Computing Kazhdan-Lusztig polynomials for arbitrary Coxeter groups

Fokko du Cloux

January 15, 2002

Institut Girard Desargues
UMR 5028 CNRS
Université Lyon-I
69622 Villeurbanne Cedex FRANCE
`ducloux@desargues.univ-lyon1.fr`

1 Introduction

Let (W, S) be a Coxeter system, *i.e.*, a group W together with a presentation of the form $\langle S \mid (st)^{m_{s,t}} = e \ (s, t \in S)\rangle$, where S is a finite set which we shall usually just consider to be $\{1, \ldots, n\}$, e is the identity element in W, and $(m_{s,t})$ is a symmetric matrix with values in $\{1, 2, \ldots\} \cup \{\infty\}$, such that $m_{s,s} = 1$ for all $s \in S$, $m_{s,t} \geq 2$ for $s \neq t$; $m_{s,t} = \infty$ simply means that the corresponding relation is to be omitted. The cardinality of S is called the *rank* of the group; sometimes we omit S from the notation and simply say that W is a Coxeter group. We refer to [14] for general information about Coxeter groups. Examples of Coxeter groups are Weyl groups of finite-dimensional or Kač-Moody semisimple Lie algebras, and finite groups generated by reflections in Euclidian space; other examples may be realized as discrete groups generated by reflections in hyperbolic space.

In their seminal paper [15], Kazhdan and Lusztig have defined for each pair of elements (x, y) in W such that $x \leq y$ in the Bruhat ordering (to be defined below), a polynomial $P_{x,y} \in \mathbf{Z}[q]$. We will recall in section 4 the recursion formula which in principle leads to the computation of $P_{x,y}$. If W is the Weyl group of a finite-dimensional or Kač-Moody Lie algebra $\mathfrak{g}$, the Kazhdan-Lusztig polynomials of W hold the key to the representation theory of $\mathfrak{g}$, and also to the geometry of the corresponding Schubert varieties. In this case, it is known that the coefficients of $P_{x,y}$ are non-negative integers. For other W, very little is known about the $P_{x,y}$ (in particular, the positivity of their coefficients remains conjectural); they probably point to a yet-to-be-discovered geometry and/or representation theory.

Due to the fundamental importance of Kazhdan-Lusztig polynomials, and to the difficulty in computing them by hand except in very special cases, great efforts have been made to implement their calculation on a computer. As far as I know, previous attempts to achieve this (including my own) have always proceeded in three stages: *(a)* implement the group W, either by way of a linear representation, or combinatorially; *(b)* deal with the Bruhat ordering; *(c)* implement the actual recursion. In practice, the main burden of the computation falls on the Bruhat order routines; but in turn, the efficiency of these routines will depend on how well we have been able to solve stage *(a)*. In view of these difficulties, most computations I am aware of have been restricted to finite Coxeter groups (for the best programs, up to a group order of about half a million, say). The exceptions are the results posted by Mark Goresky [13] on his homepage, concerning the first few hundred elements of the affine Weyl groups associated to root systems of rank ≤ 3, and $\tilde{\mathrm{A}}_4$, and Bill Casselman's Kazhdan-Lusztig programs, which he has used to compute one- and two-sided cells in various finite, affine and hyperbolic Coxeter groups (see [4] for an application, and Casselman's homepage [3]). Goresky's files contain the necessary data for the description of the singularities of the Schubert variety associated to $y \in W$, for small y; basically this involves the computation of the element c_y in the Kazhdan-Lusztig basis (cf. section 4.2), and pruning the result a little to extract the irreducible components of the stratification defined by equality of polynomials (a rough version of equisingularity.)

In this paper, we shall present an algorithm which, for any Coxeter group W, constructs directly from the Coxeter matrix $(m_{s,t})$ and a word $a = (s_1, \ldots, s_p)$ in the generators, the interval $[e, y]$ in the Bruhat ordering, where $y = s_1 \ldots s_p$ is the element of W represented by a, together with the (partially defined) left and right actions of all the generators on $[e, y]$, without any prior implementation of the group operations. This provides us with exactly the data we need to compute $P_{x,z}$ for all $x \leq z \leq y$ using the recursion formula of Kazhdan and Lusztig. The algorithm is based on the analysis of the structure of Bruhat intervals and some other Bruhat-like posets in [9]; the essential ingredient in its correctness proof is a remarkable theorem due to Matthew Dyer in his thesis [10], which we shall recall in section 3.

We have made a preliminary implementation of this algorithm in a demonstration program available from the `Coxeter` homepage [7]. This program will compute the basis element c_y, and the data which would appear in Goresky's files, describing the singularity of the corresponding Schubert cell whenever this makes sense, for an element y of moderate length in an arbitrary Coxeter group W, of rank less than 16, say (more details on the scope of the program are given in section 6.) The performance limitations of the demonstration program are mainly due to the simple-minded routines used to access the Bruhat order within the Kazhdan-Lusztig computation. By using ideas from [8], we believe that its performance (regarding both speed and memory usage) could be improved considerably — we hope to include a full-fledged implementation in some future version of `Coxeter`.

