Decomposing restrictions and tensor products of Lie algebra representations using crystal graphs

David Joyner and Roland Martin

September 4, 2003

Abstract

We use the theory of crystal graphs to give a simple graph-theoretical algorithm for determining the branching rule for decomposing a representation of a simple Lie algebra when restricted to a simple subalgebra. We also describe a computer package for determining such decompositions graphically.

1 Introduction

When modeling elementary particle interactions and symmetry breaking in physics it becomes important to understand how tensor products of representations of simple Lie algebras decompose into irreducible subrepresentations and how irreducible representations decompose when restricted to simple subalgebras. The classification of irreducible finite dimensional representations of simple Lie algebras is well-understood ([1]). In principal, this classification yields straightforward, if cumbersome, algorithms for decomposing

1. tensor products of such representations and
2. representations when restricted to simple subalgebras.

The purpose of this paper is to present simpler algorithms for solving these problems using crystal graphs ([4]). We, further, describe an implementation of these algorithms in a Maple package ([2],[3]).

2 Crystal Graphs

Let $\mathfrak{g}$ be a simple Lie algebra with highest weight λ. Let π denote a finite dimensional representation of $\mathfrak{g}$.

DEFINITION 2.1: The *crystal graph* of the representation $\pi(\mathfrak{g})$ is a colored digraph $\Gamma = (V, E)$, where

1. the vertex set V is given by the partially ordered set of all weights of $\mathfrak{g}$ oriented from left to right beginning with λ and

2. the edge set E is defined by the condition that $v \in V$ is connected to $v' \in V$ if and only if there is a simple root vector X of $\mathfrak{g}$ such that $\pi(X)v = v'$. In this case, we label or "color" this edge by X (or by the simple root associated to X).

Note that Γ is unique, $\mathrm{card}(V) = \dim(\mathfrak{g})$ and that, if Γ is connected, π is an irreducible representation of $\mathfrak{g}$ ([4]). For example, in the case $\mathfrak{g} = \mathfrak{sl}_2$ and π equal the identity representation, Γ is given by

.

Let π' denote a second finite dimensional representation of $\mathfrak{g}$ with highest weight λ' and Γ_2 its associated crystal graph.

ALGORITHM 2.1: The **crystal graph product** $\Gamma_1 \times \Gamma_2$ of these two crystal graphs is formed by the following rules.

1. Label the vertices on the vertical side on the left by Γ_1, with the bottom vertex the highest weight and the top vertex the lowest weight.

2. Label the vertices on the horizontal side on the top by Γ_2, with the left-most vertex the highest weight and the right-most vertex the lowest weight.

3. Let $i = 1$.

4. Starting from the bottom vertex of the $(i+1) - st$ column and moving up, draw an arrow up to the vertex weighted as in the left-most column in the $r - th$ vertex up $(r > 0)$ provided

   (a) there is such an arrow on the side graph Γ_1.

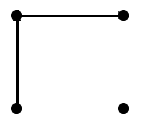
   (b) no "branch" is created with the same labels.

5. Starting from the bottom vertex of the $i - th$ column and moving up, draw an arrow over to the vertex weighted as in the top row in the $(i+s) - st$ column $(s > 0)$ provided

   (a) there is such an arrow on the top graph Γ_2,

(b) no "branch" is created with the same labels, i.e., connect vertices (v_1, v_3) and (v_2, v_3) with a labeled edge if $v_1, v_2 \in Vertices(\Gamma_1)$ are connected with an edge of that label and there does not already exist an edge of that label with head (v_2, v_3) and connect the vertices (v, v_1) and (v, v_2) with a labeled edge if $v_1, v_2 \in Vertices(\Gamma_2)$ are connected with an edge of that label and there does not already exist an edge of that label with head (v, v_2).

It is a theorem of Kashiwara ([4]) that the crystal graph product $\Gamma_1 \times \Gamma_2$ is the crystal graph associated to the tensor product representation $\pi \otimes \pi^{'}(\mathfrak{g})$. The irreducible representations of $\mathfrak{g}$ associated to the connected components of $\Gamma_1 \times \Gamma_2$ give the decomposition of $\pi \otimes \pi^{'}(\mathfrak{g})$. For example, in the case where Γ is the crystal graph associated to the identity representation of $\mathfrak{sl}_2$, $\Gamma \times \Gamma$ decomposes as



,

implying $\pi \otimes \pi(\mathfrak{g})$ decomposes into a direct sum of a trivial one dimensional representation and a three dimensional representation. Note that the three node connected component of $\Gamma \times \Gamma$ contains the vertex (λ, λ). One concludes that this three node connected component is the crystal graph of the Cartan product $\pi * \pi(\mathfrak{g})$ (i.e., the irreducible component of $\pi \otimes \pi$ containing the highest weight vector (λ, λ)) by canonically identifying (λ, λ) with the weight 2λ and by using the uniqueness property of crystal graphs. This result agrees exactly with the classical Clebsch-Gordon formula ([1]).

We remark that these definitions are essentially algorithms and are implemented in the Maple package **crystal** ([2],[3]).

3 The Branching Algorithm

We continue with the notation of §2.

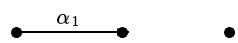
DEFINITION 3.1: A subgraph $\gamma \subset \Gamma$ is called a *branching graph* if γ is obtained from Γ by deleting all edges with labels in a subset of the set of simple roots.

Let $\mathfrak{h} \subset \mathfrak{g}$ be a subalgebra. The decomposition of $\pi(\mathfrak{g})$ when restricted to $\mathfrak{h}$ is called a branching rule. Recall that any simple Lie algebra has a Dynkin diagram labeled by its simple roots ([1]).

