Using Gröbner bases to investigate generalized n-gons and 2-designs

Douglas A. Leonard
Department of Mathematics and Statistics
Auburn University

Abstract This paper is meant primarily as a *tutorial* on how to phrase problems in association schemes in the language of Gröbner bases and use the computational results provided by those bases, though it does contain fusion scheme computations not previously found in the literature.

1 Introduction

It is instructive to see how the use of Gröbner bases can clarify certain computational and theoretic aspects of various topics. Here the choice of topics come form association schemes and fusion of flag algebras arising from generalized n-gons [2]. Three different ways of using concepts related to Gröbner bases are given as a guide for using such concepts in similar types of problems.

Often it is the case that combinatorialists have to sort out some form of truth about a given object based on its parameters. This can translate into pages and pages of manipulations of intermediate results involving polynomial equations, with the attendant derivations and justifications. But these problems can usually be easily rephrased in terms of finding common zeros of a system of polynomial equations; and knowing for what values of the parameters this happens. This is exactly what Gröbner bases are meant to accomplish. So Gröbner basis theory can provide direct computational answers without the need for further justification, once problems are phrased in terms of generators (polynomials that should be zero) for an ideal in a multivariate polynomial ring, and the need to know the corresponding varieties (set of common zeros). This should be useful either to those wishing to use Gröbner bases to clarify and/or simplify computations or to those interested in seeing how Gröbner bases can be used in "applied theoretical" settings.

Now consider the concrete problem of investigating a parameterized series of association schemes, describing the structure constants, determining fusion partitions, and investigating those fusions found. Much of this work was done by the Soviet school for metric (one parameter) schemes (see slides of Muzychuk on Linz home page).

The next natural stage is to investigae dihedral (two parameter) schemes in terminology of Zieschang, coming mainly from flag algebras, which were intoduced by a number of authors (a few credits).

The [2] paper, which started this investigation, used ad hoc methods to do the computations. So here those methods have been replaced by the systematic Gröbner basis methods, which make the actual computations invisible to the reader. There are new computations relative to *fusion* expanding on this previous work, with the MAGMA [3] code that generated them, to whet the appetite of the audience.

2 Gröbner Basis preliminaries

There are many good books covering this material, with [1] being the author's favorite. The following is a brief intro to ideals and varieties in polynomial rings for those unfamiliar with the topic. Given some variables $x_n, \ldots, x_1$ and a coefficient ring R, $R[x_n, \ldots, x_1]$ denotes the ring of (finite) R-linear combinations of (finite) products in these variables, and seems to be called either a *multivariate polynomial ring* or a *free algebra*, depending on whether the variables commute with each other or not. Even to be able to write down polynomials in a canonical way, it is necessary to have a *monomial order* (a *total order* with obvious extra properties). The (default) *lexicogrphical monomial order* is based on comparing products by considering their *indices* (that is, *exponents*) lexicographically. So, for instance, in the multivariate polynomial ring $R[x_3, x_2, x_1]$ the order looks like

$$1 \prec x_1 \prec x_1^2 \prec \cdots \prec x_2 \prec x_2x_1 \prec \cdots \prec x_2^2 \prec \cdots \prec x_3 \prec \cdots$$

which can be described by $x_3^{i_3}x_2^{i_2}x_1^{i_1} \succ x_3^{j_3}x_2^{j_2}x_1^{j_1}$ iff $(i_3 > j_3)$ or $(i_3 = j_3$ and $i_2 = j_2)$ or $(i_3 = j_3$ and $i_2 = j_2$ and $i_1 > j_1)$. The generic *total degree orders* are the *grevlex* and *glex* orders (short for graded reverse lexicographical and graded lexicographical), in which total degree is the first concern. These give orders that look like

$$1 \prec x_1 \prec x_2 \prec x_3 \prec x_1^2 \prec x_2x_1 \prec x_3x_1 \prec x_2^2 \prec \cdots$$

and

$$1 \prec x_1 \prec x_2 \prec x_3 \prec x_1^2 \prec x_2x_1 \prec x_2^2 \prec x_3x_1 \prec \cdots$$

respectively. These can be defined in ways similar to those above; but the non-singular matrices

$$A_{grevlex} := \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \; A_{glex} := \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

can be used to reduce these to the lexicographical case by converting the column vector of exponents $\begin{pmatrix} i_3 \\ i_2 \\ i_1 \end{pmatrix}$ to $\begin{pmatrix} i_3 + i_2 + i_1 \\ i_3 + i_2 \\ i_3 \end{pmatrix}$ or $\begin{pmatrix} i_3 + i_2 + i_1 \\ i_3 \\ i_2 \end{pmatrix}$ respectively.

The *leading monomial* of f, denoted here by $LM(f)$ is the largest monomial occurring in f, relative to the given order. Its coefficient is called the *leading coefficient*, denoted by $LC(f)$, and the product $LC(f)LM(f) =: LT(f)$ is called the *leading term.* (Warning: Some authors interchange the words term and monomial, and there is no uniformity of notation in the literature.)

An *ideal* I (two-sided in the case of free algebras) is a subset of a ring closed under addition of its elements and under multiplication by ring elements. Ideals are commonly given in terms of a (finite) set of *generators* (referred to as *relations* when the variables are called generators!) $I := \langle g_1, g_2, \ldots, g_m \rangle$, so that I consists of all (finite) R-linear combinations of $g_1, \ldots, g_m$.

As with vector spaces, where arbitrary generating sets are not as useful as maximal linearly independent sets, called *(vector space) bases*, it should be no surprise that arbitrary generating sets for ideals (which are unfortunately called *bases* in some of the literature) are not as important or useful as *Gröbner bases*, those generating sets B relative to which a “canonical” remainder $NormalForm(f, B)$ for f after division by the elements of B (in any order) can be defined.

A more useful computational definition of Gröbner bases is in terms of *SPolynomials.* If h is a *least common multiple* of $LM(f)$, and $LM(g)$), then $SPoly(f, g) := \frac{h}{LT(f)} f - g \frac{h}{LT(g)}$. Then B is a Gröbner basis if $SPoly(f, g)$ reduces to 0 after division by the elements of B for all choices of $f, g \in B$. In fact this is the foundation for all forms of the *Buchberger algorithm*; namely that given any generating set, it is possible to produce a Gröbner basis by computing SPolynomials, reducing them to normal form, and appending the results to the generating set if necessary, possibly reducing the other elements along the way, until the set is closed under this combined operation of computing SPolynomials and reducing them.

If one has a collection of equations of the form $f_i = h_i$ that can be written in polynomial form in terms of some (not necessarily polynomial) variables, then it is possible to use $g_i := f_i - h_i$ as generators for an ideal in a polynomial ring. Common solutions to the collection of equations then become common zeros of the elements of the ideal I or equivalently of the elements of the Gröbner basis B. These common zeros are called *points* of the *variety* $\mathcal{V}(I) := \{v : g(v) = 0 \text{ for all } g \in I\}$. This variety is generally computed by finding a Gröbner basis relative to some easy ordering, changing it to a (factored) Gröbner basis relative to a lex order, and recursively computing coordinates of the elements of the variety.

