

Quantum Coloring Problem

- This .nb is a companion to the proof in the paper by Helton Meyer Paulsen Satriano (arXiv 2017) that the ideal $I_{\text{alg}} = I(G, K_c)$ associated to any graph G with c quantum colors is proper.
-

Basic coding to produce a GB for $I(G, K_c)$ given a graph G and a natural number c .

The inputs of our function FindGB are the following :
a directed graph G ,
(for our applications we will always include $i \rightarrow j$ and $j \rightarrow i$ for any edge we are including);
 n , the number of vertices of our graph;
 c , the number of colors.

prec, the number of iterations of Mora's noncommutative Groebner basis algorithm that we are willing to run in our calculation.

This code works as follows :

Step 1 creates our variables e_{ij} for i from 1 to n and j from 1 to c ,
Step 2 declares these variables to be noncommutative
Step 3 declares a monomial ordering on these variables
Step 4 – 7 create a list of the relations generating $I(G, K_c)$
Step 8 calculates the Groebner Basis

```

<< NC`
<< NCGB`
(* This step produces a table with all
   of the variables to be used in the form e_ij *)
Step[1, G_, n_, c_, prec_] :=
  VarsMat = Table[Symbol["e" <> ToString[i] <> ToString[j]], {i, n}, {j, c}];
(* Sets the variables as non-commutative *)
Step[2, G_, n_, c_, prec_] := SetNonCommutative[Flatten[VarsMat]];
(* Sets a graded lexicographic monomial ordering given by e_11<e_12<...e_1c<e_21<...
   e_nc *)
Step[3, G_, n_, c_, prec_] :=
  SetMonomialOrder[Flatten[VarsMat]];

(* Now we proceed to make a list of relations
   which characterize quantum coloring of graphs *)

(* Next step makes a list of relations of the form e_ij*e_ik for j\neq k *)

Step[4, G_, n_, c_, prec_] :=
  Relations1 = Table[Table[{VarsMat[[i, j]] ** VarsMat[[i, k]],
    VarsMat[[i, k]] ** VarsMat[[i, j]]}, {k, j+1, c}], {i, n}, {j, c-1}];
(* Makes a list of relations of the form sum_j e_ij-1 *)
Step[5, G_, n_, c_, prec_] :=
  Relations2 = Table[Sum[VarsMat[[i, j]], {j, c}] - 1, {i, n}];
(* Makes a list of relations of the form e_ij*e_kj
   where (i,k) is an edge of G *)
Step[6, G_, n_, c_, prec_] := Relations3 = Table[
  VarsMat[[G[[e]][[1]], i]] ** VarsMat[[G[[e]][[2]], i]], {i, c}, {e, Length[G]}];
(* Puts the the relations from steps 4-6 into a single list *)
Step[7, G_, n_, c_, prec_] :=
  Relations = Flatten[{Relations1, Relations2, Relations3}];

(* Now we process the relations just constructed. *)

(* Calculate the Groebner basis for the relations *)

Step[8, G_, n_, c_, prec_] :=
  GB = NCMakeGB[Relations, prec];

(* Does all of the above steps for a graph G with n nodes,
   c colors and doing prec iterations *)
FindGB[G_, n_, c_, prec_] := Do[Step[i, G, n, c, prec], {i, 8}];
(* Reduces an expression pol using the rules rul *)
Reduc[pol_, rul_] := Block[{poln = NCReplaceRepeated[NCEExpand[pol], rul]},
  If[SameQ[pol, poln], pol, Reduc[poln, rul]]];

```

You are using the version of NCAIgebra which is found in:

C:\Users\Kyle\Desktop\NC

You can now use "<< NCAIgebra`" to load NCAIgebra or "<< NCGB`" to load NCGB.

