2017

Crossed Product Algebras over Dihedral Field Extensions

Kaitlyn Lee
Butler University

BUTLER UNIVERSITY HONORS PROGRAM

Honors Thesis Certification

Please type all information in this section:

Applicant Kaitlyn Rene Lee
(Name as it is to appear on diploma)

Thesis title Crossed Product Algebras over Dihedral Field Extensions

Intended date of commencement May 2017

Read, approved, and signed by:

Thesis adviser(s) Christopher Wilson 1 May 2017
Date

Date

Reader(s) William Johnston 1 May 2017
Date

Date

Certified by
Director, Honors Program
Date

Crossed Product Algebras over Dihedral Field Extensions

A Thesis

Presented to the Department of Mathematics

College of Liberal Arts and Sciences

and

The Honors Program

of

Butler University

In Partial Fulfillment

of the Requirements for Graduation Honors

Kaitlyn Rene Lee

April 10, 2017

1. Introduction

Algebraists are interested in classifying field extensions. To do this, they sometimes at the similarities and differences of the field extensions' automorphism groups. The structural features of a particular automorphism group tell us information about the structure present in a corresponding field extension. An object called a crossed product algebra includes both the elements of the extended field as well as the structure of the automorphism group, so it encodes a lot of information about the field extension. In this paper, we examine the multiplication operation for crossed product algebras that arise from what are called dihedral groups.

Crossed product algebras are examples of structures known as rings:

Definition 1.1. *A* ***ring*** $< R, +, \cdot >$ *is a set* R *with operations* $+, \cdot$ *satisfying the following:*

(1) $\exists 0 \in R$ *such that* $0 + a = a = a + 0$ $\forall a \in R$.

(2) $\forall a \in R$ $\exists (-a) \in R$ *such that* $a + (-a) = 0 = (-a) + a$.

(3) $a + b = b + a$ $\forall a, b \in R$.

(4) $a + (b + c) = (a + b) + c$ *and* $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ $\forall a, b, c \in R$.

(5) $a \cdot (b + c) = a \cdot b + a \cdot c$ *and* $(a + b) \cdot c = a \cdot c + b \cdot c$ $\forall a, b, c \in R$.

Definition 1.2. *A* ***field*** *is a commutative ring that has a multiplicative identity element and every nonzero element has a multiplicative inverse. When a field* F *is contained in a field* K*, we say that* K/F *("*K *over* F*") is a* ***field extension****.*

Definition 1.3. *An automorphism of the extension* K/F *is a function* $\sigma : K \to K$ *for which the following are satisfied:*

(1) $\sigma(k_1 + k_2) = \sigma(k_1) + \sigma(k_2)$ $\forall k_1, k_2 \in K$.

(2) $\sigma(k_1 \cdot k_2) = \sigma(k_1) \cdot \sigma(k_2)$ $\forall k_1, k_2 \in K$.

(3) $\sigma(f) = f$ $\forall f \in F$.

(4) σ *is one-to-one: if* $k_1 \neq k_2$ *then* $\sigma(k_1) \neq \sigma(k_2)$.

(5) σ *is onto:* $\forall y \in K$ $\exists x \in K$ *for which* $\sigma(x) = y$.

The set $G = Aut(\ K/F\ )$ *of all automorphisms forms a structure known as a group.*

2. My Set-up

Definition 2.1. *The automorphism group G is said to be **dihedral** if, for some positive integer M, $G = \{\rho^i \circ \phi^j \mid i = 0, \ldots, M-1; j = 0, 1\}$ and $\phi \circ \rho = \rho^{M-1} \circ \phi$. In this case we write $G = D_{2M}$ and say that K/F is a **dihedral field extension**.*

The interactions between the automorphisms of a dihedral automorphism group are analogous to the rotation and reflection operations that map a regular polygon onto itself via rigid motion.

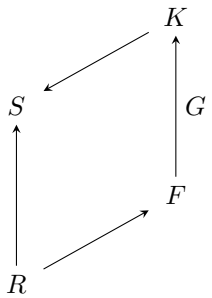
Definition 2.2. *A **dihedral field extension** is a field extension whose automorphisms act in ways that are analogous to rotating or reflecting regular polygons.*

We now define a particular field extension that gives a context for this paper.

Definition 2.3. *If k is a field, define $k[[t]] = \{a_0 + a_1 t + a_2 t^2 + \cdots \mid a_i \in k\}$. This set has a natural ring structure and is called a (formal) **power series ring**.*

From a commutative ring R that obeys the zero product property from algebra one can construct a field known as the **field of fractions** by constructing the set of all possible fractions whose numerators and denominators are elements of R.

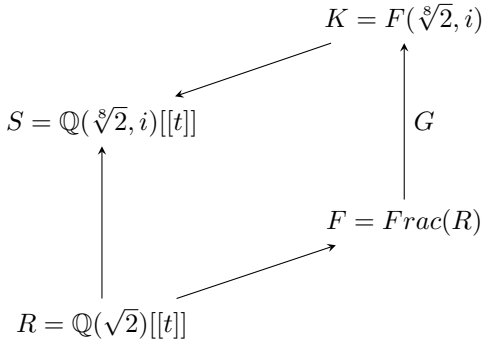
Let us fix some notation for the duration of this paper: R is a power series ring with coefficients in some field, F is a field of fractions made out of the elements of R, K is a dihedral field extension, G is a dihedral automorphism group, and S is the integral closure (defined below) of R in K.



Definition 2.4. *Let S be the **integral closure** of R in K. Then $S = \{s \in K \mid$ s satisfies a monic polynomial whose coefficients are elements of $R\}$. (A monic polynomial is one that has a coefficient of 1 on its highest degree term.)*

We are studying examples in which S turns out to be a power series ring.