2 Definition and elementary properties of the Bruhat ordering

2.1. For any poset P, and $x \leq y$ in P, we denote by $[x, y]$ the set of $z \in P$ such that $x \leq z \leq y$. Let us say that P is *locally finite*, if all intervals $[x, y]$ in P are finite; assume from now on that P is locally finite. A *chain* in P is a totally ordered subset; a chain is *maximal* if it is not properly contained in any other chain. Define the length of a chain to be its cardinality minus one. We say that P is *graded*, if for all $x \leq y \in P$, all maximal chains in $[x, y]$ have the same length; this is also sometimes called the Jordan-Dedekind condition. Let us assume that furthermore P has a smallest element $\mathbf{0}$; then we define the *length function* l on P by setting $l(x)$ to be the length of the maximal chains in $[\mathbf{0}, x]$. For $x \leq y$ in P, we also define the length of the interval $[x, y]$ to be the length of its maximal chains; it is easy to see that the length of $[x, y]$ is equal to $l(y) - l(x)$. The *atoms* of an interval $[x, y] \subset P$, $x < y$, are the $z \in [x, y]$ such that $l(z) = l(x) + 1$; similarly, the coatoms of $[x, y]$ are the $z \in [x, y]$ such that $l(z) = l(y) - 1$. For $x \in P$, we will speak about the coatoms of x instead of the coatoms of $[\mathbf{0}, x]$ and denote their set $\mathrm{coat}(x)$. A *decreasing subset* of P is a subset Q such that if $y \in Q$ and $x \leq y$, then $x \in Q$; since P has a smallest element $\mathbf{0}$, this means simply that Q is a union of intervals $[\mathbf{0}, y]$.

2.2. We denote S^* the free monoid generated by S, *i.e.*, the set of all words in the alphabet S. To avoid confusion between elements of S^* and elements of W, we will write a word $a \in S^*$ as $a = (s_1, \ldots, s_p)$; if $x = s_1 \ldots s_p$ is the corresponding element in W, we say that a is an *expression* for x. The smallest possible number of letters in an expression for x is called the *length* of x, and denoted $l(x)$; an expression for x is called *reduced*, if it has exactly

$l(x)$ letters. It is an elementary fact about Coxeter groups that for $x \in W$, $s \in S$, we have either $l(xs) = l(x) + 1$, in which case we write $xs > x$, or $l(xs) = l(x) - 1$, in which case we write $xs < x$, and of course we have an analogous statement for left multiplications.

2.3. The following proposition can be used to define the Bruhat ordering :

Proposition. — *There exists a unique ordering on W, which we call the Bruhat ordering, and denote $\leq$, such that* (a) $(W, \leq)$ *has smallest element* e (b) *for each $x \in W$, $s \in S$ such that $l(xs) < l(x)$, $[e, x]$ is the (non-necessarily disjoint) union of $[e, xs]$ and $[e, xs]s$.*

Proof. — Uniqueness is clear, since all intervals $[e, x]$ are uniquely defined by induction on the length of x, and the knowledge of these intervals is enough to determine the poset structure. For the existence, the main point is to show that the definition of $[e, x]$ makes sense, *i.e.*, that it is independent of the choice of s. This can be proved directly by an elementary argument using dihedral groups, but we will omit the proof, since it follows in any case from the usual definition of the Bruhat ordering (see the Proposition in [14], section 5.9.)

2.4. The Bruhat ordering satisfies the following properties:

- *(a)* The previous usage of $<$ in section 2.2 is compatible with Proposition 2.3: if $x \in W$ and $s \in S$ are such that $l(xs) < l(x)$, then $xs \in [e, x]$, hence $xs < x$ for the Bruhat order.
- *(b)* If $a = (s_1, \ldots, s_p)$ is any reduced expression for x, the interval $[e, x]$ is the set of all elements $z \in W$ of the form $z = s_{j_1} \ldots s_{j_q}$, where $1 \leq j_1 < \ldots < j_q \leq p$ (the elements corresponding to the so-called *subexpressions* of a.) This follows easily from *(b)* in proposition 2.3 by induction on the length of x; in particular, our definition of the Bruhat ordering is equivalent to the usual one (see for instance [14], section 5.9.)
- *(c)* $x \to x^{-1}$ is an automorphism of the Bruhat ordering (this follows from *(b)* above).
- *(d)* The Bruhat ordering also satisfies the property analogous to *(b)* in 2.3 for left multiplications (this follows from *(c)*).
- *(e)* If $x \in W$, $s \in S$ are such that $xs < x$, the interval $[e, x]$ is stable under right multiplication by s; this is clear from the equality $[e, x] = [e, xs] \cup [e, xs]s$. The analogous property holds for left multiplications.

2.5. Two other essential properties of the Bruhat ordering lie slightly deeper than the previous observations. The first one is that the Bruhat ordering is *graded* (see [14], section 5.11); moreover, the length function on W as a poset coincides with the length function previously defined for elements of W. The second one is that the poset W is *Eulerian*. This means that we have $\chi_{[x,y]} = 0$ for each $x < y$ in W, where $\chi_{[x,y]}$ is the "Euler characteristic" defined by

$$\chi_{[x,y]} = \sum_{x \leq z \leq y} (-1)^{l(z)-l(x)}$$

Equivalently, the Möbius function of W is given by $\mu(x,y) = (-1)^{(l(y)-l(x))}$ for all $x < y$ in W (see [16], sections 3.14 and 3.7, for more details on Eulerian posets and Möbius functions.) In this form, this was proved by Verma [17].

An elementary consequence is that all intervals $[x,y]$ for which $l(y) - l(x) = 2$ have exactly four elements: x, y, and exactly two intermediary elements z and z'. Also, it follows that for any $x < y$ in W, the cardinality of $[x,y]$ is *even*.

3 Dyer's theorem

3.1. A *dihedral* Coxeter group is simply a Coxeter group of rank 2, *i.e.*, for which the generating set S has two elements s, t. If $m = m_{s,t}$ is finite, then W is a finite group of order $2m$; if m is infinite, then W is the infinite dihedral group, isomorphic to the (non-trivial) semidirect product of $\mathbf{Z}$ with $\mathbf{Z}/2\mathbf{Z}$. The Bruhat ordering of a dihedral group is particularly easy to describe : there are exactly two elements in each length j such that $0 < j < m$, one in length 0, and one in length m if $m < \infty$; and all elements in length $j > 0$ are comparable to all elements in length $j - 1$. For example, if $m = 3$, we get the familiar picture of the Bruhat ordering on the Weyl group of type A_2 (also the symmetric group on three letters), shown in Figure 1.

