



Technische
Universität
Braunschweig

Technische Universität Braunschweig
Institut Computational Mathematics

Verbände getwisteter Involutionen in Coxetergruppen

Christian Hoffmeister

July 13, 2012

Contents

1	Misc	3
2	Zweizykel in der getwisteten schwachen Ordnung	4
3	Algorithmus zur Berechnung der getwisteten schwachen Ordnung	5

1 Misc

Question 1.1. Let (W, S) be a Coxeter system, $\theta : W \rightarrow W$ an automorphism of W with $\theta^2 = \text{id}$ and $\theta(K) = K$, and $K \subset S$ a subset of S generating a finite subgroup of W . Furthermore let $T, S_1, S_2, S_3 \subset S$ be four pairwise disjoint sets of generators. For which Coxeter group W does the implication

$$w \in w_K C_{T \cup S_i}, i = 1, 2, 3 \Rightarrow w \in w_K C_T \quad (1.1.1)$$

hold for any possible $K, \theta, T, S_1, S_2, S_3$ and w ?

Proposition 1.2. Let (W, S) be a Coxeter system and K, T, S_1, S_2, S_3 be like in 1.1. Suppose we have $w \in W$ and $a_1, \dots, a_n \in T \cup S_1, b_1, \dots, b_n \in T \cup S_2, c_1, \dots, c_n \in T \cup S_3$ with

$$\begin{aligned} w &= w_K \underline{a_1 \cdots a_n} \\ &= w_K \underline{b_1 \cdots b_n} \\ &= w_K \underline{c_1 \cdots c_n} \end{aligned}$$

and (1.1.1) does not hold for these three expressions, i.e. $w \notin w_K C_T$. Then there exist $t_1, \dots, t_m \in T$ and $a'_1, \dots, a'_{n-m} \in T \cup S_1, b'_1, \dots, b'_{n-m} \in T \cup S_2, c'_1, \dots, c'_{n-m} \in T \cup S_3$ such that

$$\begin{aligned} w \underline{t_1 \cdots t_m} &= w_K \underline{a'_1 \cdots a'_{n-m}} \\ &= w_K \underline{b'_1 \cdots b'_{n-m}} \\ &= w_K \underline{c'_1 \cdots c'_{n-m}} \end{aligned}$$

with $a'_{n-m}, b'_{n-m}, c'_{n-m} \notin T$.

Proof. Suppose at least one element of a_n, b_n, c_n to be in T , for example $a_n \in T$. Then we can apply $\underline{a_n}$ to all three expressions. Since $\rho(w \underline{a_n}) < \rho(w)$ the exchange condition for \mathcal{I}_θ [Hultman, 2007, Proposition 3.10] yields

$$\begin{aligned} w \underline{a_n} &= w_K \underline{a_1 \cdots a_n a_n} = w_K \underline{a_1 \cdots a_{n-1}} \\ &= w_K \underline{b_1 \cdots b_n a_n} = w_K \underline{b_1 \cdots \hat{b}_i \cdots b_n} \\ &= w_K \underline{c_1 \cdots c_n a_n} = w_K \underline{c_1 \cdots \hat{c}_j \cdots c_n} \end{aligned}$$

where $\hat{}$ means omission. The omission cannot occur within w_K since all three expressions are still of same twisted length and in the first expression we can see, that $w_K \leq w \underline{a_n}$ still holds. This step can be repeated until $w = w_K$ or $a_n, b_n, c_n \notin T$. \square

Lemma 1.3. A counterexample to 1.1 can only be found, if there is an element $u \in w_K C_T$ and three distinct generators $s_1, s_2, s_3 \in D_r(u)$ such that $u s_i \notin w_K C_T$ for $i = 1, 2, 3$.

Proof. According to proposition 1.2. \square

2 Zweizykel in der getwisteten schwachen Ordnung

Definition 2.1 (Ein- und beidseitige Wirkung). Seien (W, S) ein Coxetersystem, $w \in W$ und $s \in S$. Falls $w\bar{s} = \theta(s)ws$ ist, so sagen wir, dass s beidseitig auf w wirkt. Andernfalls sagen wir s wirkt einseitig auf w .

Definition 2.2 (Ein- und beidseitige endende Gesamtwirkung). Seien (W, S) ein Coxetersystem, $w \in W$ und $s_1, \dots, s_n \in S$. Falls $w\bar{s}_1 \cdots \bar{s}_n = \theta(s_n)(w\bar{s}_1 \cdots \bar{s}_{n-1})s_n$ ist, so sagen wir, dass $s_1 \cdots s_n$ eine beidseitig endende Gesamtwirkung auf w hat. Andernfalls sagen wir $s_1 \cdots s_n$ hat eine einseitig endende Gesamtwirkung auf w .

