Solving linearized 5-gon in emergent $\mathcal{P}$

```
SetDirectory["C:/Users/kunoy/Dropbox/MyNotes/Research/Mathematica/WithDBN"];
(* SetDirectory["/Users/kunohome/Dropbox/MyNotes/Research/Mathematica/WithDBN"];*)
<< FreeLie.m
```

```
FreeLie` implements / extends
 {*, +, **, $SeriesShowDegree, ⟨⟩, ∫, ≡, ad, Ad, adSeries, AllCyclicWords, AllLyndonWords,
  AllWords, Arbitrator, AS, ASeries, AW, b, BCH, BooleanSequence, BracketForm, BS, CC, Crop,
  cw, CW, CWS, CWSeries, D, Deg, DegreeScale, DerivationSeries, div, DK, DKS, DKSeries, EulerE,
  Exp, Inverse, j, J, JA, LieDerivation, LieMorphism, LieSeries, LS, LW, LyndonFactorization,
  Morphism, New, RandomCWSeries, Randomizer, RandomLieSeries, RC, SeriesSolve, Support,
  t, tb, TopBracketForm, tr, UndeterminedCoefficients, αMap, Γ, ι, Λ, σ, ℏ, ⌐, ⌢}.
```

```
FreeLie` is in the public domain. Dror Bar-Natan is committed
  to support it within reason until July 15, 2022. This is version 150814.
```

Preparation from algebra

1. Computing basis of the kernel of a linear map

```
BasisKer[V1_, f_, V2_] := Module[{$l, $N},
  $l = Length[V1];
  $N = NullSpace[
    Table[
      Coefficient[f[V1⟦i⟧], V2],
      {i, $l}
     ] // Transpose
   ];
  Table[
   Expand@Sum[λ⟦i⟧ V1⟦i⟧, {i, $l}],
   {λ, $N}
  ]
 ]
```

2. Partial derivation (borrowed from DBN’s “FreeLie”)

```
(* not contained in the package? *)
```

```
τ[y_LW, w_LW] /; Deg[y] == 1 := τ[y, w] = Which[
    y === w, AW[],
    Deg[w] === 1, 0,
    True, Module[{w1, w2},
     {w1, w2} = LyndonFactorization[w];
     ι[w1] ** τ[y, w2] - ι[w2] ** τ[y, w1]
    ]
   ];
τ[y_, ls_LieSeries] := τ[y, ls] = New[ASeries[as],
    as[d_] := as[d] = τ[LW[y], ls[d + 1]]
   ];
τ[y_, expr_] := Expand[expr /. w_LW :→ τ[LW[y], w]];
```

3. Basis of the free associative algebra

```
AssBasis[n_] := AW @@@ Tuples[{x, y}, n]
```

4. Powers in free associative algebras

```
AssPowers[1, u_] := u;
AssPowers[n_, u_] := u ** AssPowers[n - 1, u];
```

```
AssPowers[2, AW[x] - AW[y]]
```

```
AW[x, x] - AW[x, y] - AW[y, x] + AW[y, y]
```

5. The antipode

```
antipode[w_AW] := (-1)^Length[w] Reverse[w];
antipode[expr_] := Expand[expr /. w_AW :→ antipode[w]]
```

6. Morphism in the free associative algebra

```
AssMorphism_rules_[u_AW] := (LieMorphism[rules] @u) @Length[u]
AssMorphism_rules_[expr_] := Expand[expr /. w_AW :→ AssMorphism_rules[w]]
```

```
AssMorphism_{LW[x]:→LW[x]+2 LW[y]}[AW[x, x] + AW[x, y, z]]
```

```
AW[x, x] + 2 AW[x, y] + 2 AW[y, x] + 4 AW[y, y] + AW[x, y, z] + 2 AW[y, y, z]
```

