# Computer Verifications of Regular Representations of Groups of Orders Smaller than $33$ via $k$-Hypergraphs

Dominika Mihálová

*Comenius University, Faculty of Mathematics, Physics and Informatics, Department of Applied Informatics, Bratislava, Slovakia*

Abstract

Given a finite group $G$, the hypergraphical regular representation problem asks about the existence of a hypergraph whose full automorphism group is equal to $G$ acting regularly on the set of its vertices. In our paper, we work with $k$-hypergraphs which are hypergraphs in which each hyperedge is of the same size $k$. Since the existence or non-existence of hypergraphical regular representations for groups of the order exceeding the value $32$ has been proved theoretically, our focus is on a computational verification of hypergraphical regular representation via $k$-hypergraphs for groups of order smaller than or equal to $32$. For all groups of order less than or equal to $32$, except for the group $\mathbb{Z}_2^5$, we computationally proved a conjecture stating that if a group $G$ admits a hypergraphical regular representation via some $n$-hypergraph, it admits the hypergraphical regular representation for every $k$-hypergraph in the range $n \leq k \leq |G| - n$. To obtain our results, we created and implemented algorithms in the computational system GAP.

**Keywords**

regular representation, k-kypergraphs, computer verification

## 1. Introduction

Throughout this paper, we consider all groups to be finite. The automorphism of a graph $\Gamma$ is a permutation $\phi$ of the vertex set $V(\Gamma)$, such that the pair of vertices $(u, v)$ form an edge if and only if the pair $(\phi(u), \phi(v))$ also form an edge. The set of all automorphisms of $\Gamma$ together with the operation of composition form the automorphism group $Aut(\Gamma)$ of the given graph $\Gamma$. The graphical regular representation (GRR) of a group $G$ is a graph $\Gamma$ with the set of vertices $V(\Gamma)$ and the set of edges $E(\Gamma)$ having the property that the automorphism group of the graph $Aut(\Gamma)$ is the group $G$ in its regular action. Frucht in [1] proved that every finite group $G$ has a graphical representation whose automorphism group is isomorphic to $G$, but the representation is not necessarily regular.

The GRR problem has been intensely studied over the years. First, Sabidussi [2] and Chao [3] proved the non-existence of GRR for abelian groups with exponent greater than 2. The list of groups with no GRR's was supplemented by McAndrew [4] and Imrich [5] for groups $\mathbb{Z}_2^n$, where $n = 2, 3, 4$. Later, Imrich and Watkins [6] showed the existence of GRR's for abelian groups with exponent equal to 2.

The search for the GRR's for non-abelian groups started with Nowitz [7], later joined by Watkins [8, 9] who established the result that a group $G$ has a GRR if $G$ is a non-abelian cyclic extension of an abelian group and the order of the group is relatively prime to 6. Their method was generalised by Imrich [10] who discovered that non-abelian groups whose order is odd and not less than $3^7 \cdot 5^4$ admit a GRR. Later, Watkins [11] summarised the previous findings that were subsequently supplemented by Godsil [12]. Gradually, a list of groups that do not admit a GRR has been formed and today we have a complete classification in the form of the following list: abelian groups with exponent greater than 2, generalised dicyclic groups and groups isomorphic to one of 13 groups whose order is not greater than 32 $\mathbb{Z}_2^2$, $\mathbb{Z}_2^3$, $\mathbb{Z}_2^4$, $\mathbb{D}_3$, $\mathbb{D}_4$, $\mathbb{D}_5$, $\mathbb{A}_4$, $\mathbb{Q} \times \mathbb{Z}_3$, $\mathbb{Q} \times \mathbb{Z}_4$, $\langle a, b, c \mid a^2 = b^2 = c^2 = 1, abc = bca = cab\rangle$, $\langle a, b \mid a^8 = b^2 = 1, b^{-1}ab = a^5\rangle$, $\langle a, b, c \mid a^3 = b^3 = c^2 = 1, ab = ba, (ac)^2 = (bc)^2 = 1\rangle$, $\langle a, b, c \mid a^3 = b^3 = c^3 = 1, ac = ca, bc = cb, b^{-1}ab = ac\rangle$.

A variation of the GRR problem is a digraphical regular representation (DRR) problem, which deals with directed graphs. A directed graph $X$ is a pair of the set of vertices $V(X)$ and the set of edges $E(X)$, where each edge $e \in E(X)$ is an ordered pair of vertices $u, v \in V(X)$. The DRR of a group $G$ is a directed graph $X$ whose automorphism group $Aut(X)$ preserves the direction of the edges and is isomorphic to the group $G$ acting regularly on the set of vertices of $X$. The problem was studied by Babai [13], who showed that every finite group admits a DRR except for the five groups $\mathbb{Z}_2^2, \mathbb{Z}_2^3, \mathbb{Z}_2^4, \mathbb{Z}_3^2, \mathbb{Q}_8$. Just as a side note we would like to point out a consequence which will be the subject of future research. Babai proved that for a group $G$ of order $n$ there exists a commutative semigroup of order less than or equal to $2n + 2$ with an automorphism group isomorphic to $G$.

