MAGMA
SUPPLEMENTARY FUNCTIONS

SEAN ROSTAMI

Abstract.

Contents

1. Compute Pairing of Roots/Coroots (basic version) 2
2. Compute Pairing of Roots/Coroots (extended version) 3
3. Compute Barycenter of Fundamental Alcove 4
4. Compute Length of Dominant Translation 5
5. Check Whether Translation is Minuscule 6
6. Compute Length-Set of Element of Weyl Group 7
7. Interface with Fokko du Cloux’s Coxeter3 8
1. Compute Pairing of Roots/Coroots (basic version)

1.1. Notation. Let $\Phi \subset V$ be a root system, $W$ its Weyl group, $\Phi^\vee \subset V^*$ the coroot system of $\Phi$, $\alpha^\vee \in \Phi^\vee$ the coroot associated with $\alpha \in \Phi$, and $\langle -, - \rangle$ the canonical pairing on $V \times V^*$.

1.2. Purpose. For some reason, MAGMA does not offer a function to compute the natural pairing $\langle \alpha, \beta^\vee \rangle$ of $\alpha, \beta \in \Phi$ but does offer the function `CoxeterForm` to compute a $W$-invariant pairing $(\alpha|\beta)$. The formula relating the two pairings, $\langle \alpha, \beta^\vee \rangle = 2(\alpha|\beta)/(\beta|\beta)$ is well-known and simple. However, due to typecasting issues, it is surprisingly annoying to implement this formula “on the fly”.

1.3. Function. Here is a small function which implements the formula:

```magma
CanonicalPairing := function( RS, a, b )
  return (2*Matrix(a)*CoxeterForm(RS)*Transpose(Matrix(b)))"[1][1] / 
           (Matrix(b)*CoxeterForm(RS)*Transpose(Matrix(b)))"[1][1];
end function;
```

1.4. Input/Output. RS is a root system, i.e. a MAGMA object of type `RootSys`. a and b are elements of RS, i.e. coordinate tuples in the sequences output by MAGMA functions like `Roots` or `PositiveRoots`. The value returned is $(a, b^\vee)$.

1.5. Remarks.

---

1 Almost unbelievably, the MAGMA function `SimpleRoots` outputs a different type of object than `Roots` and `PositiveRoots`: a matrix whose rows are the coordinate tuples.
2. Compute Pairing of Roots/Coroots (extended version)

2.1. Notation. Let the notation be the same as the previous section.

2.2. Purpose. One defect of the previous function \texttt{ CanonicalPairing} is that it accepts only genuine roots and coroots. For example, it will not compute \( \langle \alpha, x \rangle \) for arbitrary \( x \in V^* \), which is a perfectly legitimate and frequently necessary thing to do. For example, one might want to compute \( \langle \alpha, \omega \rangle \) for \( \alpha \in \Phi \) and \( \omega \) a fundamental coweight.

2.3. Function. Here is a modification of \texttt{ CanonicalPairing} which allows this:

\begin{verbatim}
CanonicalPairingExtended := function( RS, a, x )
  return 2*( Matrix( Rationals(), 1, Rank( RS ), ElementToSequence( x ) )*DualCoxeterForm( RS )*Transpose( Matrix( Rationals(), 1, Rank( RS ), ElementToSequence( Coroot( RS, RootPosition( RS, a ) ) ) ) ) )[1,1] / ( Matrix( Rationals(), 1, Rank( RS ), ElementToSequence( Coroot( RS, RootPosition( RS, a ) ) ) ) )[1,1];
end function;
\end{verbatim}

2.4. Input/Output. \( RS \) and \( a \) are as in the previous section, but now \( x \) can be an arbitrary element of the “coroot space”. Formally, this means that \( x \) can be any element of the object produced by the MAGMA function \texttt{ CorootSpace}. Mathematically, the object produced by the MAGMA function \texttt{ CorootSpace} is the \( \mathbb{Q} \)-span of \( \Phi^\vee \).

2.5. Remarks.
3. Compute Barycenter of Fundamental Alcove

3.1. Notation. Let $\Phi \subset V$ be a root system, assumed to be reduced and irreducible, with Weyl group $W$. Let $A$ be the affine apartment, i.e. the affine space $V^*$ with hyperplane structure coming from $\Phi + \mathbb{Z}$ thought of as affine functionals on $V^*$. If $\Delta = \{\alpha_1, \ldots, \alpha_r\} \subset \Phi$ is a simple system and $\eta$ is the highest root then the fundamental alcove $A \subset A$ is bounded by the hyperplanes of $\alpha_0, \alpha_1, \ldots, \alpha_r$ (here, $\alpha_0 = 1 - \eta$).