Definition 2.3. Seien (W, S) ein Coxetersystem und $s, t \in S$ zwei verschiedene Erzeuger. Wir definieren:

$$[st]^n := \begin{cases} (st)^{\frac{n}{2}}, & n \text{ gerade} \\ (st)^{\frac{n-1}{2}}s, & n \text{ ungerade} \end{cases}$$

Definition 2.4 (Zweizykel). Seien (W, S) ein Coxetersystem und $s, t \in S$ zwei verschiedene Erzeuger. Dann nennen wir $wC_{\{s,t\}}$ den von s und t erzeugten Zweizykel bezüglich w .

Assumption 2.5. Seien (W, S) ein Coxetersystem und $s, t \in S$ zwei verschiedene Erzeuger von W . Dann gilt:

1. Sei $m = \text{ord}(st) < \infty$. Falls $w[st]^n \neq w$ ist für alle $n \in \mathbb{N}, n < 2m$, dann gilt $w(st)^{2m} = w$.
2. In $wC_{\{s,t\}}$ existieren keine drei Elemente derselben getwisteten Länge.
3. Falls s einseitig auf w wirkt, dann gilt $w\bar{s}t < w\bar{s}$ oder $w\bar{t} > w$.
4. Sei $w[st]^n = w$ für ein $n \in \mathbb{N}$. Dann ist n gerade und es gilt eine der beiden folgenden Eigenschaften:
 - a) Für jedes $m \in \mathbb{N}$ hat das Element $[st]^m$ genau dann eine beidseitig endende Gesamtwirkung auf w , wenn $[st]^{n/2+m}$ eine beidseitig endende Gesamtwirkung auf w hat.
 - b) Für jedes $m \in \mathbb{N}$ hat das Element $[st]^m$ genau dann eine beidseitig endende Gesamtwirkung auf w , wenn $[st]^{n-m+1}$ eine beidseitig endende Gesamtwirkung auf w hat.

Remark 2.6. Vermutung 2.5.2 bedeutet, dass Zweizykel in einem gewissen Sinne konkav sind. Vermutung 2.5.3 bedeutet, dass innerhalb eines Zweizykels einseitige Wirkungen ausschließlich am bzgl. der getwisteten Länge oberen oder unteren Ende auftreten können. Vermutung 2.5.4 bedeutet, dass in einem Zweizykel die ein- und beidseitigen Wirkungen achsen- oder punktsymmetrisch verteilt sind.

Lemma 2.7 (Assumption 2.5.2). *Proof.* Let (W, S) be a Coxeter system, $w \in W$ with $\text{rank } w = k$, $s, t \in S$ with $s \neq t$. Without loss of generality we can choose w such that $w < w\bar{s}$ and $w < w\bar{t}$. Assume the existence of an element $u \in wC_{\{s,t\}}$ with $u\bar{s} < u$ and $u\bar{t} < u$. Then [Hultman, 2007, Lemma 3.8] yields $s, t \in D_R(u)$. By using [Hultman, 2007, Lemma 3.9] we conclude that $w\bar{s} \leq u$ and $w\bar{t} \leq u$. Hence there cannot exist more than two Elements of same twisted length.

If no such u exists, then $wC_{\{s,t\}} = w \cup \{w[st]^n : n \in \mathbb{N}\} \cup \{w[ts]^n : n \in \mathbb{N}\}$ and the assumption still holds. \square

3 Algorithmus zur Berechnung der getwisteten schwachen Ordnung

Wir wollen nun einen Algorithmus zur Berechnung der getwisteten schwachen Ordnung $Wk(\theta)$ einer beliebigen Coxetergruppe W erarbeiten. Also Ausgangspunkt werden wir den Algorithmus aus [Haas and Helmnick, 2012, Algorithm 3.1.1] verwenden, der im wesentlichen benutzt, dass für jede getwistete Involution $w \in \mathcal{I}_\theta$ entweder $w\underline{s} < w$ oder aber $w\underline{s} > w$ gilt.

Algorithm 3.1 (Algorithmus 1).

```

1: procedure TWISTEDWEAKORDERINGALGORITHM1( $W$ )                                ▷  $W$  sei die Coxetergruppe
2:    $V \leftarrow \{(e, 0)\}$ 
3:    $E \leftarrow \{\}$ 
4:   for  $k \leftarrow 0$  to  $k_{\max}$  do
5:     for all  $(w, k_w) \in V$  with  $k_w = k$  do
6:       for all  $s \in S$  with  $\nexists(\cdot, w, s) \in E$  do                                ▷ Nur die  $s$ , die nicht schon nach  $w$  führen
7:          $y \leftarrow ws$ 
8:          $z \leftarrow \theta(s)y$ 
9:         if  $z = w$  then
10:           $x \leftarrow y$                                                          ▷  $s$  operiert ungetwistet auf  $w$ 
11:           $t \leftarrow s$ 
12:         else
13:           $x \leftarrow z$                                                          ▷  $s$  operiert getwistet auf  $w$ 
14:           $t \leftarrow \underline{s}$ 
15:         end if
16:          $isNew \leftarrow \mathbf{true}$ 
17:         for all  $(w', k_{w'}) \in V$  with  $k_{w'} = k + 1$  do                        ▷ Prüfen, ob  $x$  nicht schon in  $V$  liegt
18:           if  $x = w'$  then
19:              $isNew \leftarrow \mathbf{false}$ 
20:           end if
21:         end for
22:         if  $isNew = \mathbf{true}$  then
23:            $V \leftarrow V \cup \{(x, k + 1)\}$ 
24:            $E \leftarrow E \cup \{(w, x, t)\}$ 
25:         else
26:            $E \leftarrow E \cup \{(w, x, t)\}$ 
27:         end if
28:       end for
29:     end for
30:      $k \leftarrow k + 1$ 
31:   end for
32:   return  $(V, E)$                                                          ▷ The poset graph
33: end procedure

