorithm for Gröbner Bases," in *Proc. SYMSAC '86*, ed. B.W. Char, pp. 233-238, ACM Press, 1986.
6. S.R. Czapor, "Solving Algebraic Equations via Buchberger's Algorithm," in *Proc. EUROCAL '87, Lecture Notes in Computer Science* **378**, ed. J.H. Davenport, pp. 260-269, Springer-Verlag, 1987.
7. S.R. Czapor, "Gröbner Basis Methods for Solving Algebraic Equations," Ph.D. Thesis, University of Waterloo, Dept. of Applied Math., 1988.
8. S.R. Czapor, "Solving algebraic equations: combining Buchberger's algorithm with multivariate factorization," *J. Symbolic Comp.*, vol. 7, pp. 49-53, 1989.
9. J.C. Faugère, P. Gianni, D. Lazard, and T. Mora, "Efficient Computation of Zero-Dimensional Gröbner Bases by Change of Ordering," Preprint, 1990.
10. G.W. Gear, in *Numerical Initial Value Problems in Ordinary Differential Equations*, ed. G. Forsythe, p. 35, Prentice-Hall, 1971.
11. R. Gebauer and H.M. Möller, "On an Installation of Buchberger's Algorithm," *J. Symbolic Comp.*, vol. 6, pp. 275-286, 1988.
12. A. Giovini and G. Niesi, *CoCoA User's Manual*, University of Genova, Dept. of Math., 1989.
13. K. Rimey, "A System of Polynomial Equations and a Solution by an Unusual Method," *ACM SIGSAM Bull.*, vol. 18, pp. 30-32, 1984.
14. C. Traverso and L. Donati, "Experimenting the Gröbner basis algorithm with the AlPI system," in *Proc. ISSAC '89*, ed. G.H. Gonnet, pp. 192-198, ACM Press, 1989.
15. F. Winkler, "A p-adic Approach to the Computation of Gröbner Bases," *J. Symbolic Comp.*, vol. 6, pp. 287-304, 1988.