We say that an interval $[x,y]$ of length > 1 in an arbitrary Coxeter group W is dihedral, if it is isomorphic as a poset to the Bruhat ordering on a (finite) dihedral group. Let us also make the convention that intervals of length 0 (one-element sets) and of length 1 (two-element sets) are dihedral. Then it is easy to see that any subinterval in a dihedral interval is again dihedral. We shall say that $y \in W$ is dihedral, if the interval $[e,y]$ is dihedral; equivalently, there exist s,t in S such that y belongs to the subgroup of W generated by s and t.

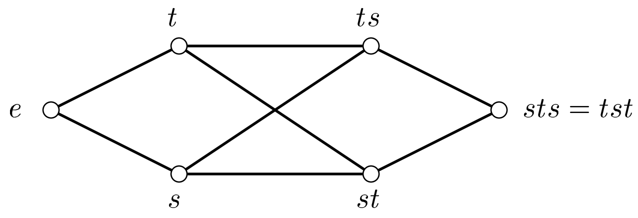

Figure 1: Bruhat ordering in type A_2

3.2. We wish to explain now a striking result due to Matthew Dyer (one of the many striking results contained in his thesis [10]), which imposes very important restrictions on the Bruhat ordering of a Coxeter group. For its proof (which we reproduce here since this result of Dyer's apparently has not been included in his publications) we first need to review some of his other results.

The *reflections* in W are defined to be the conjugates of the elements of S; so the set T of reflections is a finite union of conjugacy classes of elements of order 2 in W. The *canonical geometrical realisation* of W is defined as follows. Consider the real vector space $V = \mathbf{R}^S$, of dimension n, and its standard basis $(e_s)_{s\in S}$. Endow V with the unique symmetric bilinear form $\langle\ ,\ \rangle$ for which $\langle e_s, e_t\rangle = -\cos(\frac{\pi}{m_{s,t}})$. Then there is a unique injective homomorphism from W into the orthogonal group of V taking s to the orthogonal reflection with respect to the hyperplane $e_s^\perp$ ([14], section 5.3); this is the canonical geometrical realisation. We define the *roots* of W to be the conjugates under W of the basis elements e_s, and denote the set of roots by Φ. Since $s.e_s = -e_s$, we have $\Phi = -\Phi$. Reflections in W will correspond to orthogonal reflections with respect to elements of Φ; more precisely, if for each $t \in T$ we define its reflection line L_t to be the (-1)-eigenspace of the action of t in V, then $t \to L_t \cap \Phi$ is a bijection from T to pairs of opposite elements in Φ.

We will define a dihedral subgroup of W to be any subgroup generated by two distinct reflections. We will say that two dihedral subgroups are *commensurate*, if they are contained in a common dihedral subgroup. For any dihedral group D, it is easy to see that for any choice of reflections t_1, t_2 generating D, $D \cap T$ is exactly the set of elements of odd length with respect to the chosen generators, and also the set of conjugates of t_1 and t_2 in D. Hence all the reflection lines L_t, $t \in D \cap T$ lie in the two-dimensional subspace V_D of V spanned by L_{t_1} and L_{t_2}; this shows immediately that V_D does not depend on the choice of generators. If a dihedral subgroup is contained in another, it is clear that the corresponding two-dimensional subspaces are the same; hence the same is true for commensurate dihedral subgroups. The converse also holds, and follows from the following Lemma :

3.3 Lemma. ([10] Corollary 3.18) — *Each dihedral subgroup of W is contained in a unique maximal one.*

Proof. — Let α, β be two non-proportional elements of Φ, and let V_1 be the two-dimensional subspace of V spanned by α and β. Let $\Phi_1 = \Phi \cap V_1$, and let D be the subgroup of W generated by the reflections r_γ, $\gamma \in \Phi_1$, where for each unit vector $u \in V$, we denote by r_u the reflection $x \to x - 2\langle u, x\rangle u$. It will suffice to show that D is dihedral. If $\langle\ ,\ \rangle_1$ is the restriction of $\langle\ ,\ \rangle$ to V_1, there are three cases to consider : *(a)* $\langle\ ,\ \rangle_1$ is positive definite; *(b)* $\langle\ ,\ \rangle_1$ is non-zero positive degenerate; *(c)* $\langle\ ,\ \rangle_1$ has signature $(1, -1)$. In cases *(a)* and *(c)*, take $V_2 = V_1^\perp$; in case *(b)*, choose a subspace $V_2 \subset V_1^\perp$ such that $V = V_1 \oplus V_2$. Then in all cases, D acts trivially on V_2. So D is included in $\mathbf{O}(V_1) \times \{\mathrm{Id}_{V_2}\} \subset \mathbf{O}(V)$.

But it is well-known ([1], chapter V, n^o 4.4, Corollary 3) that W is a discrete subgroup of $\mathbf{O}(V)$; hence, D may be identified with a discrete subgroup of $\mathbf{O}(V_1)$. Moreover, as a Lie group $\mathbf{O}(V_1)$ always has the form $\mathbf{Z}_2 \ltimes A$, where A is isomorphic to $\mathbf{R}/\mathbf{Z}$ in case *(a)*, to $\mathbf{R}$ in case *(b)*, and to $\mathbf{R}^\times$ in case *(c)*, with $\mathbf{Z}_2$ acting by inversion; in case *(c)*, D is contained in $\mathbf{Z}_2 \ltimes \mathbf{R}_+^*$. Then from elementary topological considerations, one sees that in all three cases D is a dihedral subgroup of W, finite in case *(a)*, infinite in the two other cases.