Example 2.5. *Example of a dihedral field extension involving power series rings.*



Let k be the field $\mathbb{Q}(\sqrt{2}) = \{a + b\sqrt{2} \mid a, b \in \mathbb{Q}\}$. Construct the power series ring $R = \mathbb{Q}(\sqrt{2})[[t]] = k[[t]]$. Let F be the field of fractions of R. Let $K = F(\sqrt[8]{2}, i)$ be the smallest subfield of $\mathbb{C}$ that contains F, $\sqrt[8]{2}$, and i. In this case, the integral closure of R in K turns out to be the power series ring $S = \mathbb{Q}(\sqrt[8]{2}, i)[[t]]$ (The coefficient field $\mathbb{Q}(\sqrt[8]{2}, i)$ for S is the smallest field that contains $\mathbb{Q}$, $\sqrt[8]{2}$, and i.)

$(\sqrt[8]{2} + 3i) + (\frac{5}{2}\sqrt[8]{2})t + (i+1)t^2 + \cdots$ is an example of an element of S.

One can show (see [1]*, p. 561) that the automorphism group K/F is*

$$G = \left\{ \begin{array}{ll} \phi: & i \mapsto -i, \quad \sqrt[8]{2} \mapsto \sqrt[8]{2} \\ \rho: & i \mapsto i, \quad \sqrt[8]{2} \mapsto -i\sqrt[8]{2} \end{array} \text{ and compositions of } \phi, \rho \right\} = D_8$$

If you think about the complex plane, then you could view ϕ as a reflection through the x-axis and ρ as rotating certain elements by ninety degrees.

Definition 2.6. ***Crossed Product Algebra***

Given a field extension K/F with automorphism group G, form the set $A = \{\sum_{\sigma \in G} \alpha_\sigma x_\sigma \mid$ $\alpha_\sigma \in S\}$.

- *Define addition as combining like terms.*
- *Define multiplication as the following:*

(1) *Start with a "coefficient function"* $f : G \times G \to S - \{0\}$.

(2) *Define* $x_\rho a = \rho(a)x_\rho$ *and* $x_\rho x_\tau = f(\rho, \tau)x_{\rho\circ\tau}$ *for* $a \in S$ *and for automorphisms* $\rho, \tau \in G$. *We assume* $f(1_G, \sigma) = f(\sigma, 1_G) = 1_S \ \forall\sigma$. [1]

(3) *To multiply sums of terms, use the distributive property.*

We require the coefficient function f *to be chosen such that* $(x_\sigma)(x_\rho x_\tau) = (x_\sigma x_\rho)x_\tau$ $\Leftrightarrow f(\sigma, \tau) \cdot f(\sigma\tau, \rho) = \sigma f(\tau, \rho) \cdot f(\sigma, \tau\rho) \ \forall\sigma, \rho, \tau \in G$.

This requirement for the coefficient function f is equivalent to associativity, therefore A is a ring. We will refer to the equation above as the associativity condition. (In some literature, a function f satisfying the associativity condition is called a 2-cocycle.)

Definition 2.7. *In the notation from the beginning of this section, define the valuation function* $v : K \to \mathbb{Z}$ *by v(a power series) = (degree of the lowest-degree term) and define v(fraction of power series)=v(numerator)-v(denominator).*

Example 2.8. *Let R, F, K, and S be as in Example 2.5. Then, for example:* $v(\frac{it^7 + \text{ higher degree terms } + \cdots}{(\sqrt[8]{2} + \frac{3}{2})t^5 + \text{ higher degree terms } + \cdots}) = 7 - 5 = 2$

3. Research Problem

Research Problem: Let f be the coefficient function for a crossed product algebra. What can $v \circ f$ look like if G is dihedral? In the examples in this paper, we will explore this question by considering different possible choices for the following subgroup of G:

Definition 3.1. *The* ***inertial group*** *is defined to be*

$$H = \{\sigma \in G \mid v \circ f(\sigma, \sigma^{-1}) = 0\}.$$

[1] This is always possible to arrange by selecting new basis elements $\{x'_\sigma\}$ that are K-multiples of any given basis for A, so each $x'_\sigma = b_\sigma x_\sigma$ for some $b_\sigma \in K$.

In [3], [2], [5], the associativity condition

$$v \circ f(g_1, g_2) + v \circ f(g_1 g_2, g_3) = v \circ f(g_2, g_3) + v \circ f(g_1, g_2 g_3)$$

is used to prove the following:

Theorem 3.2. *H is a group.*

Example 3.3. *Inertial Group Example. Suppose* $G = D_8 = \langle \rho, \phi \mid \rho^4 = \phi^2 = 1, \rho\phi = \phi\rho^2 \rangle$ *and* $v \circ f$ *takes the values shown in Table 1.*

$v \circ f(G \times G)$	1	ρ	ρ^2	ρ^3	ϕ	$\rho\phi$	$\rho^2\phi$	$\rho^3\phi$
1	0	0	0	0	0	0	0	0
ρ	0	0	0	1	0			
ρ^2	0	0	1	1	0			
ρ^3	0	1	1	1	0			
ϕ	0	0	0	0	0	0	0	0
$\rho\phi$	0				0			
$\rho^2\phi$	0				0			
$\rho^3\phi$	0				0			

TABLE 1. $v \circ f(G \times G)$ for D_8

Then $\rho \notin H$ *because* $v \circ f(\rho, \rho^3) = 1$ *(since* $\rho^3 = \rho^{-1}$*), but* $\phi \in H$ *because* $\phi = \phi^{-1}$ *and* $v \circ f(\phi, \phi) = 0$.

4. Results

Theorem 4.1. *If G is dihedral and* $v \circ f(G \times G) \in \{0, 1\}$*, then H cannot be trivial.*

Proof. Suppose $v \circ f(G \times G) \in \{0, 1\}$. By Theorem 2.15 in [5], G/H is cyclic. Since G is dihedral, $G/\{1\}$ cannot be cyclic. Therefore $H \neq \{1\}$. □

In 2016, Sam Good proved this result in the special case $G = D_6$. An open question is if G is dihedral and $v \circ f(G \times G) \geqslant 0$, can H be trivial? ($G/H$ is not guaranteed to be cyclic if $v \circ f$ table values > 1).