In the present work, we focused on the problem of

*ITAT'22: Information technologies – Applications and Theory, September 23–27, 2022, Zuberec, Slovakia*

dominika.mihalova@fmph.uniba.sk (D. Mihálová)

hypergraphical regular representation motivated by the original GRR problem of which it is a generalization. The hypergraphical regular representation problem is interesting mainly due to the more complex structure of the hypergraphs. Section 2 covers the definition of the hypergraphical regular representation problem with a brief review of all necessary concepts and an overview of the previous results. At the end of the section, we present the hypergraphical regular representations problem that we computationally verified and a conjecture about the spectrum of possible parameters. In Section 3, we describe our computational approach in details. We specify the used computational system GAP that is important for reaching our proposed goals. Further, we present an implementation of algorithms in the computational system with a description of the theoretical background for each operation. In Section 4, we present different types of applied optimizations used to decrease the computational time and the amount of used memory of the implemented algorithms. We describe each optimization with the theoretical background supporting it. Further, we introduce pseudocode that implements operations from the mentioned optimizations. Section 5 presents the results of the computational verification of the two main goals introduced in Section 2. The first goal is the verification of the existence or non-existence of HRR's via a $k$-hypergraphs for groups of orders not exceeding 32. The second goal is the proof of the veracity of the conjecture about the spectrum for groups of orders not exceeding 32. We also describe interesting observations concerning our computations with regard to the minimal needed number of combinations of orbits to acquire a HRR of a group. We compare the runnings of our algorithm in computational systems GAP and MAGMA in terms of the computational time and the memory usage.

## 2. Preliminaries

A *hypergraph* $H$ is a combinatorial structure defined as an ordered pair of the set of its vertices $V(H)$ and the set of its hyperedges $E(H)$, sometimes called blocks, which are sets of the vertices. Each *hyperedge* $e \in E(H)$ is a nonempty subset of the set of vertices of $H$ and in general, it contains any number of hypergraph vertices. A *degree* of a vertex is the number of hyperedges to which the vertex belongs. A hypergraph is called a $k$*-uniform* if all hyperedges are of the same size $k$, i.e. $\forall e \in E(H) : |e| = k$. In our paper, we focus on $k$-uniform hypergraphs to which we will refer in short as $k$-hypergraphs from now on. A *regular* $k$*-hypergraph* is a $k$-hypergraph, where each vertex has the same degree. The *automorphism group* of $k$-hypergraph $H$ is the group of permutations of $V(H)$ that preserve the $k$-hyperedges, i.e., permutations $\phi \in Sym_V$ with the property $\forall e \in E(H) : \phi(e) \in E(H)$. Based on previous definitions, the 2-hypergraph is also the non-oriented graph $\Gamma$ from Section 1. A $k$-hypergraph is a *regular representation* (HRR) of a group $G$ for $0 \leq k \leq |G|$ if and only if for every two vertices $u, v \in V(H)$ there exists exactly one automorphism $\varphi(u) = v$ from the automorphism group of the $k$-hypergraph $Aut(H)$, i.e. $Aut(H)$ acts regularly on the set of vertices $V(H)$. An important concept that will be used is a *group action* of a group $G$ on a set $A$ which is a map $\cdot : G \times A \mapsto A$ (written as $g \cdot a$, $\forall g \in G, a \in A$) such that $\forall g_1, g_2 \in G, a \in A : g_1 \cdot (g_2 \cdot a) = (g_1 g_2) \cdot a$ and $\forall a \in A : 1_G \cdot a = a$. The *orbit* of a hyperedge $e \in E(H)$ is the set of hyperedges in $E(H)$ to which $e$ can be moved by the elements of group $G$, where $G$ is acting on the set $E(H)$, i.e. $G(e) = \{g \cdot e \in E(H) : g \in G\}$, where $\cdot$ is the induced action of $G$ on $E(H)$.

Foldes and Singhi [14] were the first to study the HRR problem and they stated that every finite group of the odd order greater than or equal to $5^7$ has a HRR via a 3-hypergraph. Later that year, Foldes [15] proved that cyclic groups $\mathbb{Z}_n$ for $n \neq 3, 4, 5$ have regular representation by a 3-hypergraph. In [16], Foldes and Singhi introduced a polynomial lower bound $p(k)$ such that every finite group of order greater than or equal to $p(k)$ has a HRR via a $k$-hypergraph, where $k$ is the uniform size of the $k$-hyperedges. The lower bound for the existence of regular representation by $k$-hypergraph for $k = 3 : p(3) > 2^6$ and for $k \geq 4 : p(k) > 4k + 2$. Later, Jajcay [17] studied the HRR problem for general hypergraphs. His solutions heavily depended on varying sizes of the hyperedges. It is a more general approach as hyperedges may be of different sizes. Thus the rules for admitting a HRR via hypergraphs with varying sizes of hyperedges are more relaxed than for $k$-hypergraphs. Thus, if a group does not have a HRR via a hypergraph with varying sizes of hyperedges, it can not have a HRR via a $k$-hypergraph. For hypergraphs with varying sizes of hyperedges, Jajcay moved the lower bound to $p(k) \geq 6$ and proved that only four finite groups $\mathbb{Z}_3, \mathbb{Z}_4, \mathbb{Z}_4, \mathbb{Z}_2^2$ do not have a HRR. The listed groups support the results from [15]. Afterwards, Jajcay and Jajcayova [18] published a list of groups without a HRR via 3-uniform hypergraphs consisting of the previously mentioned groups in [17] and the groups: $\mathbb{Z}_3, \mathbb{Q}_8, \mathbb{Z}_2^3, \mathbb{Z}_4^3, \mathbb{Z}_5^3, \mathbb{D}_5 \times \mathbb{Z}_5$.