3.4. In fact, Dyer [11] theorem 3.3 shows that if R is any (finite, say) subset of T, the subgroup W' of W generated by R is always a Coxeter group in its own right (this was proved independently by Deodhar in [6]). More precisely, he defines a canonical subset $S' \subset T \cap W'$ such that S' is a set of Coxeter generators for W'; he shows that $|S'| \leq |R|$, although it is certainly possible that $|S'| > |S|$. In particular, when W' is dihedral, S' is a two-element set. We shall call such a subgroup W' a *reflection subgroup* of W.

It is a well-known and useful fact in the theory of Coxeter groups ([14] section 5.12) that if I is any subset of S, and if W_I is the subgroup of W generated by I, then any (left, say) coset $W_I x$ of W_I in W contains a unique element of minimal length. This was extended in [11] corollary 3.4 to the case of an arbitrary reflection subgroup W'.

Let $x < y$ in W be such that $l(x) = l(y) - 1$. Then it is an easy consequence of 2.4 *(b)* that if $y = s_1 \dots s_p$ is a reduced decomposition, x may be obtained by erasing a single s_j from the decomposition (and in fact, j is unique.) This may also be expressed by saying that there exists a reflection $t \in T$ such that $x = yt$ (take $t = s_p \dots s_{j+1} s_j s_{j+1} \dots s_p$); so we have $y^{-1}x = x^{-1}y \in T$.

The following theorem regroups the main results of Dyer's study of the reflection subgroups arising from maximal chains in subintervals:

3.5 Theorem. ([12] Proposition 2.1) — *Let $x < y$ in W, and let $x = x_0 < x_1 < \dots < x_m = y$ be a maximal chain in $[x,y]$, so that $m = l(y) - l(x)$. For $1 \leq j \leq m$, set $t_j = x_{j-1}^{-1} x_j$, and let $W' \subset W$ be the subgroup generated by the t_j. Then*

(a) *W' is independent of the choice of the maximal chain.*

(b) *Let z be the unique element of minimal length in $W'x$; then $[x,y]$ is contained in the left coset $W'z$, and the map $u \to uz^{-1}$ defines a poset isomorphism from $[x,y]$ to $[x',y'] \subset W'$, where $x' = xz^{-1}$, $y' = yz^{-1}$, and $[x',y']$ is the interval in the Bruhat ordering defined by the canonical generating set S' of W'.*

3.6. The above theorem is the essential ingredient in the proof of the theorem on which our algorithm is based, and which may be stated as follows :

Theorem. ([10] Proposition 7.25) — *Let (W,S) be an arbitrary Coxeter system, and let $[x,y]$ be a Bruhat interval in W of length at least two. Then the following are equivalent :*

(i) *$[x,y]$ has two atoms;*

(ii) *$[x,y]$ has two coatoms;*

(iii) *$[x,y]$ is dihedral.*

Proof. — Of course *(iii)⇒(i)* and *(iii)⇒(ii)* are trivial. Let us prove, for instance that *(ii)⇒(iii)*. We argue by induction on $l(y) - l(x)$. If $l(y) - l(x)$ is two, there is nothing to prove. So we may assume that $l(y) - l(x)$ is at least three.

Let t, t' be the two reflections of W taking y to the two coatoms of $[x,y]$ (see 3.4). From Lemma 3.3, the dihedral subgroup $\langle t, t' \rangle$ is contained in a unique maximal dihedral subgroup D. From Theorem 3.5, it suffices to prove that the subgroup generated by all the $u^{-1}v$, $u < v$ in $[x,y]$, $l(u) = l(v) - 1$, is dihedral; and since any reflection subgroup of a dihedral group containing at least two reflections is again dihedral, it will suffice to show that $u^{-1}v \in D$ for all such u, v.

We argue by induction on $l(y) - l(u)$. If $l(y) - l(u) = 1$, we have $u^{-1}v \in \{t, t'\}$, so there is nothing to prove. Assume $l(y) - l(u) > 1$, and let z be an atom of $[v, y]$, so that $l(z) - l(u) = 2$. Write $[u, z] = \{u, v, v', z\}$, and let $t_1 = u^{-1}v$, $t_2 = v^{-1}z$, $t_2' = v'^{-1}z$. Then we know from 3.5 that the two dihedral subgroups $\langle t_1, t_2\rangle$ and $\langle t_2, t_2'\rangle$ are commensurate (in fact the latter is contained in the former); but from the inductive hypothesis, $\langle t_2, t_2'\rangle \subset D$; so $\langle t_1, t_2\rangle \subset D$ as well, and in particular $t_1 \in D$. The proof of *(i)⇒(iii)* is entirely similar.

3.7 Corollary. — *Let $y \in W$ be non-dihedral (see 3.1), and let $s \in S$. Then if $zs < s$ for all $z \in \mathrm{coat}(y)$ except at most one, we have $ys < y$; the analogous property holds for left multiplications.*

Proof. — We consider the case of right multiplications. If $y = e$, there is nothing to prove. So let $y > e$, and assume that $ys > y$. If we would have $zs < z$ for *all* $z \in \mathrm{coat}(y)$, then from 2.4 *(e)* the half-open interval $[e, y[$ would be stable under right multiplication by s; but this defines on $[e, y[$ an involution without fixed points, contradicting the fact that $|[e, y[|$ is odd, since $|[e, y]|$ is even from 2.5.