The importance of all this to a field such as *algebraic combinatorics* is that in the study of a collection of objects relative to a set of paramaters, it may be that questions may be phrased in terms of what parameter values are necessary or sufficient to guarantee the feasibility or existence of such objects. Then equations satisfied by the parameters can be turned into generators for an ideal for which the variety gives such parameter values. That means that rather than haphazardly sorting through a collection of equations, looking for results about the parameters, it is possible to merely compute a Gröbner basis and a variety, letting the Gröbner basis theory replace all the intermediate results, derivations, and justifications.

Sometimes there are unexpected added benefits beyond this, in that proper phrasing of problems in terms of Gröbner bases may give insight into the combinatorial structures, definitions, or other aspects of a problem. This will now be exemplified by considering *generalized n-gons.*

3 Definitions

Start with finite sets $\mathcal{V}$, of v *vertices*, and $\mathcal{B}$, of b *blocks*. The basic combinatorial structure called a *1-design* is merely a subset $\mathcal{D}$ of $\mathcal{V} \times \mathcal{B}$ such that $r := \#\{l \in \mathcal{B} : (p, l) \in \mathcal{D}\}$ is independent of $p \in \mathcal{V}$ and $k := \#\{p \in \mathcal{B} : (p, l) \in \mathcal{D}\}$ is independent of $l \in \mathcal{V}$. Elements $(p, l) \in \mathcal{D}$ are called *flags*; and there are $vr = bk$ such.

A 1-design such that any two vertices are incident with *at most one* block and any two blocks are incident with at most one vertex is called a *partial linear space.* (Geometers tend to use the word *point* in place of vertex, the word *line* in place of block, and the paramaters $s := k - 1$ and $t := r - 1$ instead of k and r respectively.) Normally the study of vertices is linked to the study of the (symmetric) $(0, 1)$ *adjacency matrix*, A, with

$$A(p_i, p_j) = 1 \; if \; and \; only \; if (p_i, l), (p_j, l) \in \mathcal{D} \; for \; some \; l \in B.$$

But here the focus is initially on flags; so let L denote the (symmetric) $(0, 1)$ *collinearity matrix* indexed by the flags, with

$$L((p_1, l_1), (p_2, l_2)) = 1 \textit{ if and only if } l_1 = l_2,\ p_1 \neq p_2;$$

and let N denote the (symmetric) $(0, 1)$ *concurrence matrix* indexed by the flags, with

$$N((p_1, l_1), (p_2, l_2)) = 1 \textit{ if and only if } p_1 = p_2,\ l_1 \neq l_2.$$

These matrices already satisfy the conditions

$$L^2 = (s - 1)L + sI,\ N^2 = (t - 1)N + tI$$

(with I denoting the $vr \times vr$ identity matrix).

Generalized n-gons were first introduced by Tits[4] in 1959. Standard definitions are in terms of *distance* and *adjacency*, and are geoemtric in flavor, descriptive of the fact that each point should be contained in an n-gon and in no smaller polygon. In terms of flags and the matrices L and N above this translates into the following matrix-theoretic definition, which obscures the geoemtric origin. Let

$$A_{4i} := (NL)^i,\ A_{4i+1} := L(NL)^i,\ A_{4i+2} := N(LN)^i,\ A_{4i+3} := (LN)^{i+1},$$

denote the various *flag adjacency matrices* (an possibly unexpected benefit of this non-geometric approach). Then $\mathcal{D}$ will be called a *generalized n-gon* if $(A_j,\ 0 \leq j \leq 2n - 1)$ is a linearly independent set over $\mathbf{Z}$, and hence a *basis* for the $\mathbf{Z}$-module it generates; but $A_{2n} = A_{2n-1}$. (It is a straighforward, instructive exercise to see that this really corresponds to the geometric concept above.)

Since

1. $A_0 = I$;

2. $A^T_{4i+1} = A_{4i+1}$, $A^T_{4i+2} = A_{4i+2}$, $A^T_{4i+3} = A_{4i+4}$;

3. $\sum_{j=0}^{2n-1} A_j = J$, the all 1's matrix;

the ordered set $(A_j : 0 \leq j \leq 2n-1)$ is a basis for the $\mathbf{Z}$-module it generates, lacking only a multiplication to be the *adjacency algebra* of an *association scheme.*

If the two matrices L and N are *not* known, and the parameters t and s are *not* known either, then it is possible to study generalized n-gons by starting with the *coefficient ring* $\mathbf{Z}[t,s]$ (with a *lex* monomial order $t \succ s$), and then a *free algebra* $\mathbf{Z}[t,s][y,x]$ in two *non-commutative* variables (with a *total degree* monomial order with $y \succ x$). The former means that when writing polynomials in the variables t and s, $t^i s^j$ should be treated as larger than $t^k s^l$ if either $(i > k)$ or $(i = k$ and $j > l)$ (and that one shouldn't write $s^j t^i$). The latter means that when writing polynomials in the (non-commuting) variables y and x monomials, a string such as $yxxyx$ should be treated as larger than either yxy or $yxxxy$, the first because the total degree is greater, the second because the total degree is the same but at the leftmost difference in the strings $y \succ x$. (One of the first important lessons one learns when working with multivariate polynomial rings is that a proper choice of monomial order is perhaps the most critical step in any problem, and the step most easily ignored completely by those not used to this area.)

Then this should be made into a *finitely-presented algebra* (quotient ring in two non-commuting variables, representing the two non-commuting matrices N and L respectively)

$$\mathbf{Z}[t,s][y,x]/\langle f_1, f_2, f_3\rangle$$

with $f_1 := x^2 - (s-1)x - s$, $f_2 := y^2 - (t-1)y - t$, and $f_3 := (yx)^l - (xy)^l$ if $n = 2l$ or $f_3 := y(xy)^l - (xy)^l x$ if $n = 2l+1$. The import here is that the *ideal* $I := \langle f_1, f_2, f_3\rangle$ consists of all those polynomials that are supposed to be equal to zero under the given assumptions.

4 An application of SPolynomials and reduction

Buchberger's algorithm for computing *Gröbner bases* for ideals (such as $\langle f_1, f_2, f_3\rangle$ above) is based on the central observation above that *bases* (as opposed to *generating sets*) for ideals should be *closed* under the operation of *reduction of SPolynomials*. Consider an example of this type of computation for the n-gon case with $n = 2l+1$. Since $LM(yf_3) = y^2(xy)^l = LM(f_2(xy)^l)$, the corresponding (non-commutative) *SPolynomial* would be

$$SPolynomial(f_3, f_2) := yf_3 - f_2(xy)^l = -y(xy)^l x + (t-1)y(xy)^l + t(xy)^l,$$

which would then be *reduced* to a *remainder*

$$(yf_3 - f_2(xy)^l) + f_3 x + (xy)^l f_1 - (t-1)f_3 = (t-s)\left((xy)^l x + (xy)^l\right)$$

using the explicit division by the set $\{f_1, f_2, f_3\}$ given.

