

Pensieve Header: There is no Conway Blobs quotient. Why did I have to wait for the run-time error in the line “Diag[hs /. h → (Length[ind1[[i]]*h)]” within PutOn to figure that out?

For the ArrowRules, see testing at “AlexanderBlobs-U(12D) Comparison.nb”.

```

ArrowRules = {
  Diag[hs_, lft___, ar[i_, j_], ar[k_, l_], rgt___] /;
    ! OrderedQ[{ar[i, j], ar[k, l]}] ∴ Plus[
  Diag[hs, lft, ar[k, l], ar[i, j], rgt],
  Which[
    i == k || i == j || k == l, 0,
    j == 1, Diag[h*hs, lft, ar[k, l], rgt] - Diag[h*hs, lft, ar[i, j], rgt],
    j == k && i == 1, (
      -Diag[h*hs, lft, ar[j, i], rgt] + Diag[h*hs, lft, ar[i, j], rgt] -
      Diag[h*hs, lft, ar[j, j], rgt] + Diag[h*hs, lft, ar[i, i], rgt]
    ),
    j == k, -Diag[h*hs, lft, ar[k, l], rgt] + Diag[h*hs, lft, ar[i, l], rgt],
    i == 1, -Diag[h*hs, lft, ar[k, j], rgt] + Diag[h*hs, lft, ar[i, j], rgt],
    True, 0
  ]
]
];

If[Head[$DegreeStack] != List, $DegreeStack = {Infinity}];
$ModDegree = First[$DegreeStack];
SetAttributes[ModDegree, HoldRest];
ModDegree[m_, expr_] := Module[{res},
  PrependTo[$DegreeStack, $ModDegree = m];
  res = expr;
  $DegreeStack = Rest[$DegreeStack];
  $ModDegree = First[$DegreeStack];
  res
];

Deg[Diag[hs_, ars___]] := Length[{ars}] + Exponent[hs, h];
Deg[Diag[h, up[2, 2], ar[2, 3]]]
3

Unprotect[NonCommutativeMultiply];
0**_ = 0;
_**0 = 0;
(c_?(FreeQ[#, Diag] &) * a_) ** b_ := Expand[c * (a ** b)];
a_ ** (c_?(FreeQ[#, Diag] &) * b_) := Expand[c * (a ** b)];
a_Plus ** b_ := (# ** b) & /@ a;
a_ ** b_Plus := (a ** #) & /@ b;
d1_Diag ** d2_Diag /; Deg[d1] + Deg[d2] ≥ $ModDegree := 0;
Diag[hs1_, ars1___] ** Diag[hs2_, ars2___] := FixedPoint[
  Expand[# /. ArrowRules] &,
  Diag[hs1*hs2, ars1, ars2]
];
b[x_, y_] := x**y - y**x;

```

```

(* Diag[hs1_, ars1____] ** Diag[hs2_, ars2____] := Module[
  {res},
  res=FixedPoint[
    Expand[# /. ArrowRules]&,
    Diag[hs1*hs2, ars1, ars2]
  ];
  If[OrderedQ[{ars1}] && OrderedQ[{ars2}], Diag[hs1, ars1]**Diag[hs2,ars2]=res,res]
]; *)

DPower[expr_, p_Integer] /; p > 0 := NonCommutativeMultiply @@ Table[expr, {p}];
DExp[expr_] := Module[
  {total, term, k},
  k = 0;
  total = term = Diag[1];
  While[term != 0,
    ++k;
    total += (term = Expand[term**expr / k])
  ];
  total
];
DInvert[Diag[1] + expr_] := Module[
  {total, term},
  total = term = Diag[1];
  While[term != 0,
    total += (term = Expand[-term**expr])
  ];
  total
];

r[i_, j_] := Diag[1, ar[i, j]];
r12 = r[1, 2]; r21 = r[2, 1];
R[i_, j_] := DExp[r[i, j]];

b[r[1, 2], r[1, 3]] + b[r[1, 2], r[2, 3]]
Diag[h, ar[1, 3]] - Diag[h, ar[2, 3]]
b[r[1, 2], r[1, 3]] + b[r[1, 2], r[2, 3]] + b[r[1, 3], r[2, 3]]
0

ModDegree[7, DExp[r[1, 2]]]

Diag[1] + Diag[1, ar[1, 2]] +  $\frac{1}{2}$  Diag[1, ar[1, 2], ar[1, 2]] +
 $\frac{1}{6}$  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]] +
 $\frac{1}{24}$  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]] +
 $\frac{1}{120}$  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]] +
 $\frac{1}{720}$  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]]

