KL Fat C Algorithm

Tim Sprowl, tms7qy@virginia.edu

Terminology

Weight

We are interested in computing C_γ where γ is a certain "weight". A weight is a $(n-1)$-tuple composed of positive integers. For example $(0,0,0)$, $(1,0,1)$, and $(2,1,0)$ are all examples of weights (you can look at graph-4-4.film for more examples). Each weight has an associated length. For the previous examples, their weights are 6, 7, and 8 respectively. There is not an obvious relationship between a weight and its length. We can perform one of n operations to a weight to get a possibly different weight. If the resulting weight is different, then its length will increase or decrease. These operations are denoted as $s_1, ..., s_n$ or as $s_0, s_1, ..., s_{n-1}$ where $s_n = s_0$. The graph-n,n.film file uses the first notation, which is also the one used in the rest of this document. Each weight also has an associated weyl string, denoted as $w_0 w$. We write $\gamma = w_0 w \cdot -2\rho$ to formally state the relationship between the weyl string $w_0 w$ and γ.

Weyl String

A weyl string is a concatenation of the previous defined operations. One example is $w_0 = s_1 s_2 s_1 s_3 s_2 s_1$ and $w = s_4 s_1$, which together make $w_0 w = s_1 s_2 s_1 s_3 s_2 s_1 s_4 s_1$, the weyl string for $(2,1,0)$. There is an obvious relationship between a weyl string and its length. The length of the weyl string is the number of operations in the string, assuming that the string is "reduced". Thus, for the previous example, the string is known to be reduced and so the length is 7, and is denoted $\ell(w_0 w)$. All weyl strings for a given n begin with the same w_0. Note that it is not necessarily the case that $w_0 w s_i$ is a valid weyl string with length one greater than $w_0 w$ as it may not be reduced. In order to determine this, you will need to look at already determined relationships that are recorded in graph-n,n.film.

Graph File

If we look at graph-4,4.film, we see three sections. The first denotes the actual weight. The second denotes the length for that weight. The third denotes the relationship among the weights. The three sections fit together based on the line number in each section. That is, the first weight in the weight section has the first length in the length section and has the relationships shown on the first line of the relationship section. If we look at line 19, we see the following: 1112. This means that for weight $(0,0,0)$ applying s_4 (also denoted as s_0) gives us the second weight $(1,0,1)$. Applying s_1, s_2, or s_3 gives us the first weight $(0,0,0)$. If we wanted to know what applying s_4 to $(1,0,1)$ would do, we first observe that $(1,0,1)$ is the second line of the weight section, then we look at the second line of the relationship section. We see 3241. Since we are considering s_4, we look at the fourth column and see that we go back to the weight on line 1, which is $(0,0,0)$. Applying s_3 would have given us $(0,1,2)$.

T's

Weights and weyl strings are used to index T's. We write T_γ to denote that T has index γ. Different T's are indexed by different weights. The coefficient of a T is a polynomial in terms of t^2 such as $2t^8 + 6t^6 + 7t^4 + 5t^2 + 1t^0$ or in terms of t^{-1}. There are special rules for multiplying T's based off their respective weights. These are:

$$
\begin{aligned}
T_\nu T_{s_i} &= T_{\nu s_i} && (\ell(\nu s_i) > \ell(\nu)) \\
T_\nu T_{s_i} &= (t^2 - 1)T_\nu + t^2 T_{\nu s_i} && (\ell(\nu s_i) > \ell(\nu)) \\
T_\nu T_{s_i} &= t^2 T_\nu && (\ell(\nu s_i) = \ell(\nu))
\end{aligned}
$$

C's

C's are a linear combination of T's. For example, $C_{s_i} = (1/t)(1 + T_{s_i})$. Since C_{s_i} is just a linear combination of T's. We can modify the above rules for multiplication.

$$
\begin{aligned}
T_\nu C_{s_i} &= (1/t)(T_{\nu s_i} + T_\nu) && (\ell(\nu s_i) > \ell(\nu)) \\
T_\nu C_{s_i} &= t(T_{\nu s_i} + T_\nu) && (\ell(\nu s_i) > \ell(\nu)) \\
T_\nu C_{s_i} &= (t + t^{-1})T_\nu && (\ell(\nu s_i) = \ell(\nu))
\end{aligned}
$$

These are the rules used by multiplyBy() function called by the TPoly constructor.

Algorithm

We want to find C_γ.

Calculating $FatC_\gamma$

Start with $C_{w_0} = T_{w_0}$. Compute the product $FatC_\gamma = C_{w_0} C_{s_i} C_{s_j} ... C_{s_k}$ by converting to the T notation and then multiplying as one would normally, noting that there are special rules for multiplying T_a and T_b .

