

Arrow Diagrams and Lie Bialgebras - Program

Joint project of Louis Leung and Dror Bar-Natan.

Pensieve Header: This will be the final version of the programs to carry out computations with arrow diagrams and Lie-bialgebra weight systems as in the PhD thesis by Louis Leung.

```
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]}]
]];
Place[{R6T}, {i_, j_, k_}] :=
  Permutations[{i, j, k}] /. {i1_, j1_, k1_} => Diag[R6T[i1, j1, k1]];
Place[{R6T, objs_}, {i_, rest_}] := Flatten[Table[
  Outer[Join,
    Place[{R6T}, {i, {rest}[[j]], {rest}[[k]]}],
    Place[{objs}, Delete[{rest}, {j}, {k}]]
  ],
  {k, 2, Length[{rest]}}, {j, 1, k-1}
]];
Diagrams[k_. * ar] := Place[Table[ar, {k}], Range[2 k]];
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 := {};
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]]
)
```

```

+NormalizedDiag [Diag[lft, ar[i, k], ar[j, k + 0.5], rgt]]
-NormalizedDiag [Diag[lft, ar[i, k], ar[i + 0.5, j], rgt]]
-NormalizedDiag [Diag[lft, ar[i, j + 0.5], ar[j, k], rgt]]
-NormalizedDiag [Diag[lft, ar[i, k + 0.5], ar[j, k], rgt]]
);
DimAArrow[m_] /; m < 2 := Length[Diagrams[m ar]];
DimAArrow[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]
];
BasisAArrow[m_] /; m < 2 := Diagrams[m ar];
BasisAArrow[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]}
  ];
  diags[[
    Complement[Range[Length[diags]],
      Flatten[
        Position[#, 1, {1}, 1] & /@ DeleteCases[
          RowReduce[mat],
          {Repeated[0]}
        ]
      ]
  ]

```

```

]
]
]]
]

```

Next challenge: given a diag generate all the $gl(N)$ -inspired equalities and inequalities corresponding to it, and count the solutions by counting permutations.

```

OrderTypes[0] = {{}};
OrderTypes[1] = {{1}};
OrderTypes[l_List] := Module[{nl, snl},
  (
    snl = Union[nl = Append[l, #]];
    nl /. Thread[snl → Range[Length[snl]]]
  ) & /@ Range[1/2, 1/2 + Max[l], 1/2]
];
OrderTypes[n_Integer] := OrderTypes[n] = Join @@ (OrderTypes /@ OrderTypes[n - 1]);
OrderTypes[n_Integer, {}] := OrderTypes[n];
OrderTypes[n_Integer, {{i_, j_}, rest___}] := OrderTypes[n, {{i, j}, rest}] = (
  Print["Computing ", "OrderTypes"[n, {{i, j}, rest]]];
  Select[
    OrderTypes[n, {rest}], #[[i]] < #[[j]] &
  ]
);
Clear[Wgl];
Wgl[diag_Diag] := Wgl[diag] = Module[
  {p, eq, ltheq, rule, l, indices, ineqs, i0, ot, res, n, k},
  p = Append[
    diag /. ar[i_, j_] → Sequence[eq[i, j - 1], eq[i - 1, j], ltheq[i - 1, i]],
    s[0]
  ];
  While[! FreeQ[p, eq],
    p = (p
      /. Cases[p, eq[i_, j_] → (i → j), Infinity, 1]
      /. {
        eq[i_, i_] → Sequence[]
      }
    )
  ];
  l = Length[indices = Union @@ Cases[{p}, ltheq[i_, j_] → {i, j}, Infinity]];
  ineqs = p /. Thread[indices → Range[l]];
  {i0} = Cases[ineqs, s[i_] → i, {1}, 1];

```

```

ineqs = DeleteCases[ineqs, _s];
res = Expand[Sum[
  ot = OrderTypes[1][[i]];
  Times[
    (Times @@ ineqs) /. ltheq[i_, j_] => Switch[
      Order[ot[[i]], ot[[j]],
      1, 1,
      0, 1/2,
      -1, 0
    ],
    Binomial[k-1, ot[[i0]]-1],
    Binomial[n-k, Max[ot]-ot[[i0]]]
  ],
  {i, Length[OrderTypes[1]]}
]];
Function@@{(res /. {n -> #1, k -> #2})}
];
RankWgl[m_] := Module[{v, n, k},
  v = Wgl[#][n, k] & /@ Diagrams[m*ar];
  MatrixRank[Table[
    v /. {n -> Random[], k -> Random[]},
    {Length[v]}
  ]]
]
base = 10;
ij[alpha_Integer] := IntegerDigits[alpha, base, 2];
alpha[{i_, j_}] := base * i + j;
alpha[i_, j_] := base * i + j;
b[e[alpha1_], e[alpha2_]] := b[e[alpha1], e[alpha2]] = Module[
  {i, j, k, l},
  {i, j} = ij[alpha1];
  {k, l} = ij[alpha2];
  If[j == k, e[alpha[i, l]], 0] - If[i == l, e[alpha[k, j]], 0]
];
Unprotect[NonCommutativeMultiply];
_**0 = 0;
0**_ = 0;
(a_Plus)**w_RW := (#**w) & /@ a;
e[alpha_]** (a_Plus) := (e[alpha]**#) & /@ a;
(c_*e[alpha_])**w_RW := Expand[c*(e[alpha]**w)];
e[alpha_]** (c_*w_RW) := Expand[c*(e[alpha]**w)];