We find the following theorem from my own research. The proof of this theorem suggests an algorithm that makes it easy to generate examples of coefficient functions.

Theorem 4.2. *If $G = D_{2M}$ is the dihedral group $\langle \rho, \phi \mid \rho^M = 1, \phi^2 = 1, \rho\phi = \phi\rho^{M-1} \rangle$ of order 2M, and if $v \circ f \geq 0$, then $\forall g_1, g_2 \in G$ the value $v \circ f(g_1, g_2)$ is determined by $\{v \circ f(\rho, \rho^i),\ v \circ f(\phi, \rho^i),\ v \circ f(\rho^i, \phi),\ v \circ f(\phi, \phi) \mid i \geqslant 0\}$.*

These values are starred in on Table 2 and the six stages to fill out the table are shown on Table 3.

$v \circ f$	1	ρ	ρ^2	$\cdots$	ρ^{M-1}	ϕ	$\rho\phi$	$\rho^2\phi$	$\cdots$	$\rho^{M-1}\phi$
1	0	0	0	0	0	0	0	0	0	0
ρ	0	$*$	$*$	$\cdots$	$*$	$*$				
ρ^2	0					$*$				
$\vdots$	$\vdots$					$\vdots$				
ρ^{M-1}	0					$*$				
ϕ	0	$*$	$*$	$\cdots$	$*$	$*$				
$\rho\phi$	0									
$\rho^2\phi$	0									
$\vdots$	$\vdots$									
$\rho^{M-1}\phi$	0									

TABLE 2. $v \circ f(x_{g_1}, x_{g_2})$

$v \circ f$	1	ρ	ρ^2	$\cdots$	ρ^{M-1}	ϕ	$\rho\phi$	$\rho^2\phi$	$\cdots$	$\rho^{M-1}\phi$
1										
ρ		$*$	$*$	$\cdots$	$*$	$*$	**3**	**3**	$\cdots$	**3**
ρ^2		**1**	**1**	$\cdots$	**1**	$*$	**3**			**3**
$\vdots$		$\vdots$			$\vdots$	$\vdots$	$\vdots$			$\vdots$
ρ^{M-1}		**1**	**1**	$\cdots$	**1**	$*$	**3**	**3**	$\cdots$	**3**
ϕ		$*$	$*$	$*$	$\cdots$	$*$	**4**	**4**	$\cdots$	**4**
$\rho\phi$		**2**	**2**	$\cdots$	**2**	**6**	**5**	**5**	$\cdots$	**5**
$\rho^2\phi$		**2**			**2**	**6**	**5**			**5**
$\vdots$		$\vdots$			$\vdots$	$\vdots$	$\vdots$			$\vdots$
$\rho^{M-1}\phi$		**2**	**2**	$\cdots$	**2**	**6**	**5**	**5**	$\cdots$	**5**

TABLE 3. $v \circ f$ Stages to fill out cocycle table

Proof. Using the additive associativity condition

$$v \circ f(g_1, g_2) + v \circ f(g_1 g_2, g_3) = v \circ [f(g_2, g_3)^{g_1}] + v \circ f(g_1, g_2 g_3)$$

we have the following:

Stage 1: $(\rho, \rho^{R-1}, \rho^C)$.

Put $(g_1, g_2, g_3) = (\rho, \rho^{R-1}, \rho^C)$ in the associativity condition and then solve

$$\underbrace{v \circ f(\rho, \rho^{R-1})}_{\text{Given}} + \underbrace{v \circ f(\rho^R, \rho^C)}_{1} = \underbrace{v \circ f(\rho^{R-1}, \rho^C)}_{\text{Given}} + \underbrace{v \circ f(\rho, \rho^{R-1+C})}_{\text{Given}}$$

for $v \circ f(\rho^R, \rho^C)$ to determine $v \circ f(\rho^R, \rho^C)$ recursively in terms of table values from previously determined table rows (R = row number in table and C = column number in table).

Stage 2: Use $(g_1, g_2, g_3) = (\phi, \rho^{M-R}, \rho^C)$ and solve

$$\underbrace{v \circ f(\phi, \rho^{M-R})}_{\text{Given}} + \underbrace{v \circ f(\rho^R\phi, \rho^C)}_{2} = \underbrace{v \circ f(\rho^{M-R}, \rho^C)}_{1} + \underbrace{v \circ f(\phi, \rho^{M-R+C})}_{\text{Given}}$$

for $v \circ f(\rho^R\phi, \rho^C)$ to obtain all of the values in "Block 2" of Table 3.

Stage 3: Use $(g_1, g_2, g_3) = (\rho^a, \rho^b, \phi)$

$$\underbrace{v \circ f(\rho^a, \rho^b)}_{1} + \underbrace{v \circ f(\rho^a\rho^b, \phi)}_{\text{Given}} = \underbrace{v \circ f(\rho^b, \phi)}_{\text{Given}} + \underbrace{v \circ f(\rho^a, \rho^b\phi)}_{3}$$

Solve for $v \circ f(\rho^a, \rho^b\phi)$.

Stage 4: Use $(g_1, g_2, g_3) = (\phi, \phi, \rho^a)$

$$\underbrace{v \circ f(\phi, \phi)}_{\text{Given}} + \underbrace{v \circ f(1, \rho^a)}_{0} = \underbrace{v \circ f(\phi, \rho^a)}_{\text{Given}} + \underbrace{v \circ f(\phi, \phi\rho^a)}_{4}$$

Solve for $v \circ f(\phi, \phi\rho^a)$.

Stage 5: Use $(g_1, g_2, g_3) = (\rho^a, \phi, \rho^b\phi)$

$$\underbrace{v \circ f(\rho^a, \phi)}_{\text{Given}} + \underbrace{v \circ f(\rho^a\phi, \rho^b\phi)}_{5} = \underbrace{v \circ f(\phi, \rho^b\phi)}_{4} + \underbrace{v \circ f(\rho^a, \phi\rho^b\phi)}_{\text{Given}}$$

Solve for $v \circ f(\rho^a\phi, \rho^b\phi)$.