You have already loaded NCGB.m

Examples of Groebner basis calculations for various graphs

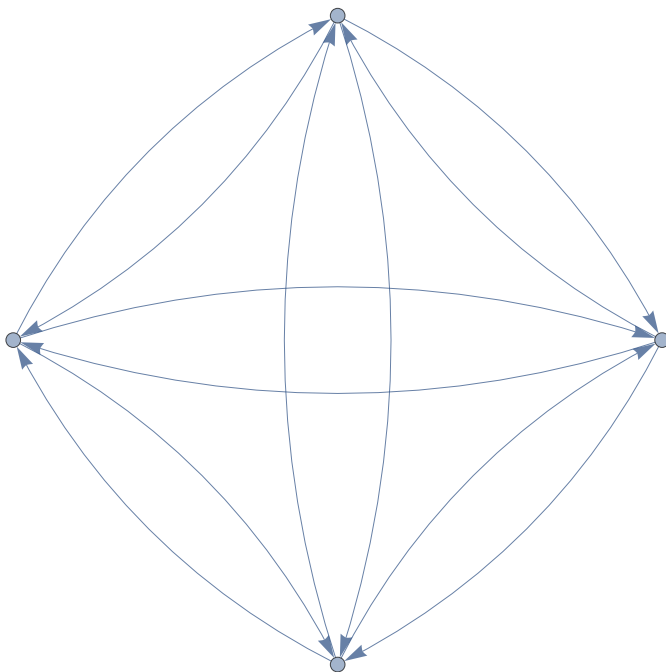
The following graphs G1, G2, G3, G4 are not 3 algebraically colorable since 1 is in their GB.

G1 = {1 → 2, 2 → 1, 1 → 3, 3 → 1, 1 → 4, 4 → 1, 2 → 3, 3 → 2, 2 → 4, 4 → 2, 3 → 4, 4 → 3};

Graph[G1]

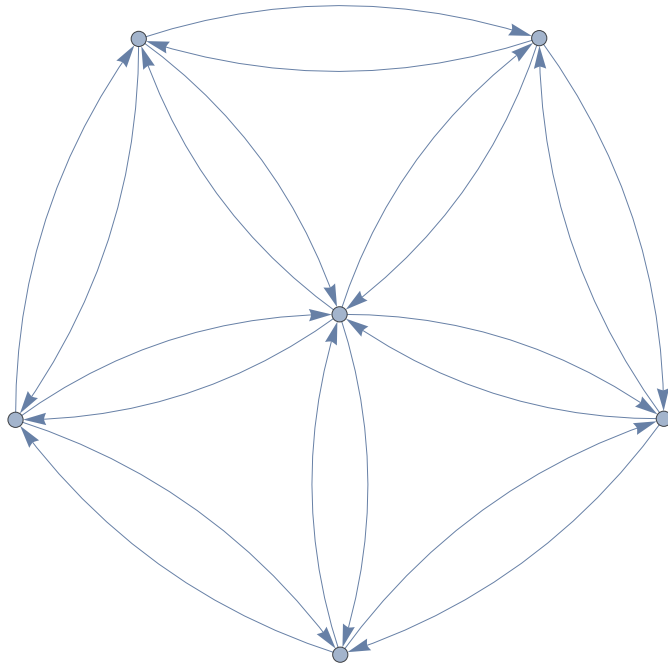
FindGB[G1, 4, 3, 5]

Print[GB]



{1}

```
G2 = {1 → 2, 2 → 1, 1 → 3, 3 → 1, 1 → 4, 4 → 1, 1 → 5, 5 → 1, 1 → 6,  
      6 → 1, 2 → 3, 3 → 2, 3 → 4, 4 → 3, 4 → 5, 5 → 4, 5 → 6, 6 → 5, 6 → 2, 2 → 6};  
Graph[G2]  
FindGB[G2, 6, 3, 5]  
Print[GB]
```