```

```

e[alpha_] ** RW[] := RW[alpha];
e[alpha_] ** RW[beta_, rest___] /; alpha <= beta := RW[alpha, beta, rest];
e[alpha_] ** RW[beta_, rest___] /; alpha > beta := Plus[
  e[beta] ** (e[alpha] ** RW[rest]),
  b[e[alpha], e[beta]] ** RW[rest]
];

CanonicalForm[W[]] = RW[];
CanonicalForm[W[alpha_, rest___]] := Expand[
  e[alpha] ** CanonicalForm[W[rest]]
];
CanonicalForm[SW[]] = RSW[];
CanonicalForm[SW[alpha_, rest___]] := If[OrderedQ[{alpha, rest}, RSW[alpha, rest]],
  If[Length[{rest}] <= 3,
    CanonicalForm[SW[alpha, rest]] = Expand[se[alpha] ** CanonicalForm[SW[rest]]],
    Expand[se[alpha] ** CanonicalForm[SW[rest]]]
  ]
];

(*d for odd*)
CanonicalForm[SdW[]] = RSdW[];
CanonicalForm[SdW[alpha_, rest___]] := If[OrderedQ[{alpha, rest}, RSdW[alpha, rest]],
  If[Length[{rest}] <= 3,
    CanonicalForm[SdW[alpha, rest]] = Expand[sde[alpha] ** CanonicalForm[SdW[rest]]],
    Expand[sde[alpha] ** CanonicalForm[SdW[rest]]]
  ]
];

(*Sp for symplectic*)
CanonicalForm[SpW[]] = RSpW[];
CanonicalForm[SpW[alpha_, rest___]] := If[OrderedQ[{alpha, rest}, RSpW[alpha, rest]],
  If[Length[{rest}] <= 3,
    CanonicalForm[SpW[alpha, rest]] = Expand[spe[alpha] ** CanonicalForm[SpW[rest]]],
    Expand[spe[alpha] ** CanonicalForm[SpW[rest]]]
  ]
];

CanonicalForm[expr_] := Expand[expr /. {w_W -> CanonicalForm[w],
  w_SW -> CanonicalForm[w], w_SdW -> CanonicalForm[w], w_SpW -> CanonicalForm[w}}];

(* CanonicalForm[expr_] := Expand[expr /. {
  W[] -> RW[],
  W[alpha_, rest___] -> e[alpha]**CanonicalForm[W[rest]],

```

```

    SW[] → RSW[],
    SW[alpha_, rest___] ⇒ se[alpha]**CanonicalForm[SW[rest]]
  ]]; *)

UGL2[diag_Diag] := Module[
  {PutAt, w},
  Expand[CanonicalForm[
    Distribute[diag /.
      ar[i_, j_] ⇒ 1/2 PutAt[i, j, 11, 11] + PutAt[i, j, 12, 21] + 1/2 PutAt[i, j, 22, 22]
    ] /. d_Diag ⇒ Times[
      Times @@ (d /. _PutAt → 1),
      (
        w = Table[0, {2 Length[d]}];
        d /. PutAt[i_, j_, alpha1_, alpha2_] ⇒ (
          w[[i]] = alpha1; w[[j]] = alpha2
        );
        W@@w
      )
    ]
  ]];
UGL2[expr_] := Expand[expr /. diag_Diag ⇒ UGL2[diag]];

UGL2BiAlg[diag_Diag] := Module[
  {PutAt, w, i, j},
  Expand[CanonicalForm[
    Distribute[diag /. ar[i_, j_] ⇒ Expand[
      PutAt[i, j, 12, 21] + 1/2 (
        PutAt[i, j, 11, 11] + PutAt[i, j, 22, 22]
        - h[1] PutAt[i, 11] - h[2] PutAt[i, 22]
        + h[1] PutAt[j, 11] + h[2] PutAt[j, 22]
        - h[1]^2 * PutAt[] - h[2]^2 * PutAt[]
      )
    ]
    ] /. d_Diag ⇒ Times[
      Times @@ (d /. _PutAt → 1),
      (
        w = Table[1, {2 Length[d]}];
        d /. {
          PutAt[i_, alpha_] ⇒ (w[[i]] = alpha),
          PutAt[i_, j_, alpha1_, alpha2_] ⇒ (
            w[[i]] = alpha1; w[[j]] = alpha2
          )
        }
      ];
      W@@DeleteCases[w, 1]
    ]
  ]];
UGL2BiAlg[expr_] := Expand[expr /. diag_Diag ⇒ UGL2BiAlg[diag]];