Stage 6: Use $(g_1, g_2, g_3) = (\rho^a\phi, \phi, \rho^b)$

$$\underbrace{v \circ f(\rho^a\phi, \phi)}_{6} + \underbrace{v \circ f(\rho^a, \rho^b)}_{1} = \underbrace{v \circ f(\phi, \rho^b)}_{\text{Given}} + \underbrace{v \circ f(\rho^a\phi, \phi\rho^b)}_{5}$$

Solve for $v \circ f(\rho^a\phi, \phi)$. □

Note 4.3. *In Appendix 5, we give the C++ code that implements this algorithm and checks that the associativity condition is satisfied for any set of integers inputed into the starred locations of Table 2.*

We find two facts when considering (g, h, h^{-1}) and (h^{-1}, h, g) in the context of a cocycle f satisfying $v \circ f(g_1, g_2) \geq 0$. Applying v to the cocycle equation $f(g_1, g_2)f(g_1g_2, g_3) = f(g_2, g_3)^{g_1}f(g_1, g_2g_3)$, we have $v \circ f(g_1, g_2) + v \circ f(g_1g_2, g_3) = v \circ [f(g_2, g_3)^{g_1}] + v \circ f(g_1, g_2g_3)$. Recall that $H = \{\sigma \in G \mid v \circ f(\sigma, \sigma^{-1}) = 0\}$. From the assumption that $f(1, g) = f(g, 1) = 1_K$ we have $v \circ f(1, g) = v \circ f(g, 1) = 0$, we find the following facts:

Fact 4.4. *If $v \circ f \geq 0$ and $h \in H$, then*

$$f(g, h)f(gh, h^{-1}) = f(h, h^{-1})^g f(g, hh^{-1})$$

$$\Rightarrow f(g, h)f(gh, h^{-1}) = 1^g f(g, 1)$$

where g leaves 1 alone ($1 \in F$).

$$\Rightarrow f(g, h)f(gh, h^{-1}) = 1$$

$$\Rightarrow v \circ f(g, h) + v \circ f(gh, h^{-1}) = 0$$

Because the summands are nonnegative, we conclude the following:

$$v \circ f(g, h) = 0 \ \forall h \in H, \ \forall g \in G.$$

Fact 4.5. *If $v \circ f \geq 0$, then for $h \in H$*

$$f(h^{-1}, h)f(h^{-1}h, g) = f(h, g)^{h^{-1}} f(h^{-1}, hg)$$

$$\Rightarrow f(1,g) = f(h,g)^{h^{-1}} f(h^{-1},hg)$$

$$\Rightarrow 1 = f(h,g)^{h^{-1}} f(h^{-1},hg)$$

$$\Rightarrow 0 = v \circ f(h,g) + v \circ f(h^{-1},hg)$$

$$\Rightarrow 0 = v \circ f(h,g)$$

Remark 4.6. *If S is the power series ring $\overline{S}[[t]]$ over the field $\overline{S}$, then $v(g(t)) = v(t) = 1$ for all $g \in G$* [2] *by results from* [4]*; see also Theorem 3.1 of* [5].

When evaluating the associativity condition at (h, g_1, g_2) we have the following:

$$f(h,g_1)f(hg_1,g_2) = f(g_1,g_2)^h f(h,g_1g_2)$$

$$\Rightarrow v \circ f(h,g_1) + v \circ f(hg_1,g_2) = v \circ f(g_1,g_2)^h + v \circ f(h,g_1g_2)$$

By Fact 4.4, Fact 4.5, and Remark 4.6, we have

$$v \circ f(hg_1,g_2) = v \circ f(g_1,g_2)^h = v \circ f(g_1,g_2)$$

Evaluating the associativity condition at (g_1, g_2, h) produces

$$f(g_1,g_2)f(g_1g_2,h) = f(g_2,h)^{g_1} f(g_1,g_2h)$$

$$\Rightarrow v \circ f(g_1,g_2) + v \circ f(g_1g_2,h) = v \circ f(g_2,h)^{g_1} + v \circ f(g_1,g_2h)$$

By Facts 4.4 and 4.5, we have $v \circ f(g_1g_2,h) = v \circ f(g_2,h)^{g_1} = 0$.
Since h and g_1 do not change the value of $f(g_1,g_2)$ and $f(g_2,h)$ respectively, we have

$$v \circ f(hg_1,g_2) = v \circ f(g_1,g_2).$$

[2]Except possibly in unusual cases where S/R is said to be wildly ramified, the definition of which is beyond the scope of this paper; see [4].

From this we see that we can apply h either on the left of the input g_1 in $f(g_1, g_2)$ or on the right of the input g_2 without changing the value $v \circ f(g_1, g_2)$. We collect these facts in the following lemma.

Lemma 4.7. *Assume $v \circ f(g_1, g_2) \geq 0$. Then for every $g \in G$ and $h \in H$,*

(1) $v \circ f(g, h) = v \circ f(h, g) = 0$, *and*

(2) $v \circ f(hg_1, g_2) = v \circ f(g_1, g_2) = v \circ f(g_1, g_2 h)$.