Hence there is a unique $z \in \mathrm{coat}(y)$ such that $zs > z$. Now any $x < y$ other than z satisfies $x \leq z'$ for some $z' \in \mathrm{coat}(y)$, $z' \neq z$: this is clear if $x \in \mathrm{coat}(y)$, and otherwise follows from the fact that $[x, y]$ has at least two coatoms if $l(y) - l(x) > 1$. Hence $[e, y] \setminus \{z, y\}$ is stable under right multiplication by s, and $[e, ys]$ contains exactly two elements not already in $[e, y]$, *viz.* zs and ys; in particular, $\mathrm{coat}(ys) = \{y, zs\}$, which from the Theorem implies that $[e, ys]$ is dihedral; but then $[e, y]$ is dihedral as well, contradicting our assumption on y.

4 Kazhdan-Lusztig polynomials

4.1. We refer to the original paper [15] or to [14], chapter 7, for the proper definition and proofs of the basic properties of the Kazhdan-Lusztig polynomials. Our goal here is to recall the recursion formulæ from [15] which we use to compute these polynomials, in order to assess the data which will be needed in the process.

Let q be an indeterminate. Then it is clear that there is at most one family $P_{x,y}$ of elements of $\mathbf{Z}[q]$, defined for $x \leq y$ in W, satisfying the following requirements :

(a) $P_{x,x} = 1$ for all $x \in W$;

(b) if $x < y$, and $s \in S$, are such that $ys < y$ and $xs > x$, $P_{x,y} = P_{xs,y}$;

(b') if $x < y$, and $s \in S$, are such that $sy < y$ and $sx > x$, $P_{x,y} = P_{sx,y}$;

(c) if $x < y$, and $s \in S$, are such that $ys < y$ and $xs < x$, and x is not comparable to ys, $P_{x,y} = P_{xs,ys}$;

(d) if $x < y$, and $s \in S$, are such that $ys < y$, $xs < x$, and $x \leq ys$, we have :

$$P_{x,y} = P_{xs,ys} + qP_{x,ys} - \sum_{\substack{x \leq z < ys \\ zs < z}} q^{\frac{1}{2}(l(y)-l(z))} \mu(z, ys) P_{x,z}$$

where $\mu(z,ys)$ is the coefficient of degree $\frac{1}{2}(l(ys) - l(z) - 1)$ in $P_{z,ys}$ (defined to be 0 if $l(ys) - l(z)$ is even); it is not hard to show by induction that in fact $P_{x,y}$ is at most of degree $\frac{1}{2}(l(y) - l(x) - 1)$ when $x < y$.

4.2. In fact, the $P_{x,y}$ are (up to a degree shift) the coordinates of the elements of a remarkable basis of the Hecke algebra of W, the so-called Kazhdan-Lusztig basis. There is one such basis element c_y for each y in W, and in many applications it is the knowledge of such a c_y which is required. In other words, a common requirement will be the computation of the $P_{x,y}$ for a fixed y, and all $x \leq y$. The whole recursion then takes place in the interval $[e, y]$, which is finite even if the group is infinite. We see that in order to carry out the recursion, we will need the following :

(a) an enumeration of the interval $[e, y]$, and a description of the Bruhat ordering on it;

(b) in this enumeration, the action of the generators $s \in S$ on the left and on the right.

Note that the action of each $s \in S$ on $[e, y]$ is only partially defined. In fact, it follows from 2.3 that it is everywhere defined (on the right, say) if and only if $ys < y$; in other words, if and only if s belongs to the so-called right descent set of y (see 4.3 below). Otherwise, xs remains within $[e, y]$ if and only if there exists $z \geq x$ in $[e, y]$ such that $zs < z$. We will say that this is the *domain* of the right action of s; it is a (possibly empty) decreasing subset of $[e, y]$.

4.3. For each $y \in W$ we define $L(y)$ (resp. $R(y)$) to be the set of $s \in S$ such that $sy < y$ (resp. $ys < y$); we set $LR(y) = L(y) \coprod R(y) \subset S \coprod S$. We say that $L(y)$ (resp. $R(y)$, $LR(y)$) is the left (resp. right, two-sided) descent set of y. These descent sets play an important role in the theory.

We notice that the knowledge of $LR(x)$ for each $x \leq y$ suffices to determine the left and right actions of all generators on $[e, y]$ (this remark could be used if a compact data encoding is required, but it is likely to be unpractical for systematic computations.) Indeed, encoding $LR(x)$ as a sequence of $2|S|$ bits in the obvious way, and looking at the bit position for, say, the right action of a generator s, we see that if the bit for x is set, meaning $xs < x$, there will be exactly one among the coatoms z of x for which the corresponding bit is *not* set; indeed, $[e, x]$ is stable under right multiplication by s, so $zs > z$ happens if and only if $zs = x$, hence $z = xs$ (these are precisely the remarks underlying the axiomatization in [9].)

4.4. Assuming we have solved *(a)* and *(b)* above, the only further (and obvious) idea used in our program is to remember the values of all the polynomials already computed. More precisely, assume that a polynomial $P_{x,z}$ is required, for $x \leq z \leq y$. The program first reduces to the case where $LR(x) \supset LR(z)$ (this amounts to putting ourselves in cases *(c)* or *(d)* of 4.1). Then it looks up a list of such x; the first time this occurs for a given z, it actually has to make the list, which involves extracting the subinterval $[e, z]$—we will come back to this problem, which is probably the most time-consuming part of the computation, in the next section. If the polynomial had already been computed, it will find it there; otherwise, it uses the recursion formula, potentially triggering many other computations, finds the requested $P_{x,z}$, and writes it down.