Thus a single SPolynomial computation, and the use of the assumption that $(xy)^l x$ and $(xy)^l$ are linearly independent, already forces $t - s = 0$. So it follows that:

Propostion 1 Generalized (2l+1)-gons with parameter pairs (t, s) can only exist if $t = s$.

5 Structure constants of association schemes

In light of the proposition above, consider only generalized 2l-gons in their finitely-presented algebra form. If a *total degree* monomial order (with $a_i \succ a_j$ for $i > j$) is used, then a *minimal, reduced* Gröbner basis (that is, one with a minimal number of elements, and no leading monomial of one dividing any monomial of another) for the ideal of relations, $\mathcal{I}$, will have *all* its elements of the form

$$a_i a_j - NormalForm(a_i a_j, \mathcal{I})$$

with

$$NormalForm(a_i a_j, \mathcal{I}) = \sum_{k=0}^{2n-1} p_{i,j,k} a_k,$$

describing the multiplication in the algebra, as well as determiniming the *structure constants* $p_{i,j,k}$ that make this the *adjacency algebra* of an *association scheme.*

Thus a simple Gröbner basis computation relative to an appropriate total degree monomial order gives constructively the following result (with concrete examples below):

Proposition 2

$$\mathbf{Z}[t, s][a_{4l-1}, \ldots, a_1]/\mathcal{I}$$

with $\mathcal{I}$ the *ideal of relations* with generating set containing the basis relations from the definition of a generalized 2l-gon:

$$f_1 := a_1^2 - (s-1)a_1 - s,$$

$$f_2 := a_2^2 - (t-1)a_2 - t,$$

$$f_3 := (a_2 a_1)^l - (a_1 a_2)^l;$$

together with the relations gotten from the definitions of the a_j's:

$(a_2 a_1)^i - a_{4i}$, $(a_1(a_2 a_1)^i - a_{4i+1}$, $a_2(a_1 a_2)^i - a_{4i+2}$, $(a_1 a_2)^{i+1} - a_{4i+3}$, $0 \leq i < l$, corresponds to the adjacency algebra of an association scheme. (Note that $a_0 = 1$ is implicit in the calculations, and sometimes explicit in the theory.)

Moreover, if a *total degree* monomial order (with $a_i \succ a_j$ for $i > j$) is used, then a *minimal, reduced* Gröbner basis for $\mathcal{I}$ will have *all* its elements of the form

$$a_i a_j - NormalForm(a_i a_j, \mathcal{I})$$

with

$$NormalForm(a_i a_j, \mathcal{I}) = \sum_{k=0}^{2l-1} p_{i,j,k} a_k,$$

which corresponds to a complete description of the algebra multiplication and a determination of the *structure constants* $p_{i,j,k}$.

Proof: Given that the a_i are linearly independent, there can be no relations with leading monomial of degree 1. Given that the a_i's form an algebra, $a_i a_j$ must be expressible as a linear combination $\sum_k p_{i,j,k} a_k$, since all elements of the algebra are of this form. Hence $a_i a_j - \sum_{k=0}^{2l-1} p_{i,j,k} a_k$ must be basis elements in any total degree monomial order Gröbner basis. And any monomial of total degree greater than 2 is divisible by one of degree 2, so can't be a leading monomial in any *minimal* total degree Gröbner basis.

6 Fusion

Now consider a *partition* Π of the set $\{0, \ldots, 4l - 1\}$, and the corresponding *fusion* of classes

$$B_\gamma := \sum \{A_k : k \in \gamma\}, \; \gamma \in \Pi.$$

This could conceivably determine a *fusion scheme*; that is, be an association scheme with fewer classes than the original, if:

1. $\{0\}$ is a part;
2. $\gamma' := \{k' : k \in \gamma, \; A_k^T = A_{k'}\}$ is a part for each $\gamma \in \Pi$;
3. the structure constant $P_{\alpha,\beta,\gamma} := \sum_{i\in\alpha} \sum_{j\in\beta} p_{i,j,k}$ is independent of the choice of $k \in \gamma$, for all parts α, β, and γ in Π.

This can be viewed as another Gröbner basis problem, but this time in the *mutivariate polynomial coefficient ring* $\mathbf{Z}[t, s]$ (with two *commuting* variables t and s, the *integer* parameters). The ideal in question is generated by the relations (forced by the third item above):

$$\sum_{i\in\alpha} \sum_{j\in\beta} p_{i,j,k_1} - \sum_{i\in\alpha} \sum_{j\in\beta} p_{i,j,k_2}$$

for all α, β, and γ in Π and all $k_1, k_2 \in \gamma$, provided Π is a good partition (that is, satisfying the first two items above). Then the *variety* of this ideal determines all possible parameter pairs (t, s) for which the good partition Π produces a fusion scheme. Of course, it is necessary to restrict the variety to pairs of *positive integers*, and probably to ignore the *trivial* case $t = 1 = s$ in which the original generalized 2l-gon is merely a 2l-gon. The varieties involved are computed most easily using a *lexicographical* monomial order on $\mathbf{Z}[t, s]$ and a *factored, minimal, reduced* Gröbner basis.

7 2-(v,k,1) designs

Consider a similar problem of *flag adjacency matrices* for $2-(v, k, 1)$ *designs*; that is, 1-designs for which every 2 vertices determine an unique block. The matrices for this are similar in flavor to those for the generalized n-gons:

$A_0 := I,\ A_1 := L,\ A_2 := N,\ A_3 := LN,\ A_4 := NL,\ A_5 := LNL,\ A_6 := NLN{-}LNL$

with

$$LNLN = sA_6 + (s-1)A_5 + sA_4,\ NLNL = sA_6 + (s-1)A_5 + sA_3.$$

Since

1. $A_0 = I$;
2. $A_1^T = A_1,\ A_2^T = A_2,\ A_3^T = A_4,\ A_5^T = A_5,\ A_6^T = A_6$;
3. $\sum_{j=0}^{6} A_j = J$, the all 1's matrix;

the ordered set $(A_j : 0 \leq j \leq 6)$ is a basis for the $\mathbf{Z}$-module it generates, lacking only a multiplication to be the *adjacency algebra* of an *association scheme.*

Then this should be made into a *finitely-presented algebra* as before

$$\mathbf{Z}[t, s][y, x]/\langle f_1, f_2, f_3\rangle$$

with $f_1 := x^2-(s-1)x-s$, $f_2 := y^2-(t-1)y-t$, and $f_3 := (yx)^2-s*(yxy-xyx)-(s-1)*xyx)-s*xy)$, $f_4 := (xy)^2 - s(yxy-xyx)-(s-1)xyx-syx$.