```

```

Clear[UGLnBiAlg];
UGLnBiAlg[diag_Diag] := Module[
  {allterms, s, LTAt, EQAt, CAT, m1, m2, dirs, coeff, k, ltpairs, w, allots, i, j},
  allterms = List@@Expand[(Times @@ diag) /. ar[p_, q_] =>
    LTAt[p, q] + 1/2 (EQAt[p, q] - CAT[p] + CAT[q] - CAT[])
  ];
  s = Sum[
    dirs = allterms[[m1]];
    coeff = dirs /. (_CAT | _EQAt | _LTAt) -> 1;
    k = 0;
    w = Table[1, {2 Length[diag]}];
    ltpairs = {};
    dirs /. {
      LTAt[p_, q_] => (
        w[[p]] = e[i = ++k, j = ++k];
        w[[q]] = e[j, i];
        AppendTo[ltpairs, {i, j}];
      ),
      EQAt[p_, q_] => (w[[p]] = w[[q]] = e[i = ++k, i]);
      CAT[p_] => (
        w[[p]] = e[i = ++k, i];
        coeff *= h[i];
      ),
      CAT[]^p_ => Do[coeff *= h[++k]^2, {p}]
    };
  w = DeleteCases[w, 1];
  allots = OrderTypes[k, ltpairs];
  Sum[
    ot = allots[[m2]];
    Times[
      (* Binomial[n, Max[ot]], *)
      n^Max[ot],
      coeff /. h[i_] => h[ot[[i]]],
      w@@w /. e[i_, j_] => alpha[ot[[i]], ot[[j]]]
    ],
    {m2, Length[allots]}
  ], {m1, Length[allterms]}
];
If[Length[diag] <= 3,
  UGLnBiAlg[diag] = CanonicalForm[Expand[s]],
  CanonicalForm[Expand[s]]
]
];
UGLnBiAlg[expr_] := Expand[expr /. diag_Diag => UGLnBiAlg[diag]];

```

```

Clear[UGLnBiAlg0];
UGLnBiAlg0[diag_Diag] := Module[
  {allterms, s, LTAT, EQAT, CAT, m1, m2, dirs, coeff, k, ltpairs, w, allots, i, j},
  allterms = List@@Expand[(Times @@ diag) /. ar[p_, q_] =>
    LTAT[p, q] + 1/2 (EQAT[p, q] - CAT[p] + CAT[q])
  ];
  s = Sum[
    dirs = allterms[[m1]];
    coeff = dirs /. (_CAT | _EQAT | _LTAT) -> 1;
    k = 0;
    w = Table[1, {2 Length[diag]}];
    ltpairs = {};
    dirs /. {
      LTAT[p_, q_] => (
        w[[p]] = e[i = ++k, j = ++k];
        w[[q]] = e[j, i];
        AppendTo[ltpairs, {i, j}];
      ),
      EQAT[p_, q_] => (w[[p]] = w[[q]] = e[i = ++k, i]);,
      CAT[p_] => (
        w[[p]] = e[i = ++k, i];
        coeff *= h[i];
      ),
      CAT[]^p_ => Do[coeff *= h[++k]^2, {p}]
    };
  w = DeleteCases[w, 1];
  allots = OrderTypes[k, ltpairs];
  Sum[
    ot = allots[[m2]];
    Times[
      (* Binomial[n, Max[ot]], *)
      n^Max[ot],
      coeff /. h[i_] => h[ot[[i]]],
      w@@w /. e[i_, j_] => alpha[ot[[i]], ot[[j]]]
    ],
    {m2, Length[allots]}
  ], {m1, Length[allterms]}
];
If[Length[diag] <= 3,
  UGLnBiAlg[diag] = CanonicalForm[Expand[s]],
  CanonicalForm[Expand[s]]
]
];
UGLnBiAlg0[expr_] := Expand[expr /. diag_Diag => UGLnBiAlg0[diag]];
Coproduct[diag_Diag, l_] := Total[
  T[
    NormalizeDiag[#],
    NormalizeDiag[Complement[diag, #]]
  ] & /@ Subsets[diag, {1}]
]

