A GAP PACKAGE FOR COMPUTATION WITH COHERENT CONFIGURATIONS

DMITRII V. PASECHNIK, KESHAV KINI

1. Introduction

Coherent configurations (CCs, for short) appear in the study of permutation groups, and can be considered a topic in algebraic combinatorics. Specifically, they generalize the centralizer algebra of a permutation representation of a finite group. A special class of CCs known as association schemes (cf. e.g. [1]) is also extensively studied. The need to compute with CCs and their representations arises outside their immediate domain[1]. For instance, they play an important role in study of combinatorial properties of symmetric graphs, arising for example in coding theory. The area was founded by the seminal work of Delsarte, Lovasz, and Schrijver in the 1970s, and later received a new lease on life thanks to developments in semidefinite programming (SDP, for short) solvers, that made it possible to numerically optimize linear functions on cones of positive semidefinite elements of matrix algebras, subject to linear constraints. The relevant literature is extensive—we can only cite a small part of it here [11, 4, 3, 14].

One more application we can mention concerns nonassociative algebras related to the Monster sporadic simple group; see e.g. [8].

In a nutshell, CCs allow for "dimension reduction". For example, in coding theory one reduces the computational problem in the space of $2^d \times 2^d$ real symmetric matrices to the space of $d^k \times d^k$ matrices, for a small fixed k.

Another important application domain for CCs is the graph isomorphism problem. For instance, a recent polynomial-time algorithm for isomorphism of graphs with a common forbidden minor [7] constructs a certain CC, which is a generalization of a *stabilization* procedure due to Leman and Weisfeiler [15], who pioneered this approach.

We are currently developing an add-on package for the computer algebra system GAP [6]. This package will enable a user to create, manipulate, and otherwise represent CCs within the GAP system (in particular, by linear representations), and to use upon them other tools GAP provides. A prototype is available [10]. As SDP solvers, needed for applications mentioned above, are difficult to interface with GAP, we are also developing a Sage [13] package that pulls CC data from GAP and calls one of the SDP solvers available there, such as CVXOPT.

Date: 2010-05-11.

1991 *Mathematics Subject Classification.* Primary 05-04; Secondary 20-04, 20B40, 05E30.

Key words and phrases. GAP, coherent configuration, association scheme, permutation group, GRAPE, Sage, semidefinite programming, centralizer ring.

[1]Due to space constraints we will not review here the many works where association schemes and CCs are used in the theory of permutation groups; see e.g. [2], or very recent [9]. We can mention that jointly with Csaba Schneider we are using [10] to work on *synchronizing* permutation groups; see `http://www.maths.qmul.ac.uk/~pjc/LTCC-2010-intensive3/`.

2. Coherent configurations

A *coherent configuration* (or CC) of *degree* n and *dimension* d is a finite collection $\mathcal{A} = \{A_1, \ldots, A_d\}$ of d nonzero 0-1 matrices of dimensions $n \times n$, satisfying the following four axioms [2].

(1) $\sum_{i=1}^{d} A_i = J_n$, the all-1 matrix
(2) It is possible to find some subset of $\mathcal{A}$ whose sum is I_n.
(3) If $A \in \mathcal{A}$, then $A^\top \in \mathcal{A}$
(4) There exist natural numbers $p_{i,j}^k$, $0 < i, j, k \leq d$, such that $A_i A_j = \sum_{k=1}^{d} p_{i,j}^k A_k$ holds for all i, j.

Two CCs are called *isomorphic* if there exists an bijection between them of the form $\varphi : A \mapsto PAP^{-1}$ for some permutation matrix P. The numbers $p_{i,j}^k$ stipulated in the fourth axiom are called the *intersection numbers* of the CC, and are invariant under isomorphism.

Note that a CC can be equally regarded as a partition of Ω^2 for a set Ω of size n. By the second axiom, every CC induces a partition of Ω; the parts are called the *fibers* of the CC, and a CC with only one fiber is called *homogeneous*. For a CC $\mathcal{A}$, each $A \in \mathcal{A}$ is associated with a *left* and a *right fiber*, with the nonzero entries of A occurring only in rows indexed by the left fiber and columns indexed by the right fiber.

In the algebra $M_n(\mathbb{C})$ of all $n \times n$ matrices, $\mathcal{A}$ forms a linearly independent set. This, with the fourth axiom, shows that $\mathcal{A}$ spans a subalgebra $\mathbb{C}[\mathcal{A}]$, which is called its *basis algebra*. This algebra has special properties: it is self-adjoint, it contains I, it contains J (the all-1 matrix), and it is closed under the entrywise (or Hadamard) matrix product.

A concrete example of a CC is the *Schurian CC* arising from a permutation group (G, Ω) [2]. Let the *orbitals* of (G, Ω) be the orbits of G acting on Ω^2 in the natural way; then the characteristic arrays of the orbitals together form a CC. Further, the orbits of G on Ω are the fibers of the CC. The basis algebra of the CC is exactly the *centralizer algebra* of the permutation representation of G in GL_n, considered as a set of matrices in the matrix algebra – that is, the algebra of matrices which commute with all elements of G in the representation.

3. GAP package

3.1. **Functionality.**

We have embarked upon a project to build a package for the computer algebra system GAP that allows users to handle CCs as encapsulated objects. The eventual goal of the project is to incorporate it into the main GAP distribution as a package that implements a wide variety of operations and computations that can be done with CCs.

