Representations package

A subpackage for QuantumGroups v2.
Version 2.0, July 30, 2005, Scott Morrison

Introduction

This package implements a weight multiplicity formula and a tensor product decomposition algorithm (the underlying implementation is by Littelmann paths, but you don't need to know that, if you don't care!), and can calculate the weight multiplicities of tensor products and direct sums.

Implementation

```
BeginPackage["QuantumGroups`Representations`",
  {"QuantumGroups`", "QuantumGroups`RootSystems`", "QuantumGroups`Algebra`",
   "QuantumGroups`WeylGroups`", "QuantumGroups`LittelmannPaths`",
   "QuantumGroups`Steinberg`", "QuantumGroups`Utilities`Debugging`"}];
```

```
WeightMultiplicities::usage = "";
```

```
WeightMultiplicity::usage = "";
```

```
DecomposeRepresentation::usage = "";
```

```
Weights::usage = "";
```

```
WeightDiameter::usage = "";
```

```
PositiveWeights::usage = "";
```

```
qDimension::usage = "";
```

```
MultiplicityFreeQ::usage =
  "MultiplicityFreeQ[Γ,V] determines whether every weight space in the
    representation V of Γ has dimension 1. In the case that this is false,
    MultiplicityFreeQ is much faster than WeightMultiplicities[Γ,V].";
```

```
DualRepresentation;
```

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

```
DualRepresentation[X_DirectSum] := DualRepresentation /@ X
DualRepresentation[X_CircleTimes] := DualRepresentation /@ (Reverse[X])
DualRepresentation[Irrep[Γ_][V_]] := Irrep[Γ][-LongestWord[Γ].V]
```

```
q = Global`q;
```

Weight and tensor product multiplicities, using the Littelmann path model.

```
WeightMultiplicities[Γ_, ℂ] := {{ZeroVector[Rank[Γ]], 1}}
```

```
WeightMultiplicities[Γ_, CircleTimes[V_]] := WeightMultiplicities[Γ, V]
```

Use the functions provided by the LittelmannPath package.

```
WeightMultiplicities[Γ_, Irrep[Γ_][λ_]] := WeightMultiplicities[Γ, Irrep[Γ][λ]] =
  LittelmannPathWeightMultiplicities[Γ, Irrep[Γ][λ]]
```

```
GreatestMultiplicityAfterLowerings[Γ_, λ_, k_] :=
 Max[Last /@ Tally[LittelmannPathEndpoint /@
     Nest[LittelmannPathOneStepLowerings, {LittelmannPath[Γ][{λ}]}, k]]]
```

```
MultiplicityFreeQ[Γ_, Irrep[Γ_][λ_]] := Module[{m, k = 1},
  While[(m = GreatestMultiplicityAfterLowerings[Γ, λ, k]) == 1, ++k];
  m == -∞
 ]
```

```
DecomposeRepresentation[Γ_][Irrep[Γ_][λ_] ⊗ Irrep[Γ_][μ_]] :=
 DecomposeRepresentation[Γ][Irrep[Γ][λ] ⊗ Irrep[Γ][μ]] =
  LittelmannPathDecomposeRepresentation[Γ][Irrep[Γ][λ] ⊗ Irrep[Γ][μ]]
```

```
DecomposeRepresentation[Γ_, l_][Irrep[Γ_][λ_] ⊗ Irrep[Γ_][μ_]] :=
 SteinbergDecomposeRepresentation[Γ, l][Irrep[Γ][λ] ⊗ Irrep[Γ][μ]]
```

```
Weights[Γ_, V_] := Weights[Γ, V] = First /@ WeightMultiplicities[Γ, V]
```

```
WeightMultiplicity[Γ_, V_, μ_] :=
 WeightMultiplicity[Γ, V, μ] = Plus @@ Cases[WeightMultiplicities[Γ, V], {μ, n_} :> n]
```

```
WeightDiameter[Γ_, V_] :=
 WeightDiameter[Γ, V] = 2 Max[√(KillingForm[Γ][#, #] & /@ Weights[Γ, V])]
```

```
DecomposeRepresentation[Γ_][Irrep[Γ_][λ_]] := Irrep[Γ][λ]
```

```
DecomposeRepresentation[Γ_][V_DirectSum] :=
 SortWeights[Γ][DecomposeRepresentation[Γ] /@ V]
```

```
DecomposeRepresentation[Γ_][V_DirectSum⊗W_] :=
 SortWeights[Γ][DecomposeRepresentation[Γ][Distribute[V⊗W, DirectSum]]]
```

```
DecomposeRepresentation[Γ_][V_⊗W_DirectSum] :=
 SortWeights[Γ][DecomposeRepresentation[Γ][Distribute[V⊗W, DirectSum]]]
```

```
DecomposeRepresentation[Γ_][(U_⊗V_)⊗W__] :=
 SortWeights[Γ][DecomposeRepresentation[Γ][DecomposeRepresentation[Γ][U⊗V]⊗W]]
```

```
DecomposeRepresentation[Γ_][U_⊗(V_⊗W__)] :=
 SortWeights[Γ][DecomposeRepresentation[Γ][U⊗DecomposeRepresentation[Γ][V⊗W]]]
```

