LittelmannPath package

A subpackage for QuantumGroups v2.
Version 2.0, June 18, 2005, Scott Morrison

Introduction

This package implements Littelmann paths, for calculating weight diagrams, and decomposing tensor products of irreps.

Implementation

```
BeginPackage["QuantumGroups`LittelmannPaths`",
  {"QuantumGroups`", "QuantumGroups`RootSystems`"}];
```

```
LittelmannPath::usage =
  "LittelmannPath[Γ][{λ1,λ2,λ3,...}] represents a Littelmann path
    in the weight lattice for Γ with segments λ1,λ2,λ3...";
```

```
LittelmannPathDecomposeRepresentation::usage =
  "LittelmannPathDecomposeRepresentation[Γ][Irrep[Γ][λ]⊗Irrep[Γ][μ]]
    gives the direct sum decomposition of Irrep[Γ][λ]⊗Irrep[Γ][μ]
    into irreducibles, using the Littelmann path model.";
```

```
LittelmannPathWeightMultiplicities::usage =
  "LittelmannPathWeightMultiplicities[Γ,Irrep[Γ][λ]] gives
    a list of pairs; each pair consists of a weight and its
    multiplicity in Irrep[Γ][λ], using the Littelmann path model.";
```

```
LittelmannPathOneStepLowerings;
LittelmannPathLowerings;
LittelmannPathEndpoint;
```

```
Begin["`Internals`"];
```

```
LittelmannPathVertices;
LittelmannPathInnerProducts;
LowerLittelmannPath;
ComposeLittelmannPaths;
LittelmannPathDominantQ;
```

```
End[];
```

```
Begin["`Private`"];
```

```
AppendTo[$ContextPath, "QuantumGroups`LittelmannPaths`Internals`"];
```

```
LittelmannPathAlgebra[LittelmannPath[Γ_][l_List]] := Γ
```

```
LittelmannPathEndpoint[LittelmannPath[Γ_][l_List]] := Plus @@ l
```

```
LittelmannPathVertices[LittelmannPath[Γ_n_][l_List]] := FoldList[Plus, ZeroVector[n], l]
```

```
RedefineLittelmannPathInnerProducts[] := (Clear[LittelmannPathInnerProducts];
  LittelmannPathInnerProducts[LittelmannPath[Γ_][l_List], i_Integer] :=
   LittelmannPathInnerProducts[LittelmannPath[Γ][l], i] =
    LittelmannPathVertices[LittelmannPath[Γ][l]].Inverse[Transpose[CartanMatrix[Γ]]].
     DiagonalMatrix[CartanFactors[Γ]].SimpleRoots[Γ][[i]])
```

```
RedefineLittelmannPathInnerProducts[];
```

```
LowerLittelmannPath[0, i_] := 0
LowerLittelmannPath[0, i_, m_] := 0
```

```
LowerLittelmannPath[lp_, i_] := LowerLittelmannPath[lp, i, 1]
```

```
VectorsPositivelyProportionalQ[v1_, v2_] := Simplify[(v1.v2)^2 - v1.v1 v2.v2] == 0 ∧ v1.v2 > 0
```

```
SimplifyLittelmannPath[LittelmannPath[Γ_][l_List]] := LittelmannPath[Γ][Plus @@ # & /@
    Split[DeleteCases[l, ZeroVector[Rank[Γ]]], VectorsPositivelyProportionalQ]]
```

```
LowerLittelmannPath[lp_, {}] := lp
LowerLittelmannPath[lp_, d_List] :=
 LowerLittelmannPath[LowerLittelmannPath[lp, Last[d]], Drop[d, -1]]
```

```
SimpleRootLength[Γ_][i_] :=
 SimpleRootLength[Γ][i] = 1/2 KillingForm[Γ][SimpleRoots[Γ][[i]], SimpleRoots[Γ][[i]]]