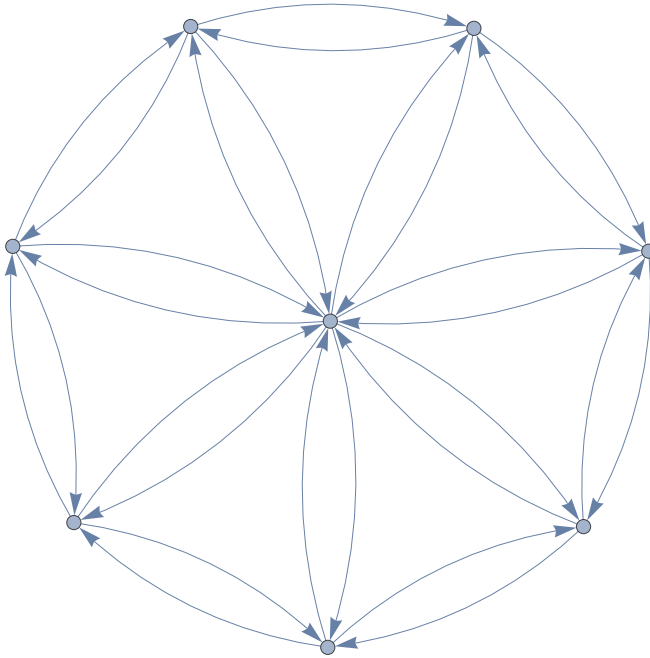


```
{1}
```

```

G3 = {1 → 2, 2 → 1, 1 → 3, 3 → 1, 1 → 4, 4 → 1, 1 → 5, 5 → 1,
      1 → 6, 6 → 1, 1 → 7, 7 → 1, 1 → 8, 8 → 1, 2 → 3, 3 → 2, 3 → 4, 4 → 3,
      4 → 5, 5 → 4, 5 → 6, 6 → 5, 6 → 7, 7 → 6, 7 → 8, 8 → 7, 8 → 2, 2 → 8};
Graph[G3]
FindGB[G3, 8, 3, 5]
Print[GB]

```

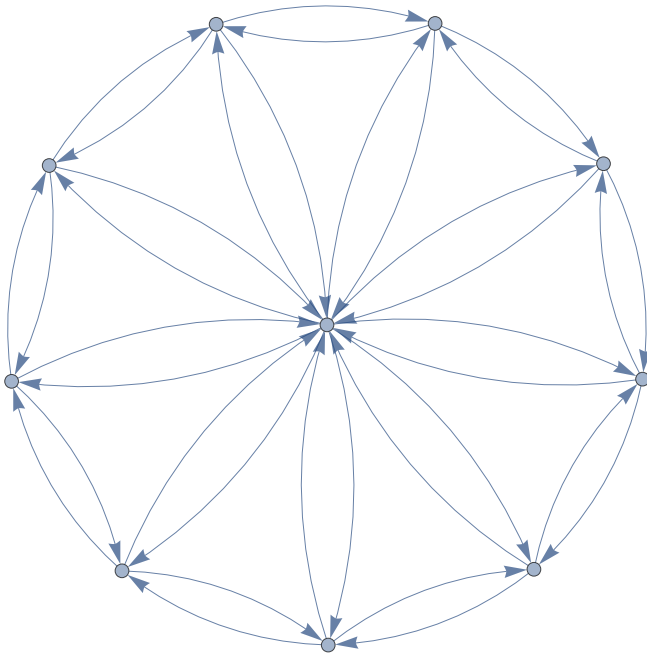


```
{1}
```

```

G4 = {1 → 2, 2 → 1, 1 → 3, 3 → 1, 1 → 4, 4 → 1, 1 → 5, 5 → 1, 1 → 6, 6 → 1, 1 → 7, 7 → 1, 1 → 8,
      8 → 1, 1 → 9, 9 → 1, 1 → 10, 10 → 1, 2 → 3, 3 → 2, 3 → 4, 4 → 3, 4 → 5, 5 → 4, 5 → 6,
      6 → 5, 6 → 7, 7 → 6, 7 → 8, 8 → 7, 8 → 9, 9 → 8, 9 → 10, 10 → 9, 10 → 2, 2 → 10};
Graph[G4]
FindGB[G4, 10, 3, 5]
Print[GB]

```



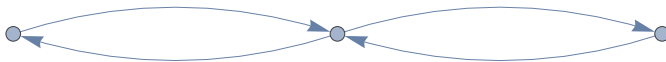
```
{1}
```

The graph G5 is 2 algebraically colorable since 1 is not in its GB

```

G5 = {1 → 2, 2 → 1, 1 → 3, 3 → 1};
Graph[G5]
FindGB[G5, 3, 2, 5]
Print[GB]