```



```

BasisAArrow[4] = {Diag[ar[5, 1], ar[6, 4], ar[7, 3], ar[8, 2]],
  Diag[ar[3, 2], ar[6, 1], ar[7, 5], ar[8, 4]], Diag[ar[4, 2], ar[6, 1], ar[7, 5], ar[8, 3]],
  Diag[ar[4, 7], ar[5, 2], ar[6, 1], ar[8, 3]], Diag[ar[5, 2], ar[6, 1], ar[7, 4], ar[8, 3]],
  Diag[ar[4, 3], ar[6, 1], ar[7, 5], ar[8, 2]], Diag[ar[4, 7], ar[5, 3], ar[6, 1], ar[8, 2]],
  Diag[ar[5, 3], ar[6, 1], ar[7, 4], ar[8, 2]], Diag[ar[4, 5], ar[6, 1], ar[7, 3], ar[8, 2]],
  Diag[ar[5, 4], ar[6, 1], ar[7, 3], ar[8, 2]], Diag[ar[1, 7], ar[4, 2], ar[6, 3], ar[8, 5]],
  Diag[ar[1, 7], ar[3, 6], ar[4, 8], ar[5, 2]], Diag[ar[1, 7], ar[3, 6], ar[5, 2], ar[8, 4]],
  Diag[ar[1, 7], ar[5, 2], ar[6, 3], ar[8, 4]], Diag[ar[1, 7], ar[3, 8], ar[5, 2], ar[6, 4]],
  Diag[ar[1, 7], ar[5, 2], ar[6, 4], ar[8, 3]], Diag[ar[1, 7], ar[2, 6], ar[5, 3], ar[8, 4]],
  Diag[ar[1, 7], ar[4, 3], ar[6, 2], ar[8, 5]], Diag[ar[1, 7], ar[3, 5], ar[4, 8], ar[6, 2]],
  Diag[ar[1, 7], ar[3, 5], ar[6, 2], ar[8, 4]], Diag[ar[1, 7], ar[4, 8], ar[5, 3], ar[6, 2]],
  Diag[ar[1, 7], ar[5, 3], ar[6, 2], ar[8, 4]], Diag[ar[1, 7], ar[3, 8], ar[4, 5], ar[6, 2]],
  Diag[ar[1, 7], ar[3, 8], ar[5, 4], ar[6, 2]], Diag[ar[1, 7], ar[4, 5], ar[6, 2], ar[8, 3]],
  Diag[ar[1, 7], ar[5, 4], ar[6, 2], ar[8, 3]], Diag[ar[1, 7], ar[2, 8], ar[5, 3], ar[6, 4]],
  Diag[ar[1, 7], ar[2, 8], ar[5, 4], ar[6, 3]], Diag[ar[1, 7], ar[3, 5], ar[6, 4], ar[8, 2]],
  Diag[ar[1, 7], ar[4, 6], ar[5, 3], ar[8, 2]], Diag[ar[1, 7], ar[5, 3], ar[6, 4], ar[8, 2]],
  Diag[ar[1, 7], ar[5, 4], ar[6, 3], ar[8, 2]], Diag[ar[3, 2], ar[4, 8], ar[6, 5], ar[7, 1]],
  Diag[ar[3, 2], ar[6, 5], ar[7, 1], ar[8, 4]], Diag[ar[4, 2], ar[5, 3], ar[6, 8], ar[7, 1]],
  Diag[ar[4, 2], ar[5, 3], ar[7, 1], ar[8, 6]], Diag[ar[4, 2], ar[5, 8], ar[6, 3], ar[7, 1]],
  Diag[ar[4, 2], ar[6, 3], ar[7, 1], ar[8, 5]], Diag[ar[2, 5], ar[4, 8], ar[6, 3], ar[7, 1]],
  Diag[ar[2, 5], ar[6, 3], ar[7, 1], ar[8, 4]], Diag[ar[3, 4], ar[5, 2], ar[6, 8], ar[7, 1]],
  Diag[ar[3, 4], ar[5, 2], ar[7, 1], ar[8, 6]], Diag[ar[4, 3], ar[5, 2], ar[6, 8], ar[7, 1]],
  Diag[ar[4, 3], ar[5, 2], ar[7, 1], ar[8, 6]], Diag[ar[3, 6], ar[4, 8], ar[5, 2], ar[7, 1]],
  Diag[ar[3, 6], ar[5, 2], ar[7, 1], ar[8, 4]], Diag[ar[4, 8], ar[5, 2], ar[6, 3], ar[7, 1]],
  Diag[ar[5, 2], ar[6, 3], ar[7, 1], ar[8, 4]], Diag[ar[3, 8], ar[4, 6], ar[5, 2], ar[7, 1]],
  Diag[ar[3, 8], ar[5, 2], ar[6, 4], ar[7, 1]], Diag[ar[4, 6], ar[5, 2], ar[7, 1], ar[8, 3]],
  Diag[ar[5, 2], ar[6, 4], ar[7, 1], ar[8, 3]], Diag[ar[2, 6], ar[4, 3], ar[5, 8], ar[7, 1]],
  Diag[ar[2, 6], ar[4, 3], ar[7, 1], ar[8, 5]], Diag[ar[2, 6], ar[3, 5], ar[4, 8], ar[7, 1]],
  Diag[ar[2, 6], ar[3, 5], ar[7, 1], ar[8, 4]], Diag[ar[2, 6], ar[4, 8], ar[5, 3], ar[7, 1]],
  Diag[ar[2, 6], ar[5, 3], ar[7, 1], ar[8, 4]], Diag[ar[2, 6], ar[3, 8], ar[5, 4], ar[7, 1]],
  Diag[ar[2, 6], ar[4, 5], ar[7, 1], ar[8, 3]], Diag[ar[2, 6], ar[5, 4], ar[7, 1], ar[8, 3]],
  Diag[ar[4, 3], ar[5, 8], ar[6, 2], ar[7, 1]], Diag[ar[4, 3], ar[6, 2], ar[7, 1], ar[8, 5]],
  Diag[ar[3, 5], ar[4, 8], ar[6, 2], ar[7, 1]], Diag[ar[3, 5], ar[6, 2], ar[7, 1], ar[8, 4]],
  Diag[ar[4, 8], ar[5, 3], ar[6, 2], ar[7, 1]], Diag[ar[5, 3], ar[6, 2], ar[7, 1], ar[8, 4]],
  Diag[ar[3, 8], ar[4, 5], ar[6, 2], ar[7, 1]], Diag[ar[3, 8], ar[5, 4], ar[6, 2], ar[7, 1]],
  Diag[ar[4, 5], ar[6, 2], ar[7, 1], ar[8, 3]], Diag[ar[5, 4], ar[6, 2], ar[7, 1], ar[8, 3]],
  Diag[ar[2, 8], ar[3, 5], ar[6, 4], ar[7, 1]], Diag[ar[2, 8], ar[4, 6], ar[5, 3], ar[7, 1]],
  Diag[ar[2, 8], ar[5, 3], ar[6, 4], ar[7, 1]], Diag[ar[2, 8], ar[3, 6], ar[4, 5], ar[7, 1]],
  Diag[ar[2, 8], ar[3, 6], ar[5, 4], ar[7, 1]], Diag[ar[2, 8], ar[4, 5], ar[6, 3], ar[7, 1]],
  Diag[ar[2, 8], ar[5, 4], ar[6, 3], ar[7, 1]], Diag[ar[3, 5], ar[6, 4], ar[7, 1], ar[8, 2]],
  Diag[ar[4, 6], ar[5, 3], ar[7, 1], ar[8, 2]], Diag[ar[5, 3], ar[6, 4], ar[7, 1], ar[8, 2]],
  Diag[ar[3, 6], ar[4, 5], ar[7, 1], ar[8, 2]], Diag[ar[3, 6], ar[5, 4], ar[7, 1], ar[8, 2]],