Example 4.8. *The following tables are examples of possible values taken by the function $v \circ f$ when G is the dihedral group $G = D_8$ of order 8. In Tables 4, 6, and 7, the inertial group is $H = \langle \rho^2, \phi \rangle$ and $G/H = \langle \rho H \rangle$. In Table 5, $H = \langle \rho \rangle$ and $G/H = \langle \phi \rangle$.*

$v \circ f$	1	ρ	ρ^2	ρ^3	ϕ	$\rho\phi$	$\rho^2\phi$	$\rho^3\phi$
1	0	0	0	0	0	0	0	0
ρ	0	1	0	1	0	1	0	1
ρ^2	0	0	0	0	0	0	0	0
ρ^3	0	1	0	1	0	1	0	1
ϕ	0	0	0	0	0	0	0	0
$\rho\phi$	0	1	0	1	0	1	0	1
$\rho^2\phi$	0	0	0	0	0	0	0	0
$\rho^3\phi$	0	1	0	1	0	1	0	1

TABLE 4. $v \circ f(x_{g_1}, x_{g_2})$

$v \circ f$	1	ρ	ρ^2	ρ^3	ϕ	$\rho\phi$	$\rho^2\phi$	$\rho^3\phi$
1	0	0	0	0	0	0	0	0
ρ	0	0	0	0	0	0	0	0
ρ^2	0	0	0	0	0	0	0	0
ρ^3	0	0	0	0	0	0	0	0
ϕ	0	0	0	0	1	0	1	1
$\rho\phi$	0	0	0	0	1	1	1	1
$\rho^2\phi$	0	0	0	0	1	1	1	1
$\rho^3\phi$	0	0	0	0	1	1	1	1

TABLE 5. $v \circ f(x_{g_1}, x_{g_2})$

$v \circ f$	1	ρ	ρ^2	ρ^3	ϕ	$\rho\phi$	$\rho^2\phi$	$\rho^3\phi$
1	0	0	0	0	0	0	0	0
ρ	0	4	0	4	0	4	0	4
ρ^2	0	0	0	0	0	0	0	0
ρ^3	0	4	0	4	0	4	0	4
ϕ	0	0	0	0	0	0	0	0
$\rho\phi$	0	4	0	4	0	4	0	4
$\rho^2\phi$	0	0	0	0	0	0	0	0
$\rho^3\phi$	0	4	0	4	0	4	0	4

TABLE 6. $v \circ f(x_{g_1}, x_{g_2})$

$v \circ f$	1	ρ	ρ^2	ρ^3	ϕ	$\rho\phi$	$\rho^2\phi$	$\rho^3\phi$
1	0	0	0	0	0	0	0	0
ρ	0	4	0	4	0	4	0	4
ρ^2	0	0	0	0	0	0	0	0
ρ^3	0	4	0	4	0	4	0	4
ϕ	0	0	0	0	4	0	4	4
$\rho\phi$	0	4	0	4	4	8	4	8
$\rho^2\phi$	0	0	0	0	4	4	4	4
$\rho^3\phi$	0	4	0	4	4	8	4	8

TABLE 7. $v \circ f(x_{g_1}, x_{g_2})$

Example 4.9. *Table 8 provides an example of possible values taken by the function $v \circ f$ when G is the dihedral group $G = D_{12}$ of order 12.*

$v \circ f$	1	ρ	ρ^2	ρ^3	ρ^4	ρ^5	ϕ	$\rho\phi$	$\rho^2\phi$	$\rho^3\phi$	$\rho^4\phi$	$\rho^5\phi$
1	0	0	0	0	0	0	0	0	0	0	0	0
ρ	0	1	0	1	0	1	0	1	0	1	0	1
ρ^2	0	0	0	0	0	0	0	0	0	0	0	0
ρ^3	0	1	0	1	0	1	0	1	0	1	0	1
ρ^4	0	0	0	0	0	0	0	0	0	0	0	0
ρ^5	0	1	0	1	0	1	0	1	0	1	0	1
ϕ	0	0	0	0	0	0	0	0	0	0	0	0
$\rho\phi$	0	1	0	1	0	1	0	1	0	1	0	1
$\rho^2\phi$	0	0	0	0	0	0	0	0	0	0	0	0
$\rho^3\phi$	0	1	0	1	0	1	0	1	0	1	0	1
$\rho^4\phi$	0	0	0	0	0	0	0	0	0	0	0	0
$\rho^5\phi$	0	1	0	1	0	1	0	1	0	1	0	1

TABLE 8. $v \circ f(x_{g_1}, x_{g_2})$

The computer program in the appendix was very useful for verifying that these examples satisfy the associativity condition.

Definition 4.10. *If H_0 is a subgroup of G, then for each $g \in G$ the set $gH_0 = \{gh \mid h \in H_0\}$ is called a left coset of H_0 in G. The set of all left cosets of H_0 in G is denoted G/H_0 ("G mod H_0").*

If $v \circ f(G \times G) \subseteq \{0, 1\}$ and H is the inertial subgroup, then G/H also has a group structure with multiplication given by $g_1H \cdot g_2H = (g_1g_2)H$.

Remark 4.11. *If $v \circ f \in \{0, 1\}$, ρH is of order m in G/H (meaning that m is the smallest positive integer for which $(\rho H)^m = 1_G H = H$), and ρH is minimal among nonidentity cosets of G/H, then*

$$v \circ f(\rho, \rho^j) = \begin{cases} 1 \text{ if } j \equiv -1 \mod (m) \\ 0 \text{ otherwise.} \end{cases}$$

Remark 4.11 is due to the fact that G/H is partially ordered by the relation

$$g_1H \leq g_2H$$

if and only if

$$v \circ f(g_1, g_1^{-1}g_2) = 0$$

(see [3] [2] [5]) If $v \circ f(G \times G) \subseteq \{0, 1\}$, then $\leq$ becomes a total ordering [2] [5]. For example, in Table 4, we have $H = \langle \rho^2, \phi \rangle$ and $H \leq \rho H$.

