Classifying the non-synchronizing primitive permutation groups

Leonard Soicher

Queen Mary University of London

Computational Group Theory, Oberwolfach, 2021

Introduction

I want to talk about a classification problem for finite primitive permutation groups, which arose from the theory of deterministic finite automata.

This interesting permutation group problem turns out to have strong connections to certain problems in combinatorics, finite geometry, and computation.

Google: synchronizing permutation groups

Here, I concentrate on computation.

Non-synchronizing graphs

Let Γ be a (finite, simple) graph.

Definition

The *chromatic number* $\chi(\Gamma)$ of Γ is the smallest number of colours required to make a *proper vertex-colouring* of Γ, that is, a colouring of the vertices so that adjacent vertices have different colours. The *clique number* $\omega(\Gamma)$ of Γ is the size of a largest set of pairwise adjacent vertices.

Note that $\chi(\Gamma) \geq \omega(\Gamma)$.

Definition

We say that Γ is *non-synchronizing* if Γ is non-null, non-complete, and $\chi(\Gamma) = \omega(\Gamma)$.

Non-synchronizing groups

Let G be a permutation group on a finite set Ω.

Definition

We say that G is *non-synchronizing* if there is a non-synchronizing graph Γ with vertex-set Ω, such that $G \leq \mathrm{Aut}(\Gamma)$.

It is easy to see that if $|\Omega| > 2$ and G is intransitive or imprimitive then G is non-synchronizing. Hence the interest in the non-synchronizing **primitive** groups.

I have classified the non-synchronizing primitive permutation groups of degree up to 464. It turns out that, amongst the 3667 primitive permutation groups having degree $n \in \{2, \ldots, 464\}$, exactly 1093 are non-synchronizing.

Outline of the classification

A list $\mathcal{NS}$ is maintained of known or discovered non-synchronizing graphs for primitive groups. For example, for $d > 1, q > 2$, the Hamming graph $H(d, q)$ is non-synchronizing, having clique number $=$ chromatic number $= q$.

The GAP library of primitive groups is used to provide the primitive groups G of degree n (up to conjugacy in S_n), and for each such G, its O'Nan-Scott type. For a given degree $n \leq 464$, the groups are processed in order of non-increasing size.

If G is 2-set transitive (in particular if G is 2-transitive) then G is synchronizing (i.e. not non-synchronizing).

Primitive groups of prime degree are synchronizing, and primitive groups of prime-squared or prime-cubed degree are mostly handled using a specialized approach due to Peter Cameron.

Suppose now G is a primitive group on $\Omega = \{1, \ldots, n\}$, and we need to consider the non-null non-complete graphs Γ with vertex-set Ω, such that $G \le \mathrm{Aut}(\Gamma)$. These are the non-null non-complete "generalized orbital graphs" for G, and are efficiently constructed and stored using the GRAPE package for GAP.

It is checked (using nauty via GRAPE) whether any such Γ is isomorphic to a graph on the current list $\mathcal{NS}$ of non-synchronizing graphs, and if so, then G is non-synchronizing and we are finished with G.

Otherwise, it needs to be determined whether any such Γ is non-synchronizing.

Now if our primitive G is non-synchronizing, acting as a group of automorphisms of a non-synchronizing graph Γ, then necessarily

$$|\Omega| = \omega(\Gamma)\alpha(\Gamma),$$

where $\alpha(\Gamma)$ is the size of a largest set of pairwise non-adjacent vertices of Γ (equivalently, $\alpha(\Gamma) = \omega(\bar{\Gamma})$, the clique number of the complement of Γ).

For the non-null non-complete graphs Γ under consideration, the condition $|\Omega| = \omega(\Gamma)\alpha(\Gamma)$ can usually be checked using the powerful GRAPE package machinery for cliques (which exploits graph symmetry).

The following result can then be applied unless G is almost simple (recall that G is *almost simple* if there is a non-abelian simple group T such that $T \leq G \leq \mathrm{Aut}(T)$).

Theorem (J.N. Bray, Q. Cai, P.J. Cameron, P. Spiga, and H. Zhang (2019))

Suppose G is a primitive permutation group on a finite set Ω and G is **not** *almost simple. Then G is non-synchronizing if and only if there is a non-null non-complete graph Γ with vertex-set Ω, $G \leq \mathrm{Aut}(\Gamma)$, and $|\Omega| = \omega(\Gamma)\alpha(\Gamma)$.*

Suppose now our primitive G is almost simple, Γ is a non-null non-complete graph with vertex-set Ω, $G \leq \mathrm{Aut}(\Gamma)$, and $|\Omega| = \omega(\Gamma)\alpha(\Gamma)$.

In that case, the (relatively new) proper-vertex colouring machinery in GRAPE, which exploits the automorphism group of Γ, can be applied to determine whether Γ has a proper vertex-colouring using $\omega(\Gamma)$ colours, and hence whether Γ is non-synchronizing.

However, this can take too much time for difficult cases.

For the difficult cases where Γ is in fact non-synchronizing, I usually found (using the GRAPE or DESIGN package) a proper vertex-colouring of Γ using $\omega(\Gamma)$ colours, such that the set of colour-classes is invariant under some small chosen non-trivial subgroup of $\mathrm{Aut}(\Gamma)$.

Parallel computing

For just two cases, to prove that a given graph is synchronizing, I made use of a new hybrid GAP/GRAPE/C program for computing the cliques with given vertex-weight sum in a graph whose vertices are weighted with non-zero d-vectors of non-negative integers.

GAP and GRAPE are used to exploit graph symmetry and to build the top of the search tree. Then, a new C program is used on the Apocrita HPC cluster at QMUL to handle the lower branches of the search tree in parallel.

The most difficult case handled was a graph Δ having 315 vertices, degree 250, automorphism group $J_2.2$, $\omega(\Delta) = 63$, and $\alpha(\Delta) = 5$. The problem was to determine whether the vertices of $\bar{\Delta}$ could be partitioned into 63 of its 1008 cliques of size 5.

The answer (no) took about 33 hours runtime using 100 cores, and showed that Δ has chromatic number greater than its clique number.