8 Code and output

There are also technical difficulties to be considered when using existing computer algebra packages. For instance, in MAGMA, the coefficient ring of a finitely-presented algebra needs to be a field. So it is necessary to use $\mathbf{Q}(t,s)$, the function field in two variables over the rational field $\mathbf{Q}$, with a total degree monomial order, and then map results to $\mathbf{Q}[t,s]$ with the desired *lex* monomial order, where factorization can be done.

The MAGMA code below has been automated so that the only parameter input necessary is l. The output is a list of partitions giving rise to fusion schemes, together with a factored Gröbner basis from which it is relatively easy to read the corresponding parameter pairs (t,s), if such positive integer pairs exist.

Although the computations are (for better or worse) no longer visible to the reader, it is instructive to have some idea of what is happening. As an example from the generalized quadrangle case, the partition $\Pi := [[1,2,3,4,7][5,6]]$ could only give a fusion scheme if ts^2-2ts, t^2-2t-s^2+2s, t^2s-ts^2, and $t^2-ts^2+3ts-4t+s^2-4s+4$ (gotten from the fusion condition 3 above) are all zero. An *interreduction* of these generators already gives a Gröbner basis $(t^2-2t, ts-2t, s^2-2s)$ for the ideal they generate. And a recursive calculation of s and t gives $s=0, t=0$, $s=2, t=0$, or $s=2, t=2$. Of these 3 rational points in the variety, clearly only $(t,s)=(2,2)$ is useful in this fusion context. (Currently the code actually produces a modified Gröbner basis $(t-2, s-2)$ by removing factors such as t and s which can't ever lead to positive integer solutions.) There are examples for which the solutions must be done by hand. For instance, in partition 48 in the hexagon output below, one factor is $T^2-TS+2T+1$ which gives a one dimensional variety $s=(t+1)^2/t$ over the rationals, but only the single positive integer solution $s=4, t=1$.

The code is written to search for a good partition, compute differences of coefficients that should be equal for fusion to occur, find a Gröbner basis for ideal of all such differences, and output same, at least in the cases in which there might conceivably be an element in the variety with all coordinates positive. In most cases, it is relatively easy to read off any parameter pairs with both entries positive integers, and to ignore those with only the trivial solution $t=1=s$.