Weights and weight multiplicities for repesentations

```
HighestWeight[Γ_, Irrep[Γ_][λ_]] := λ
```

```
WeightMultiplicities[Γ_, V_] /; MinisculeRepresentationQ[Γ, V] :=
 WeightMultiplicities[Γ, V] =
  {#, 1} & /@ SortWeights[Γ][WeylOrbit[Γ, HighestWeight[Γ, V]]]
```

```
WeightMultiplicities[Γ_, Irrep[Γ_][μ_]] /; ZeroVectorQ[μ] := {{ZeroVector[Rank[Γ]], 1}}
```

```
TensorWeightMultiplicities[{λ_, a_}, {μ_, b_}] := {λ + μ, a b}
```

```
CombineWeightMultiplicities[Γ_, l_] :=
 {#[[1, 1]], Plus @@ (Last /@ #)} & /@ Split[SortWeightMultiplicities[Γ][l], #1[[1]] == #2[[1]] &]
```

```
WeightMultiplicities[Γ_, CircleTimes[V_, W_]] := WeightMultiplicities[Γ, V⊗W] =
  CombineWeightMultiplicities[Γ, Flatten[Outer[TensorWeightMultiplicities,
     WeightMultiplicities[Γ, W], WeightMultiplicities[Γ, V], 1], 1]]
```

```
WeightMultiplicities[Γ_, V_DirectSum] := WeightMultiplicities[Γ, V] =
  CombineWeightMultiplicities[Γ, Join @@ (WeightMultiplicities[Γ, #] & /@ (List @@ V))]
```

```
WeightMultiplicity[Γ_, V_DirectSum, λ_] :=
 Plus @@ (WeightMultiplicity[Γ, #, λ] & /@ (List @@ V))
```

```
PositiveWeights[Γ_, V_] :=
 PositiveWeights[Γ, V] = Select[Weights[Γ, V], PositiveWeightQ[Γ]]
```

```
qWeylDimension[Γ_][Irrep[Γ_][λ_]] := qWeylDimension[Γ][Irrep[Γ][λ]] = Expand[
    Together[Product[qInteger[KillingForm[Γ][α, λ + ρ[Γ]]][q] / qInteger[KillingForm[Γ][α, ρ[Γ]]][q], {α, PositiveRoots[Γ]}]]]
```

```
qDimension[Γ_][V_DirectSum] := Plus @@ (qDimension[Γ] /@ (List @@ V))
qDimension[Γ_][V_TensorProduct] := Times @@ (qDimension[Γ] /@ (List @@ V))
qDimension[Γ_][Irrep[Γ_][λ_]] := qWeylDimension[Γ][Irrep[Γ][λ]]
```

```
(*qDimension[Γ_][V_]:=qDimension[Γ][V]=With[{ρ=Table[1,{Rank[Γ]}]},
    Plus@@(WeightMultiplicities[Γ,V]/.{λ_,n_Integer}:>n q^KillingForm[Γ][2ρ,λ])]*)
```

```
End[];
```

```
EndPackage[];
```