Based on the previous theoretical results, we know that groups with orders greater than 32 admit a HRR. The existence of HRR is not proven for groups with orders less than or equal to 32 except for the few groups mentioned above. We computationally verified which groups of the order less than or equal to 32 have or do not have a HRR via a $k$-hypergraph. Simultaneously, we partially proved a conjecture mentioned by Jajcayova [19] in a generalised version. It states that if a group $G$ admits a HRR via a $n$-hypergraph then it admits HRRs via $k$-hypergraphs for all

$n \leq k \leq |G| - n$. The conjecture predicts a continuous and symmetric spectrum of possible parameters $k$ for which a group $G$ admits a HRR via $k$-hypergraphs.

## 3. Methods

For the computational verification of group regular representations via $k$-hypergraphs, we decided to program our algorithms in the system for computational discrete algebra - GAP [20]. The system is free, open-source and widely used in similar computational problems concerning work with groups, graphs and other combinatorial structures. It provides its own programming language and implements functions for multiple algebraic algorithms in importable packages. Throughout the implementation of our algorithms, we were using different types of packages: *GRAPE, loops* and *DESIGN*. In the rest of this section, I will give some technical information about the used packages and commands. The algorithms were implemented in the GAP with version *4.11.1*.

The name of the package *GRAPE* [21] is an abbreviation for *GRaph Algorithms using PErmutation groups* and is designed for computations, constructions and analysis of graphs with relations to groups. The package connects the graph structure with the group structure. Each graph is stored as a structure with several components. One of the components is a group of a graph which is a specifically chosen subgroup of the graph automorphism group. The *loops* package [22] allows computing with algebraic structures: quasigroups and loops. We chose to use this package because of the implementation of group-theoretical algorithms, which were important in our experiments. More precisely, we used an algorithm for the left multiplication action. We firstly transformed a group object into a quasigroup object and then performed the action of the left multiplication. Lastly, the *DESIGN* package [23] is made for constructing, classifying, partitioning and studying block designs. In their definition, a block design is an ordered pair of the set of points and the set of blocks, where a point is an element and a block is a set of elements. The condition for using the package is an imported *GRAPE* package.

To find the implementation solution to the HRR problem, we firstly focused on proposing an algorithm to detect the existence or non-existence of HRR's for groups via 2-hypergraphs. We tested our algorithm and verified its correctness on the known theoretical results. We have the list of all groups with orders not exceeding 32 admitting the GRR from the [12], which is the HRR via a 2-hypergraph in our case. After verifying the proposed algorithm for groups on 2-hypergraphs, we modified the algorithm to compute a HRR of groups via 3-hypergraphs for the same set of groups of orders less than or equal to 32. We observed the changes in the list of groups admitting the HRR's. Lastly, we generalized the algorithm for any given $k$ to obtain the list of groups with HRR via any $k$-hypergraph to confirm the conjecture given by Jajcayova [19].

As stated before, a group has a HRR if and only if there is a $k$-hypergraph whose automorphism group acts regularly on the set of its vertices. We implemented methods described and proved in [18]. The authors defined $P_k(G)$ as a set of all $k$-element subsets of a group $G$ and $G_L$ as the left multiplication of a group $G$. By [18], a group $G$ has a HRR via a $k$-hypergraph whenever there exists a $k$-hypergraph $H$ with the set of $k$-hyperedges $E(H) \subseteq P_k(G)$ whose $Aut(H) = G_L$, where $V(H)$ are elements of $G$.

