Some Dimensions of Arrow Diagram Spaces

By Dror Bar-Natan; with some code borrowed from a joint project with Louis Leung, Arrow_Diagrams_and_gl (N).

Pensieve Header: Same as Dimensions.nb, but on a computer with more RAM.

Program

Diagrams

```
SetAttributes[Diag, Orderless];
Place[{ar}, {i_, j_}] := {Diag[ar[i, j]], Diag[ar[j, i]]};
Place[{ar, objs__}, {i_, rest__}] := Flatten[Table[
    Outer[Join,
     Place[{ar}, {i, {rest}[[k]]}],
     Place[{objs}, Delete[{rest}, k]]
    ],
    {k, Length[{rest}]}
   ]];
Diagrams[k_. * ar] := Place[Table[ar, {k}], Range[2 k]];
```

The Circle

```
DiagramRotateLeft[d_Diag] := Module[
   {labels = Union @@ (List @@ Apply[List, d, {1}])},
   d /. Thread[Rule[labels, RotateLeft[labels]]]
  ];
DiagramRotateToMinimal[expr_] := expr /. d_Diag :> Module[
     {bd = d, rd = DiagramRotateLeft[d]},
     While[rd =!= d,
      bd = First[Sort[{bd, rd}]];
      rd = DiagramRotateLeft[rd]];
     bd
    ];
ClosedDiagrams[dir_] := ClosedDiagrams[dir] = Union[DiagramRotateToMinimal[Diagrams[dir]]]
```

Relators

```
Place[{r : (TC | R6T), objs__}, {i_, rest__}] := Flatten[Table[
    Outer[Join,
     Place[{r}, {i, {rest}[[j]], {rest}[[k]]}],
     Place[{objs}, Delete[{rest}, {{j}, {k}}]]
    ],
    {k, 2, Length[{rest}]}, {j, 1, k - 1}
   ]];
```

```
Place[{R6T}, {i_, j_, k_}] :=
  Permutations[{i, j, k}] /. {i1_, j1_, k1_} :> Diag[R6T[i1, j1, k1]];
Diagrams[R6T] := Place[{R6T}, {1, 2, 3}];
Diagrams[R6T + k_. * ar] /; k > 0 := Flatten[
   Place[#, Range[2 k + 3]] & /@ Permutations[Table[ar, {k}] ~ Append ~ R6T]
  ];
Diagrams[R6T + k_. * ar] /; k < 0 := {};

Place[{TC}, {i_, j_, k_}] := Diag /@ {TC[i, j, k], TC[j, k, i], TC[k, i, j]};
Diagrams[TC] := Place[{TC}, {1, 2, 3}];
Diagrams[TC + k_. * ar] /; k > 0 := Flatten[
   Place[#, Range[2 k + 3]] & /@ Permutations[Table[ar, {k}] ~ Append ~ TC]
  ];
Diagrams[TC + k_. * ar] /; k < 0 := {};
```

Framing Independence

```
DiagramsSansFI[specs_] := Select[Diagrams[specs],
   FreeQ[#, ar[i_, j_] /; Abs[i - j] == 1] &
  ];
ClosedDiagramsSansFI[specs_] := Select[ClosedDiagrams[specs],
  FreeQ[#, ar[i_, j_] /; Abs[i - j] == 1] &
 ]
```