```

Dieser Algorithmus berechnet alle getwisteten Involutionen und deren getwistete Länge (w, k_w)

und deren Relationen (w', w, s) bzw. (w', w, \underline{s}) . Zu bemerken ist, dass zur Berechnung der getwisteten Involutionen der Länge k nur die Knoten aus V benötigt werden, mit der getwisteten Länge $k - 1$ und k sowie die Kanten aus E , die Knoten der Länge $k - 2$ und $k - 1$ bzw. $k - 1$ und k verbinden. Alle vorherigen Ergebnisse können schon persistiert werden, so dass nie das komplette Ergebnis im Speicher gehalten werden muss.

Eine Operation, die hier als elementar angenommen wurde ist der Vergleich von Elementen in W . Für bestimmte Gruppen wie z.B. die A_n , welche je isomorph zu $Sym(n + 1)$ sind, lässt sich der Vergleich von Element effizient implementieren. Will man jedoch mit Coxetergruppen im Allgemeinen arbeiten, so liegt W als frei präsentierte Gruppe vor und der Vergleich von Element ist eine sehr aufwendige Operation. Bei Algorithmus 3.1 muss jedes potentiell neue Element x mit allen schon bekannten w' von gleicher getwisteter Länge verglichen werden um zu bestimmen, ob x wirklich ein noch nicht bekanntes Element aus \mathcal{I}_θ ist.

Algorithm 3.2 (Algorithmus 2).

```

1: procedure TWISTEDWEAKORDERINGALGORITHM2( $W$ )           ▷  $W$  sei die Coxetergruppe
2:    $V \leftarrow \{(e, 0)\}$ 
3:    $E \leftarrow \{\}$ 
4:   for  $k \leftarrow 0$  to  $k_{\max}$  do
5:     TODO
6:   end for
7:   return  $(V, E)$                                        ▷ The poset graph
8: end procedure

```

Im Anhang findet sich eine Implementierung der Algorithmen 3.1 und 3.2 in GAP 4.5.4. Tabelle 3 zeigt ein Benchmark anhand von fünf ausgewählten Coxetergruppen.

W	A_9	A_{10}	A_{11}	E_6	E_7	E_8
$ Wk(\text{id}, W) $	9496	35696	140152	892	10208	199952
$\max \rho(w)$	25	30	36	20	35	64
TWOA1	00:02.180	00:31.442	11:04.241	00:03.044	06:11.728	–
TWOA2	00:01.372	00:06.276	00:29.830	00:00.268	00:02.840	11:03.278
TWOA1 element compares	13.531.414	185.791.174	2.778.111.763	85.857	7.785.186	–
TWOA2 element compares	42.156	173.356	737.313	2.347	29.687	682.227

Table 3.1: Benchmark

Dabei sind die A_n als symmetrische Gruppen implementiert und die E_n als frei präsentierte Gruppen. Ausgeführt wurden die Messungen auf einem Intel Core i5-3570k mit vier Kernen zu je 3,40 GHz. Der Algorithmus ist dabei aber nur single threaded und kann so nur auf einem Kern laufen. Um die Messergebnisse nicht durch Limitierungen des Datenspeichers zu beeinflussen, wurden die Daten in diesem Benchmark nicht stückweise persistiert sondern ausschließlich berechnet. Wie zu erwarten ist der Geschwindigkeitsgewinn bei den Coxetergruppen vom Typ E_n deutlich größer, da in diesem Fall die Elementvergleiche deutlich aufwendiger sind als bei Gruppen vom Typ A_n .

References

- R. Haas and A. G. Helmnick. Algorithms for twisted involutions in weyl groups. *Algebra Colloquium* 19, 2012.
- A. Hultman. The combinatorics of twisted involutions in coxeter groups. *Transactions of the American Mathematical Society, Volume 359*, pages 2787–2798, 2007.
- J. E. Humphreys. *Reflection groups and Coxeter groups*. Cambridge University Press, 1992.