```

```
LowerLittelmannPath[lp_, i_, m_] := Module[{ip, min, lm, ms, l, v, Γ, α, v1, v2},
  ip = LittelmannPathInnerProducts[lp, i];
  min = Min[ip];
  lm = Last[Position[ip, min]][[1]];
  If[lm > Length[lp[[1]]], Return[0]];
  Γ = LittelmannPathAlgebra[lp];
  α = SimpleRootLength[Γ][i];
  ms = (Min[Cases[Drop[ip, lm], _?(# < α &)]] - min)/α;
  If[ms < m, Return[LowerLittelmannPath[LowerLittelmannPath[lp, i, ms], i, m - ms]]];
  If[(l = (ip[[lm + 1]] - ip[[lm]])/α) < m,
   Return[LowerLittelmannPath[LowerLittelmannPath[lp, i, l], i, m - l]]];
  v = lp[[1, lm]];
  v1 = Simplify[m/l v - m SimpleRoots[Γ][[i]]];
  v2 = Simplify[(l - m)/l v];
  SimplifyLittelmannPath[
   LittelmannPath[Γ][Take[lp[[1]], lm - 1] ~Join~ {v1, v2} ~Join~ Drop[lp[[1]], lm]]
 ]
```

```
LittelmannPathOneStepLowerings[{}] = {};
LittelmannPathOneStepLowerings[paths_List] :=
 Module[{n = Rank[LittelmannPathAlgebra[paths[[1]]]]},
  UnsortedUnion[DeleteCases[
    Flatten[Table[LowerLittelmannPath[#, i] & /@ paths, {i, 1, n}]], 0]]
 ]
```

```
LittelmannPathLowerings[paths_List] := Module[{result}, result =
   UnsortedUnion[Flatten[FixedPointList[LittelmannPathOneStepLowerings, paths]]];
  RedefineLittelmannPathInnerProducts[];
  result
 ]
```

```
LittelmannPathLowerings[lp_] := LittelmannPathLowerings[{lp}]
```

```
LittelmannPathLowerings[Irrep[Γ_][λ_]] := LittelmannPathLowerings[Irrep[Γ][λ]] =
  LittelmannPathLowerings[LittelmannPath[Γ][{λ}]]
```

```
LittelmannPathWeightMultiplicities[Γ_, Irrep[Γ_][λ_]] := {#[[1]], Length[#]} & /@ Split[
   SortWeights[Γ][LittelmannPathEndpoint /@ LittelmannPathLowerings[Irrep[Γ][λ]]]]
```

```
ComposeLittelmannPaths[LittelmannPath[Γ_][l1_], LittelmannPath[Γ_][l2_]] :=
 LittelmannPath[Γ][l1 ~Join~ l2]
```

```
LittelmannPathDominantQ[lp_] :=
 And @@ (PositiveWeightQ[LittelmannPathAlgebra[lp]] /@ LittelmannPathVertices[lp])
```

```
LittelmannPathDecomposeRepresentation[Γ_][Irrep[Γ_][λ_] ⊗ Irrep[Γ_][μ_]] :=
 Module[{lp, compositions},
  lp = LittelmannPath[Γ][{λ}];
  compositions =
   ComposeLittelmannPaths[lp, #] & /@ LittelmannPathLowerings[Irrep[Γ][μ]];
  DirectSum @@ SortWeights[Γ][Cases[compositions,
     lp_ ? LittelmannPathDominantQ :> Irrep[Γ][LittelmannPathEndpoint[lp]]]]
 ]
```


```
LittelmannPathDecomposeRepresentation[Γ_][Irrep[Γ_][λ_] ⊗ Irrep[Γ_][μ_]] :=
 Module[{lp, compositions},
  lp = LittelmannPath[Γ][{λ}];
  compositions =
   ComposeLittelmannPaths[lp, #] & /@ LittelmannPathLowerings[Irrep[Γ][μ]];
  DirectSum @@ SortWeights[Γ][Cases[compositions,
     lp_ ? LittelmannPathDominantQ :> Irrep[Γ][LittelmannPathEndpoint[lp]]]]
 ]
```


```
End[];
```

```
EndPackage[];
```