We needed to search groups of small orders to use the theory from [18] in computational implementation. The above-mentioned theoretical result confirms the existence of $k$-hypergraphs, which are the HRR's, for groups exceeding the order of 32, thus our proposed algorithm needed to go through all groups of order up to and including 32. We used the for-loops `for order` $\leq 32$ to go through all orders less than or equal to 32 and `for i` $\leq$ `NrSmallGroups(order)` to get all possible groups of given order, where `NrSmallGroups()` returns the number of groups with the given order. Subsequently, we created a group object with the command `G := SmallGroup(order, i)`, which returns the $i$-th group of the given order. The results above imply that the automorphism group of the $k$-hypergraph must be equal to the left multiplication of the group from which the $k$-hypergraph was constructed. According to Cayley's theorem, we assigned to each element of the group $g_i$ the permutation of the left multiplication, such that $g_i \cdot g_j = g_h$ where $g_j$ and $g_h$ are also elements of the group. We used commands from the *loops* package to get the left multiplication. With the command `quasi := IntoQuasigroup(G)`, we transformed the default group object into a quasigroup object from the *loops* package. We performed `perm := LeftMultiplicationGroup(quasi)` to obtain the permutation of the left multiplication for each element of $G$. Based on the computed permutations, we created orbits of $k$-hyperedges with the command `orb := Orbits(perm, Combinations ([1..order], k), OnSets)`, which returns a duplicate-free list of orbits where each orbit is a set of $k$-hyperedges. The command `Combinations()` creates all possible $k$-hyperedges that are divided into orbits depending on the permutations of the left multiplication. Since the orbit for each $k$-hyperedge contains a set of equivalent $k$-hyperedges, the orbits are disjoint. By creating the combinations of orbits with `Combinations(orb)`, we obtained a $k$-hypergraph, which either admits or does not admit a HRR for a given group. The GAP system does not have a specific object for a $k$-hypergraph. Thus, we decided to represent a $k$-hypergraph as an incidence structure which preserves symmetries and its automorphism group is isomorphic to the automorphism group of the $k$-hypergraph. We depicted the incidence structure $I$ (Fig. 1) as a bipartite graph where the left (black) set of vertices $V_L(I)$ are the elements of a given group and the right (white) set of vertices $V_R(I)$ are $k$-hyperedges of the $k$-hypergraph, i.e. each $v \in V_R(I)$ is a $k$-subset of $V_L(I)$. An edge between two vertices $u \in V_L(I), v \in V_R(I)$ is constructed if and only if $u \cap v \neq \emptyset$. In the GAP, we constructed the graph

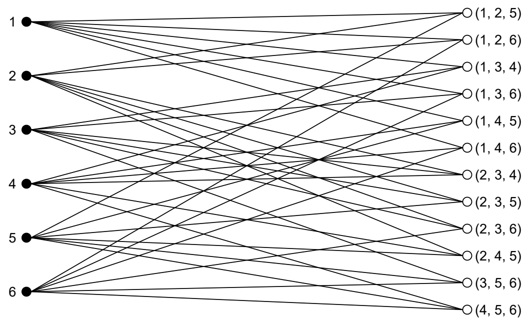

**Figure 1:** Incidence structure of the 3-hypergraph generated from group of order 6

object, i.e. the incidence structure of a $k$-hypegraph, with the command `I := Graph(Group(()), [1..Size(vertices)], OnPoints, function (x,y)return ((Length(vertices[x])= 1 and Length(vertices[y])<> 1)or (Length( vertices[x])<> 1 and Length(vertices[y])= 1))and Length(IntersectionSet(vertices[ x], vertices[y]))>= 1; end, true)` from the *GRAPE* package, where `vertices` $= V_L(I) \cup V_R(I)$. Consequently, we got the automorphism group of the incidence structure with `AutomorphismGroup(I)`. If the automorphism group has the same order as the given group, it means the group admits a HRR otherwise the group does not admit a HRR.

## 4. Optimizations

We introduce the optimization methods used to improve the computational time and the memory usage of the original algorithms proposed in Section 3.

### 4.1. Left vs. right action

In the original algorithms, we used the outcome of the left multiplication of the group given by `LeftMultiplicationGroup()` to get the orbits consisting of $k$-hyperedges. The left and the right group actions are equivalent. Therefore, the orbits corresponding to the left and to the right multiplication actions are isomorphic. The difference between the actions is in the order in which $g_i$ acts on $g_j$. The orbits of the $k$-hyperedges and the effectiveness of the final algorithm are more important. Instead of using two commands `IntoQuasigroup()` and `LeftMultiplication()` from the *loops* package in the original algorithms, we used one command `Action(G , AsList(G), OnRight)` which is by default in GAP, i.e. we did not need to import additional package. The command performs the group action of the right multiplication on the elements of the group. The results of these two approaches are distinct in the arrangement of vertices in the built graphs which is not relevant as all the constructed graphs from the first approach are isomorphic to all the graphs constructed by the second approach.

### 4.2. Creating orbits

The original algorithms use the command `Orbits()` to obtain a list of all orbits for all $k$-hyperedges without duplications. With the increasing size of $k$ and the increasing order of the group, the list of orbits takes a lot of computational time. Instead of the `Orbits()` command, we used the command `OrbitsDomain(perm, Combinations([1..order], k), OnSets)`. The difference between these two commands is in the approach to the set of $k$-hyperedges built by command `Combinations([1..order], k)`. The `Orbits()` works with the set of $k$-hyperedges as seeds compare to the `OrbitsDomain()` that works with the set of $k$-hyperedges as a domain. The domain is a structured set in GAP, which is closed under the action of the group $G$.