Note that the code produces a Gröbner basis describing the orignal adjacency algebra before tackling the fusion problem (an expected benefit); so it is possible to see the structure constants, not in a table or even in the form $a_ia_j = \sum_k p_{i,j,k}a_k$, but in the form $a_ia_j - \sum_k p_{i,j,k}a_k$.

```
//common code to search for fusion

R<T,S>:=PolynomialRing(Q,2);

co:=function(f,mon) return MonomialCoefficient(f,mon); end function;

fusion:=function(J) return &+[A.(N+1-j): j in J]; end function;

prod:=function(I,J) return NormalForm(fusion(I)*fusion(J),ID); end function;

relations:=function(I,J,K)
   W:=prod(I,J);
   return [(co(W,A.(N+1-K[1]))-co(W,A.(N+1-k)))@hom<FF->R|T,S>: k in K];
end function;

RELATIONS:=function(part)
   rel:=[];
   for i in [1..#part] do for j in [1..#part] do for k in [1..#part] do
      rel:=rel cat relations(part[i],part[j],part[k]);
   end for; end for; end for;
   return rel;
end function;

max:=function(PV,j)
   if j gt 1 then
      return Maximum({PV[i]: i in [1..j-1]});
   else
      return -1;
   end if;
end function;

partno:=0; goodpartno:=0; vector:=[1:i in [1..N]]; v1:=N;
while v1 gt 1 do
   Bound:=max(vector,N+1);
   B:=[[]: i in [1..Bound]];
   for i in [1..N] do Append(~B[vector[i]],i); end for;
   symmetric:=true;
   for i in [1..Bound] do
      if #{vector[AT[j]]: j in B[i]} ne 1 then symmetric:=false; break; end if;
   end for;
   if symmetric then
      partno+:=1;
      id:=ideal<R|RELATIONS(B)>;
      gb:=GroebnerBasis(id);
      pos_sol:=true;
      if #gb ne 0 then
         for i in [1..#gb] do
            m:=Minimum(Coefficients(gb[i]));
            if m gt 0 then pos_sol:=false; break; end if;
         end for;
      end if;
      if pos_sol then
         goodpartno+:=1;
         partno, "partition" cat IntegerToString(goodpartno) cat "=",B;
         if #gb ne 0 then
            for i in [1..#gb] do Factorization(gb[i]); end for;
         end if;
      end if;
   end if;
   v2:=v1;
   bound:=1+max(vector,v1);
   if vector[v1] ge bound then
      while vector[v1] ge bound and v1 gt 1 do
         vector[v1]:=1;
         v1-:=1;
         v2:=N;
         if v1 gt 1 then bound:=1+max(vector,v1); end if;
      end while;
      if v1 gt 1 then
         vector[v1]+:=1;
         v1:=v2;
      end if;
   else
      if v1 gt 1 then
         vector[v1]+:=1;
      end if;
   end if;
end while;
```

```
//preamble code for generalized n-gons
l:=2;//the only parameter that needs to be changed from one run to the next
//n-gons for n=2l, as rank N+1 association schemes of flags

N:=4*l-1;
AT:=[i : i in [1..N]];
for i in [1..l-1] do AT[4*i]:=4*i-1; AT[4*i-1]:=4*i; end for;

Q:=RationalField();
FF<t,s>:=FunctionField(Q,2);
A:=FreeAlgebra(FF,N);
AssignNames(~A,["a" cat IntegerToString(N+1-i): i in [1..N]]);
a1:=A.N;a2:=A.(N-1);
rel1:=a1^2-s-(s-1)*a1;
rel2:=a2^2-t-(t-1)*a2;
rel3:=(a2*a1)^l-(a1*a2)^l;

def1:=[(a1*a2)^i-A.(N-4*i+2): i in [1..l]];
def2:=[(a2*a1)^i-A.(N-4*i+1): i in [1..l-1]];
def3:=[a1*(a2*a1)^i-A.(N-4*i): i in [1..l-1]];
def4:=[a2*(a1*a2)^i-A.(N-4*i-1): i in [1..l-1]];

ID:=ideal<A|rel1,rel2,rel3,def1,def2,def3,def4>;
G:=GroebnerBasis(ID);G;#G;
```

```
//Gr\"obner basis for generalized quadrangles///////////////////////////////

a7^2 -(t^2s^2-t^2s+t^2-ts^2-t+s^2-s+1)*a7-(t^2s^2-t^2s-ts^2+ts+s^2-s)*a6
      -(t^2s^2-t^2s+t^2-ts^2+ts-t)*a5-(t^2s^2-t^2s-ts^2+ts)*a4
      -(t^2s^2-t^2s-ts^2+ts)*a3-(t^2s^2-ts^2)*a2-(t^2s^2-t^2s)*a1-(t^2s^2)*a0,
a7*a6-(t^2s-t^2-ts+t+s-1)*a7-(t^2s-2ts+s)*a6-(t^2s-t^2-ts+t)*a5-(t^2s-ts)*a4
      -(t^2s-ts)*a3-(t^2s)*a1,
a7*a5-(ts^2-ts+t-s^2+s-1)*a7-(ts^2-ts-s^2+s)*a6-(ts^2-2ts+t)*a5-(ts^2-ts)*a4
      -(ts^2-ts)*a3-(ts^2)*a2,
a7*a4-(ts-t-s+1)*a7-(ts-s)*a6-(ts-t)*a5-(ts)*a3,
a7*a3-(ts-t-s+1)*a7-(ts-s)*a6-(ts-t)*a5-(ts)*a4,
a7*a2-(t-1)*a7-(t)*a5,
a7*a1-(s-1)*a7-(s)*a6,
a6*a7-(t^2s-t^2-ts+t+s-1)*a7-(t^2s-2ts+s)*a6-(t^2s-t^2-ts+t)*a5
      -(t^2s-ts)*a4-(t^2s-ts)*a3-(t^2s)*a1,
a6^2 -(t^2-2t+1)*a7-(ts-t)*a6-(t^2-t)*a5-(t^2s-ts)*a2-(t^2s)*a0,
a6*a5-(ts-t-s+1)*a7-(ts-s)*a6-(ts-t)*a5-(ts)*a3,
a6*a4-(t-1)*a7-(ts-t)*a4-(ts)*a2,
a6*a3-(t-1)*a7-(t)*a5,
a6*a2-(t-1)*a6-(t)*a4,
a6*a1-(1)*a7,
a5*a7-(ts^2-ts+t-s^2+s-1)*a7-(ts^2-ts-s^2+s)*a6
      -(ts^2-2ts+t)*a5-(ts^2-ts)*a4-(ts^2-ts)*a3-(ts^2)*a2,
a5*a6-(ts-t-s+1)*a7-(ts-s)*a6-(ts-t)*a5-(ts)*a4,
a5^2 -(s^2-2s+1)*a7-(s^2-s)*a6-(ts-s)*a5-(ts^2-ts)*a1-(ts^2),