```



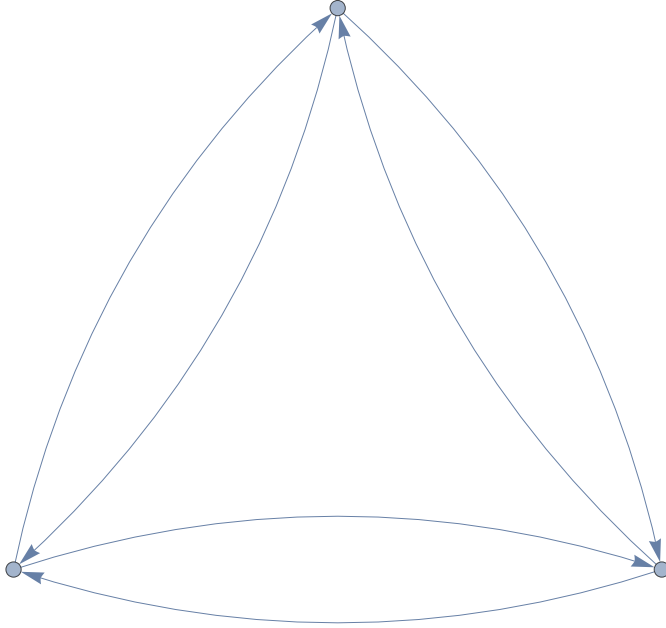
```
{-1 + e11 + e12, -e11 + e11 ** e11, -1 + e11 + e21, -1 + e11 + e31, -e11 + e22, -e11 + e32}
```

The graph G6 is 3 algebraically colorable

```

G6 = {1 → 2, 2 → 1, 1 → 3, 3 → 1, 2 → 3, 3 → 2};
Graph[G6]
FindGB[G5, 3, 3, 5]
Print[GB]

```



```

{e11 ** e12, e12 ** e11, e21 ** e22, e22 ** e21, e31 ** e32, e32 ** e31,
 -1 + e11 + e12 + e13, -1 + e21 + e22 + e23, -1 + e31 + e32 + e33, e11 ** e21,
 e21 ** e11, e11 ** e31, e31 ** e11, e12 ** e22, e22 ** e12, e12 ** e32, e32 ** e12,
 -e11 + e11 ** e11, -e12 + e12 ** e12, -e21 + e21 ** e21, -e22 + e22 ** e22,
 -e31 + e31 ** e31, -e32 + e32 ** e32, 1 - e11 - e12 - e21 - e22 + e11 ** e22 + e12 ** e21,
 1 - e11 - e12 - e31 - e32 + e11 ** e32 + e12 ** e31, 1 - e11 - e12 - e21 - e22 + e11 ** e22 + e21 ** e12,
 -e22 + e32 + e11 ** e22 - e11 ** e32 - e21 ** e32 + e22 ** e31,
 1 - e11 - e12 - e31 - e32 + e11 ** e32 + e31 ** e12, -e11 ** e22 + e22 ** e11,
 -e11 ** e32 + e32 ** e11, e22 - e32 - e11 ** e22 + e11 ** e32 - e31 ** e22 + e32 ** e21,
 e22 - e11 ** e22 - e31 ** e22 - e32 ** e22 + e11 ** e32 ** e22,
 e32 - e11 ** e32 - e21 ** e32 - e22 ** e32 + e11 ** e22 ** e32,
 e31 ** e21 ** e32 + e31 ** e22 ** e32, e21 ** e31 ** e22 + e21 ** e32 ** e22}

```

Now we turn to calculations cited in HMPSarXiv2017 in the proof that every graph is algebraically 4 colorable.

CLAIM: Any syzygy between any two of the relations denoted “rels” listed below reduces to zero when reduced