The current version is based around Schurian CCs, and can do all the obvious tasks, such as computing the Schurian CC arising from a given permutation group, computing the intersection numbers, etc. While packages such as the well-known GRAPE [12] are able to do this in the homogeneous case, we have no such restriction. Since we store CCs in a nontrivial way (actually by indexing CC elements by their left and right fibers and a third parameter), the package also provides full emulation of the surface form of the CC, being able to output the CC's elements directly or in a sparse matrix form. This functionality is also available in the standalone system CoCo by I. Faradjev [5].

The array of intersection numbers $\{p_{i,j}^k\}$ of a CC $\mathcal{A}$ can be separated along the j dimension to produce a set of d many $d \times d$ matrices called the *intersection matrices* of $\mathcal{A}$. It turns out that these matrices span an algebra isomorphic to the basis algebra, called the *intersection algebra* or sometimes the *regular representation* of the CC, and denoted $\operatorname{reg}\mathcal{A}$ [2]. In fact, it is easy to turn it into a *-isomorphism, i.e. a mapping that also preserves the conjugate transpose operation, cf. [4]. This turns out to be crucial for the SDP-related applications. Our package provides functions to compute $\operatorname{reg}\mathcal{A}$ and to output its basis directly or in a sparse form.

We are also able to use the Schurian CC $\mathcal{A}$ arising from a permutation group (G,Ω) to efficiently compute the class sum $\sum_{c\in C}\rho(c)$, where ρ is the permutation representation of (G,Ω) and C is some conjugacy class of G. As the class sums over all C span the center of $\operatorname{span}\rho(G)$, we can compute that algebra as well. Crucially, the latter, together with the knowledge of the irreducible characters $\operatorname{Irr}(G)$ of G, allows us to compute the irreducible representations of $\operatorname{reg}\mathcal{A}$. More precisely, we now have functionality to compute surjections $\phi_\chi : \operatorname{reg}\mathcal{A} \to \operatorname{reg} M_{m_\chi}(\mathbb{C})$, where m_χ is the multiplicity of the irreducible character χ in ρ.

3.2. **Ongoing and future work.** Presently we are developing a method to compute the aforementioned irreducible representations of $\operatorname{reg}\mathcal{A}$, starting from $\phi_\chi(\mathcal{A})$, which is an m_χ^2-dimensional subalgebra of $M_{m_\chi^2}(\mathbb{C})$ isomorphic to $M_{m_\chi}(\mathbb{C}) \otimes I_{m_\chi}$. This will allow us to construct the explicit isomorphism $\operatorname{reg}\mathcal{A} \to \oplus_{\chi\in\operatorname{Irr}(G)} M_{m_\chi}(\mathbb{C})$.

Every CC has a refinement that is Schurian – if nothing else, the discrete partition of Ω^2 is the Schurian CC arising from the permutation group $(1,\Omega)$, and refines all CCs. Thus if we adapt our code to consider coarsenings of Schurian CCs, we can handle CCs in full generality.

References

[1] E. Bannai and T. Ito. *Algebraic Combinatorics I: Association Schemes*. Benjamin/Cummings, 1984.

[2] P. J. Cameron. *Permutation groups*, volume 45 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1999.

[3] E. de Klerk, D. V. Pasechnik, J. Maharry, B. Richter, and G. Salazar. Improved bounds for the crossing numbers of $K_{m,n}$ and K_n. *SIAM J. Discr. Math.*, 20:189–202, 2006.

[4] E. de Klerk, D. V. Pasechnik, and A. Schrijver. Reduction of symmetric semidefinite programs using the regular *-representation. *Math. Prog. B*, 109:613–624, 2007.

[5] I. A. Faradzev and M. H. Klin. Computer package for computations with coherent configurations. In *Proc. ISSAC'91* , 219–221. ACM, 1991.

[6] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.4.12*, 2008. (http://www.gap-system.org).

[7] M. Grohe. Fixed-point definability and polynomial time on graph with excluded minors. In *25th IEEE LICS'10*, to appear, 2010.

[8] A. A. Ivanov, D. V. Pasechnik, A. Seress, and S. Shpectorov. Majorana representations of the symmetric group of degree 4. *J. of Algebra*, submitted.

[9] I. Miyamoto. Computation of isomorphisms of coherent configurations. *Ars Mathematica Contemporanea*, 3(1), 2010.

[10] D. Pasechnik and K. Kini. Cohcfg, a GAP package for coherent configurations (preliminary version). 2010. http://www1.spms.ntu.edu.sg/~dima/software/cohcfg-a1.tgz.

[11] A. Schrijver. New code upper bounds from the Terwilliger algebra and semidefinite programming. *IEEE Trans. Inform. Theory*, 51(8):2859–2866, 2005.

[12] L. H. Soicher. GRAPE: a system for computing with graphs and groups. In L. Finkelstein and W. Kantor, editors, *Groups and Computation*, vol. 11 *DIMACS Series in Discrete Mathematics and Theoretical CS*, 287–291. AMS, 1993.

[13] W. Stein et al. *Sage Mathematics Software (Version 4.4)*. The Sage Development Team, 2010. (http://www.sagemath.org).

[14] F. Vallentin. Symmetry in semidefinite programs. *Linear Algebra Appl.*, 430(1):360–369, 2009.

[15] B. Weisfeiler, editor. *On Construction and Identification of Graphs*, vol. 558 *Lecture Notes in Mathematics*. Springer, Berlin, 1976.

Department of Mathematical Sciences, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore

E-mail address: dima@ntu.edu.sg, kini@member.ams.org