The map R

1. The map R: FL --> FA (symmetric - horizontal)

```
R_A_ [0] = 0;
R_A_ [u_LW] := R_A[u] = Which[
    Deg[u] === 1, 0,
    True, Module[{w1, w2},
      {w1, w2} = LyndonFactorization[u];
      Expand[b[ι[w1], R_A[w2]] + b[R_A[w1], ι[w2]]
        + Sum[(1 / 2) * (τ[LW[α], w2] ** AW[α] ** antipode[τ[LW[α], w1]] -
            τ[LW[α], w1] ** AW[α] ** antipode[τ[LW[α], w2]]),
          {α, A}]
      ]
    ]
  ];
R_A_ [expr_] := Expand[expr /. LW[seq__] ⧴ R_A[LW[seq]]];
```

```
R_{x,y}[LW[x, x, x, y]]
```

−AW[x, x, y] + AW[y, x, x]

```
R_{x,y,z}[b[b[LW[x], LW[y]], b[LW[x], LW[z]]]]
```

$\frac{1}{2}$ AW[y, x, z] − $\frac{1}{2}$ AW[z, x, y]

2. |R(x, -x-y)y + R(y, -x-y)x|

```
S[expr_] := tr[AssMorphism_{LW[x]⧴LW[x],LW[y]⧴-LW[x]-LW[y]}[expr] ** AW[y] +
    AssMorphism_{LW[x]⧴LW[y],LW[y]⧴-LW[x]-LW[y]}[expr] ** AW[x]]
```

```
R_{x,y}[LW[x, x, x, x, y]]
S[R_{x,y}[LW[x, x, x, x, y]]]
```

−$\frac{3}{2}$ AW[x, x, x, y] + AW[x, x, y, x] + AW[x, y, x, x] − $\frac{3}{2}$ AW[y, x, x, x]

$\widehat{xxxxy} + 3\,\widehat{xxxyy} - 2\,\widehat{xxyxy} + 3\,\widehat{xxyyy} - 2\,\widehat{xyxyy} + \widehat{xyyyy}$

R(x,y) + R(y,x)

```
Rxyyx[expr_] := Expand[R_{x,y}[expr] + AssMorphism_{LW[x]⧴LW[y],LW[y]⧴LW[x]}[R_{x,y}[expr]]];
```

```
Rxyyx[LW[x, z, x, y]]
```

−$\frac{1}{2}$ AW[x, y, z] − $\frac{1}{2}$ AW[y, x, z] + $\frac{1}{2}$ AW[z, x, y] + $\frac{1}{2}$ AW[z, y, x]

Equations

1. linearized pentagon eq. in emergent $\mathcal{P}$

```
P3[ψ_LW] := Module[{$d, $pd},
    $d = Deg[ψ];
    $pd = τ[LW[y], ψ];
    Expand[
     $pd + (LieMorphism[{LW[x] :> LW[y], LW[y] :> 0}]@$pd) @ ($d - 1) -
       (LieMorphism[{LW[x] :> LW[x] + LW[y], LW[y] :> 0}]@$pd) @ ($d - 1) - 2 R_{x,y}[ψ]
    ]
   ];
P3[expr_] := Expand[expr /. ψ_LW :> P3[ψ]];
```

2. Solutions to twist equations

```
(* ψ(x,y)+ψ(y,x)=0 *)
```

```
TwistSol[n_] := Module[{$al, $altw, $Ntest},
  $al = AllLyndonWords[n, {LW[x], LW[y]}];
  $altw = $al + Table[
     (LieMorphism[{LW[x] :> LW[y], LW[y] :> LW[x]}]@$al[[i]])@n, {i, Length[$al]}];
  $Ntest = NullSpace[
    Table[
      Coefficient[$altw[[i]], $al], {i, Length[$al]}
     ] // Transpose
   ];
  Table[
   Expand@Sum[λ[[i]] $al[[i]], {i, Length[$al]}],
   {λ, $Ntest}]
 ]
```