Relations

```
NormalizeDiag[diag_Diag] := Module[
   {indices = Union @@ (List @@ diag /. ar → List)},
   diag /. Thread[indices → Range[Length[indices]]]
  ];
R[Diag[lft___, R6T[i_, j_, k_], rgt___]] := (
   + NormalizeDiag[Diag[lft, ar[i, j], ar[i + 0.5, k], rgt]]
    + NormalizeDiag[Diag[lft, ar[i, j], ar[j + 0.5, k], rgt]]
    + NormalizeDiag[Diag[lft, ar[i, k], ar[j, k + 0.5], rgt]]
    - NormalizeDiag[Diag[lft, ar[i, k], ar[i + 0.5, j], rgt]]
    - NormalizeDiag[Diag[lft, ar[i, j + 0.5], ar[j, k], rgt]]
    - NormalizeDiag[Diag[lft, ar[i, k + 0.5], ar[j, k], rgt]]
  );
(* ooops, TC is HC; needs fixing *)
R[Diag[lft___, TC[i_, j_, k_], rgt___]] := (
   + NormalizeDiag[Diag[lft, ar[i, k], ar[j, k + 0.5], rgt]]
    - NormalizeDiag[Diag[lft, ar[i, k + 0.5], ar[j, k], rgt]]
  );
RSansFI[d_Diag] := (R[d] /. Diag[lft___, ar[i_, j_], rgt___] /; Abs[i - j] == 1 :> 0)
```

Dimensions

```
DimAvLong[m_] /; m < 2 := Length[Diagrams[m ar]];
DimAvLong[m_] /; m ≥ 2 := Module[
   {diags, rels, mat, rel, i},
   diags = Diagrams[m ar];
   rels = R /@ Diagrams[R6T + (m - 2) ar];
   mat = SparseArray[
     Join @@ Table[
       rel = rels[[i]];
       {i, Position[diags, #][[1, 1]]} → Coefficient[rel, #] & /@
        Cases[{rel}, diag_Diag, Infinity],
       {i, Length[rels]}
      ],
     {Length[rels], Length[diags]}
    ];
   Length[diags] - MatrixRank[mat]
  ];

DimAvrLong[m_] /; m < 2 := Length[DiagramsSansFI[m ar]];
DimAvrLong[m_] /; m ≥ 2 := Module[
   {diags, rels, diagtoindex, mat, rel, i},
   diags = DiagramsSansFI[m ar];
   rels = RSansFI /@ DiagramsSansFI[R6T + (m - 2) ar];
   diagtoindex = Dispatch[Thread[Rule[diags, Range[Length[diags]]]]];
   mat = SparseArray[
     Join @@ Table[
       rel = rels[[i]];
       {i, # /. diagtoindex} → Coefficient[rel, #] & /@
        Cases[{rel}, diag_Diag, Infinity],
       {i, Length[rels]}
      ],
     {Length[rels], Length[diags]}
    ];
   Length[diags] - MatrixRank[mat]
  ];

DimAvClosed[m_] /; m < 2 := Length[ClosedDiagrams[m ar]];
DimAvClosed[m_] /; m ≥ 2 := Module[
  {diags, rels, mat, rel, i},
  diags = ClosedDiagrams[m ar];
  rels = DiagramRotateToMinimal[R /@ ClosedDiagrams[R6T + (m - 2) ar]];
  mat = SparseArray[
    Join @@ Table[
      rel = rels[[i]];
      {i, Position[diags, #][[1, 1]]} → Coefficient[rel, #] & /@
       Cases[{rel}, diag_Diag, Infinity],
      {i, Length[rels]}
     ],
    {Length[rels], Length[diags]}
   ];
  Length[diags] - MatrixRank[mat]
 ]
```

```
DimAvrClosed[m_] /; m < 2 := Length[ClosedDiagramsSansFI[m ar]];
DimAvrClosed[m_] /; m ≥ 2 := Module[
   {diags, rels, mat, rel, i},
   diags = ClosedDiagramsSansFI[m ar];
   rels = Union[
     DiagramRotateToMinimal[RSansFI /@ ClosedDiagramsSansFI[R6T + (m - 2) ar]] /.
      Diag[lft___, ar[i_, j_], rgt___] /; Abs[i - j] == 1 :> 0
    ];
   mat = SparseArray[
     Join @@ Table[
       rel = rels[[i]];
       {i, Position[diags, #][[1, 1]]} → Coefficient[rel, #] & /@
        Cases[{rel}, diag_Diag, Infinity],
       {i, Length[rels]}
      ],
     {Length[rels], Length[diags]}
    ];
   Length[diags] - MatrixRank[mat]
  ];

DimAwLong[m_] /; m < 2 := Length[Diagrams[m ar]];
DimAwLong[m_] /; m ≥ 2 := Module[
   {diags, rels, mat, rel, i},
   diags = Diagrams[m ar];
   rels = R /@ Join[Diagrams[R6T + (m - 2) ar], Diagrams[TC + (m - 2) ar]];
   mat = SparseArray[
     Join @@ Table[
       rel = rels[[i]];
       {i, Position[diags, #][[1, 1]]} → Coefficient[rel, #] & /@
        Cases[{rel}, diag_Diag, Infinity],
       {i, Length[rels]}
      ],
     {Length[rels], Length[diags]}
    ];
   Length[diags] - MatrixRank[mat]
  ];
```

```
DimAwClosed[m_] /; m < 2 := Length[ClosedDiagrams[m ar]];
DimAwClosed[m_] /; m ≥ 2 := Module[
   {diags, rels, mat, rel, i},
   diags = ClosedDiagrams[m ar];
   rels = DiagramRotateToMinimal[
     R /@ Join[Diagrams[R6T + (m - 2) ar], Diagrams[TC + (m - 2) ar]]
    ];
   mat = SparseArray[
     Join @@ Table[
       rel = rels[[i]];
       {i, Position[diags, #][[1, 1]]} → Coefficient[rel, #] & /@
        Cases[{rel}, diag_Diag, Infinity],
       {i, Length[rels]}
      ],
     {Length[rels], Length[diags]}
    ];
   Length[diags] - MatrixRank[mat]
  ];
```