by “rels”. To see this, it is sufficient (and necessary) to see that “rels” form a Groebner basis, specifically we show that they form a reduced Groebner basis for the ideal $I(K_6, K_4)$


```

(* Each Relsj creates a table of relations of type j,
then Rels is list with all of these relations*)
Rels1 = Table[Table[{VarsMat[[v, i]] ** VarsMat[[v, j]],
  VarsMat[[v, j]] ** VarsMat[[v, i]]}, {j, i + 1, 3}], {v, 6}, {i, 2}];
Rels2 = Table[VarsMat[[v, i]] ** VarsMat[[v, i]] - VarsMat[[v, i]],
  {v, 1, 6}, {i, 1, 3}];
Rels3 = Table[Sum[VarsMat[[v, i]], {i, 4}] - 1, {v, 6}];
Rels4 = Table[Table[{VarsMat[[v, i]] ** VarsMat[[w, i]],
  VarsMat[[w, i]] ** VarsMat[[v, i]]}, {v, w + 1, 6}], {w, 5}, {i, 3}];
Rels5 = Table[Table[VarsMat[[v, 3]] ** VarsMat[[w, 2]] +
  VarsMat[[v, 3]] ** VarsMat[[w, 1]] + VarsMat[[v, 2]] ** VarsMat[[w, 3]] +
  VarsMat[[v, 2]] ** VarsMat[[w, 1]] + VarsMat[[v, 1]] ** VarsMat[[w, 3]] +
  VarsMat[[v, 1]] ** VarsMat[[w, 2]] - VarsMat[[v, 3]] - VarsMat[[v, 2]] -
  VarsMat[[v, 1]] - VarsMat[[w, 3]] - VarsMat[[w, 2]] - VarsMat[[w, 1]] + 1,
  {v, Drop[{1, 2, 3, 4, 5, 6}, {w}]}], {w, {1, 2, 3, 4, 5, 6}}];
Rels6 = Table[Table[VarsMat[[v, 3]] ** VarsMat[[w, 1]] ** VarsMat[[v, 2]] -
  VarsMat[[v, 2]] ** VarsMat[[w, 3]] ** VarsMat[[v, 1]] -
  VarsMat[[v, 2]] ** VarsMat[[w, 1]] ** VarsMat[[v, 3]] -
  VarsMat[[v, 1]] ** VarsMat[[w, 3]] ** VarsMat[[v, 1]] -
  VarsMat[[v, 1]] ** VarsMat[[w, 2]] ** VarsMat[[v, 3]] -
  VarsMat[[v, 1]] ** VarsMat[[w, 2]] ** VarsMat[[v, 1]] +
  VarsMat[[v, 2]] ** VarsMat[[w, 3]] + VarsMat[[v, 2]] ** VarsMat[[w, 1]] +
  VarsMat[[v, 1]] ** VarsMat[[w, 3]] + VarsMat[[v, 1]] ** VarsMat[[w, 2]] +
  VarsMat[[w, 3]] ** VarsMat[[v, 1]] + VarsMat[[w, 2]] ** VarsMat[[v, 3]] +
  VarsMat[[w, 2]] ** VarsMat[[v, 1]] + VarsMat[[w, 1]] ** VarsMat[[v, 3]] -
  VarsMat[[v, 3]] - VarsMat[[v, 2]] - VarsMat[[v, 1]] -
  VarsMat[[w, 3]] - VarsMat[[w, 2]] - VarsMat[[w, 1]] + 1,
  {v, Drop[{1, 2, 3, 4, 5, 6}, {w}]}], {w, {1, 2, 3, 4, 5, 6}}];
Rels7 = Table[Table[Table[VarsMat[[v, 3]] ** VarsMat[[w, 1]] ** VarsMat[[x, 2]] -
  VarsMat[[v, 2]] ** VarsMat[[w, 3]] ** VarsMat[[x, 1]] -
  VarsMat[[v, 2]] ** VarsMat[[w, 1]] ** VarsMat[[x, 3]] -
  VarsMat[[v, 1]] ** VarsMat[[w, 3]] ** VarsMat[[x, 1]] -
  VarsMat[[v, 1]] ** VarsMat[[w, 2]] ** VarsMat[[x, 3]] -
  VarsMat[[v, 1]] ** VarsMat[[w, 2]] ** VarsMat[[x, 1]] +
  VarsMat[[v, 3]] ** VarsMat[[x, 1]] + VarsMat[[v, 2]] ** VarsMat[[w, 3]] +
  VarsMat[[v, 2]] ** VarsMat[[w, 1]] + 2 VarsMat[[v, 2]] ** VarsMat[[x, 3]] +
  2 VarsMat[[v, 2]] ** VarsMat[[x, 1]] + VarsMat[[v, 1]] ** VarsMat[[w, 3]] +
  VarsMat[[v, 1]] ** VarsMat[[w, 2]] + 2 VarsMat[[v, 1]] ** VarsMat[[x, 3]] +
  VarsMat[[v, 1]] ** VarsMat[[x, 2]] + VarsMat[[w, 3]] ** VarsMat[[x, 1]] +
  VarsMat[[w, 2]] ** VarsMat[[x, 3]] + VarsMat[[w, 2]] ** VarsMat[[x, 1]] +
  VarsMat[[w, 1]] ** VarsMat[[x, 3]] - VarsMat[[v, 3]] - 2 VarsMat[[v, 2]] -
  2 VarsMat[[v, 1]] - VarsMat[[w, 3]] - VarsMat[[w, 2]] - VarsMat[[w, 1]] -
  2 VarsMat[[x, 3]] - VarsMat[[x, 2]] - 2 VarsMat[[x, 1]] + 2,
  {v, Drop[Drop[{1, 2, 3, 4, 5, 6}, {Max[w, x]}], {Min[w, x]}]}],
  {w, Drop[{1, 2, 3, 4, 5, 6}, {x}]}], {x, {1, 2, 3, 4, 5, 6}}];
Rels = Flatten[{Rels1, Rels2, Rels3, Rels4, Rels5, Rels6, Rels7}];