```

```

Diag[ar[4, 5], ar[6, 3], ar[7, 1], ar[8, 2]], Diag[ar[5, 4], ar[6, 3], ar[7, 1], ar[8, 2]],
Diag[ar[1, 8], ar[5, 2], ar[6, 4], ar[7, 3]], Diag[ar[1, 8], ar[2, 6], ar[4, 7], ar[5, 3]],
Diag[ar[1, 8], ar[2, 6], ar[5, 3], ar[7, 4]], Diag[ar[1, 8], ar[4, 3], ar[5, 7], ar[6, 2]],
Diag[ar[1, 8], ar[4, 3], ar[6, 2], ar[7, 5]], Diag[ar[1, 8], ar[3, 5], ar[4, 7], ar[6, 2]],
Diag[ar[1, 8], ar[3, 5], ar[6, 2], ar[7, 4]], Diag[ar[1, 8], ar[4, 7], ar[5, 3], ar[6, 2]],
Diag[ar[1, 8], ar[5, 3], ar[6, 2], ar[7, 4]], Diag[ar[1, 8], ar[3, 7], ar[4, 5], ar[6, 2]],
Diag[ar[1, 8], ar[3, 7], ar[5, 4], ar[6, 2]], Diag[ar[1, 8], ar[4, 5], ar[6, 2], ar[7, 3]],
Diag[ar[1, 8], ar[5, 4], ar[6, 2], ar[7, 3]], Diag[ar[1, 8], ar[2, 7], ar[3, 5], ar[6, 4]],
Diag[ar[1, 8], ar[2, 7], ar[4, 6], ar[5, 3]], Diag[ar[1, 8], ar[2, 7], ar[5, 3], ar[6, 4]],
Diag[ar[1, 8], ar[2, 7], ar[3, 6], ar[4, 5]], Diag[ar[1, 8], ar[2, 7], ar[3, 6], ar[5, 4]],
Diag[ar[1, 8], ar[2, 7], ar[4, 5], ar[6, 3]], Diag[ar[1, 8], ar[2, 7], ar[5, 4], ar[6, 3]],
Diag[ar[1, 8], ar[3, 5], ar[6, 4], ar[7, 2]], Diag[ar[1, 8], ar[4, 6], ar[5, 3], ar[7, 2]],
Diag[ar[1, 8], ar[5, 3], ar[6, 4], ar[7, 2]], Diag[ar[1, 8], ar[3, 6], ar[4, 5], ar[7, 2]],
Diag[ar[1, 8], ar[3, 6], ar[5, 4], ar[7, 2]], Diag[ar[1, 8], ar[4, 5], ar[6, 3], ar[7, 2]],
Diag[ar[1, 8], ar[5, 4], ar[6, 3], ar[7, 2]], Diag[ar[5, 2], ar[6, 4], ar[7, 3], ar[8, 1]],
Diag[ar[2, 6], ar[4, 7], ar[5, 3], ar[8, 1]], Diag[ar[2, 6], ar[5, 3], ar[7, 4], ar[8, 1]],
Diag[ar[4, 3], ar[5, 7], ar[6, 2], ar[8, 1]], Diag[ar[4, 3], ar[6, 2], ar[7, 5], ar[8, 1]],
Diag[ar[3, 5], ar[4, 7], ar[6, 2], ar[8, 1]], Diag[ar[3, 5], ar[6, 2], ar[7, 4], ar[8, 1]],
Diag[ar[4, 7], ar[5, 3], ar[6, 2], ar[8, 1]], Diag[ar[5, 3], ar[6, 2], ar[7, 4], ar[8, 1]],
Diag[ar[3, 7], ar[4, 5], ar[6, 2], ar[8, 1]], Diag[ar[3, 7], ar[5, 4], ar[6, 2], ar[8, 1]],
Diag[ar[4, 5], ar[6, 2], ar[7, 3], ar[8, 1]], Diag[ar[5, 4], ar[6, 2], ar[7, 3], ar[8, 1]],
Diag[ar[2, 7], ar[3, 5], ar[6, 4], ar[8, 1]], Diag[ar[2, 7], ar[4, 6], ar[5, 3], ar[8, 1]],
Diag[ar[2, 7], ar[5, 3], ar[6, 4], ar[8, 1]], Diag[ar[2, 7], ar[3, 6], ar[4, 5], ar[8, 1]],
Diag[ar[2, 7], ar[3, 6], ar[5, 4], ar[8, 1]], Diag[ar[2, 7], ar[4, 5], ar[6, 3], ar[8, 1]],
Diag[ar[2, 7], ar[5, 4], ar[6, 3], ar[8, 1]], Diag[ar[3, 5], ar[6, 4], ar[7, 2], ar[8, 1]],
Diag[ar[4, 6], ar[5, 3], ar[7, 2], ar[8, 1]], Diag[ar[5, 3], ar[6, 4], ar[7, 2], ar[8, 1]],
Diag[ar[3, 6], ar[4, 5], ar[7, 2], ar[8, 1]], Diag[ar[3, 6], ar[5, 4], ar[7, 2], ar[8, 1]],
Diag[ar[4, 5], ar[6, 3], ar[7, 2], ar[8, 1]], Diag[ar[5, 4], ar[6, 3], ar[7, 2], ar[8, 1]];

