Doubly Even Codes and Adinkras

Zack Baker, Nicholas Hemleben, Benedict Mondal, Brian Tran

PIMS

Doubly Even $\iff$ Odd Dashing

Theorem:
Let C be a code and let I^n be the n-cube. Then, a chromotopology $A = I^n/C$ admits an odd dashing $\iff$
C is doubly-even, that is $\forall c \in C, 4|wt(c)$

proof:
($\Longrightarrow$) Suppose $A = I^n/C$ is a well-dashed chromotopology. Let $q_I : V(A) \to V(A)$ be the I^{th} color map, and let $C = \{(x_1 \cdots x_n) \in \mathbb{Z}_2^n | \sum_{I=1}^n q_I^{x^I} = \mathbb{I}\}$

Let $x = (x_1 \cdots x_n) \in C$. Then, observe

$$Q_1^{x_1} \cdots Q_n^{x_n} T = k\partial^{wt(x)/2} T \qquad (1)$$

for some $k \in \{\pm i, \pm 1\}$
Then, $Q_1^{x_1} \cdots Q_n^{x_n} Q_1^{x_1} \cdots Q_n^{x_n} T = k^2 \partial^{wt(x)} T$, i.e.

$$(-1)^{\binom{wt(x)}{2}} Q_1^{2x_1} \cdots Q_n^{2x_n} T = k^2 \partial^{wt(x)} T$$

$$(-1)^{\binom{wt(x)}{2}} (i)^{wt(x)} \partial^{wt(x)} T = k^2 \partial^{wt(x)} T$$

Observe $(-1)^{\binom{wt(x)}{2}} = (i)^{wt(x)}$, thus

$$(i)^{wt(x)} (i)^{wt(x)} \partial^{wt(x)} T = k^2 \partial^{wt(x)} T$$

$$k^2 = (i)^{2wt(x)}$$

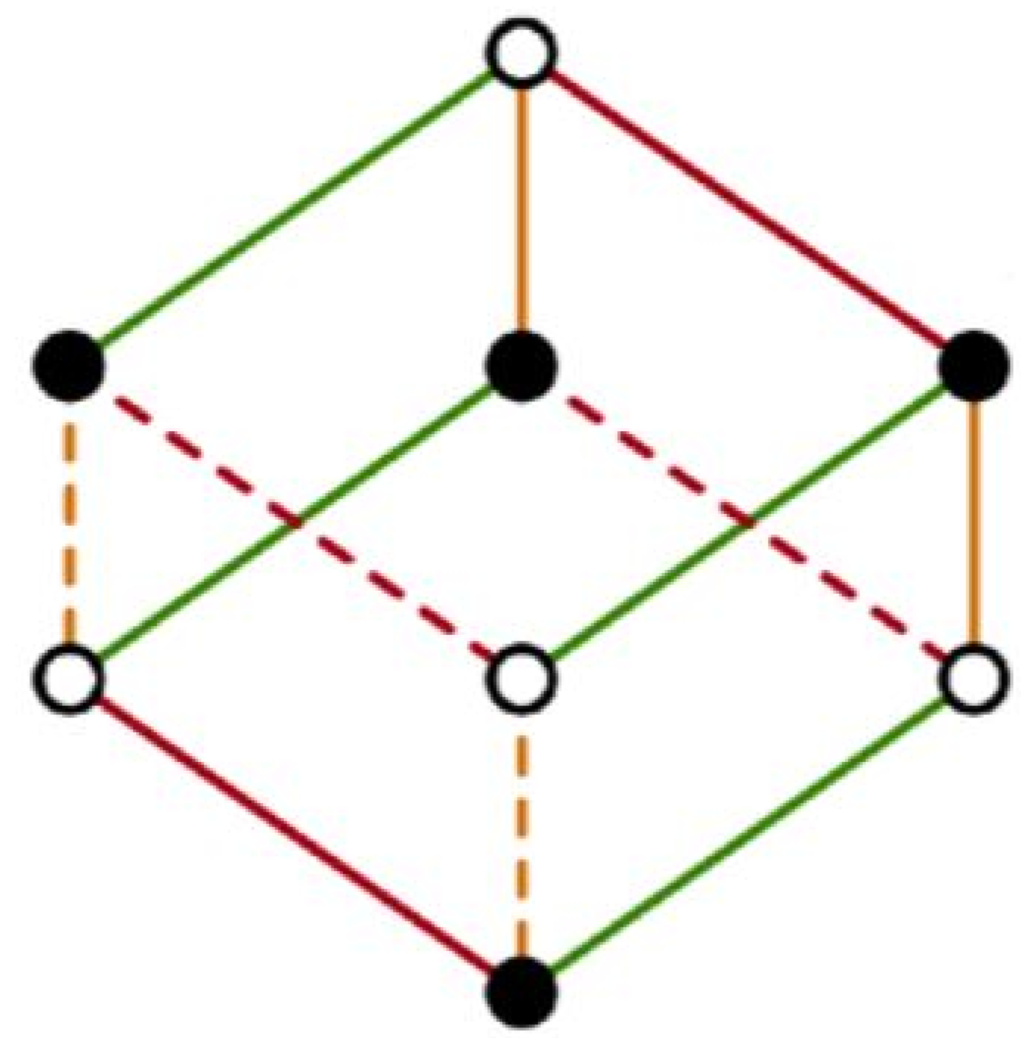
Since A is a chromotopology, A is bipartite and so, $wt(x)$ is even. Thus, $k^2 = 1$, so $k = \pm 1$. Note that $Q_I Q_J$ produces one factor of i, but from equation (1), this implies that $wt(x) = 0$ mod 4.