From the Dimensions of the Primitives to the Total Dimensions

```
PrimitivesToAll[p_List] := Module[{x},
  CoefficientList[Series[
    Product[(1 - x^i)^(-p[[i]]), {i, Length[p]}],
    {x, 0, Length[p]}], x]
 ]
```

Results

```
PrimitivesToAll[{1, 1, 1, 2, 3, 5, 8}]
```

```
{1, 1, 2, 3, 6, 10, 19, 33}
```

```
Timing[DimAvLong /@ {1, 2, 3, 4}]
```

```
{28.8066, {2, 7, 27, 139}}
```

```
PrimitivesToAll[{2, 4, 15, 82}]
```

```
{1, 2, 7, 27, 139}
```

```
Timing[DimAvLong[5]]
```

```
No more memory available.
Mathematica kernel has shut down.
Try quitting other applications and then retry.
```

```
Timing[DimAwLong /@ {1, 2, 3, 4}]
```

```
{32.665, {2, 4, 7, 12}}
```

```
PrimitivesToAll[{2, 1, 1, 1, 1, 1, 1}]
```

```
{1, 2, 4, 7, 12, 19, 30, 45}
```

```
PrimitivesToAll[{0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}]
```

```
{1, 0, 1, 1, 2, 2, 4, 4, 7, 8, 12, 14, 21}
```

```
Timing[DimAvrLong /@ {1, 2, 3, 4}]
```

```
{1.42078, {0, 2, 7, 42}}
```

```
Timing[DimAvrLong[5]]
```

```
{3195.44, 246}
```

```
Timing[DimAvClosed /@ {1, 2, 3, 4}]
```

```
{2.85857, {1, 2, 5, 19}}
```

```
Timing[DimAvClosed[5]]
```

```
{577.641, 77}
```

```
Timing[DimAwClosed /@ {1, 2, 3, 4}]
```

```
{11.2013, {1, 1, 1, 1}}
```

```
Timing[DimAwClosed[5]]
```

```
No more memory available.
Mathematica kernel has shut down.
Try quitting other applications and then retry.
```

```
Timing[DimAvrClosed /@ {1, 2, 3, 4, 5}]
```

```
{104.895, {0, 0, 1, 4, 17}}
```