```

```

ModDegree[3, DExp[r[1, 2]] ** DExp[r[1, 3]] ** DExp[r[2, 3]]]
Diag[1] + Diag[1, ar[1, 2]] + Diag[1, ar[1, 3]] + Diag[1, ar[2, 3]] +
   $\frac{1}{2}$  Diag[1, ar[1, 2], ar[1, 2]] + Diag[1, ar[1, 2], ar[1, 3]] + Diag[1, ar[1, 2], ar[2, 3]] +
   $\frac{1}{2}$  Diag[1, ar[1, 3], ar[1, 3]] + Diag[1, ar[1, 3], ar[2, 3]] +  $\frac{1}{2}$  Diag[1, ar[2, 3], ar[2, 3]]
t1 = ModDegree[3, DExp[r[2, 3]] ** DExp[r[1, 3]]]
Diag[1] + Diag[1, ar[1, 3]] + Diag[1, ar[2, 3]] + Diag[h, ar[1, 3]] - Diag[h, ar[2, 3]] +
   $\frac{1}{2}$  Diag[1, ar[1, 3], ar[1, 3]] + Diag[1, ar[1, 3], ar[2, 3]] +  $\frac{1}{2}$  Diag[1, ar[2, 3], ar[2, 3]]
ModDegree[3, t1 ** DExp[r[1, 2]]]
Diag[1] + Diag[1, ar[1, 2]] + Diag[1, ar[1, 3]] + Diag[1, ar[2, 3]] +
   $\frac{1}{2}$  Diag[1, ar[1, 2], ar[1, 2]] + Diag[1, ar[1, 2], ar[1, 3]] + Diag[1, ar[1, 2], ar[2, 3]] +
   $\frac{1}{2}$  Diag[1, ar[1, 3], ar[1, 3]] + Diag[1, ar[1, 3], ar[2, 3]] +  $\frac{1}{2}$  Diag[1, ar[2, 3], ar[2, 3]]
ModDegree[7, DExp[r[1, 2]] ** DExp[r[1, 3]] ** DExp[r[2, 3]] -
  DExp[r[2, 3]] ** DExp[r[1, 3]] ** DExp[r[1, 2]]]
0
ModDegree[7, DInvert[Diag[1] + r[1, 2]]]
Diag[1] - Diag[1, ar[1, 2]] + Diag[1, ar[1, 2], ar[1, 2]] -
  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]] + Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]] -
  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]] +
  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]]
Adjoint[Diag[hs_, ars___]] := (-1)^Length[{ars}] * Reverse[Diag[ars, hs]];
Adjoint[expr_] := Expand[expr /. diag_Diag => Adjoint[diag]];
Cap[Diag[hs_]] := Diag[hs];
Cap[Diag[_, _]] := 0;
Cap[expr_] := Expand[expr /. diag_Diag => Cap[diag]];
PutOn[ind___, Diag[hs_, ars___]] := Module[
  {ind1},
  ind1 = Flatten[{#}] & /@ {ind};
  Expand[
    Distribute[Join[
      Diag[hs /. h -> (Length[ind1][[i]] * h)],
      Diag[ars] /. ar[i_, j_] => Total[Outer[ar, ind1[[i]], ind1[[j]]], 2]
    ] /. Diag[c_Integer * hs1_, ars1___] => c * Diag[hs1, ars1] /. ArrowRules
  ]
];
PutOn[ind___, expr_] := Expand[expr /. diag_Diag => PutOn[ind, diag]];
ModDegree[5, DExp[r[1, 2]]]
Diag[1] + Diag[1, ar[1, 2]] +  $\frac{1}{2}$  Diag[1, ar[1, 2], ar[1, 2]] +
   $\frac{1}{6}$  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]] +  $\frac{1}{24}$  Diag[1, ar[1, 2], ar[1, 2], ar[1, 2], ar[1, 2]]

```

```
PutOn[2, 3, Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]]]
```

Part::pspec: Part specification i is neither an integer nor a list of integers. >>

```
Diag[1, ar[2, 3], ar[2, 3], ar[2, 3]]
```

```
PutOn[1, {2, 3}, Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]]]
```

```
Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]] + 3 Diag[1, ar[1, 2], ar[1, 2], ar[1, 3]] +
  3 Diag[1, ar[1, 2], ar[1, 3], ar[1, 3]] + Diag[1, ar[1, 3], ar[1, 3], ar[1, 3]]
```

```
PutOn[{1, 2}, 3, Diag[1, ar[1, 2], ar[1, 2], ar[1, 2]]]
```

```
Diag[h[1]^2, ar[2, 3]] - Diag[h[1] h[2], ar[1, 3]] - Diag[h[1] h[2], ar[2, 3]] +
  Diag[h[2]^2, ar[1, 3]] - 3 Diag[h[1], ar[1, 3], ar[2, 3]] - 3 Diag[h[1], ar[2, 3], ar[2, 3]] +
  3 Diag[h[2], ar[1, 3], ar[1, 3]] + 3 Diag[h[2], ar[1, 3], ar[2, 3]] +
  Diag[1, ar[1, 3], ar[1, 3], ar[1, 3]] + 3 Diag[1, ar[1, 3], ar[1, 3], ar[2, 3]] +
  3 Diag[1, ar[1, 3], ar[2, 3], ar[2, 3]] + Diag[1, ar[2, 3], ar[2, 3], ar[2, 3]]
```

```
R4Eqn[V_] := V ** DExp[r[1, 3] + r[2, 3]] - R[1, 3] ** R[2, 3] ** V;
```

```
TwistEqn[V_] := PutOn[2, 1, V] ** DExp[1/2 (r[1, 2] + r[2, 1])] - R[1, 2] ** V;
```

```
SetAttributes[PullDiags, Listable];
```

```
PullDiags[expr_] /; Head[expr] != List := Module[
  {diags, res},
  diags = Union[Cases[expr, _Diag, Infinity]];
  res = Total[(Coefficient[expr, #] #) & /@ diags];
  res + Expand[expr - res]
]
```

```
PullDiags[
   $\frac{1}{24}$  Diag[h[1]^2, ar[2, 3]] + 2 c21 Diag[h[1]^2, ar[2, 3]] - 2 c23 Diag[h[1]^2, ar[2, 3]]
   $\left(\frac{1}{24} + 2 c21 - 2 c23\right)$  Diag[h[1]^2, ar[2, 3]]
]
```