### 4.3. Creating and accessing orbit combinations

In the original algorithms, we created all possible combinations of orbits with `Combinations(orb)`. Then, we iterated through all of the combinations in a for-loop until we found a regular representation via a $k$-hypergraph for a given group. The orbits are represented as sets of $k$-hyperedges thus the combinations of orbits are stored as a set of sets of sets of $k$-hyperedges. The resulting object of the `Combinations()` command is demanding on the computational memory and becomes more complex with the increasing $k$ which makes the approach not effective in the way of the memory usage.

The first optimization changed the combinations of orbits to the combinations of indexes of orbits. We used command `Combinations([1..Size(orb)])` that creates combinations on elements $\{1, .., |orb|\}$ representing the indexes of orbits in the `orb` object. The optimization led to lower memory load since combinations of orbits are saved as a set of sets. A disadvantage of the algorithm is that it generates the combinations of orbits that are not needed if we find the HRR for a group in the earlier combinations.

We optimized the generated combinations of orbits by decreasing the amount of computed and not-used combinations. We gradually created all combinations of size $c$, shortly called $c$-combinations, of indexes one by one, where $c$ is in the range $1 \leq c \leq |orb|$. In the beginning, we generated all $c$-combinations of orbits for $c = 1$. We looked if a group admits a HRR via a $k$-hypergraph constructed from one of the $c$-combinations. If we did not find a HRR, we moved to $(c + 1)$-combinations of orbits. We went one by one through the combinations of orbits until we either found a HRR for a given group or we reached $c + 1 > |orb|$. With the optimization, we saved the computational memory and the computational time as we did not have to compute the combinations of orbit's indexes above the $c$-combinations, where we found the HRR of a group. Let us point out that we generated all $c$-combinations, which is not necessary. If we found a combination resulting in a HRR of a group earlier in the $c$-combinations, the rest of the combinations are not used.

To avoid the computation of not-used combinations of orbits, we decided on another optimization with the command `IteratorOfCombinations(orb)` which returns an iterator object through all combinations of the set of orbits. The iterator provides the possibility to loop over the combinations without the repetition and without the need to store all the combinations of orbits. With the use of the iterator, we did not need to precompute the combinations of orbits which makes a significant improvement in the usage of the computational memory and the computational time of the algorithm.

## 4.4. Reducing number of orbit combinations

As the graphs are being constructed based on the combinations of orbits, we easily computed the number of all possible graphs given by $c$-combinations of orbits with $\binom{|orb|}{c}$, where $|orb|$ is the total number of orbits. After observing the original algorithms printouts for groups that admit HRR's, we noticed a given group starts to admit a HRR from some $c$-combinations of orbits and stops to admit a HRR from $(|orb| - c)$-combinations of orbits. The smallest number of $k$-hypergraphs that are the regular representations of a group is at the starting $c$-combinations of orbits, i. e. the $c$-combination from which the group starts to admit a HRR. The number of $k$-hypergraphs which are a HRR's increases from the starting $c$ up to $\frac{|orb|}{2}$ and decreases from $\frac{|orb|}{2}$ to $|orb| - c$, where it is also on its minimum. The maximum number of HRR's via $k$-hypergraphs are obtained at the $\frac{|orb|}{2}$-combinations of orbits. Following these observations, we conjectured that if a given group did not admit a HRR at the $\frac{|orb|}{2}$-combinations of orbits, it could not be found at any $c$-combinations of orbits, i.e. the given group does not have a HRR. Based on the conjecture, we reduced the number of constructed graphs from $2^{|orb|}$ to $\binom{|orb|}{\frac{|orb|}{2}}$ which improved the computational time, in particular for groups not admitting HRR's. We computed the specific $c$-combinations of orbits with the iterator object mentioned in Section 4.3 by specifying the extra parameter $c$: `IteratorOfCombinations(orb, c)`.

## 4.5. Correcting the graph object

To create the graph object expressing the incidence structure in the original algorithms, we used the `Graph()` command from the *GRAPE* package. However, we observed a significant flaw with using the command as it interchanges the sets of vertices $V_L(I)$ and $V_R(I)$. With the interchange, the original algorithms found more automorphisms of the created graph, i.e. automorphism group with a bigger order, as it should be.

We decided to use a command `BlockDesign(order, hyperedges)` from the *DESIGN* package where `order` is the set $V_L(I)$ and `hyperedges` is the set $V_R(I)$. Even though we needed to import an extra package, the new command prevented: interchanging the sets of vertices $V_L(I)$ and $V_R(I)$, mistakes in writing the correct condition in the graph creating function and was more efficient in the computational time.

## 4.6. Optimized algorithm

Regarding the above-mentioned optimizations, we were able to implement a more effective algorithm shown in Algorithm 1. The input to the algorithm is a parameter $k$ for the $k$-hypergraph. The optimization from Section 4.1 concerning the change in the group action is used in Operation 5. Next optimization in the way of computing the orbits from Section 4.2 was applied in Operation 6. Optimizations considering the combinations of orbits from Sections 4.3 and 4.4 is showed in Operation 7. The last optimization of constructing the graph object is presented in Operation 8.