a5*a4-(s-1)*a7-(s)*a6,
a5*a3-(s-1)*a7-(ts-s)*a3-(ts)*a1,
a5*a2-(1)*a7,
a5*a1-(s-1)*a5-(s)*a3,
a4*a7-(ts-t-s+1)*a7-(ts-s)*a6-(ts-t)*a5-(ts)*a3,
a4*a6-(t-1)*a7-(t)*a5,
a4*a5-(s-1)*a7-(ts-s)*a4-(ts)*a1,
a4^2 -(1)*a7,
a4*a3-(s-1)*a6-(ts-s)*a2-(ts)*a0,
a4*a2-(1)*a6,
a4*a1-(s-1)*a4-(s)*a2,
a3*a7-(ts-t-s+1)*a7-(ts-s)*a6-(ts-t)*a5-(ts)*a4,
a3*a6-(t-1)*a7-(ts-t)*a3-(ts)*a2,
a3*a5-(s-1)*a7-(s)*a6,
a3*a4-(t-1)*a5-(ts-t)*a1-(ts)*a0,
a3^2 -(1)*a7,
a3*a2-(t-1)*a3-(t)*a1,
a3*a1-(1)*a5,
a2*a7-(t-1)*a7-(t)*a5,
a2*a6-(t-1)*a6-(t)*a3,
a2*a5-(1)*a7,
a2*a4-(t-1)*a4-(t)*a1,
a2*a3-(1)*a6,
a2^2 -(t-1)*a2-(t)*a0,
a2*a1-(1)*a4,
a1*a7-(s-1)*a7-(s)*a6,
a1*a6-(1)*a7,
a1*a5-(s-1)*a5-(s)*a4,
a1*a4-(1)*a5,
a1*a3-(s-1)*a3-(s)*a2,
a1*a2-(1)*a3,
a1^2 -(s-1)*a1-(s)*a0

//edited partitions for generalized 4-gons eliminating those with only t=1=s
partition1= [[ 1, 2, 3, 4, 5, 6, 7 ]]
[]
partition3= [[ 1, 2, 3, 4, 5, 7 ],[ 6 ]]
[T-1]
partition4= [[ 1, 2, 3, 4, 6, 7 ],[ 5 ]]
[S-1]
partition5= [[ 1, 2, 3, 4, 7 ],[ 5, 6 ]]
[T-2,S-2]
partition10= [[ 1, 2, 7 ],[ 3, 4 ],[ 5, 6 ]]
[T-2,S-2]
partition11= [[ 1, 2 ],[ 3, 4 ],[ 5, 6 ],[ 7 ]]
[T-S]
partition13= [[ 1, 3, 4, 5, 6, 7 ],[ 2 ]]
[]
partition14= [[ 1, 3, 4, 6 ],[ 2, 5, 7 ]]
[S-1]
partition15= [[ 1, 3, 4, 6 ],[ 2, 5 ],[ 7 ]]
[S-1]
partition17= [[ 1, 3, 4, 6 ],[ 2, 7 ],[ 5 ]]
[S-1]
partition18= [[ 1, 3, 4, 6 ],[ 2 ],[ 5, 7 ]]
```

```
[]
partition19= [[ 1, 3, 4, 6 ],[ 2 ],[ 5 ],[ 7 ]]
[S-1]
partition20= [[ 1, 6, 7 ],[ 2, 3, 4, 5 ]]
[T-1]
partition21= [[ 1, 6 ],[ 2, 3, 4, 5 ],[ 7 ]]
[T-1]
partition22= [[ 1 ],[ 2, 3, 4, 5, 6, 7 ]]
[]
partition23= [[ 1, 7 ],[ 2, 3, 4, 5 ],[ 6 ]]
[T-1]
partition24= [[ 1 ],[ 2, 3, 4, 5 ],[ 6, 7 ]]
[]
partition25= [[ 1 ],[ 2, 3, 4, 5 ],[ 6 ],[ 7 ]]
[T-1]
partition27= [[ 1, 5, 6 ],[ 2 ],[ 3, 4, 7 ]]
[T-1,S^2-4*S+3]=[T-1,(S-3)(S-1)]
partition29= [[ 1, 6, 7 ],[ 2, 5 ],[ 3, 4 ]]
[T-1,S-3]
partition30= [[ 1, 6 ],[ 2, 5, 7 ],[ 3, 4 ]]
[T-3,S-1]
partition32= [[ 1 ],[ 2, 5, 6 ],[ 3, 4, 7 ]]
[T^2-4*T+3,S-1]=[(T-3)(T-1),S-1]
partition33= [[ 1 ],[ 2, 5, 7 ],[ 3, 4, 6 ]]
[S-1]
partition34= [[ 1 ],[ 2, 5 ],[ 3, 4 ],[ 6 ],[ 7 ]]
[S-1]
partition35= [[ 1, 6, 7 ],[ 2 ],[ 3, 4, 5 ]]
[T-1]
partition36= [[ 1, 6 ],[ 2 ],[ 3, 4 ],[ 5 ],[ 7 ]]
[T-1]
partition38= [[ 1 ],[ 2 ],[ 3 ],[ 4 ],[ 5 ],[ 6 ],[ 7 ]]
[]
```

```
//fusion partitions for gen. hexagons other than those with only  t=1=s
partition1= [[ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 ]]
[]
partition7= [[ 1, 2, 3, 4, 5, 7, 8, 9, 10, 11 ],[ 6 ]]
[T-1]
partition8= [[ 1, 2, 3, 4, 6, 7, 8, 9, 10, 11 ],[ 5 ]]
[S-1]
partition26= [[ 1, 2 ],[ 3, 4 ],[ 5, 6 ],[ 7, 8 ],[ 9, 10 ],[ 11 ]]
[T-S]
partition30= [[ 1, 3, 4, 5, 6, 7, 8, 9, 10, 11 ],[ 2 ]]
[]
partition34= [[ 1, 3, 4, 5, 6, 7, 8, 10 ],[ 2 ],[ 9, 11 ]]
[T-S]
partition37= [[ 1, 3, 4, 6, 9, 11 ],[ 2, 5, 7, 8, 10 ]]
[S-1]
partition41= [[ 1, 3, 4, 6, 9, 11 ],[ 2, 5, 10 ],[ 7, 8 ]]
[T^2-5*T+4,S-1]=[(T-1)(T-4),S-1]
partition42= [[ 1, 3, 4, 6, 9, 11 ],[ 2, 5 ],[ 7, 8 ],[ 10 ]]
[S-1]
partition45= [[ 1, 3, 4, 6, 9, 11 ],[ 2, 7, 8, 10 ],[ 5 ]]
[S-1]
partition46= [[ 1, 3, 4, 6, 9, 11 ],[ 2, 7, 8 ],[ 5, 10 ]]
[T-2,S-1]
partition47= [[ 1, 3, 4, 6, 9, 11 ],[ 2, 10 ],[ 5, 7, 8 ]]
[T-2,S-1]
partition48= [[ 1, 3, 4, 6, 9, 11 ],[ 2 ],[ 5, 7, 8, 10 ]]
[T^2*S-T^2-T*S^2+3*T*S-2*T+S-1]=[(S-1)(T^2-TS+2T+1)] so S=1 or (T=1,S=4)
partition49= [[ 1, 3, 4, 6 ],[ 2 ],[ 5, 7, 8, 10 ],[ 9, 11 ]]
[]
partition54= [[ 1, 3, 4, 6, 9, 11 ],[ 2 ],[ 5 ],[ 7 ],[ 8 ],[ 10 ]]
[S-1]
partition55= [[ 1, 3, 4, 9 ],[ 2, 7, 8, 10 ],[ 5 ],[ 6, 11 ]]
[S-1]
partition58= [[ 1, 6, 7, 8, 9 ],[ 2, 3, 4, 5, 10, 11 ]]
[T-1]
partition62= [[ 1, 6, 9 ],[ 2, 3, 4, 5, 10, 11 ],[ 7, 8 ]]
[T-1,S^2-5*S+4]=[T-1,(S-4)(S-1)]
partition63= [[ 1, 6 ],[ 2, 3, 4, 5, 10, 11 ],[ 7, 8 ],[ 9 ]]
[T-1]
partition67= [[ 1 ],[ 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 ]]
```

```
[]
partition69= [[ 1 ],[ 2, 3, 4, 5, 6, 7, 8, 9 ],[ 10, 11 ]]
[T-S]
partition72= [[ 1, 7, 8, 9 ],[ 2, 3, 4, 5, 10, 11 ],[ 6 ]]
[T-1]
partition73= [[ 1, 7, 8 ],[ 2, 3, 4, 5, 10, 11 ],[ 6, 9 ]]
[T-1,S-2]
partition74= [[ 1, 9 ],[ 2, 3, 4, 5, 10, 11 ],[ 6, 7, 8 ]]
[T-1,S-2]
partition75= [[ 1 ],[ 2, 3, 4, 5, 10, 11 ],[ 6, 7, 8, 9 ]]
[T^2*S-T*S^2-3*T*S-T+S^2+2*S+1]=[(T-1)(TS-S^2-2S-1)] so T=1 or (S=1,T=4)
partition76= [[ 1 ],[ 2, 3, 4, 5 ],[ 6, 7, 8, 9 ],[ 10, 11 ]]
[]
partition81= [[ 1 ],[ 2, 3, 4, 5, 10, 11 ],[ 6 ],[ 7 ],[ 8 ],[ 9 ]]