ALGORITHM 3.1: The *branching rule* for $\pi(\mathfrak{g})$ with respect to $\mathfrak{h}$ is determined by the branching graph γ obtained from Γ by deleting all edges labeled with simple roots not contained in the Dynkin diagram of $\mathfrak{h}$.

As an example, consider the identity representation of $\mathfrak{sl}_3$ and the subalgebra $\mathfrak{sl}_2 \subset \mathfrak{sl}_3$. Denote by α_1, α_2 the simple roots of $\mathfrak{sl}_3$. The associated crystal graph Γ of $\mathfrak{sl}_3$ is, then,

An examination of the Dynkin diagrams for $\mathfrak{sl}_2$ and $\mathfrak{sl}_3$ implies that the branching rule for $\mathfrak{sl}_3$ with respect to $\mathfrak{sl}_2$ is determined by the branching graph



obtained from Γ by deleting the edge labeled by α_2. In particular, $\mathfrak{sl}_3$ decomposes into a direct sum of $\mathfrak{sl}_2$ and a trivial, one dimensional representation when restricted to $\mathfrak{sl}_2$.

To check our result, recall that every representation π of $\mathfrak{sl}_n$ is obtained from n fundamental representations by means of the Cartan product ([1]). It follows that a representation π of $\mathfrak{sl}_3$ is uniquely determined by a triplet of non-negative integers (a, b, c). The identity representation of $\mathfrak{sl}_3$ is determined by $(1, 0, 0)$. The classical formula for the branching rule of $\mathfrak{sl}_3$ with respect to $\mathfrak{sl}_2$ is

$$\mathrm{Res}^{\mathfrak{sl}_3}_{\mathfrak{sl}_2}((1,0,0)) = \oplus(a', b')$$

where (a', b') determines a unique representation of $\mathfrak{sl}_2$ with $1 \geq a' \geq 0 \geq b' \geq 0$ ([1]). Since the identity representation of $\mathfrak{sl}_2$ is determined by $(1, 0)$ our result checks immediately with the classical formula.

We again remark that this algorithm has been implemented in the Maple package **crystal** [3]. For a proof of its validity, see [2].

4 The crystal Package

The **crystal** package ([3]) contains programs to compute the crystal graph of a multiplicity free, irreducible representation associated to the highest weight of a fundamental representation, compute the crystal graph product of two crystal graphs, compute branching graphs and display crystal graphs. The **crystal** package is, by design, compatible with the **coxeter** and **weyl** packages of John Stembridge ([5]), and implicitly uses both of those packages.

The utility of this package is further enhanced by existing Maple commands. For example, $\mathfrak{so}_{10} \otimes \mathfrak{so}_{10}$ can be shown to decompose into a direct sum of a 54 dimensional representation, a 45 dimensional representation and a trivial representation by computing the crystal graph of $\mathfrak{so}_{10}$, computing

the crystal graph product of $\mathfrak{so}_{10}$ with itself, and using the existing **networks**[components] and **networks**[vertices] commands.

The **crystal** package also provides the means of quickly deriving branching rules such as

$$\mathrm{Res}^{\mathfrak{sl}_{n+m}}_{\mathfrak{sl}_n}((1,0,\ldots,0)) = \oplus(a_1^{'},\ldots,a_n^{'})$$

where $1 \geq a_1^{'} \geq 0 \geq a_2^{'} \geq \ldots \geq 0$.

4.1 Example

The following commands show how to compute the triple tensor product of the identity representation of $\mathfrak{su}_3$ with itself.

```
with(share);with(plots):
readshare(coxeter,algebra);
read(`d:/maplev4/share/algebra/crystal/crystal25.mpl`);
init_crystal():
weyl[weights](A2);
L1:=crystal[weight_system](-(1/3*e2-2/3*e1+1/3*e3),A2);
weyl[weyl_dim](1/3*e2-2/3*e1+1/3*e3,A2);
L2:=crystal[weight_system](-(1/3*e2-2/3*e1+1/3*e3),A2);
crystal[graphrep](L1,A2,G1);
crystal[showgraph](G1,1,4);
crystal[graphrep](L2,A2,G2);
crystal[showgraph](G2,1,4);
crystal[graphprodrep](G1,G2,G3);
crystal[showgraphprod](G3,1,3,1,3);
crystal[linsubgraphrep](2*e1,A2,G3,G4);
crystal[showgraph](G4,1,6);
crystal[branch]([2],G4,G5);
crystal[showgraph](G5,1,6);
crystal[linsubgraphrep](v1X2,A2,G3,G6);
crystal[graphprodrep](G1,G6,G7);
crystal[showgraphprod](G7,1,3,1,3);
crystal[graphprodrep](G1,G4,G8);
crystal[showgraphprod](G8,1,3,1,6);
```

The corresponding MAPLE worksheet and crystal package **crystal25.mpl** with further documentation may be downloaded from the site [3].

Some of the graphs may be viewed at the web page

`http://web.usna.navy.mil/~wdj/crystal2.htm`

For elementary background on the crystal package, see also

`http://web.usna.navy.mil/~wdj/crystal.htm`

5 References

[1] J. E. Humphreys, Introduction to Lie Algebras and Representation theory, Springer Verlag, New York, 1972.

[2] D. Joyner and R. Martin, *A Constructive Algorithm for Determining Branching Rules of Lie Group Representations*, in preparation.

[3] D. Joyner and R. Martin, the Maple package **crystal** available at

`http://web.usna.navy.mil/~wdj/papers.html`

[4] Masaki Kashiwara, *Crystalizing the q-analog of universal enveloping algebras*, Comm. Math. Physics, **133** (1990) pp. 249-260.

[5] John Stembridge, *A Maple package for root systems and finite Coxeter groups*, preprint, 1992.