## 5. Results

The algorithms were implemented and executed in the GAP computational system. From several runnings of the algorithms, we were able to get results described in the following subsections.

**Algorithm 1** Pseudocode: Optimized algorithm

```
function (k)
    for $order \leq 32$ do
        for $i \leq$ NrSmallGroup(order) do
            G = SmallGroup(order, i)
            perm = Action(G, AsList(G), OnRight)
            orb = OrbitsDomain(perm, Combinations([1..order], k), OnSets)
            for comb in IteratorOfCombinations(orb, Int(Size(orb)/2)) do
                graph = BlockDesign(order, comb)
                if Size(AutomorphismGroup(graph)) = order then
                    return 'group has HRR'
                    break
                end if
            end for
            return 'group does not have HRR'
        end for
    end for
end function
```

## 5.1. Groups with or without HRR

The first goal of our computational verification was to find which groups of orders less than or equal to 32 admit or do not admit HRR's via $k$-hypergraphs, where $k$ is in the range $0 \leq k \leq |G|$. We obtained results for all groups of orders less than or equal to 32 with their respective values of $k$ except for the group $\mathbb{Z}_2^5$. We were able to compute the existence or non-existence of HRR via $k$-hypergraph for group $\mathbb{Z}_2^5$ only for $k = 3, 4, 5, 28, 29, 30, 31, 32$. All mentioned groups with an associated value of $k$ admit a HRR except groups that are shown in Table 1. With the increasing $k$ and the increasing order of the group, the computations became more complex considering the computational time and the computational memory. The most challenging was to compute a HRR of a group for the $k$ around the middle of the range, i.e., for $k$ around $\frac{|G|}{2}$, due to a large number of orbits. Especially, the computations for the group $\mathbb{Z}_2^5$ got exhaustive, because of an enormous number of orbits. We are still working on computing the results for the group $\mathbb{Z}_2^5$ and values of $k$ in the range $6 \leq k \leq 27$.

## 5.2. Proved conjecture

One of the main goals was to prove a conjecture by Jajcayova [19]. Based on the results from the previous subsection and Table 1, we proved the conjecture for all groups of orders less than or equal to 32 except the group $\mathbb{Z}_2^5$ as we do not have results for all values of $k$ in the range $0 \leq k \leq |G|$. Thus if a group $G$ of order not exceeding 32, except $\mathbb{Z}_2^5$, admits a HRR via a $n$-hypergraph, it also admits HRR's via $k$-hypergraphs, where the $k$ is in the range $n \leq k \leq |G| - n$. In other words, if a group starts to admit a HRR via some $n$-hypergraph, it admits HRR's via $k$-hypergraphs until the $k \geq |G| - n$. For example, the group $\mathbb{Z}_7$ admits a HRR via a $k$-hypergraph for $k = 3, 4$ and does not admit a HRR via a $k$-hypergraph for $k = 0, 1, 2, 5, 6, 7$. The range from the conjecture applied to the group $\mathbb{Z}_7$ gives the range $3 \leq k \leq 4$ which satisfies the conjecture. We continue with our experiments for the remaining open case of the group $\mathbb{Z}_2^5$.

**Table 1**
Groups of order less than or equal to 32 not admitting HRR's via $k$-hypergraphs for $k$ in the range $0 \leq k \leq |G|$, where $|G|$ is an order of a given group