```

We compare GB, the Groebner basis for $I(K_6, K_4)$, to the relations enumerated above, Rels, and find that they

are the same.

```
(* G is a complete graph on 6 vertices *)
G = {1 → 2, 2 → 1, 1 → 3, 3 → 1, 1 → 4, 4 → 1, 1 → 5, 5 → 1, 1 → 6,
      6 → 1, 2 → 3, 3 → 2, 2 → 4, 4 → 2, 2 → 5, 5 → 2, 2 → 6, 6 → 2, 3 → 4, 4 → 3,
      3 → 5, 5 → 3, 3 → 6, 6 → 3, 4 → 5, 5 → 4, 4 → 6, 6 → 4, 5 → 6, 6 → 5};
(* FindGB calculates the Groebner basis for G and stores it in GB *)
FindGB[G, 6, 4, 5]

(* Checks that GB and Rels have the same number of elements *)
Length[GB] == Length[Rels]

True

330

(* Checks that each element of Rels is in GB,
and since they have the same number of elements they are equal as sets *)
GBMemQ[rel_] := MemberQ[GB, rel];
AllTrue[Rels, GBMemQ]

True
```

This completes the proof of the CLAIM.

For completeness, we list GB below.

GB

```
{e11 ** e12, e12 ** e11, e11 ** e13, e13 ** e11, ... 323 ...,
 2 - 2 e31 - 2 e32 - e33 - 2 e51 - e52 - 2 e53 - e61 - e62 - e63 + e31 ** e52 + 2 e31 ** e53 +
 e31 ** e62 + e31 ** e63 + 2 e32 ** e51 + 2 e32 ** e53 + e32 ** e61 + e32 ** e63 + e33 ** e51 +
 e61 ** e53 + e62 ** e51 + e62 ** e53 + e63 ** e51 - e31 ** e62 ** e51 - e31 ** e62 ** e53 -
 e31 ** e63 ** e51 - e32 ** e61 ** e53 - e32 ** e63 ** e51 + e33 ** e61 ** e52,
 2 - 2 e41 - 2 e42 - e43 - 2 e51 - e52 - 2 e53 - e61 - e62 - e63 + e41 ** e52 + 2 e41 ** e53 +
 e41 ** e62 + e41 ** e63 + 2 e42 ** e51 + 2 e42 ** e53 + e42 ** e61 + e42 ** e63 + e43 ** e51 +
 e61 ** e53 + e62 ** e51 + e62 ** e53 + e63 ** e51 - e41 ** e62 ** e51 - e41 ** e62 ** e53 -
 e41 ** e63 ** e51 - e42 ** e61 ** e53 - e42 ** e63 ** e51 + e43 ** e61 ** e52,
 1 - e51 - e52 - e53 - e61 - e62 - e63 + e51 ** e62 + e51 ** e63 + e52 ** e61 + e52 ** e63 +
 e61 ** e53 + e62 ** e51 + e62 ** e53 + e63 ** e51 - e51 ** e62 ** e51 - e51 ** e62 ** e53 -
 e51 ** e63 ** e51 - e52 ** e61 ** e53 - e52 ** e63 ** e51 + e53 ** e61 ** e52}
```

large output

show less

show more

show all

set size limit...