($\Longleftarrow$) Let $\gamma_1, ..., \gamma_n$ be generators for the Clifford algebra, i.e. which satisfy $\{\gamma_i, \gamma_j\} = 2\delta_{ij}\mathbb{I}$. Let C be a code and $c_i = (c_i(1) \cdots c_i(n)), i = 1, ..., k$ be generators for C. We can associate to each codeword c_i an element of the Clifford Algebra: $g(c_i) = \prod_k \gamma_k^{c_i(k)}$. Note these elements, with sign i.e. $\pm g(c_i)$, form a multiplicative group G.

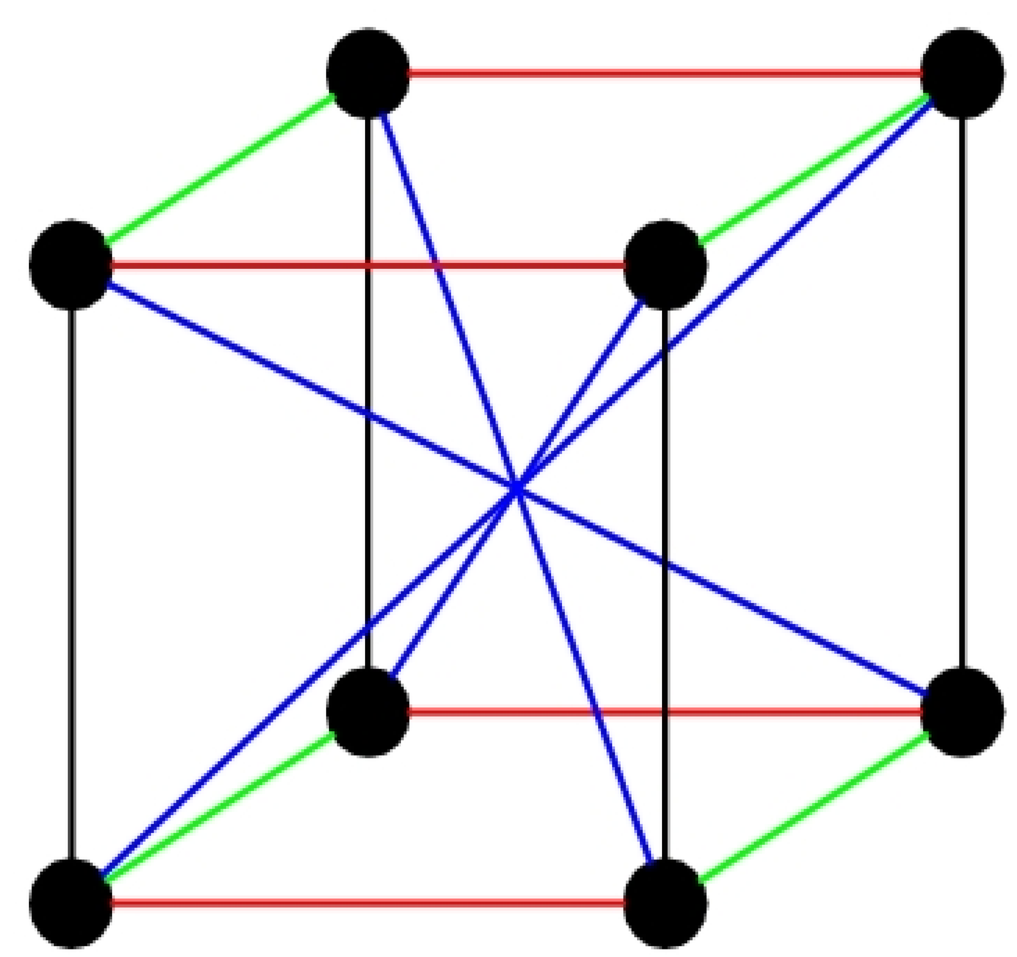
Now, suppose C is a doubly-even code and consider the sign function, $s(c) \in \{\pm 1\}\ \forall c \in C$, with specified action on a basis of b_i of C: $s(b_i) = 1$. Then, define $G_s = \{s(c)g(c) | c \in C\}$ which is a subgroup of G.

