Program of the GAP Days 2014, August 25–29

Version from August 19, 2014 at 15:12

Talks are at Pontdriesch 14/16 in room 008, coding sessions in rooms 003 and 103.

Monday, August 25

10:00	Coding session and discussion
14:00	Welcome session (room 008)
15:00	Vinay Wagh *LessGenerators – finding small generating sets for modules (part of the homalg project)*
15:30	Martin Bies *String theory, sheaf cohomology and the homalg package*
16:00	Johannes Hahn *Coxeter groups and Kazhdan-Lusztig theory in GAP*
16:30	Chris Jefferson *Ferret – a modern C++ rewrite of Partition Backtracking in GAP*
17:00	Max Horn *libsing – an interface between Singular and GAP*
17:30	Christof Söger *NormalizInterface – an interface between normaliz and GAP*

Tuesday, August 26

10:00	Sebastian Gutsche & Max Horn *How to make a GAP package*
16:00	Pedro A. García-Sánchez *Recent progress in the NumericalSgps package*
16:30	Manual Delgado *intpic – a package for drawing integers, by emphasizing some subsets.*
17:00	Hebert Perez-Roses *Graph construction via voltage assignment with GAP*
17:30	Delaram Kahrobaei *TBA*

Wednesday, August 27

10:00	Reimer Behrends *HPC-GAP: Design and Implementation of a Concurrency Model for GAP*
16:20	Markus Pfeiffer *Two (HPC)GAP infrastructure packages in the making: GAPData and Matrix*
16:50	Sebastian Gutsche & Sebastian Posur *CategoriesForHomalg - A GAP-based meta language for category theory based computations & ToolsForHomalg - Tools for caching and propagation*
17:30	Thomas Breuer *Recent progress concerning the GAP packages AtlasRep, CTblLib, CTBlocks, MFER*
19:00	Dinner at the "Labyrinth"

Thursday, August 28

10:00	Alexander Konovalov *Continuous integration, package update mechanism and release management in GAP*

Friday, August 29

10:00	Open discussion: Your wishes for the future of GAP
13:30	Open discussion: Results of the meeting, feedback

http://gapdays2014.coxeter.de/

Abstracts of talks and sessions

Sebastian Gutsche & Max Horn (TU Kaiserslautern & JLU Gießen)

"*How to make a GAP package*"

In this hands-on workshop, we will explain the basic requirements for creating a simple GAP package from scratch. After a brief introduction, participants can immediately apply this with our help. For this, participants should bring their laptops and, if present, some code they want to publish in a package. We also plan to cover more advanced aspects of creating and maintaining a GAP package. Which topics are covered in part also depends on requests by participants. Some possibilities include:

- "Package manuals done right: GAPDoc and AutoDoc" (this will definitely be covered)
- Integrating C / C++ code into a GAP package
- Using GitHub pages as website for your package
- Automating the package release process with GitHub
- The importance of package tests and continuous integration
- Example for automated testing using GitHub and Jenkins
- ...

Hebert Perez-Roses (University of Lleida, Spain)

"*Graph construction via voltage assignment with GAP*"

The voltage assignment technique takes a directed "base" graph B, and a group G, and constructs another graph L with $|L| = |B||G|$ vertices, where $|B|$ is the number of vertices of $|B|$. L is usually called the lift of B by G, and is a generalization of Cayley graphs. The voltage assignment technique has been very successful in the construction of large graphs with small degree and diameter. We are now working on the implementation of this technique in GAP, and we would like to bring into consideration of the GAP community the algorithms and data structures used, as well as to discuss the best alternatives for an efficient implementation.

Vinay Wagh (IIT Guwahati, India)

"*LessGenerators – finding small generating sets for modules (part of the homalg project)*"

A GAP package called "LessGenerators" has been developed by Mohamed Barakat and myself, to implement the Quillen-Suslin algorithm in computer algebra systems SINGULAR and GAP. The package is part of the homalg project. The aim of this package is to provide a tool for finding a minimal generating set for a given module. The package provides universal implementation in the sense of CASs, i.e. it can use any CAS supported by the homalg project for ring arithmetic.