3. Solutions to KV1

```
ATν[ψ_] := b[LW[x], LieMorphism[{LW[x] :> -LW[x] - LW[y], LW[y] :> LW[x]}]@ψ] +
    b[LW[y], LieMorphism[{LW[x] :> -LW[x] - LW[y], LW[y] :> LW[y]}]@ψ];
```

```
Table[ATν[ψ]@5, {ψ, AllLyndonWords[4, {LW[x], LW[y]}]}]
```

$$\{[x,[x,[x,[x,y]]]] - [x,[x,[[x,y],y]]] - [[x,[x,y]],[x,y]] - [x,[[[x,y],y],y]] - 2\,[[x,y],[[x,y],y]] + [[[[x,y],y],y],y],$$
$$[x,[x,[x,[x,y]]]] - [x,[x,[[x,y],y]]] - [x,[[[x,y],y],y]] - [[x,y],[[x,y],y]] + [[[[x,y],y],y],y],\ [x,[x,[x,[x,y]]]] + [[[[x,y],y],y],y]\}$$

```
SolKV1[n_] := Module[{$alln, $allnplus, $im⋎, $Ntest},
  $alln = AllLyndonWords[n, {LW[x], LW[y]}];
  $allnplus = AllLyndonWords[n + 1, {LW[x], LW[y]}];
  $im⋎ = Table[AT⋎[$alln⟦i⟧]@(n + 1), {i, Length[$alln]}];
  $Ntest = NullSpace[
    Table[
      Coefficient[$im⋎⟦i⟧, $allnplus], {i, Length[$alln]}
     ] // Transpose
   ];
  Table[
   Expand@Sum[λ⟦i⟧ $alln⟦i⟧, {i, Length[$alln]}],
   {λ, $Ntest}]
 ]
```

```
TwistSol[4]
```

{x x xy + xyy y, x xyy}

4. Twist and KV1

```
TwistKV1Sol[n_] := Module[{$alln, $allnplus, $alltwkv1, $Ntest},
  $alln = AllLyndonWords[n, {LW[x], LW[y]}];
  $allnplus = AllLyndonWords[n + 1, {LW[x], LW[y]}];
  $alltwkv1 = $alln + Table[
      (LieMorphism[{LW[x] ⧴ LW[y], LW[y] ⧴ LW[x]}]@$alln⟦i⟧)@n, {i, Length[$alln]}] +
    Table[AT⋎[$alln⟦i⟧]@(n + 1), {i, Length[$alln]}];
  $Ntest = NullSpace[
    Table[
      Coefficient[$alltwkv1⟦i⟧, Join[$alln, $allnplus]], {i, Length[$alln]}
     ] // Transpose
   ];
  Table[
   Expand@Sum[λ⟦i⟧ $alln⟦i⟧, {i, Length[$alln]}],
   {λ, $Ntest}]
 ]
```

```
TwistSol[6]
TwistKV1Sol[6]
```

{x x x x xy + xyy yyy, x x xy xy + xy xyy y, x x x xyy − 2 x x xy xy + x xyy yy, −2 x x xyy y + 3 x xyy xy, 2 x x xyy y + 3 x xy xyy}

{}

```
Table[Length@TwistKV1Sol[k], {k, 1, 11}]
```

{1, 0, 1, 0, 1, 0, 2, 1, 4, 2, 9}

5. Duflo function

```
Duflo[n_] := tr[AssPowers[n, AW[x]] + AssPowers[n, AW[y]] - AssPowers[n, AW[x] + AW[y]]]
```

```
Duflo[5]
```

$-5\,\overline{xxxxy} - 5\,\overline{xxxyy} - 5\,\overline{xxyxy} - 5\,\overline{xxyyy} - 5\,\overline{xyxyy} - 5\,\overline{xyyyy}$

```
Timing[TwistSol[5]]
```

{0.001328, {-x x x xy + xy y yy, -x xy xy + xy xy y, -x x xy y + x xy xy + x xy y y}}

Computations

1. Checking if twist eq implies |R(x, -x-y)y + R(y, -x-y)x| = Duflo.

```
k = 2;
test = TwistSol[k];
l = Length[test];
Do[
 Print[
  Part[test, i], " and ",
  S[R{x,y}@Part[test, i]]
 ],
 {i, l}
]
```

$\overline{xy}$ and 0

```
k = 5;
test = TwistSol[k];
l = Length[test];
Do[
 Print[
  Part[test, i], " and ",
  S[R{x,y}@Part[test, i]]
 ],
 {i, l}
]
```

-x x x xy + xy y yy and $\overline{xxxxy} - 5\,\overline{xxxyy} + 10\,\overline{xxyxy} - 5\,\overline{xxyyy} + 10\,\overline{xyxyy} + \overline{xyyyy}$