5 Description of the main algorithm

5.1. We shall now describe an algorithm which takes as input an arbitrary word over the alphabet S, and produces as output the poset $[e, y]$, and for each $x \in [e, y]$, the "shift-table" of x recording the result of the left and right actions of the generators on x. The only other datum that this algorithm needs is the Coxeter matrix of W, which is assumed to have been somehow read into memory. In other words, for s, t in $\{1, \ldots, n\}$, we may call a function $\texttt{Cox}(s, t)$, which will return $m_{s,t}$ (with some convention for representing ∞; in our program, it is represented by 0).

An enumeration of $[e, y]$ simply means an identification of $[e, y]$ with the numbers 0 to $N - 1$, where N is the cardinality of $[e, y]$. We view this as a function ν from $[e, y]$ to $[0, N[\subset \mathbf{N}$. We shall always require that the enumeration be *increasing*: *i.e.*, if $x < z$ in $[e, y]$, we want to have

$\nu(x) < \nu(z)$. Increasing enumerations exist for any finite poset. On the other hand, we do not insist that the enumeration be length-first; in other words, we do not require that $l(x) < l(z)$ implies $\nu(x) < \nu(z)$. It is always possible to do a length-sort if and when this becomes necessary (for instance, in order to have pleasant output.) In particular, the fact that ν is increasing implies that $\nu(e) = 0$.

In order to represent a graded poset, it is enough to give for each x in $[e, y]$ the list of coatoms of x; this will be empty if and only if $x = e$. To summarize, we wish to find the cardinality N of $[e, y]$, and construct the following data :

(a) for each $x \in [0, N[$, the list $\texttt{coat}[x]$ of the elements in $[0, x[$ which correspond to coatoms of x;

(b) for each $x \in [0, N[$, the length $\texttt{length}[x]$;

(c) for each $x \in [0, N[$, the shift table $\texttt{shift}[x]$; this is the datum of $2n$ elements of $[0, N[\cup\infty$, corresponding to the left and right action of the generators, where ∞ is a special value, indicating that the corresponding shift is undefined (in practice, it is convenient to take for ∞ a value which is larger than any legal value for x).

(d) for each $x \in [0, N[$, the descent set $\texttt{LR}[x]$; of course these are trivially deduced from the shift tables, and we have seen that the converse is also true; however, it is convenient to have them both available.

Notice that these data imply the enumeration of the poset : clearly, if all left shifts are known, it is a trivial matter to write down for each $x \in [0, N[$ a reduced expression for the corresponding group element, and indeed the `ShortLex` normal form (this is by definition the lexicographically smallest reduced expression, corresponding to the ordering of S implied by its identification with $\{1, \dots, n\}$.) Conversely, if we are given a reduced expression of an element of $[e, y]$, following the right shifts from the identity we find the corresponding $x \in [0, N[$ (in fact, this works for any expression, reduced or not, provided the path does not take us outside of $[e, y]$.) So we will leave ν implicit in the sequel.

At the beginning of the algorithm, the data are initialized to their values for $y = e$: we have $N = 1$, the coatom list of $x = 0$ is empty, the length of $x = 0$ is 0, the shifts are all set to ∞, and the descent set is empty. We shall describe in the next sections the main loop of the algorithm, passing from the data for $[e, y]$ to the data for $[e, ys]$, where s is a new letter in our input word. A very nice feature of the construction is that it fully preserves the data already constructed for $[e, y]$, at least if $ys > y$ (which should be the "hard" case, and always happens if the word is reduced.) So once we have determined the size of the bigger interval, we simply resize our lists and fill in the new part.

5.2. It is easy to find out how many new elements have to be added to our interval. Indeed, from section 2.4 *(e)* we see that for each $x \in [e, ys]$ we have $x \leq y$ or $xs \leq y$ or both; hence the new elements are the $x = x's$, where x' runs through the elements in $[e, y]$ for which $x's$ is not already in $[e, y]$, *i.e.*, for which $\texttt{shift}[x'][s]$ is undefined. From this we get the new value of N, and we can fill in the length function and the right shifts by s (note that right shift by s is everywhere defined on $[e, ys]$.)

Now we may construct the coatom lists of the new elements as follows ([9], section 2.9 and proposition 2.14.) Traverse the list of new elements in increasing order. For each x, let $x' = xs$; then the coatoms of x are x', and the $z = z's$, where z' runs through the coatoms of x' in $[e, y]$ for which $z's > z'$. Since the coatoms of x are all represented by strictly smaller integers, it is clear that the enumeration is indeed increasing.

5.3. It remains to explain how to find the shifts other than right multiplication by s. Let σ be such a shift, *i.e.*, σ is either right multiplication by some $t \neq s$, or left multiplication by any t. Before we start, all the $\sigma(x)$ for the newly created x are set to undefined. Let $\Delta \subset [e, y]$ be the previous domain of σ, and let $\Delta' \subset [e, ys]$ be the new one. Then we need to define σ on $\Delta' \setminus \Delta$; this will involve some of the newly created elements, and some already existing elements for which σ was previously undefined. But we notice that in fact $\Delta' \setminus \Delta = \coprod\{\sigma(x), x\}$, where x runs through the set of newly created elements such that $\sigma(x) < x$. So if for each newly created element x we are able to decide whether $\sigma(x) < x$ or $\sigma(x) > x$, we are done : if $\sigma(x) > x$ the corresponding shift is left undefined for the time being; if $\sigma(x) < x$, there is a unique coatom z of x such that $\sigma(z) > z$ (in fact, $\sigma(z)$ will have the value ∞ at this point), and we set $\sigma(x) = z$, $\sigma(z) = x$.