| k | Groups |
|---|---|
| 3 | groups of orders $3, 4, 5$, and $\mathbb{Q}_8, \mathbb{Z}_2^3$ |
| 4 | groups of orders $4, 5, 6$ |
| 5 | groups of orders $5, 6, 7$, and $\mathbb{Q}_8, \mathbb{Z}_2^3$ |
| 6 | groups of orders $6, 7, 8$ |
| 7 | groups of orders $7, 8, 9$ |
| 8 | groups of orders $8, 9, 10$ |
| 9 | groups of orders $9, 10, 11$ |
| 10 | groups of orders $10, 11$, and $\mathbb{Z}_{12}, \mathbb{Z}_3 \rtimes \mathbb{Z}_4, \mathbb{A}_4, \mathbb{Z}_2 \times \mathbb{Z}_6$ |
| 11 | groups of orders $11, 12, 13$ |
| 12 | groups of orders $12, 13$, and $\mathbb{Z}_{14}$ |
| 13 | groups of orders $13, 14, 15$ |
| 14 | groups of orders $14, 15$, and $\mathbb{Z}_{16}, \mathbb{Z}_4^2, \mathbb{Z}_4 \rtimes \mathbb{Z}_4, \mathbb{Z}_2 \times \mathbb{Z}_8, \mathbb{Z}_8 \rtimes_3 \mathbb{Z}_2, \mathbb{Q}_{16}, \mathbb{Z}_2^2 \times \mathbb{Z}_4, \mathbb{Z}_2 \times \mathbb{Q}_8, \mathbb{Z}_4 \circ \mathbb{D}_4, \mathbb{Z}_2^4$ |
| 15 | groups of $order = 15, 16, 17$ |
| 16 | groups of $order = 16, 17$ and $\mathbb{Z}_{18}, \mathbb{Z}_3 \rtimes, \mathbb{S}_3, \mathbb{Z}_3 \times \mathbb{Z}_6$ |
| 17 | groups of $order = 17, 18, 19$ |
| 18 | groups of $order = 18, 19$ and $\mathbb{Z}_{20}, \mathbb{Z}_5 \rtimes_2 \mathbb{Z}_4, \mathbb{Z}_2 \times \mathbb{Z}_{10}$ |
| 19 | groups of $order = 19, 20$ and $\mathbb{Z}_{21}$ |
| 20 | groups of $order = 20, 21$ and $\mathbb{Z}_{22}$ |
| 21 | groups of $order = 21, 22, 23$ |
| 22 | groups of $order = 22, 23$ and $\mathbb{Z}_{24}, \mathbb{Z}_3 \rtimes \mathbb{Q}_8, \mathbb{Z}_2 \times (\mathbb{Z}_3 \rtimes \mathbb{Z}_4), \mathbb{Z}_2 \times \mathbb{Z}_{12}, \mathbb{Z}_3 \times \mathbb{Q}_8, \mathbb{Z}_2^2 \times \mathbb{Z}_6$ |
| 23 | groups of $order = 23, 24, 25$ |
| 24 | groups of $order = 24, 25$ and $\mathbb{Z}_{26}$ |
| 25 | groups of $order = 25, 26$ and $\mathbb{Z}_{27}, \mathbb{Z}_3 \times \mathbb{Z}_9, \mathbb{Z}_3^2 \rtimes \mathbb{Z}_3, \mathbb{Z}_3^3$ |
| 26 | groups of $order = 26, 27$ and $\mathbb{Z}_{28}, \mathbb{Z}_7 \rtimes \mathbb{Z}_4, \mathbb{Z}_2 \times \mathbb{Z}_{14}$ |
| 27 | groups of $order = 27, 28, 29$ |
| 28 | groups of $order = 28, 29$ and $\mathbb{Z}_{30}$ |
| 29 | groups of $order = 29, 30, 31$ |
| 30 | groups of $order = 30, 31$ and $\mathbb{Z}_{32}, \mathbb{Z}_4 \times \mathbb{Z}_8, \mathbb{Z}_2.\mathbb{Z}_8, \mathbb{Z}_2 \times \mathbb{Z}_{16}, \mathbb{Q}_{32}, \mathbb{Z}_2 \times \mathbb{Z}_4^2, \mathbb{Z}_2 \times \mathbb{Z}_4 \rtimes \mathbb{Z}_4, \mathbb{Z}_2 \times (\mathbb{Z}_8 \rtimes_3 \mathbb{Z}_2), \mathbb{Z}_4 \rtimes \mathbb{Q}_8, \mathbb{Z}_2^2 \times \mathbb{Z}_8, \mathbb{Z}_2 \times \mathbb{Q}_{16}, \mathbb{Z}_2^3 \times \mathbb{Z}_4, \mathbb{Z}_2^2 \times \mathbb{Q}_8$ |
| 31 | groups of $order = 31, 32$ |
| 32 | groups of $order = 32$ |

## 5.3. Minimal c-combinations of orbits needed for HRR

We performed several runs of the implemented algorithm and analysed the printouts about the minimal needed $c$-combinations, where we can find the first $k$-hypergraph satisfying the HRR conditions for a given group. From Section 5.2, we know that if a group starts to have a HRR via a $n$-hypergraph, it admits a HRR via $k$-hypergraph as far as $k \leq |G| - n$. The $c$ for the minimal needed $c$-combinations to construct a HRR is at its maximum for the $n$-hypergraph, i.e. the starting $k$-hypergraph, and also for the $(|G| - n)$-hypergraph. The development of the $c$ decreases from $n$ to $\frac{|G|}{2}$ and consequently increases from $\frac{|G|}{2}$ to $|G| - n$. The greatest starting $c$ for $c$-combinations we obtain so far is $c = 6$ for the group $\mathbb{Z}_4^2.\mathbb{Z}_2$. In most of the cases, groups admit HRR 's with $c$-combinations where $c = 1, 2$. Based on the small $c$, we speeded up our computation for some groups and instead of having $\binom{|orb|}{\frac{|orb|}{2}}$ possible combinations of orbits, we have only $\binom{|orb|}{1}$ or $\binom{|orb|}{2}$ combinations of orbits.

An interesting observation was made with regard to the cyclic groups, where we noticed that cyclic groups of $order = 6, 7, 8$ start to admit HRR's with a 2-combinations of orbits and cyclic groups of orders greater than or equal to 9 start to admit HRR's with a 1-combinations of orbits. To obtain HRR's for the dihedral groups via a 3-hypergraph, we always needed a 2-combinations of orbits.