```

se[1,i,j] stands for A_{ij} ;

se[2,i,j] stands for B_{ij} ;

se[3,i,j] stands for D_{ij} .

Commutation relations come from <http://katlas.math.toronto.edu/drorbn/bbs/show?shot=Leung-090227-132523.jpg>

```

sbase = 10;
sij[salphi_Integer] := IntegerDigits[salphi, sbase, 3];
salphi[{t_, i_, j_}] := sbase^2*t + sbase*i + j;
salphi[t_, i_, j_] := sbase^2*t + sbase*i + j;
b[se[salphi1_], se[salphi2_]] := b[se[salphi1], se[salphi2]] = Module[
  {d, t1, t2, i, j, k, l, s = 1},
  d[i_, j_] := If[i == j, 1, 0];
  {t1, i, j} = sij[salphi1];

```

```

{t2, k, l} = sij[salpha2];
If[t2 < t1, s = -1; {{t1, i, j}, {t2, k, l}} = {{t2, k, l}, {t1, i, j}}];
{
  {
    d[j, k] se[1, i, l] - d[i, l] se[1, k, j],
    d[j, k] se[2, i, l] - d[j, l] se[2, i, k],
    d[i, k] se[3, l, j] + d[l, i] se[3, j, k]
  },
  {
    Null,
    0,
    d[j, l] se[1, i, k] + d[i, k] se[1, j, l] - d[j, k] se[1, i, l] - d[i, l] se[1, j, k]
  },
  {
    Null,
    Null,
    0
  }
}[[{t1, t2}]] /. {
se[2 | 3, i_, i_] → 0,
se[2, i_, j_] /; i > j → -se[2, j, i],
se[3, i_, j_] /; i > j → -se[3, j, i]
} /. se[tij_] → s * se[salpha[tij]]
];
(a_Plus) ** w_RSW := (#**w) & /@ a;
se[alpha_] ** (a_Plus) := (se[alpha] ** #) & /@ a;
(c_ * se[alpha_]) ** w_RSW := Expand[c * (se[alpha] ** w)];
se[alpha_] ** (c_ * w_RSW) := Expand[c * (se[alpha] ** w)];
se[alpha_] ** RSW[] := RSW[alpha];
se[alpha_] ** RSW[beta_, rest___] /; alpha ≤ beta := RSW[alpha, beta, rest];
se[alpha_] ** RSW[beta_, rest___] /; alpha > beta := Plus[
  se[beta] ** (se[alpha] ** RSW[rest]),
  b[se[alpha], se[beta]] ** RSW[rest]
];

Clear[USO];
USO[diag_Diag] := Module[
  {allterms, s, LTAt, BDAt, EQAt, CAT,
   m1, m2, dirs, coeff, k, ltpairs, w, allots, tc = 0, i, j},
  allterms = List @@ Expand[(Times @@ diag) /. ar[p_, q_] →
    LTAt[p, q] + BDAt[p, q] + 1 / 2 (EQAt[p, q] - CAT[p] + CAT[q] - CAT[])
  ];
];

```

```

s = Sum[
  dirs = allterms[[m1]]; (* "dirs" is "directions" *)
  coeff = dirs /. (_CAT | _EQat | _LTat | _BDat) → 1;
  k = 0;
  w = Table[1, {2 Length[diag]}];
  ltpairs = {};
  dirs /. {
    LTat[p_, q_] ⇒ (
      w[[p]] = se[1, i = ++k, j = ++k];
      w[[q]] = se[1, j, i];
      AppendTo[ltpairs, {i, j}];
    ),
    BDat[p_, q_] ⇒ (
      w[[p]] = se[2, i = ++k, j = ++k];
      w[[q]] = se[3, i, j];
      AppendTo[ltpairs, {i, j}];
    ),
    EQat[p_, q_] ⇒ (w[[p]] = w[[q]] = se[1, i = ++k, i]);),
  CAT[p_] ⇒ (
    w[[p]] = se[1, i = ++k, i];
    coeff *= h[i];
  ),
  CAT[ ] ^ p_. ⇒ Do[coeff *= h[++k]^2, {p}]
];
w = DeleteCases[w, 1];
allots = OrderTypes[k, ltpairs];
Sum[
  ot = allots[[m2]];
  ++tc;
  CanonicalForm[Expand[Times[
    n^Max[ot],
    coeff /. h[i_] ⇒ h[ot[[i]]],
    SW@w /. se[t_, i_, j_] ⇒ salpha[t, ot[[i]], ot[[j]]]
  ]],
  {m2, Length[allots]}
], {m1, Length[allterms]}
];
Print[{tc, Length[s]};
If[Length[diag] ≤ 3, USO[diag] = s, s]
];
USO[expr_] := Expand[expr /. diag_Diag ⇒ USO[diag]];