Theorem 4.12. *If G is dihedral, $v \circ f(G \times G) \subseteq \{0, 1\}$, and $\rho \in H$ is the minimal (with respect to the partial ordering described above) nonidentity coset for G/H then ρH must have order 2 in G/H and $v \circ f(\rho, \rho) \neq 0$.*

Proof. Because G/H is cyclic when $v \circ f \in \{0, 1\}$ [5] we must have $\phi \in H$ as well (otherwise G/H is not cyclic). Now assume $f(\rho, \rho) = 0$. Use Stage 1 of the algorithm in Theorem 4.2 to get values for $v \circ f(\rho^R, \rho^C)$. In particular, $v \circ f(\rho^{\alpha-1}, \rho^{\alpha-1}) = 1$, where α is the order of ρH in G/H. By Facts 4.4 and 4.5, we have $v \circ f(\rho^{\alpha-1}\phi, \rho\phi) = v \circ f(\underbrace{\phi}_{\phi \in H} \rho, \rho\phi) = v \circ f(\rho, \rho \underbrace{\phi}_{\phi \in H}) = v \circ f(\rho, \rho) = 0$. But if ρH is minimal nonidentity coset for $\leq$ on G/H, then by Lemma 4.7(1)

(with $h = \phi$), we have $v \circ f(\rho^{\alpha-1}, \phi) = v \circ f(\underbrace{\phi}_{\phi \in H}, \rho\phi) = 0$, so we also have (using the associativity condition on $(\rho^{\alpha-1}, \phi, \rho\phi)$) : $v \circ f(\rho^{\alpha-1}, \phi) + v \circ f(\rho^{\alpha-1}\phi, \rho\phi) = v \circ f(\phi, \rho\phi) + v \circ f(\rho^{\alpha-1}, \rho^{\alpha-1}) \Rightarrow 0 + v \circ f(\rho^{\alpha-1}\phi, \rho\phi) = 0 + v \circ f(\rho^{\alpha-1}, \rho^{\alpha-1})$. But then $0 = v \circ f(\rho^{\alpha-1}\phi, \rho\phi) = v \circ f(\rho^{\alpha-1}, \rho^{\alpha-1}) = 1$, a contradiction.

Thus, $v \circ f(\rho, \rho^{-1}\rho^2) \neq 0$ so that $\rho H \not\leq \rho^2 H$; because ρH is minimal and G/H is totally ordered, $\rho^2 H$ must be H. Thus ρH has order $\alpha = 2$ in G/H. □