## 5.4. Magma vs. GAP

We had the opportunity to run our algorithm on the same computer in Magma and GAP. Magma is a computational algebra system that provides an environment to work with various mathematical structures and preprogrammed mathematical algorithms. When we ran our algorithm in Magma, the algorithm did not have yet incorporated an optimization from Section 4.4. Nevertheless, we executed the not fully optimized algorithm through Magma and GAP on the same computer. We found out that Magma has a better computational time than GAP. However, Magma has a problem completing the computation of the existence or non-existence of HRR via 3-hypergraph for a group of order 30, because it ran out of the computational memory.

## Acknowledgments

I would like to thank Ján Karabáš from the Matej Bel University for his advice about managing orbits in Section 4.3 and for the possibility to work with Magma in Section 5.4. I would also like to thank Grahame Erskine from the Open University for his very useful suggestion to use the command `BlockDesign()` to describe the graph object in Section 4.5. I would like to thank my supervisor Tatiana Jajcayová for her guidance, suggestions and corrections throughout the writing process. The work was partially supported by G-22-173-00 and VEGA 1/0423/20.

## References

[1] R. Frucht, Herstellung von Graphen mit vorgegebener abstrakter Gruppe, Compositio Mathematica 6 (1939) 239–250. URL: http://eudml.org/doc/88709.

[2] G. Sabidussi, Vertex-transitive Graphs, Monatshefte für Mathematik 68 (1964) 426–438.

[3] C.-Y. Chao, On a theorem of Sabidussi, 1964.

[4] M. A. McAndrew, On graphs with transitive automorphism, Notices of the American Mathematical Society 12 (1965) 575.

[5] W. Imrich, Graphs with transitive abelian automorphism group, Combinat. Theory Appl., Colloq. Math. Soc. János Bolyai 4, 651-656 (1970)., 1970.

[6] M. E. Watkins, W. Imrich, On automorphism groups of Cayley graphs, Periodica Mathematica Hungarica 7 (1976) 243–258.

[7] L. A. Nowitz, On the non-existence of graphs with transitive generalized dicyclic groups, Journal of Combinatorial Theory 4 (1968) 49–51. URL: https://www.sciencedirect.com/science/article/pii/S0021980068800869. doi:https://doi.org/10.1016/S0021-9800(68)80086-9.

[8] L. A. Nowitz, M. E. Watkins, Graphical Regular Representations of Non-Abelian Groups, I, Canadian Journal of Mathematics 24 (1972) 993–1008. doi:10.4153/CJM-1972-101-5.

[9] L. A. Nowitz, M. E. Watkins, Graphical Regular Representations of Non-Abelian Groups, II, Canadian Journal of Mathematics 24 (1972) 1009–1018. doi:10.4153/CJM-1972-102-3.

[10] W. Imrich, On graphical regular representations of groups, Infinite finite Sets, Colloq. Honour Paul Erdös, Keszthely 1973, Colloq. Math. Soc. Janos Bolyai 10, 905-925 (1975)., 1975.

[11] M. E. Watkins, The state of the GRR problem, Recent Adv. Graph Theory, Proc. Symp. Prague 1974, 517-522 (1975)., 1975.

[12] C. D. Godsil, GRR's for non-solvable groups, Algebraic methods in graph theory, Vol. I, Conf. Szeged 1978, Colloq. Math. Soc. Janos Bolyai 25, 221-239 (1981)., 1981.

[13] L. Babai, Finite digraphs with given regular automorphism groups, Periodica Mathematica Hungarica, 1980.

[14] S. Foldes, N. M. Singhi, Regular representation of

Abelian groups by 3-uniform hypergraphs, Ars Comb. 3 (1977) 15–20.

[15] S. Foldes, Symmetries, 1977.

[16] S. Foldes, N. M. Singhi, Regular Representation of Finite Groups by Hypergraphs, Canadian Journal of Mathematics 30 (1978) 946–960. doi:10.4153/CJM-1978-082-9.

[17] R. Jajcay, Representing Finite Groups As Regular Automorphism Groups Of Combinatorial Structures, Ars Comb. 62 (2002).

[18] R. Jajcay, T. Jajcayová, k-hypergraphs with regular automorphism groups, Acta Mathematica Universitatis Comenianae 88 (2019) 835–840. URL: http://www.iam.fmph.uniba.sk/amuc/ojs/index.php/amuc/article/view/1257.

[19] T. Jajcayová, Regular actions of groups and inverse semigroups on combinatorial structures, URL: https://ciencias.ulisboa.pt/sites/default/files/fcul/public/CSA2016-Jajcayova.pdf, 2016.

[20] GAP system for computational discrete algebra, 1986. URL: https://www.gap-system.org/index.html.

[21] L. H. Soicher, GRAPE, 1993. URL: https://www.gap-system.org/Packages/grape.html.

[22] G. P. Nagy, P. Vojtěchovský, loops, 2015. URL: https://www.gap-system.org/Packages/loops.html.

[23] L. H. Soicher, DESIGN, 2006. URL: https://www.gap-system.org/Packages/design.html.