In[17]:=
```
SetDirectory[NotebookDirectory[]];
<< "OperatorGB.m"
```

```
Package OperatorGB version 1.4.2
Copyright 2019, Institute of Mathematics, University of Kassel
by Clemens Hofstadler, clemens.hofstadler@mathematik.uni-kassel.de
```

Computations included in the paper "*Signature Gröbner bases, bases of syzygies and cofactor reconstruction in the free algebra*" by Clemens Hofstadler and Thibaut Verron.

Computations of Example 31

Computations of Example 31 over the coefficient ring ℚ.

In[19]:=
```
(* Defining the generators and the monomial ordering as in the paper *)
f1 = x ** y ** x - x ** y;
f2 = y ** x ** y;
f3 = x ** y ** y - x ** x ** y;
f4 = x ** x ** y;
SetUpRing[{x, y}]
```

In[24]:=
```
(* First, we compute a partial signature
 Gröbner basis w.r.t. the generators f1,f2,f3 *)
{G1, H1} = SigGB[{f1, f2, f3}, 20];
```

In[25]:=
```
(* Then, we recover a partial labelled Gröbner basis *)
{G, H} = SyzygyRecovery[G1, H1, {f1, f2, f3}, IsFullBasis → False];
```

In[26]:=
```
(* We already start to see the infinite
 structure starting with the 5th element *)
G
```

Out[26]=
```
{{-x ** y + x ** y ** x, e1}, {y ** x ** y, e2}, {-x ** x ** y + x ** y ** y, e3},
 {x ** x ** y, -e3 - e1 ** y + x ** e2}, {y ** x ** x ** y, e2 ** y - y ** e3},
 {y ** x ** x ** x ** y, e2 ** y ** y - y ** e3 ** y - y ** x ** e3}, {y ** x ** x ** x ** x ** y,
  e2 ** y ** y ** y - y ** e3 ** y ** y - y ** x ** e3 ** y - y ** x ** x ** e3},
 {y ** x ** x ** x ** x ** x ** y, e2 ** y ** y ** y ** y - y ** e3 ** y ** y ** y -
   y ** x ** e3 ** y ** y - y ** x ** x ** e3 ** y - y ** x ** x ** x ** e3},
 {y ** x ** x ** x ** x ** x ** x ** y, e2 ** y ** y ** y ** y ** y -
   y ** e3 ** y ** y ** y ** y - y ** x ** e3 ** y ** y ** y -
   y ** x ** x ** e3 ** y ** y ** y - y ** x ** x ** x ** e3 ** y ** y -
   y ** x ** x ** x ** x ** e3 ** y - y ** x ** x ** x ** x ** x ** e3},
 {y ** x ** x ** x ** x ** x ** x ** x ** y, e2 ** y ** y ** y ** y ** y ** y -
   y ** e3 ** y ** y ** y ** y ** y - y ** x ** e3 ** y ** y ** y ** y -
   y ** x ** x ** e3 ** y ** y ** y - y ** x ** x ** x ** x ** e3 ** y ** y -
   y ** x ** x ** x ** x ** x ** e3 ** y - y ** x ** x ** x ** x ** x ** x ** e3}}
```

In[27]:=
```
(* Now, we add the fourth generator... *)
{G1, H1} = SigGB[{f1, f2, f3, f4}, 20];
(* and recover the labelled Gröbner basis *)
{G, H} = SyzygyRecovery[G1, H1, {f1, f2, f3, f4}, IsFullBasis → True];
```

In[29]:=
```
(* In this case, G is finite containing only the five claimed elements *)
G
```

Out[29]=
```
{{-x ** y + x ** y ** x, e1}, {y ** x ** y, e2}, {-x ** x ** y + x ** y ** y, e3},
 {x ** x ** y, e4}, {y ** x ** x ** y, e2 ** y - y ** e3}}
```

Computations of Section 6

To do the computations of Section 6 including the counting of (zero) reductions, we first have to extend the functions SigGB and Groebner.

Extension of SigGB to count (zero) reductions

Extension of Groebner to count (zero) reductions

Actual computations

In[∘]:=
```
(* Define all benchmark examples as in the paper *)
braid3 = {y ** x ** y - z ** y ** z, x ** y ** x - z ** x ** y,
   z ** x ** z - y ** z ** x, x ** x ** x + y ** y ** y + z ** z ** z + x ** y ** z};
lp1 = {z ** z ** z ** z + y ** x ** y ** x - x ** y ** y ** x - 3 * z ** y ** x ** z,
   x ** x ** x + y ** x ** y - x ** y ** x, z ** y ** x - x ** y ** z + z ** x ** z};
lv2 = {x ** y + y ** z, x ** x + x ** y - y ** x - y ** y};
tri1 = {x ** x ** x - 1, y ** y - 1, (y ** x ** y ** x ** y ** x ** x ** y ** x ** x) **
     (y ** x ** y ** x ** y ** x ** x ** y ** x ** x) - 1};
tri3 = {x ** x ** x - 1, y ** y ** y - 1, (y ** x ** y ** x ** x) ** (y ** x ** y ** x ** x) - 1};
```

In[∘]:=
```
(* Define a monomial ordering as in the paper *)
SetUpRing[{x, y, z}]
```

SigGB

```
reductions = zeroReductions = 0;
G = ExtendedSigGB[braid3, 100 000, MaxDeg → 10];
reductions
zeroReductions
```

Out[∘]= 1053

Out[∘]= 40

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedSigGB[lp1, 100 000, MaxDeg → 11];
reductions
zeroReductions
```

Out[∘]= 155

Out[∘]= 0

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedSigGB[lv2, 100 000, MaxDeg → 100];
reductions
zeroReductions
```
Out[∘]= 201

Out[∘]= 0

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedSigGB[tri1, 100 000];
reductions
zeroReductions
```
Out[∘]= 335

Out[∘]= 164

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedSigGB[tri3, 100 000];
reductions
zeroReductions
```
Out[∘]= 252

Out[∘]= 136

BB vanilla

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, braid3, 100, MaxDeg → 10, Criterion → False];
reductions
zeroReductions
```
Out[∘]= 1154

Out[∘]= 661

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, lp1, 100, MaxDeg → 11, Criterion → False];
reductions
zeroReductions
```
Out[∘]= 205

Out[∘]= 130

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, lv2, 100, MaxDeg → 100, Criterion → False];
reductions
zeroReductions
```
Out[∘]= 9702

Out[∘]= 4990

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, tri1, 100, Criterion → False];
reductions
zeroReductions
```
Out[∘]= 9435

Out[∘]= 8897

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, tri3, 100, Criterion → False];
reductions
zeroReductions
```
Out[∘]= 2705

Out[∘]= 2573

BB optimized

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, braid3, 100, MaxDeg → 10];
reductions
zeroReductions
```
Out[∘]= 1121

Out[∘]= 634

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, lp1, 100, MaxDeg → 11];
reductions
zeroReductions
```
Out[∘]= 198

Out[∘]= 125

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, lv2, 100, MaxDeg → 100];
reductions
zeroReductions
```
Out[∘]= 9702

Out[∘]= 4990

In[∘]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, tri1, 100];
reductions
zeroReductions
```
Out[∘]= 3480

Out[∘]= 3288

In[ ]:=
```
reductions = zeroReductions = 0;
G = ExtendedGroebner[cofactors, tri3, 100];
reductions
zeroReductions
```

Out[ ]= 1060

Out[ ]= 979