5. Appendix-Coefficient Function Generator Computer Code

```
////////////////////////////////////////////////
//                                            //
//          Cocycle Generator                 //
//                                            //
//       Kaitlyn Lee - 27 January 2017        //
//                                            //
////////////////////////////////////////////////


#include <iostream>
#include <string>

using namespace std;

const int MODULUS = 4;
//set MODULUS = order of rotation subgroup;  rotation subgroup = <rho>

struct cocycleValue
{
    int i;      //power on rho^i in f(rho^i phi^j , rho^h phi^k)
    int j;      //power on phi^j
```

```
    int h;   //power on rho^h
    int k;   //power on phi^k
    int omega;  //power on omega (the MODULUS-th root of unity)
    int u ;     //  power on unit
    int t ;    // power on uniformizer of the DVR S
};


//Declare functions that will be defined below.

void displayCocycleValue(cocycleValue f);
void initializeCocycle(cocycleValue f[MODULUS][2][MODULUS][2]);
void generateCocycle(cocycleValue f[MODULUS][2][MODULUS][2]);
cocycleValue productOfCocycleValues
(cocycleValue v_1, cocycleValue v_2, int I, int J, int H, int K);
cocycleValue quotientOfCocycleValues
(cocycleValue v_1, cocycleValue v_2, int I, int J, int H, int K);
cocycleValue applySigma(int power, cocycleValue v);
cocycleValue applySigma(cocycleValue v);
cocycleValue copyOf(cocycleValue v);
bool isEqual(cocycleValue v, cocycleValue w);
bool isCocycle(cocycleValue f[MODULUS][2][MODULUS][2]);

////////////////////////////////////////////////////////////
///MAIN    MAIN    MAIN    MAIN    MAIN    MAIN    MAIN    ///
///MAIN    MAIN    MAIN    MAIN    MAIN    MAIN    MAIN    ///
///MAIN    MAIN    MAIN    MAIN    MAIN    MAIN    MAIN    ///
////////////////////////////////////////////////////////////
```

```
int main()
{
    cocycleValue f[MODULUS][2][MODULUS][2];
    //A cocycle is a 2-dimesional array of cocycleValues.

    initializeCocycle(f);
    //Pass in the name of the array of cocycleValues.
    //This fills the cocycle with the value 1_K.



    //Define values for the f(sigma, sigma^i) row
    //of the multiplicaion table:
    //f[1][0][2][0].t =  1 ;       //f(sigma, sigma^5) := t^5
    f[1][0][3][0].t =  1 ;
    // f[1][23].u = 1;         //f(sigma, sigma^23) := u*t^5

    generateCocycle(f);
    //Determine cocycle values based on values of
    //f(rho^i phi^j, rho^h phi^k.  Calling this function
    //also lists these values.

    isCocycle(f);
    //Determine if resulting multiplication table
    //gives an associative multiplication.  Also
    //determine if any f(g1, g2) has
    //negative S-adic value.

    cout << "Press ENTER to continue..." << endl;
    cin.get();
```

```
    //Pause before closing output window.

    return 0;

}

////////////////////////////////////////////////
////////////////////////////////////////////////
// END OF MAIN - END OF MAIN - END OF MAIN  ////
////////////////////////////////////////////////
////////////////////////////////////////////////

void displayCocycleValue(cocycleValue f)
{
    cout << "f(rhoˆ" << f.i << "phiˆ" << f.j << ",rhoˆ" << f.h << "phiˆ" << f.k
    << ") = tˆ" << f.t << endl ;
    return ;
}

void initializeCocycle
(cocycleValue f[MODULUS][2][MODULUS][2])
{
    for (int i = 0 ; i <= MODULUS - 1; i++)
    {
        for (int j = 0; j<= 1; j++)
        {for (int k = 0; k<= MODULUS - 1; k++)
        {for (int h= 0; h<=1; h++)
        {
            f[i][j][k][h].i = i ;
```

```
            f[i][j][k][h].j = j ;
            f[i][j][k][h].h = h ;
            f[i][j][k][h].k = k ;
            f[i][j][k][h].u = 0 ;
            f[i][j][k][h].omega = 0 ;
            f[i][j][k][h].t = 0 ;
        }
        }
        }
    }
    return;
}


// generateCocycle assumes that
// 1.  initializeCocycle has been called.
// 2.  the "i=1 row" f(rho^1, rho^h)
//     has been chosen.
// 3.  The f(phi, phi) value choses

void generateCocycle
(cocycleValue f[MODULUS][2][MODULUS][2])
{

    ////////////////////////////////////////////////////
    // STAGE 1 // STAGE 1 //STAGE 1 // STAGE 1 // STAGE 1 //
    ////////////////////////////////////////////////////
```

```
for (int row = 2 ; row <= MODULUS - 1 ; row++)
{
    for (int col = 1 ; col <= MODULUS - 1 ; col++)
    {
        f[row][0][col][0] =
        quotientOfCocycleValues
        (productOfCocycleValues
         (applySigma(f[row - 1][0][col][0]),
          f[1][0][ (row + col - 1) % MODULUS][0], row, 0, col, 0) ,
         f[1][0][row - 1][0], row, 0, col, 0);
        //Use f(g^r, g^c) = f(g^(r-1), g^c)^g f(g, g^(r-1+c))
        //                                  / f(g, g^(r-1)
        //to generate next row of cocycle.

        displayCocycleValue(f[row][0][col][0]);
    }
    cin.get();   //Pauses display of output
}

//////////////////////////////////////////////////////
// STAGE 2 // STAGE 2 //STAGE 2 // STAGE 2 // STAGE 2 //
//////////////////////////////////////////////////////

for (int row = 1 ; row <= MODULUS - 1 ; row++)
{
    for (int col = 1 ; col <= MODULUS - 1 ; col++)
    {
        f[row][1][col][0] =
        quotientOfCocycleValues
```

```
            (productOfCocycleValues
             (applySigma(f[MODULUS - row][0][col][0]),
              f[0][1][ (MODULUS - row + col) % MODULUS][0], row, 1, col, 0) ,
             f[0][1][MODULUS - row][0], row, 1, col, 0);
            //Use f(g^r, g^c) = f(g^(r-1), g^c)^g f(g, g^(r-1+c))
            //                                  / f(g, g^(r-1)
            //to generate next row of cocycle.

            displayCocycleValue(f[row][1][col][0]);
        }
        cin.get();   //Pauses display of output
    }

    ////////////////////////////////////////////////////
    // STAGE 3 // STAGE 3 //STAGE 3 // STAGE 3 // STAGE 3 //
    ////////////////////////////////////////////////////

    for (int row = 1 ; row <= MODULUS - 1 ; row++)
    {
        for (int col = 1 ; col <= MODULUS - 1 ; col++)
        {
            f[row][0][col][1] =
            quotientOfCocycleValues
            (productOfCocycleValues
             (applySigma(f[row][0][col][0]),
              f[( row + col )% MODULUS ][0][0][1], row, 0, col, 1) ,
             f[col][0][0][1], row, 0, col, 1);
            //Use f(g^r, g^c) = f(g^(r-1), g^c)^g f(g, g^(r-1+c))
            //                                  / f(g, g^(r-1)
```

```
        //to generate next row of cocycle.

        displayCocycleValue(f[row][0][col][1]);
    }
    cin.get();   //Pauses display of output
}

////////////////////////////////////////////////////
// STAGE 4 // STAGE 4 //STAGE 4 // STAGE 4 // STAGE 4 //
////////////////////////////////////////////////////

for (int col = 1 ; col <= MODULUS - 1 ; col++)
{
    f[0][1][MODULUS - col][1] =
    quotientOfCocycleValues
