Software for finding and classifying cliques of given size

Leonard Soicher

Queen Mary University of London

10th Slovenian Conference on Graph Theory, June 2023

Introduction

I will describe my hybrid **GAP/GRAPE/C** software for finding and classifying the cliques of given size in a graph, making effective use of both the symmetries of the graph and parallel computation.

This software has been instrumental in my recent research, including the classification of certain combinatorial designs, the study of certain finite geometries, and the classification of the synchronizing permutation groups (currently up to degree 494).

GAP, GRAPE, and C

GAP is an internationally developed, freely available, open-source computer system for algebra and discrete mathematics, with an emphasis on computational group theory. See `https://www.gap-system.org`

GRAPE is a **GAP** package for computing with graphs together with associated groups of automorphisms, which are used to store and compute with the graphs efficiently. **GRAPE** is designed for applications in algebraic graph theory, and for studying graphs arising from permutation groups, combinatorial designs, and finite geometries. In particular, **GRAPE** has extensive functionality for clique finding and classification. See `https://gap-packages.github.io/grape/`

C is a fast, compiled programming language.

All graphs in this talk are finite, undirected, with no loops and no multiple edges. A **clique** in a graph is a set of pairwise adjacent vertices.

For simplicity of exposition, I will only consider finding and classifying k-cliques (cliques of size k) in an unweighted graph, although the software described can find and classify cliques with given vertex-weight sum in a graph whose vertices are weighted with non-zero d-vectors of non-negative integers, and also maximal such cliques.

The problem

Let Γ be a graph, with associated group $G \leq \mathrm{Aut}(\Gamma)$, and let k be a non-negative integer.

The problem is to determine a set of k-cliques of Γ, containing at least one representative from each G-orbit of the k-cliques of Γ.

First, a **GAP/GRAPE** function (based on the **GRAPE** function `CompleteSubgraphsOfGivenSize`) is used to perform a partial backtrack search, exploiting G, and using dynamic re-ordering of the search tree to generate a sequence

$$(P_1, A_1), (P_2, A_2), \ldots, (P_t, A_t),$$

where each P_i is a clique of size $\leq k$, and its corresponding A_i is a set of vertices of Γ disjoint from P_i, where if G_i is the (setwise) stabilizer of P_i in G, the following hold:

- for each i, A_i is G_i invariant and each vertex of A_i is adjacent to each vertex of P_i
- each G-orbit of k-cliques of Γ has at least one representative consisting of some P_i extended by $k - |P_i|$ elements belonging to the corresponding A_i
- for each i, we have $|P_i| = k$, or the image of G_i acting on A_i has order $\leq c_1$ (the default is $c_1 = 1$, i.e. the image is trivial) and A_i has size $\leq c_2$ (the use of c_2 is to attempt load balancing in the case of parallel computation).

We next use a **C** program, based on the **GRAPE** clique classifier, except that it does not exploit any graph symmetry, but does use dynamic search tree re-ordering, and also partial proper vertex-colouring to avoid searching in certain induced subgraphs containing no k-clique.

For each tuple (P_i, A_i) above, we use this **C** program (in parallel on the QMUL Apocrita cluster if desired) to determine the set B_i of all the cliques of size $k - |P_i|$ in the subgraph of Γ induced on A_i, so $K_i := \{P_i \cup B \mid B \in B_i\}$ is the set of cliques of size k of Γ which are extensions of P_i by elements from A_i.

Now let K be the union of the sets K_i. Then K is a set of k-cliques of Γ containing at least one representative from each G-orbit of the k-cliques of Γ.

If required, one could then use a program, such as Steve Linton's `SmallestImageSet` (in parallel if desired), to determine a subset of K containing exactly one representative from each G-orbit of the k-cliques of Γ.

An example

Now let Γ be the **non**collinearity graph of the Cohen-Tits near octagon. Then Γ has 315 vertices, is regular with degree 304, and its automorphism group is isomorphic to $J_2{:}2$, of order 1209600.

To determine the cliques of size $k := 90$, I used $G := \mathrm{Aut}(\Gamma)$ as the associated group of automorphisms of Γ, took $c_1 := 1$, $c_2 := 240$, and then my **GAP/GRAPE** function produced $t = 14814$ cases $(P_1, A_1), (P_2, A_2), \ldots, (P_t, A_t)$ as specified above. The computation this far took about one minute on my i5 Linux laptop.

These 14814 cases were then run in parallel using my **C** program on 200 cores of the QMUL Apocrita cluster, taking about 5.5 hours of user time.

These computations returned a total of 31 90-cliques of Γ, but these all turned out to be G-equivalent, and so Γ has just one G-orbit of cliques of size 90.

An easy calculation in Γ then showed that a clique of size 90 is maximal, so Γ has no clique of size 91, and so has clique number $\omega(\Gamma) = 90$.

This shows that, in the Cohen-Tits near octagon, a largest "partial ovoid" has size 90.

To conclude

Please let me know of any interesting clique problems you may have. My software may help!

Thank you.