3.2. Purpose. There is a unique point $b \in A$, called the “barycenter” of $A$, for which $\alpha_0(b) = \alpha_1(b) = \cdots = \alpha_r(b)$. Coordinates for $b$ are deduced from the following well-known identity: If $\eta = c_1 \alpha_1 + \cdots + c_r \alpha_r$ then $c_1 + \cdots + c_r = h - 1$ for $h$ the Coxeter Number of $W$. It is desired to compute $b$.

3.3. Function. The following function computes the barycenter:

```magma
BarycenterOfFundamentalAlcove := function( RD )
    RS := RootSystem( RD );
    if not ( IsIrreducible( RS ) and IsReduced( RS ) ) then
        return 0;
    end if;
    coweightbasis := FundamentalCoweight( RD );
    prebarycenter := Zero( CorootSpace( RD ) );
    for i := 1 to Rank( RS ) do
        prebarycenter += CorootSpace( RD ) ! coweightbasis[i];
    end for;
    return ( 1 / CoxeterNumber( CoxeterGroup( GrpPermCox, RS ) ) ) * prebarycenter;
end function;
```

3.4. Input/Output. RD is a root datum, i.e. a MAGMA object of type RootDtm. BarycenterOfFundamentalAlcove requires the root system of RD to be reduced and irreducible. The value returned is the barycenter of the fundamental alcove, which is formally an element of the object returned by the MAGMA function CorootSpace.

3.5. Remarks. RD is not required to be semisimple, i.e. it could be the root datum of a group like GL.

---

2As the body of the function suggests, combining the requirement on $b$ with the well-known identity yields that $b = \frac{1}{h} \sum \omega_i$, where $\omega_i$ are the fundamental coweights. Alternatively, if $\vartheta \overset{\text{def}}{=} \sum_{\alpha > 0} \alpha^\vee$ then $b = \frac{1}{h} \vartheta$. 

4. Compute Length of Dominant Translation

4.1. Notation.

4.2. Purpose.

4.3. Function. uses CanonicalPairingExtended from above

\begin{verbatim}
LengthOfDominantTranslation := function( RS, t )
    length := 0;
    for r in PositiveRoots( RS ) do
        length +:= CanonicalPairingExtended( RS, r, t );
    end for;
    return Integers() ! length;
end function;
\end{verbatim}

4.4. Input/Output.

4.5. Remarks. The return is typecast to be in \( \mathbb{Z} \). When used correctly, the value of \( \text{length} \) will indeed be in \( \mathbb{Z} \), but formally its type will be \( \mathbb{Q} \), which can cause the user trivial but annoying MAGMA errors later. I think that the optimal behavior is to return the expected type and so \texttt{LengthOfDominantTranslation} typecasts the return prior to return. It is the user’s responsibility to guarantee that \( t \) is indeed a dominant translation.
5. Check Whether Translation is Minuscule

5.1. Notation.

5.2. Purpose. to check whether a coweight is minuscule, i.e. has product one of -1, 0, 1 when paired with each root

5.3. Function. uses CanonicalPairingExtended from above

```plaintext
IsMinusculeTranslation := function( RS, t )
for r in PositiveRoots( RS ) do
if CanonicalPairingExtended( RS, r, t ) notin -1, 0, 1 then
    return false;
end if;
end for;
return true;
end function;
```

5.4. Input/Output. RS is a root system, i.e. a MAGMA object of type RootSys. t is an arbitrary element of the “coroot space”, thought of as a translation. Formally, this means that x can be any element of the object produced by the MAGMA function CorootSpace. Mathematically, the object produced by the MAGMA function CorootSpace is the Q-span of Φ∨. The return value is true if ⟨α, t⟩ ∈ {-1, 0, +1} for all α ∈ Φ and false otherwise.

5.5. Remarks. Strictly speaking, it is not required that t be an element of the coweight lattice or any other special subgroup of V∗–the test will be performed uniformly on all inputs.
6. Compute Length-Set of Element of Weyl Group

6.1. Notation.

6.2. Purpose.

```
ComputeLengthSet := function( RS, w )
P := PositiveRoots( RS );
list := [];
for a in P do
    if IsNegative( RS, RootPosition( RS, a )^w ) then
        Append(~list, a);
    end if;
end for;
return list;
end function;
```

6.3. Function.

6.4. Input/Output. \( w \) must be an element of the Weyl group of \( RS \) in “permutation format”, i.e. an element of the object returned by \texttt{CoxeterGroup( GrpPermCox, RS )} rather than that returned by \texttt{CoxeterGroup( GrpFP Cox, RS )}.

6.5. Remarks.
7. Interface with Fokko du Cloux’s Coxeter3

7.1. Notation.

7.2. Purpose.

7.3. Function.

7.4. Input/Output.

7.5. Remarks.