We next verify the two reductions that we claimed in the

HMPSarXiv2017 proof that result in relations of types 6 and 7.

```

SetNonCommutative[ev1, ev2, ev3, ev4, ew1, ew2, ew3, ew4]
SetMonomialOrder[{ev1, ev2, ev3, ev4, ew1, ew2, ew3, ew4}]

rels =
{ev1 ** ev2, ev2 ** ev1, ev1 ** ev3, ev3 ** ev1, ev1 ** ev4, ev4 ** ev1,
 ev2 ** ev3, ev3 ** ev2, ev2 ** ev4, ev4 ** ev2, ev3 ** ev4, ev4 ** ev3,
 ew1 ** ew2, ew2 ** ew1, ew1 ** ew3, ew3 ** ew1, ew1 ** ew4, ew4 ** ew1,
 ew2 ** ew3, ew3 ** ew2, ew2 ** ew4, ew4 ** ew2, ew3 ** ew4, ew4 ** ew3,
 -ev1 + ev1 ** ev1, -ev2 + ev2 ** ev2, -ev3 + ev3 ** ev3,
 -ew1 + ew1 ** ew1, -ew2 + ew2 ** ew2, -ew3 + ew3 ** ew3,
 -1 + ev1 + ev2 + ev3 + ev4, -1 + ew1 + ew2 + ew3 + ew4,
 ev1 ** ew1, ew1 ** ev1, ev2 ** ew2, ew2 ** ev2, ev3 ** ew3, ew3 ** ev3,
 ev3 ** ew2 + ev3 ** ew1 + ev2 ** ew3 + ev2 ** ew1 + ev1 ** ew3 + ev1 ** ew2 -
 ew3 - ew2 - ew1 - ev3 - ev2 - ev1 + 1, ew3 ** ev2 + ew3 ** ev1 + ew2 ** ev3 +
 ew2 ** ev1 + ew1 ** ev3 + ew1 ** ev2 - ev3 - ev2 - ev1 - ew3 - ew2 - ew1 + 1};

Reduc[
(ev3 ** ew2 + ev3 ** ew1 + ev2 ** ew3 + ev2 ** ew1 + ev1 ** ew3 + ev1 ** ew2 - ew3 - ew2 - ew1 -
 ev3 - ev2 - ev1 + 1) ** ev2 - ev3 ** ew2 ** ev2, PolyToRule[rels]]
1 - ev1 - ev2 - ev3 - ew1 - ew2 - ew3 + ev1 ** ew2 + ev1 ** ew3 + ev2 ** ew1 + ev2 ** ew3 +
 ew1 ** ev3 + ew2 ** ev1 + ew2 ** ev3 + ew3 ** ev1 - ev1 ** ew2 ** ev1 - ev1 ** ew2 ** ev3 -
 ev1 ** ew3 ** ev1 - ev2 ** ew1 ** ev3 - ev2 ** ew3 ** ev1 + ev3 ** ew1 ** ev2