```

```

spbase = 10;
spij[spalpha_Integer] := IntegerDigits[spalpha, spbase, 3];
spalpha[{t_, i_, j_}] := spbase^2*t + spbase*i + j;
spalpha[t_, i_, j_] := spbase^2*t + spbase*i + j;
b[spe[spalpha1_], spe[spalpha2_]] := b[spe[spalpha1], spe[spalpha2]] = Module[
  {d, t1, t2, i, j, k, l, s = 1},
  d[i_, j_] := If[i == j, 1, 0];
  {t1, i, j} = spij[spalpha1];
  {t2, k, l} = spij[spalpha2];
  If[t2 < t1, s = -1; {{t1, i, j}, {t2, k, l}} = {{t2, k, l}, {t1, i, j}}];
  {
    {
      d[j, k] spe[1, i, l] - d[i, l] spe[1, k, j],
      d[j, k] spe[2, i, l] + d[j, l] spe[2, i, k],
      -d[i, k] spe[3, l, j] - d[l, i] spe[3, j, k]
    },
    {
      Null,
      0,
      d[j, l] spe[1, i, k] + d[i, k] spe[1, j, l] + d[j, k] spe[1, i, l] + d[i, l] spe[1, j, k]
    },
    {
      Null,
      Null,
      0
    }
  }
  ][[t1, t2]] /. {
  (*spe[2|3, i_, i_] → 0,*)
  spe[2, i_, j_] /; i > j ⇒ spe[2, j, i],
  spe[3, i_, j_] /; i > j ⇒ spe[3, j, i]
  } /. spe[tij_] ⇒ s * spe[spalpha[tij]]
];
(a_Plus) ** w_RSpW := (#**w) & /@ a;
spe[alpha_] ** (a_Plus) := (spe[alpha] ** #) & /@ a;
(c * spe[alpha_]) ** w_RSpW := Expand[c * (spe[alpha] ** w)];
spe[alpha_] ** (c * w_RSpW) := Expand[c * (spe[alpha] ** w)];
spe[alpha_] ** RSpW[] := RSpW[alpha];
spe[alpha_] ** RSpW[beta_, rest___] /; alpha ≤ beta := RSpW[alpha, beta, rest];
spe[alpha_] ** RSpW[beta_, rest___] /; alpha > beta := Plus[
  spe[beta] ** (spe[alpha] ** RSpW[rest]),
  b[spe[alpha], spe[beta]] ** RSpW[rest]
];

```

```
];
```

```
Clear[USp];
USp[diag_Diag] := Module[
  {allterms, s, LTAt, BDAt, BDdiagAt, EQAt,
   CAT, m1, m2, dirs, coeff, k, ltpairs, w, allots, tc = 0, i, j},
  allterms = List@@Expand[(Times @@ diag) /. ar[p_, q_] =>
    LTAt[p, q] + BDAt[p, q] + BDdiagAt[p, q] + 1/2 (EQAt[p, q] - CAT[p] + CAT[q] - CAT[])
  ];
  s = Sum[
    dirs = allterms[[m1]]; (* "dirs" is "directions" *)
    coeff = dirs /. (_CAT | _EQAt | _LTAt | _BDAt | _BDdiagAt) -> 1;
    k = 0;
    w = Table[1, {2 Length[diag]}];
    ltpairs = {};
    dirs /. {
      LTAt[p_, q_] => (
        w[[p]] = spe[1, i = ++k, j = ++k];
        w[[q]] = spe[1, j, i];
        AppendTo[ltpairs, {i, j}];
      ),
      BDAt[p_, q_] => (
        w[[p]] = spe[2, i = ++k, j = ++k];
        w[[q]] = spe[3, i, j];
        AppendTo[ltpairs, {i, j}];
      ),
      BDdiagAt[p_, q_] => (
        w[[p]] = spe[2, i = ++k, i];
        w[[q]] = spe[3, i, i];
        coeff *= 1/2;
      ),
      EQAt[p_, q_] => (w[[p]] = w[[q]] = spe[1, i = ++k, i]);
      CAT[p_] => (
        w[[p]] = spe[1, i = ++k, i];
        coeff *= h[i];
      ),
      CAT[]^p_ => Do[coeff *= h[++k]^2, {p}]
    ];
  w = DeleteCases[w, 1];
  allots = OrderTypes[k, ltpairs];
  Sum[
    ot = allots[[m2]];
  ]
];
```

```

++tc;
CanonicalForm[Expand[Times[
  n^Max[ot],
  coeff /. h[i_] => h[ot[[i]]],
  SpW@@w /. spe[t_, i_, j_] => spalpa[t, ot[[i]], ot[[j]]]
]],
  {m2, Length[allots]}
], {m1, Length[allterms]}
];
Print[{tc, Length[s]};
If[Length[diag] ≤ 3, USp[diag] = s, s]
];
USp[expr_] := Expand[expr /. diag_Diag => USp[diag]];

(*d for odd*)
sdbase = 10;
sdijs[sdalpha_Integer] := IntegerDigits[sdalpha, sdbase, 3];
sdalpha[{t_, i_, j_}] := sdbase^2*t + sdbase*i + j;
sdalpha[t_, i_, j_] := sdbase^2*t + sdbase*i + j;
b[sde[sdalpha1_], sde[sdalpha2_]] := b[sde[sdalpha1], sde[sdalpha2]] = Module[
  {d, t1, t2, i, j, k, l, s = 1},
  d[i_, j_] := If[i == j, 1, 0];
  {t1, i, j} = sdijs[sdalpha1];
  {t2, k, l} = sdijs[sdalpha2];
  If[t2 < t1, s = -1; {{t1, i, j}, {t2, k, l}} = {{t2, k, l}, {t1, i, j}}];
  {
    {
      d[j, k] sde[1, i, l] - d[i, l] sde[1, k, j],
      d[j, k] sde[2, i, l] - d[j, l] sde[2, i, k],
      d[i, k] sde[3, l, j] + d[l, i] sde[3, j, k],
      d[j, k] sde[4, i, i],
      -d[i, k] sde[5, j, j]
    },
    {
      Null,
      0,
      d[j, l] sde[1, i, k] + d[i, k] sde[1, j, l] - d[j, k] sde[1, i, l] - d[i, l] sde[1, j, k],
      0,
      -d[j, k] sde[4, i, i] + d[i, k] sde[4, j, j]
    },
    {
      Null,