    (f[0][1][0][1],
     //(productOfCocycleValues
     //(applySigma(f[a][0][b][0]),
     //  f[( a + b )% MODULUS ][0][1][0], a, 0, b, 1) ,
     f[0][1][col][0], 0, 1, MODULUS - col, 1);
    //Use f(g^r, g^c) = f(g^(r-1), g^c)^g f(g, g^(r-1+c))
    //                                  / f(g, g^(r-1)
    //to generate next row of cocycle.

    displayCocycleValue(f[0][1][col][1]);
}
cin.get();   //Pauses display of output

////////////////////////////////////////////////////
```

```
// STAGE 5 // STAGE 5 //STAGE 5 // STAGE 5 // STAGE 5 //
////////////////////////////////////////////////////////

for (int row = 1 ; row <= MODULUS - 1 ; row++)
{
    for (int col = 1 ; col <= MODULUS - 1 ; col++)
    {
        f[row][1][col][1] =
        quotientOfCocycleValues
        (productOfCocycleValues
         (applySigma(f[0][1][col][1]),
          f[row][0][MODULUS - col][0], row, 1, col, 1) ,
         f[row][0][0][1], row, 1, col, 1);
        //Use f(g^r, g^c) = f(g^(r-1), g^c)^g f(g, g^(r-1+c))
        //                                    / f(g, g^(r-1)
        //to generate next row of cocycle.

        displayCocycleValue(f[row][1][col][1]);
    }
    cin.get();   //Pauses display of output
}

////////////////////////////////////////////////////////
// STAGE 6 // STAGE 6 //STAGE 6 // STAGE 6 // STAGE 6 //
////////////////////////////////////////////////////////

for (int row = 1 ; row <= MODULUS - 1 ; row++)
{
```

```
        f[row][1][0][1] =
        quotientOfCocycleValues
        (productOfCocycleValues
         (applySigma(f[0][1][1][0]),
          f[row][1][MODULUS - 1][1], row, 1, 0, 1) ,
         f[row][0][1][0], row, 1, 0, 1);
        //Use f(g^r, g^c) = f(g^(r-1), g^c)^g f(g, g^(r-1+c))
        //                                  / f(g, g^(r-1)
        //to generate next row of cocycle.

        displayCocycleValue(f[row][1][0][1]);

        cin.get();   //Pauses display of output
    }

    return;
}

// productOfCocycleValues  and
// quotientOfCocycleValues need
// 1.  two cocycleValues
// 2.  the i and j  for the product

cocycleValue productOfCocycleValues
(cocycleValue v_1, cocycleValue v_2, int I, int J, int H, int K)
{
    cocycleValue product;
    product.u = v_1.u + v_2.u        ;
    product.t = v_1.t + v_2.t     ;
```

```
    product.omega = ((v_1.omega + v_2.omega) % MODULUS) ;
    product.i = I   ;
    product.j = J   ;
    product.h = H ;
    product.k = K  ;
    return product ;
}

cocycleValue quotientOfCocycleValues
(cocycleValue v_1, cocycleValue v_2, int I, int J, int H, int K)
{
    cocycleValue quotient;
    quotient.u = v_1.u - v_2.u ;
    quotient.omega = ((v_1.omega - v_2.omega + MODULUS) % MODULUS) ;
    quotient.t = v_1.t - v_2.t ;
    quotient.i = I ;
    quotient.j = J ;
    quotient.h = H ;
    quotient.k = K ;
    return quotient;
}

//return a copy of a given cocycleValue
cocycleValue copyOf(cocycleValue v)
{
    cocycleValue result;
    result.i = v.i;
    result.j = v.j;
    result.h = v.h;
```

```
    result.k = v.k;
    result.u = v.u;
    result.t = v.t;
    result.omega = v.omega;
    return result;
}

cocycleValue applySigma
(int power, cocycleValue v)

//Apply sigma^power to a cocycleValue v.
//by using the applySigma(cocycleValue)
//a total of *power* times
{
    cocycleValue result = copyOf(v);
    for (int i = 1 ; i <= power ; i++)
    {
        result = applySigma(result);
    }
    return result;
}

cocycleValue applySigma(cocycleValue v)     //not implemented yet
{
    cocycleValue result = copyOf(v);
    // result.omega = ((v.omega + 4*v.u + v.t ) % MODULUS);

    //Edit the preceding line according to
    //the action of sigma on u and t.
```

```
    return result;
}

bool isEqual(cocycleValue v, cocycleValue w)
{
    if (v.t!=w.t) return 0;
    //  if (v.u != w.u) return 0;
    //if ((v.omega % MODULUS) != (w.omega % MODULUS) )
    // return 0;
    //Return FALSE if any factors do not match.

    return 1;  //Otherwise return TRUE.
}

bool isCocycle
(cocycleValue f[MODULUS][2][MODULUS][2])
{
    for (int R1 = 0 ; R1 <= MODULUS - 1 ; R1++)
    {
        for  (int R2 = 0 ; R2 <= MODULUS - 1 ; R2++)
        {
            for (int R3 = 0 ; R3 <= MODULUS - 1 ; R3++)
            {
                for (int F1 = 0; F1 <= 1; F1++)
                {
                    for (int F2 = 0; F2 <= 1; F2++)
                    {
                        for (int F3 = 0; F3 <= 1; F3++)
```

```
{
    if
        (
         isEqual
         (
          productOfCocycleValues
          (f[R1][F1][R2][F2], f[(F1*(R1+MODULUS-R2)+(1-F1)*(R1+R2))
          % MODULUS][(F1 + F2)%2][R3][F3], 0 , 0, 0,0)    ,
          (
           applySigma (R1, f[R2][F2][R3][F3]) ,
           f[R1][F1][(F2*(R2 + MODULUS - R3) +(1-F2)*(R2+R3))
           % MODULUS][(F2 + F3)%2] , 0 , 0,0,0
           // applySigma has not been implemented
           )
          )
         )
        cout << "(" << R1 << "," <<  F1 << "," << R2 <<
        "," << F2 << "," << R3 << "," << F3 << " ) ok!" << endl ;

    else
    {
        cout << "FAIL:" << "(" << R1 << "," <<  F1 << "," << R2 <<
        "," << F2 << "," << R3 << "," << F3 <<")";
        //list the first failure of associativity

        return 0;   //return FALSE
    }
}
}
```

```
            }
        }
    }
}



{
    for (int R1 = 0 ; R1 <= MODULUS - 1 ; R1++)
    {
        for (int R2 = 0 ; R2 <= MODULUS - 1 ; R2++)
        {
            for (int F1 = 0 ; F1 <= 1 ; F1++)
            {
                for (int F2 = 0 ; F2 <= 1 ; F2++)
                {  if (f[R1][F1][R2][F2].t < 0)
                {
                    cout <<"FAIL: Negative S-adic value at f(rho^"
                    << R1 << ", phi" << F1 <<", rho^" << R2 <<", phi"
                    << F2 << ")" << endl;
                    return 0 ;
                    //list the first place for which
                    //f(sigma^i, sigma^j) has negative S-adic value
                    //and return FALSE.
                }
                }
            }
        }
    }
    return 1;      //return TRUE: f defines a weak
```

```
        //crossed product order A_f in Sigma_f.
    }
}
```

References

[1] D. Dummit and R. Foote, *Abstract Algebra*, Wiley, 2 ed., 1999.

[2] D. E. Haile, *Crossed-products orders over discrete valuation rings*, J. Algebra, 105 (1987), pp. 116–148.

[3] D. E. Haile, R. G. Larson, and M. E. Sweedler, *A new invariant for* $\mathbf{C}$ *over* $\mathbf{R}$*: almost invertible cohomology theory and the classification of idempotent cohomology classes and algebras by partially ordered sets with a Galois group action*, Amer. J. Math., 105 (1983), pp. 689–814.

[4] G. J. Janusz, *Algebraic Number Fields*, Academic Press, 2 ed., 1996.

[5] C. J. Wilson, *Hereditary crossed product orders over discrete valuation rings*, J. Algebra, 371 (2012), pp. 329–349.