To see how this produces an odd-dashed chromotopology A, consider G/G_s, where the cosets of G_s in G correspond to the vertices of A. The Clifford Algebra structure is preserved in G/G_s and thus the anti-commutation relation is satisfied, which produces $Q_I Q_J = -Q_J Q_I$ $(J \neq I)$, i.e. A is odd-dashed □

Examples



An odd-dashed chromotopology, by the theorem, corresponds to a doubly even code.



The graph of $I^4/\{0000, 1111\}$ can be odd-dashed, since $\{0000, 1111\}$ is doubly-even.

Algorithm

1 Search For Even Even Codes

1.1 Group Theory Observations

To begin, we noticed that all doubly even codes are normal subgroups of the groups of all codes of a given length n, We notate this $C_n \trianglelefteq \mathbb{Z}_2^n$. By Lagrange's theorem the order of all C_n must divide the order of $\mathbb{Z}_2^n$. Thus the order of all C_n must be 2^k for some $k \leq n$, since $|\mathbb{Z}_2^n| = 2^n$.

1.2 Pseudo Code

1. To find all doubly even codes, we computed all subsets of $\mathbb{Z}_2^n$ of order 2^k for all $k \leq n$. This list of sets gave us all potential subgroups of $\mathbb{Z}_2^n$.
2. Then, we computationally brute forced checking which of these subsets are closed under addition. These closed subsets are all subgroups of $\mathbb{Z}_2^n$ which are doubly even. We outline our algorithm below.

Code

1.3 Code

The following functions compose the algorithms we used to compute the closed doubly even codes:

- *gen_words(l)* generates all doubly even codewords of length l. This set is used to create all possible subsets less than l, to begin checking closure.
- *get_subsets(s,l)* generates all valid subsets of size less than l of s. Since subgroups must have order of 2^k, this function only creates subsets of size 2^k to validate.
- *b_add(a,b)* is the operation under which codes must be closed to be valid. It is defined by componentwise addition of codeword digits by $1+1 = 0+0 = 0$,$1+0 = 1$. This operation is abelian.
- *is_closed(s)*checks that the code s is closed under the function *b_add*. Since all codes are abelian, we only need to iterate over all $b > a$ to ensure that s is closed.

Algorithm Images

```
def get_subsets(s, l):
    pset = []
    for i in range(1,l):
        if pow(2,i)<=pow(2,l-2):
            #pset.append(list(itertools.combinations(s,pow(2,i)-1)))
            cur_list = list(itertools.combinations(s,pow(2,i)-1))
            for i in cur_list:
                if is_closed(i):
                    pset.append(i)
    return pset

def b_add(a,b): #add 2 binary strings mod 2 and return the result
    ab = ""
    for i in range(len(a)):
        d = (int(a[i])+int(b[i]))%2
        ab += str(d)
    return ab

def gen_words(l): #generate all doubly even binary numbers of length up to l
    words = []
    for i in range(pow(2,l)):
        code = pad_words(l,bin(i)[2:])
        if is_doubly_even(code):
            words.append(code)
    return words

def is_closed(s): #for a given code s consisting of codewords, check if it is closed under b_add
    for a in s:
        for b in [i for i in s if i>=a]:
            if b_add(a,b) in s or b_add(a,b)==n_zero(len(a)):
                pass
            else:
                #print s, "is not closed"
                return
    print s, "is closed"


#main

even_words = gen_words(int(sys.argv[1]))[1:]



pset_even = get_subsets(even_words,int(sys.argv[1]))
```

The code used to determine the closed doubly even codes of size n.

```
('001111',) is closed
('010111',) is closed
('011011',) is closed
('011101',) is closed
('011110',) is closed
('100111',) is closed
('101011',) is closed
('101101',) is closed
('101110',) is closed
('110011',) is closed
('110101',) is closed
('110110',) is closed
('111001',) is closed
('111010',) is closed
('111100',) is closed
('001111', '110011', '111100') is closed
('001111', '110101', '111010') is closed
('001111', '110110', '111001') is closed
('010111', '101011', '111100') is closed
('010111', '101101', '111010') is closed
('010111', '101110', '111001') is closed
('011011', '100111', '111100') is closed
('011011', '101101', '110110') is closed
('011011', '101110', '110101') is closed
('011101', '100111', '111010') is closed
('011101', '101011', '110110') is closed
('011101', '101110', '110011') is closed
('011110', '100111', '111001') is closed
('011110', '101011', '110101') is closed
('011110', '101101', '110011') is closed
```

Results for n=6. Note that the zero vector has been removed, to aid calculation times. The zero vector is implicitly present in all codes.

Acknowledgements

We would like to acknowledge Charles Doran, Sylvester James Gates, Jr., Kevin Iga, Ursula Whitcher, the Pacific Institute of Mathematical Sciences, and the National Science Foundation for hosting this event.