```

```

Null,
0,
d[j, k] sde[5, i, i] - d[i, k] sde[5, j, j],
0
},
{
Null,
Null,
Null,
-sde[2, i, k],
sde[1, i, k]
},
{
Null,
Null,
Null,
Null,
sde[3, i, k]
}
}[[t1, t2]] /. {
sde[2 | 3, i_, i_] → 0,
sde[2, i_, j_] /; i > j ⇒ -sde[2, j, i],
sde[3, i_, j_] /; i > j ⇒ -sde[3, j, i]
} /. sde[tij_] ⇒ s * sde[sdalpha[tij]]
];

(a_Plus) ** w_RSdW := (#**w) & /@ a;
sde[alpha_] ** (a_Plus) := (sde[alpha] ** #) & /@ a;
(c_ * sde[alpha_]) ** w_RSdW := Expand[c * (sde[alpha] ** w)];
sde[alpha_] ** (c_ * w_RSdW) := Expand[c * (sde[alpha] ** w)];
sde[alpha_] ** RSdW[] := RSdW[alpha];
sde[alpha_] ** RSdW[beta_, rest___] /; alpha ≤ beta := RSdW[alpha, beta, rest];
sde[alpha_] ** RSdW[beta_, rest___] /; alpha > beta := Plus[
sde[beta] ** (sde[alpha] ** RSdW[rest]),
b[sde[alpha], sde[beta]] ** RSdW[rest]
];

Clear[USOd];
USOd[diag_Diag] := Module[
{allterms, s, LTAt, BDAt, EQAt, CAT, LAT,
m1, m2, dirs, coeff, k, ltpairs, w, allots, tc = 0, i, j},
allterms = List @@ Expand[(Times @@ diag) /. ar[p_, q_] ⇒
LTAt[p, q] + BDAt[p, q] + LAT[p, q] + 1/2 (EQAt[p, q] - CAT[p] + CAT[q] - CAT[])
];

```



```

];
s = Sum[
  dirs = allterms[[m1]]; (* "dirs" is "directions" *)
  coeff = dirs /. (_Cat | _EQat | _LTat | _BDat | _Lat) -> 1;
  k = 0;
  w = Table[1, {2 Length[diag]}];
  ltpairs = {};
  dirs /. {
    LTat[p_, q_] -> (
      w[[p]] = sde[1, i = ++k, j = ++k];
      w[[q]] = sde[1, j, i];
      AppendTo[ltpairs, {i, j}];
    ),
    BDat[p_, q_] -> (
      w[[p]] = sde[2, i = ++k, j = ++k];
      w[[q]] = sde[3, i, j];
      AppendTo[ltpairs, {i, j}];
    ),
    Lat[p_, q_] -> (
      w[[p]] = sde[4, i = ++k, i];
      w[[q]] = sde[5, i, i];
    ),
    EQat[p_, q_] -> (w[[p]] = w[[q]] = sde[1, i = ++k, i];),
    Cat[p_] -> (
      w[[p]] = sde[1, i = ++k, i];
      coeff *= h[i];
    ),
    Cat[]^p_. -> Do[coeff *= h[++k]^2, {p}]
  };
w = DeleteCases[w, 1];
allots = OrderTypes[k, ltpairs];
Sum[
  ot = allots[[m2]];
  ++tc;
  CanonicalForm[Expand[Times[
    n^Max[ot],
    coeff /. h[i_] -> h[ot[[i]]],
    sdw@@w /. sde[t_, i_, j_] -> sdalpha[t, ot[[i]], ot[[j]]]
  ]],
  {m2, Length[allots]}
], {m1, Length[allterms]}
];

```

```
Print[{tc, Length[s]}];  
If[Length[diag] ≤ 3, USOd[diag] = s, s]  
];  
USOd[expr_] := Expand[expr /. diag_Diag => USOd[diag]];
```