[T-1]
partition83= [[ 1, 7, 8, 9 ],[ 2, 3, 4, 10 ],[ 5, 11 ],[ 6 ]]
[T-1]
partition100= [[ 1 ],[ 2, 5, 7, 8, 10 ],[ 3, 4, 6, 9, 11 ]]
[S-1]
partition105= [[ 1 ],[ 2, 5, 10 ],[ 3, 4, 11 ],[ 6, 9 ],[ 7, 8 ]]
[T^2-5*T+4,S-1]=[(T-4)(T-1),S-1]
partition106= [[ 1 ],[ 2, 5 ],[ 3, 4 ],[ 6, 9 ],[ 7, 8 ],[ 10 ],[ 11 ]]
[S-1]
partition108= [[ 1, 6, 7, 8, 9 ],[ 2 ],[ 3, 4, 5, 10, 11 ]]
[T-1]
partition113= [[ 1, 6, 9 ],[ 2 ],[ 3, 4, 11 ],[ 5, 10 ],[ 7, 8 ]]
[T-1,S^2-5*S+4]=[T-1,(S-4)(S-1)]
partition114= [[ 1, 6 ],[ 2 ],[ 3, 4 ],[ 5, 10 ],[ 7, 8 ],[ 9 ],[ 11 ]]
[T-1]
partition124= [[ 1 ],[ 2 ],[ 3 ],[ 4 ],[ 5 ],[ 6 ],[ 7 ],[ 8 ],[ 9 ],[ 10 ],[ 11 ]
[]
```

```
//non-trivial octagon fusions
partition1= [[ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 ]]
[]
partition10= [[ 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15 ],[ 6 ]]
[T-1]
partition16= [[ 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 ],[ 5 ]]
[S-1]
partition47= [[ 1, 2 ],[ 3, 4 ],[ 5, 6 ],[ 7, 8 ],[ 9, 10 ],[ 11, 12 ],[ 13, 14 ],
[T-S]
partition56= [[ 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 ],[ 2 ]]
[]
partition60= [[ 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14 ],[ 2 ],[ 13, 15 ]]
[T-S]
partition62= [[ 1, 3, 4, 5, 6, 7, 8, 10, 13, 15 ],[ 2 ],[ 9, 11, 12, 14 ]]
[[T-1,S-2]
partition63= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2, 5, 7, 8, 10, 13, 15 ]]
[S-1]
partition69= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2, 5, 7, 8, 15 ],[ 10, 13 ]]
[T-2,S-1]
partition77= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2, 5, 15 ],[ 7, 8 ],[ 10, 13 ]]
[T-2,S-1]
partition78= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2, 5 ],[ 7, 8 ],[ 10, 13 ],[ 15 ]]
[S-1]
partition83= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2, 7, 8, 10, 13, 15 ],[ 5 ]]
[S-1]
partition88= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2, 7, 8, 13 ],[ 5 ],[ 10, 15 ]]
[S-1]
partition93= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2 ],[ 5, 7, 8, 10, 13, 15 ]]
[S-1]
partition95= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2 ],[ 5, 7, 8, 10 ],[ 13, 15 ]]
[S-1]
partition99= [[ 1, 3, 4, 6, 13, 15 ],[ 2 ],[ 5, 7, 8, 10 ],[ 9, 11, 12, 14 ]]
[T-1,S-2]
partition100= [[ 1, 3, 4, 6 ],[ 2 ],[ 5, 7, 8, 10 ],[ 9, 11, 12, 14 ],[ 13, 15 ]]
[]
partition112= [[ 1, 3, 4, 6, 9, 11, 12, 14 ],[ 2 ],[ 5 ],[ 7 ],[ 8 ],[ 10 ],[ 13 ]
[S-1]
partition113= [[ 1, 3, 4, 9 ],[ 2, 7, 8, 13 ],[ 5 ],[ 6, 11, 12, 14 ],[ 10, 15 ]]
[S-1]
partition116= [[ 1, 6, 7, 8, 9, 14, 15 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ]]
```

```
[T - 1]
partition122= [[ 1, 6, 7, 8, 15 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 9, 14 ]]
[T-1,S-2]
partition127= [[ 1, 6, 15 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 7, 8 ],[ 9, 14 ]]
[T-1,S-2]
partition128= [[ 1, 6 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 7, 8 ],[ 9, 14 ],[ 15 ]]
[T-1]
partition132= [[ 1 ],[ 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 ]]
[]
partition134= [[ 1 ],[ 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13 ],[ 14, 15 ]]
[T-S]
partition136= [[ 1 ],[ 2, 3, 4, 5, 6, 7, 8, 9, 14, 15 ],[ 10, 11, 12, 13 ]]
[T-2,S-1]
partition137= [[ 1, 7, 8, 9, 14, 15 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 6 ]]
[T-1]
partition142= [[ 1, 7, 8, 14 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 6 ],[ 9, 15 ]]
[T-1]
partition147= [[ 1 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 6, 7, 8, 9, 14, 15 ]]
[T - 1]
partition149= [[ 1 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 6, 7, 8, 9 ],[ 14, 15 ]]
[T - 1]
partition152= [[ 1 ],[ 2, 3, 4, 5, 14, 15 ],[ 6, 7, 8, 9 ],[ 10, 11, 12, 13 ]]
[T-2,S-1]
partition153= [[ 1 ],[ 2, 3, 4, 5 ],[ 6, 7, 8, 9 ],[ 10, 11, 12, 13 ],[ 14, 15 ]]
[]
partition166= [[ 1 ],[ 2, 3, 4, 5, 10, 11, 12, 13 ],[ 6 ],[ 7 ],[ 8 ],[ 9 ],[ 14 ]
[T-1]
partition170= [[ 1, 7, 8, 14 ],[ 2, 3, 4, 10 ],[ 5, 11, 12, 13 ],[ 6 ],[ 9, 15 ]]
[T-1]
partition197= [[ 1 ],[ 2, 5, 7, 8, 10, 13, 15 ],[ 3, 4, 6, 9, 11, 12, 14 ]]
[S-1]
partition199= [[ 1 ],[ 2, 5, 7, 8, 15 ],[ 3, 4, 6, 9, 14 ],[ 10, 13 ],[ 11, 12 ]]
[T-2,S-1]
partition211= [[ 1 ],[ 2, 5, 15 ],[ 3, 4, 14 ],[ 6, 9 ],[ 7, 8 ],[ 10, 13 ],[ 11,
[T-2,S-1]
partition212= [[ 1 ],[ 2, 5 ],[ 3, 4 ],[ 6, 9 ],[ 7, 8 ],[ 10, 13 ],[ 11, 12 ],[ 1
[S-1]
partition217= [[ 1, 6, 7, 8, 9, 14, 15 ],[ 2 ],[ 3, 4, 5, 10, 11, 12, 13 ]]