So again we traverse the list of newly created elements in increasing order. If $l(x) = 1$ (which can happen only if x is the first new element and s is a generator which had not occurred before), then $x = s$, and the only σ other than right shift by s that can take it down is left shift by s; so we may assume that $l(x) > 1$. We have already remarked that if $\sigma(x) < x$, there is a single coatom z of x such that $\sigma(z) > z$; so if there are at least two such coatoms, we may conclude without further ado that $\sigma(x) > x$. Otherwise, from the Theorem in 3.6, we can decide if x is dihedral or not by looking at the cardinality of $\texttt{coat}[x]$. If x is non-dihedral, we conclude from

Corollary 3.7 that $\sigma(x) < x$. If x is dihedral, it is easy to conclude directly : the other generator involved in x is the unique element t in $\mathtt{R}[xs]$; since we have assumed that $\sigma(z) < z$ for one of the two coatoms of x, σ is either left multiplication by s, or right or left multiplication by t. If $l(x) = m_{s,t}$, $\sigma(x) < x$; otherwise, $xt > x$, and $sx < x$, $tx > x$ if $l(x)$ is odd, $sx > x$, $tx < x$ if $l(x)$ is even (note that this is the *only* place in the whole algorithm where the Coxeter matrix comes in.) This concludes the main loop of the algorithm.

5.4. Each time the algorithm reads a new generator from its input word, we are in the situation where the data for the interval $[e, y]$ have been constructed, and we want the data for $[e, ys]$. We have assumed so far that $ys > y$ (note that from the shift tables for y, which are available when we read s, we can immediately determine whether or not this is the case.) If, on the contrary, $ys < y$, we are in the situation where we have to restrict to a subinterval $[e, ys]$ of the already constructed interval. There is a straightforward way of extracting the subinterval "backwards" from the knowledge of the coatom lists. In our program, however, we prefer to imitate the above procedure, using for instance the `ShortLex` normal form for ys, to extract $[e, ys]$ in the form of a list of elements in $[e, y]$. Then as the construction proceeds, we have to deal with the situation where we only enlarge a *subinterval* of our current poset, which can no longer be assumed to be an interval, but rather an arbitrary finite decreasing subset of W; in fact this causes no trouble at all, and can be handled exactly as above.

Note that there are efficient methods available to construct *a priori* the normal form of an arbitrary element in an arbitrary Coxeter group (see [5] for a nice exposition of the practical implementation of the ideas from Brink and Howlett in [2]). So the trouble caused by non-reduced expressions could in principle be avoided entirely. However, as we have seen in 4.4 it will in any case be necessary to extract many subintervals of the form $[e, x]$ in the course of the Kazhdan-Lusztig computations, so this problem has to be addressed one way or another.

6 Scope and further developments

6.1. We would like to conclude with a few informal remarks on the resources required by this algorithm. In our experience, given enough memory, time has never been a factor in Kazhdan-Lusztig computations. It only becomes a problem if we are unable to keep in memory the tables described in this article, or the polynomials already computed. Assuming for simplicity that everything is represented by 32-bit unsigned integers, the memory requirements for the table constructions are not hard to evaluate. The limiting factor is the size N of the actual poset $[e, y]$ we are constructing.

The sizes of the length, descent and shift tables are exactly N, N, and $2nN$, respectively (if we assume that the rank does not exceed 16, so that the descent sets can be represented by a single word.). The size of the coatom lists is harder to predict, but a rule-of-thumb for the most common cases, where there is enough commutativity around, is that $2nN$ is a reasonable estimate for the total number of coatoms (for groups like the free Coxeter group, where all the coefficients $m_{s,t}$ are infinite, the estimate would be much larger). On top of that, there is an unavoidable overhead of $2N$ elements, because we have to indicate somehow the number of elements in each coatom list, and we need a pointer to the beginning of each list. So we end up with an estimate of $4(n+1)N$ long integers, which seems to be pretty well validated by experience.

6.2. Of course, we will still need a sizeable amount of additional memory for the Kazhdan-Lusztig computations. In order to give an idea of how big this requirement might be, we print a table obtained for a few typical cases.

We define :

$$
\begin{aligned}
y_1 &= 213213243213234321 32 \in F_4 \\
y_2 &= 21213212132124321213212343212132123432121321234321213212 \in H_4 \\
y_3 &= (321212)^8.3.(212123)^8 \in \tilde{G}_2 \quad \text{(length 97)} \\
y_4 &= (12345)^8 \in \tilde{A}_4 \quad \text{(length 40)}
\end{aligned}
$$

The elements in F_4 and H_4 are (at least in our experience) among the worst possible ones : in fact they are elements of longest length with the property of having one-element left and right descent sets, which are moreover equal. The element chosen in $\tilde{G}_2$ also has this property. The element in $\tilde{A}_4$ has one-element left and right descent sets, but they are unequal.

Here we have used the usual Coxeter matrices for F_4, H_4, $\tilde{G}_2$, and $\tilde{A}_4$ respectively :

$$
\begin{pmatrix} 1 & 3 & 2 & 2 \\ 3 & 1 & 4 & 2 \\ 2 & 4 & 1 & 3 \\ 2 & 2 & 3 & 4 \end{pmatrix}
\qquad
\begin{pmatrix} 1 & 5 & 2 & 2 \\ 5 & 1 & 3 & 2 \\ 2 & 3 & 1 & 3 \\ 2 & 2 & 3 & 1 \end{pmatrix}
\qquad
\begin{pmatrix} 1 & 6 & 2 \\ 6 & 1 & 3 \\ 2 & 3 & 1 \end{pmatrix}
\qquad
\begin{pmatrix} 1 & 3 & 2 & 2 & 3 \\ 3 & 1 & 3 & 2 & 2 \\ 2 & 3 & 1 & 3 & 2 \\ 2 & 2 & 3 & 1 & 3 \\ 3 & 2 & 2 & 3 & 1 \end{pmatrix}
$$