-x xy xy + xy xy y and $-\overline{xxxxy} + 5\,\overline{xxxyy} - 10\,\overline{xxyxy} + 5\,\overline{xxyyy} - 10\,\overline{xyxyy} - \overline{xyyyy}$

-x x xy y + x xy xy + x xy y y and $\overline{xxxxy} - 5\,\overline{xxxyy} + 10\,\overline{xxyxy} - 5\,\overline{xxyyy} + 10\,\overline{xyxyy} + \overline{xyyyy}$

```
Duflo[5]
```

$-5\,\widehat{xxxxy} - 5\,\widehat{xxxyy} - 5\,\widehat{xxyxy} - 5\,\widehat{xxyyy} - 5\,\widehat{xyxyy} - 5\,\widehat{xyyyy}$

```
k = 6;
test = TwistSol[k];
l = Length[test];
Do[
 Print[
  Part[test, i], " and ",
  𝒮[R{x,y}@Part[test, i]]
 ],
 {i, l}
]
```

```
x x x x xy + xy y yyy and 0
x x xy xy + xy xyy y and 0
x x x xyy − 2 x x xy xy + x xyy yy and 0
−2 x x xyy y + 3 x xyy xy and 0
2 x x xyy y + 3 x xy xyy and 0
```

2. Checking if twist eq. + KV1 implies |R(x, -x-y)y + R(y, -x-y)x| = Duflo.

```
(* The result is negative. *)
```

```
k = 5;
test = TwistKV1Sol[k];
l = Length[test];
Do[
 Print[
  Part[test, i], " and ",
  𝒮[R{x,y}@Part[test, i]]
 ],
 {i, l}
]
```

```
−2 x x x xy + 4 x x xyy − 3 x xy xy − 4 x xyy y − xy xyy +
 2 xyy yy and −xxxxy + 5 xxxyy − 10 xxyxy + 5 xxyyy − 10 xyxyy − xyyyy
```

3. Solving linearized pentagon in emergent $\mathcal{P}$

```
Do[
 Print[k, " and ",
  Timing[Length@BasisKer[AllLyndonWords[k, {LW[x], LW[y]}], P3, AssBasis[k - 1]]]],
 {k, 1, 14}]
```

```
1 and {0.000262, 1}
2 and {0.000099, 1}
3 and {0.000258, 1}
4 and {0.000355, 0}
5 and {0.001217, 1}
6 and {0.002399, 0}
7 and {0.012715, 1}
8 and {0.04713, 1}
9 and {0.258667, 1}
10 and {2.24417, 1}
11 and {9.39412, 2}
12 and {50.9139, 2}
13 and {348.902, 3}
14 and {2151.73, 3}
```

```
k = 3;
test = BasisKer[AllLyndonWords[k, {LW[x], LW[y]}], P3, AssBasis[k - 1]]
Duflo[k]
Table[𝓢[R_{x,y}[test[[i]]]], {i, Length[test]}]
```

$\{[x,[x,y]]\}$

$-3\,\widehat{xxy} - 3\,\widehat{xyy}$

$\{2\,\widehat{xxy} + 2\,\widehat{xyy}\}$

```
k = 5;
test = BasisKer[AllLyndonWords[k, {LW[x], LW[y]}], P3, AssBasis[k - 1]]
Duflo[k]
Table[𝓢[R_{x,y}[test[[i]]]], {i, Length[test]}]
```

$\{-2\,[x,[x,[x,[x,y]]]] + 2\,[x,[x,[[x,y],y]]] + [[x,[x,y]],[x,y]] - 2\,[x,[[[x,y],y],y]] + 2\,[[x,y],[[x,y],y]]\}$

$-5\,\widehat{xxxxy} - 5\,\widehat{xxxyy} - 5\,\widehat{xxyxy} - 5\,\widehat{xxyyy} - 5\,\widehat{xyxyy} - 5\,\widehat{xyyyy}$

$\{-6\,\widehat{xxxxy} - 6\,\widehat{xxxyy} - 6\,\widehat{xxyxy} - 6\,\widehat{xxyyy} - 6\,\widehat{xyxyy} - 6\,\widehat{xyyyy}\}$