[T-1]
partition219= [[ 1, 6, 7, 8, 15 ],[ 2 ],[ 3, 4, 5, 10, 13 ],[ 9, 14 ],[ 11, 12 ]]
```

```
[T-1,S-2]
partition235= [[ 1, 6, 15 ],[ 2 ],[ 3, 4, 13 ],[ 5, 10 ],[ 7, 8 ],[ 9, 14 ],[ 11,
[T-1,S-2]
partition236= [[ 1, 6 ],[ 2 ],[ 3, 4 ],[ 5, 10 ],[ 7, 8 ],[ 9, 14 ],[ 11, 12 ],[ 1
[T-1]
partition273= [[ 1 ],[ 2 ],[ 3 ],[ 4 ],[ 5 ],[ 6 ],[ 7 ],[ 8 ],[ 9 ],[ 10 ],[ 11 ]
[]
```

```
//preamble code for 2-(v,k,1) designs
N:=6;
AT:=[1,2,4,3,5,6];
Q:=RationalField();
FF<t,s>:=FunctionField(Q,2);
A:=FreeAlgebra(FF,N);
AssignNames(~A,["a" cat IntegerToString(N+1-i): i in [1..N]]);
a1:=A.6;
a2:=A.5;
a3:=A.4;
a4:=A.3;
a5:=A.2;
a6:=A.1;
rel1:=a1^2-s-(s-1)*a1;rel1;
rel2:=a2^2-t-(t-1)*a2;rel2;
rel3:=(a2*a1)^2-s*a6-(s-1)*a5-s*a3;rel3;
rel4:=(a1*a2)^2-s*a6-(s-1)*a5-s*a4;rel4;

def1:=a1*a2-A.4;def1;
def2:=a2*a1-A.3;def2;
def3:=a1*a2*a1-A.2;def3;
def4:=(a2*a1*a2)-(a1*a2*a1)-A.1;def4;

ID:=ideal<A|rel1,rel2,rel3,rel4,def1,def2,def3,def4>;
G:=GroebnerBasis(ID);G;#G;
```

```
a6^2 -(t^2s-2ts^2-2ts-t+s^3+3s^2+2s)*a6
     -(t^2s-2ts^2-ts+s^3+s^2)*a5-(t^2s-2ts^2-ts+s^3+s^2)*a4
     -(t^2s-2ts^2-ts+s^3+s^2)*a3-(t^2s-2ts^2-ts+s^3+s^2)*a2
     -(t^2s-ts^2)*a1-(t^2s-ts^2)*a0,
a6*a5-(ts^2-s^3-s^2)*a6-(ts^2-ts-s^3+s^2)*a5-(ts^2-ts-s^3+s^2)*a4
     -(ts^2-s^3)*a3-(ts^2-s^3)*a2,
a6*a4-(ts-s^2-s)*a6-(ts-t-s^2+s)*a5-(ts-t-s^2+s)*a4-(ts-s^2)*a3-(ts-s^2)*a2,
a6*a3-(ts-s^2-s)*a6-(ts-s^2)*a5-(ts-s^2)*a4,
a6*a2-(t-s-1)*a6-(t-s)*a5-(t-s)*a4,
a6*a1-(s)*a6,
a5*a6-(ts^2-s^3-s^2)*a6-(ts^2-ts-s^3+s^2)*a5-(ts^2-s^3)*a4-(ts^2-ts-s^3+s^2)*a3
     -(ts^2-s^3)*a2,
a5^2 -(s^3-s^2)*a6-(ts+s^3-3s^2+2s-1)*a5-(s^3-2s^2+s)*a4-(s^3-2s^2+s)*a3
     -(s^3-s^2)*a2-(ts^2-ts)*a1-(ts^2)*a0,
a5*a4-(s^2)*a6-(s^2-2s+1)*a5-(s^2-s)*a4-(s^2-s)*a3-(s^2)*a2,
a5*a3-(s^2-s)*a6-(s^2-2s+1)*a5-(s^2-s)*a4-(ts- s)*a3-(ts)*a1,
a5*a2-(s)*a6-(s-1)*a5-(s)*a4,
a5*a1-(s-1)*a5-(s)*a3,
a4*a6-(ts-s^2-s)*a6-(ts-s^2)*a5-(ts-s^2)*a3,
a4*a5-(s^2-s)*a6-(s^2-2s+1)*a5-(ts-s)*a4-(s^2-s)*a3-(ts)*a1,
a4^2 -(s)*a6-(s-1)*a5-(s)*a3,
a4*a3-(s-1)*a6-(s-1)*a5-(ts-s)*a2-(ts)*a0,
a4*a2-(1)*a6-(1)*a5,
a4*a1-(s-1)*a4-(s)*a2,
a3*a6-(ts-s^2-s)*a6-(ts-t-s^2+s)*a5-(ts-s^2)*a4-(ts-t-s^2+s)*a3-(ts-s^2)*a2,
a3*a5-(s^2)*a6-(s^2-2s+1)*a5-(s^2-s)*a4-(s^2-s)*a3-(s^2)*a2,
a3*a4-(t-1)*a5-(ts-t)*a1-(ts)*a0,
a3^2 -(s)*a6-(s-1)*a5-(s)*a4,
a3*a2-(t-1)*a3-(t)*a1,
a3*a1-(1)*a5,
a2*a6-(t-s-1)*a6-(t-s)*a5-(t-s)*a3,
a2*a5-(s)*a6-(s-1)*a5-(s)*a3,
a2*a4-(t-1)*a4-(t)*a1,
a2*a3-(1)*a6-(1)*a5,
a2^2 -(t-1)*a2-(t)*a0,
a2*a1-(1)*a4,
a1*a6-(s)*a6,
a1*a5-(s-1)*a5-(s)*a4,
a1*a4-(1)*a5,
a1*a3-(s-1)*a3-(s)*a2,
a1*a2-(1)*a3,
a1^2 -(s-1)*a1-(s)*a0
```

```
//fusion for 2-(v,k,1) designs except for those partitions with only t=1=s
partition1= [[ 1, 2, 3, 4, 5, 6 ]]
[]
%partition2= [[ 1, 2, 3, 4, 5 ],[ 6 ]]
%[T-S]
partition3= [[ 1, 2, 3, 4, 6 ],[ 5 ]]
[S-1]
%partition4= [[ 1, 2, 3, 4 ],[ 5, 6 ]]
%[T-1]
%partition6= [[ 1, 2, 5 ],[ 3, 4 ],[ 6 ]]
%[T-S,S^2-5*S+4]=[T-1,(S-4)(S-1)]
%partition7= [[ 1, 2 ],[ 3, 4 ],[ 5 ],[ 6 ]]
%[T-S]
partition9= [[ 1, 3, 4, 5, 6 ],[ 2 ]]
[]
%partition11= [[ 1, 3, 4, 5 ],[ 2 ],[ 6 ]]
%[T-S]
%partition12= [[ 1, 3, 4 ],[ 2, 5 ],[ 6 ]]
%[T-2,S-2]
partition13= [[ 1, 5 ],[ 2, 3, 4, 6 ]]
[S-2]
%partition14= [[ 1, 5 ],[ 2, 3, 4 ],[ 6 ]]
%[T-2,S-2]
partition15= [[ 1, 6 ],[ 2, 3, 4, 5 ]]
[T-S-1]
partition16= [[ 1 ],[ 2, 3, 4, 5, 6 ]]
[]
partition17= [[ 1 ],[ 2, 3, 4, 5 ],[ 6 ]]
[]
partition18= [[ 1, 6 ],[ 2, 3, 4 ],[ 5 ]]
[T-2,S-1]
partition20= [[ 1, 6 ],[ 2, 5 ],[ 3, 4 ]]
[T-4,S-3]
partition21= [[ 1 ],[ 2, 5 ],[ 3, 4 ],[ 6 ]]
[S-1]
partition22= [[ 1, 6 ],[ 2 ],[ 3, 4, 5 ]]
[T-S-1]
partition23= [[ 1 ],[ 2 ],[ 3 ],[ 4 ],[ 5 ],[ 6 ]]
[]
```

References

[1] Cox, Little, ans O'Shea, Ideals, Varieties, and Algorithms An Introduction to Computational Algebraic Geometry and Commutative Algebra, 2nd Edition, Springer Verlag, 1996

[2] Mikhail Klin, Christian Pech, and Paul-Hermann Zieschang, Flag algebras of block designs, I. Initial notions. Steiner 2-designs and generalized quadrangles, Math-AL-10-1998, Technische Universität, Dresden, November, 1998, preprint.

[3] The MAGMA Computational Algebra System for Algebra, Number Theory and Geometry. The University of Sydney Computational Algebra Group. `http://magma.maths.usyd.edu.au/magma`.

[4] J.Tits, Sur la trialité et certains groupes qui s'en déduisent, Inst. Hautes Etudes Sci. Publ. Math 2 (1959), 14-60.