As a test, calculate $FatC_\gamma$ where $\gamma = w_0 s_4 s_1$ using the information in graph-4,4.film. You should get that:

$$C_{w_0} C_{s_4} C_{s_1} = ((1/t)(T_{w_0 s_4} + T_{w_0})C_{s_1} = (1/t^2)(T_{w_0 s_4 s_1} + T_{w_0 s_4}) + (1/t)(t + t^{-1})T_{w_0}$$

Transforming $FatC_\gamma$ to C_γ

Begin with $FatC_\gamma$ and modify as follows:

Look for a coefficient of a non-negative power of t in $FatC_\gamma$ for the term $T_\nu / t^{\ell(\nu) - \ell(w_0)}$ where $\ell(\nu) < \ell(\gamma)$. If none are found, then the algorithm is finished and $C_\gamma = FatC_\gamma$. If one is found, focus on largest $\nu < \gamma$ for which it is found. Let $f(t)$ be the coefficient of $T_\nu / t^{\ell(\nu) - \ell(w_0)}$.

By assumption, $f(t) = f_{\geq 0}(t) + f_{<0}(t)$. $f_{\geq 0}(t)$ is a linear combination of non-negative powers of t, and $f_{<0}(t)$ is a linear combination of negative powers of t. Write $f_{\geq 0}(t) = z + f_{>0}(t)$, where z is an integer and $f_{>0}(t)$ is a polynomial with only positive powers.

Then, compute $FatC'_\gamma = FatC_\gamma - g(t)FatC_\nu$ where $g(t) = f_{>0}(t) + f_{>0}(t^{-1}) + z$. Note that $FatC_\nu$ has to be calculated by using techniques in the above subsection.

Lastly, $FatC_\gamma = FatC'_\gamma$

Let's transform the $FatC_\gamma$ from the previous section into C_γ. First, we need to rewrite each term as $T_\nu/t^{\ell(\nu)-\ell(w_0)}$. Thus:

$$\begin{aligned} C_{w_0}C_{s_4}C_{s_1} &= (1/t^2)(T_{w_0s_4s_1} + T_{w_0s_4}) + (1/t)(t + t^{-1})T_{w_0} \\ &= (1)(T_{w_0s_4s_1}/t^2) + (t^{-1})(T_{w_0s_4}/t) + (t^0 + t^{-2})(T_{w_0}/t^0) \end{aligned}$$

We look at the largest $\nu < \gamma$ (remember $\gamma = w_0s_4s_1$) so we only need to look at the last term, which means $\nu = w_0$. which has the non-negative power t^0, so $g(t) = t^0 = 1$. We have to now calculate $FatC_\nu$. In this case, it is easy since $\nu = w_0$ and so $FatC_\nu = T_{w_0}$. We now have everything we need to calculate $FatC'_\gamma$

$$\begin{aligned} FatC'_\gamma &= FatC_\gamma - g(t)FatC_\nu \\ &= (1)(T_{w_0s_4s_1}/t^2) + (t^{-1})(T_{w_0s_4}/t) + (t^{-2})(T_{w_0}/t^0) \end{aligned}$$

Seeing there is no more terms other than T_γ that has a positive degree of t we can conclude that $FatC'_\gamma = C_\gamma$.

Polynomials in Terms of t^2 and Output Files

You may notice in the code that there is not much discussion of negative powers of t and instead I reference t^2. Also, the output files do not contain any polynomials in terms of t^{-1}. This slight alteration in how the polynomials are represented is necessary to ease data storage. This changes the algorithm slightly. For example, factoring out t^{-2} from $2t^8+6t^6+7t^4+5t^2+1t^0$ will not result in any negative powers of t. However, in both cases, it is the same idea i.e. find a coefficient of t^x in $FatC_\gamma$, where x is greater than or equal to some known value, for the term $T_\nu/t^{\ell(\nu)-\ell(w_0)}$ where $\ell(\nu) < \ell(\gamma)$. If $P_{\nu,\gamma}(t^2)$ denotes the polynomial in terms of t^2 for the T_ν of C_γ and $P_{\nu,\gamma}(t^{-1})$ denotes the polynomial in terms of t^{-1} for the T_ν of C_γ, then the formula to convert between the two is $p_{\nu,\gamma}(t^{-1}) = (1/t^{\ell(\gamma)-\ell(\nu)})P_{\nu,\gamma}(t^2)$. Thus, the above C_γ in terms of polynomials in t^2 is $(1)(T_{w_0s_4s_1}/t^2) + (1)(T_{w_0s_4}/t) + (1)(T_{w_0}/t^0)$.

The progam would output this like:

```
( 0 , 0 , 0 ) ( 2 , 1 , 0 ) 2 + 1t^0
( 1 , 0 , 1 ) ( 2 , 1 , 0 ) 1 + 1t^0
( 2 , 1 , 0 ) ( 2 , 1 , 0 ) 0 + 1t^0
```

The first column represents ν in T^ν, second γ in C^γ, the third is the value $\ell(\gamma) - \ell(\nu)$, and the fourth are the coefficients we just computed (called Kazhdan-Lusztig polynomials).