```

```
SetNonCommutative[ev1, ev2, ev3, ev4, ew1, ew2, ew3, ew4, ex1, ex2, ex3, ex4]
SetMonomialOrder[{ev1, ev2, ev3, ev4, ew1, ew2, ew3, ew4, ex1, ex2, ex3, ex4}]
```

```
rels =
{ev1 ** ev2, ev2 ** ev1, ev1 ** ev3, ev3 ** ev1, ev1 ** ev4, ev4 ** ev1,
ev2 ** ev3, ev3 ** ev2, ev2 ** ev4, ev4 ** ev2, ev3 ** ev4, ev4 ** ev3,
ew1 ** ew2, ew2 ** ew1, ew1 ** ew3, ew3 ** ew1, ew1 ** ew4, ew4 ** ew1,
ew2 ** ew3, ew3 ** ew2, ew2 ** ew4, ew4 ** ew2, ew3 ** ew4, ew4 ** ew3,
ex1 ** ex2, ex2 ** ex1, ex1 ** ex3, ex3 ** ex1, ex1 ** ex4, ex4 ** ex1,
ex2 ** ex3, ex3 ** ex2, ex2 ** ex4, ex4 ** ex2, ex3 ** ex4, ex4 ** ex3,
-ev1 + ev1 ** ev1, -ev2 + ev2 ** ev2, -ev3 + ev3 ** ev3, -ew1 + ew1 ** ew1, -ew2 +
ew2 ** ew2, -ew3 + ew3 ** ew3, -ex1 + ex1 ** ex1, -ex2 + ex2 ** ex2, -ex3 + ex3 ** ex3,
-1 + ev1 + ev2 + ev3 + ev4, -1 + ew1 + ew2 + ew3 + ew4, -1 + ex1 + ex2 + ex3 + ex4,
ev1 ** ew1, ew1 ** ev1, ev2 ** ew2, ew2 ** ev2, ev3 ** ew3, ew3 ** ev3,
ev1 ** ex1, ex1 ** ev1, ev2 ** ex2, ex2 ** ev2, ev3 ** ex3, ex3 ** ev3,
ex1 ** ew1, ew1 ** ex1, ex2 ** ew2, ew2 ** ex2, ex3 ** ew3, ew3 ** ex3,
ev3 ** ew2 + ev3 ** ew1 + ev2 ** ew3 + ev2 ** ew1 + ev1 ** ew3 + ev1 ** ew2 -
ew3 - ew2 - ew1 - ev3 - ev2 - ev1 + 1, ew3 ** ev2 + ew3 ** ev1 + ew2 ** ev3 +
ew2 ** ev1 + ew1 ** ev3 + ew1 ** ev2 - ev3 - ev2 - ev1 - ew3 - ew2 - ew1 + 1,
ev3 ** ex2 + ev3 ** ex1 + ev2 ** ex3 + ev2 ** ex1 + ev1 ** ex3 + ev1 ** ex2 -
ex3 - ex2 - ex1 - ev3 - ev2 - ev1 + 1, ex3 ** ev2 + ex3 ** ev1 + ex2 ** ev3 +
ex2 ** ev1 + ex1 ** ev3 + ex1 ** ev2 - ev3 - ev2 - ev1 - ex3 - ex2 - ex1 + 1,
ex3 ** ew2 + ex3 ** ew1 + ex2 ** ew3 + ex2 ** ew1 + ex1 ** ew3 + ex1 ** ew2 -
ew3 - ew2 - ew1 - ex3 - ex2 - ex1 + 1, ew3 ** ex2 + ew3 ** ex1 + ew2 ** ex3 +
ew2 ** ex1 + ew1 ** ex3 + ew1 ** ex2 - ex3 - ex2 - ex1 - ew3 - ew2 - ew1 + 1};
```

```
Reduc[
(ev3 ** ew2 + ev3 ** ew1 + ev2 ** ew3 + ev2 ** ew1 + ev1 ** ew3 + ev1 ** ew2 - ew3 - ew2 - ew1 -
ev3 - ev2 - ev1 + 1) ** ex2 - ev3 ** ew2 ** ex2, PolyToRule[rels]]
```

```
2 - 2 ev1 - 2 ev2 - ev3 - ew1 - ew2 - ew3 - 2 ex1 - ex2 - 2 ex3 + ev1 ** ew2 + ev1 ** ew3 +
ev1 ** ex2 + 2 ev1 ** ex3 + ev2 ** ew1 + ev2 ** ew3 + 2 ev2 ** ex1 + 2 ev2 ** ex3 +
ev3 ** ex1 + ew1 ** ex3 + ew2 ** ex1 + ew2 ** ex3 + ew3 ** ex1 - ev1 ** ew2 ** ex1 -
ev1 ** ew2 ** ex3 - ev1 ** ew3 ** ex1 - ev2 ** ew1 ** ex3 - ev2 ** ew3 ** ex1 + ev3 ** ew1 ** ex2
```