	$y_1 \in F_4$	$y_2 \in H_4$	$y_3 \in \tilde{G}_2$	$y_4 \in \tilde{A}_4$
N	988	14 042	9 276	56 410
Hasse edges	5 244	98 357	53 925	496 734
memory cost for poset construction (bytes)	68 400	1 067 444	586 740	5 145 896
Kazhdan-Lusztig polynomials allocated	1383	379 991	572 003	2 221 661
Kazhdan-Lusztig polynomials computed	887	246 895	416 994	1 569 005
mu coefficients allocated	410	181 387	269 985	1 042 705
mu coefficients computed	146	107 148	191 284	619 255
memory cost for memory tables (bytes)	34 104	4 771 864	6 921 424	27 243 128
distinct polynomials found	135	67 864	190 860	23 946
$\sum \deg(P) + 2$	728	653 508	2 992 386	240 106
memory cost for polynomial storage (bytes)	3992	3 156 944	13 496 424	1 151 992
total cost for computation (bytes)	106 496	8 996 252	21 004 588	33 541 016

Table 1: Memory costs for some typical computations

The breakup of the corresponding memory costs has been collected in Table 1. A few explanatory comments may be in order regarding this table. The memory cost for the poset construction is computed as explained above (on a machine where pointes also have a size of 4 bytes.) The number of Hasse edges is in fact the total number of coatoms for the various lists `coat`$[x]$; notice that our estimate of $2nN$ is rather accurate in these examples.

As we have explained in 4.4, in order to record the polynomials already computed, we maintain an array of N rows, where row z holds the $P_{x,z}$ for the elements $x \leq z$ which are in "extremal position" with respect to z; a row is allocated if and when a $P_{x,z}$ is actually required. The allocation requires eight bytes for each polynomial : four to record the value of x, and four for a pointer to the actual polynomial, which is written down in full only once. The header of each row also requires eight bytes. In fact, we maintain another such table for recording μ-coefficients (when a μ-coefficient is required, we try as much as possible to compute it without unnecessarily computing full polynomials; also, μ-coefficients are essential information in many applications.) We allocate a μ-coefficient only if in addition to x being extremal with respect to z, the length difference is odd and ≥ 3. Again each allocation takes up eight bytes. Finally, we do not allocate the row for z if $z^{-1} < z$ in our enumeration; it is reasonably easy to deduce row z from row z^{-1}, if one maintains a (partially defined) table of inverses, at an additional cost of N words. The cost of the memory tables is the sum of the $4N$ words for the headers, the space for the allocation of the non-empty rows, and the space for the table of inverses.

When many distinct polynomials appear, the space used for recording them becauses an important part of the memory requirement (sometimes the dominant part.) The cost of storing them is $\deg(P) + 2$ words for each P, plus the cost of a searching structure to access the store efficiently (in our case, a hash table). The total cost of the computation is the sum of the costs for the poset, memory tables and polynomial storage.

6.3. The upshot of this analysis is that on a computer with 512 Mb of memory available, the computation of the basis vector c_y should go through for a size of $[e, y]$ of about 2^{18} to 2^{20}, if the rank is not bigger than 8, say (this will not be possible with the demonstration program, however; a more careful implementation of the Kazhdan-Lusztig computation will be needed.) In practice, this seems to correspond to elements of length around 40 or 50, unless the rank of the group is small, where lengths might get larger (but not much larger than 100, except in very easy cases.)

References

[1] N. Bourbaki. *Groupes et algèbres de Lie, chap. 4-6*. Herrmann, Paris, 1968.

[2] B. Brink and D. Howlett. A finiteness property and an automatic structure for Coxeter groups. *Math. Ann.*, **296**:179–190, 1993.

[3] W. Casselman. `http://www.math.ubc.ca/people/faculty/cass/cass.html`.

[4] W. Casselman. Verifying Kottwitz' conjecture by computer. *Representation Theory*, **4**:32–45, 2000.

[5] W. Casselman. Multiplication in Coxeter groups. preprint, 2001.

[6] V. Deodhar. A note on subgroups generated by reflections in Coxeter groups. *Arch. Math. (Basel)*, **53**(6):543–546, 1989.

[7] F. du Cloux. `Coxeter`, demonstration version. available from `http://www.desargues.univ-lyon1.fr/home/ducloux/coxeter.html`.

[8] F. du Cloux. A transducer approach to Coxeter groups. *J. Symbolic Computation*, **27**:311–324, 1999.

[9] F. du Cloux. An abstract model for Bruhat intervals. *Europ. J. Combinatorics*, **21**:197–222, 2000.

[10] M. Dyer. *Hecke algebras and reflections in Coxeter groups*. PhD thesis, University of Sydney, 1987.

[11] M. Dyer. Reflection subgroups of Coxeter systems. *J. Algebra*, **135**:57–73, 1990.

[12] M. Dyer. On the "Bruhat graph" of a Coxeter system. *Compositio Math.*, **78**:185–191, 1991.

[13] M. Goresky. Tables of Kazhdan-Lusztig polyomials. available from `http://www.math.ias.edu/~goresky`.

[14] J.E. Humphreys. *Reflection Groups and Coxeter Groups*. Cambridge University Press, Cambridge, UK, 1990.

[15] D. Kazhdan and G. Lusztig. Representations of Coxeter groups and Hecke algebras. *Invent. Math.*, **53**:165–184, 1979.

[16] R.P. Stanley. *Enumerative Combinatorics.* Cambridge University Press, Cambridge, UK, 1997.

[17] D.-N. Verma. Möbius inversion for the Bruhat ordering on a Weyl group. *Ann. Sci. Ec. Norm. Sup.*, **4**:393–398, 1971.