Groups and Certificates
(Part II)

Volker Sorge

University of Birmingham, UK

still joint work with

Arjeh Cohen, Technische Universiteit Eindhoven, The Netherlands

Scott H. Murray, Technische Universiteit Eindhoven, The Netherlands

Martin Pollet, University of Saarbrücken, Germany

Motivation

- Check plausibility/correctness of CAS results
- Have an easy and convincing argument
- Nevertheless use a “clever” argument
  - Certify mathematics like mathematicians
- Are certificates sufficient to
  - construct a formal proof?
  - plan a formal proof?

Implementation

- GAP functions give solutions + certificates
- translation of certificates into ad hoc explanations
- translation of certificates into formal proofs
- provide proof planning machinery in Omega
- abstraction to retain comprehensibility

$\Rightarrow$ integrate the certificates into the reasoning

$\Rightarrow$ construct highly hierarchical plans

Queries — Overview

- What is the order of a group G?
- Is a permutation NOT an element G?

Queries — Overview

- What is the order of a group G?
- Is a permutation NOT an element G?

- Is g an element G?
- Is H a subgroup of G?
- Determine the orbit of $x \in \Omega$ under G?
- What is the stabiliser subgroup for $x \in \Omega$ in G?
- Find a base for G?

Formalisation

- Cycle: duplicate free list of natural numbers
- Permutation: set of disjoint cycles or composition of permutations

Formalisation

- Cycle: duplicate free list of natural numbers
- Permutation: set of disjoint cycles or composition of permutations

$\Rightarrow$ properties give additional proof obligations

Formalisation

- Cycle: duplicate free list of natural numbers
- Permutation: set of disjoint cycles or composition of permutations

$\Rightarrow$ properties give additional proof obligations

- Operator @ to apply the permutations to points
- $g_1 = g_2 \Leftrightarrow \forall_{n \in \mathbb{N}} . g_1 @ n = g_2 @ n$
- Other formalisations straightforward

Annotated Constants

- Always concrete permutations
- Declaration of (a, b, c) denotes a constant with
  - annotation, that it is a cycle of the objects a, b, c
  - definition $($*cons* $a($*cons* $b($*cons* c *nil*$)))$
- Declaration of $\{a, b, c\}$ denotes a constant with
  - annotation, that it is a set containing the objects a, b, c
  - definition $\lambda x.(x = a \vee x = b \vee x = c)$

$\Rightarrow$ $\{(1, 2), (3, 4)\}$ and $\{(3, 4), (2, 1)\}$ denote the same constant

$\Rightarrow$ 'trivial' properties for free

Formalisation of Concepts

$$\textit{Orbit}(G_{\alpha\to o}, @_{\alpha\to\beta\to\beta}, x_\beta) \equiv \lambda y_\beta\centerdot \exists g{:}G\centerdot y = g@x$$

$$\textit{Stabiliser}(G_{\alpha\to o}, @_{\alpha\to\beta\to\beta}, x_\beta) \equiv \lambda g_\alpha\centerdot g \in G \wedge g@x = x$$

$$\textit{StabChain}(G, @, (a :: l)_{\textit{list}}) \equiv \textit{Stabiliser}(\textit{StabChain}(G, @, l), @, a$$

$$\textit{StabChain}(G_{\alpha\to o}, @_{\alpha\to\beta\to\beta}, ()_{\textit{list}}) \equiv G$$

$$\textit{Base}(G_{\alpha\to o}, @_{\alpha\to\beta\to\beta}, l_{\textit{list}}) \equiv \textit{StabChain}(G, @, l) = \{\textit{id}\}$$

Formalisation of Queries

How to formalise ‘Compute the Orbit of 1 under G’

Formalisation of Queries

How to formalise ‘Compute the Orbit of 1 under G’

$\exists x. x = 1G$

Formalisation of Queries

How to formalise ‘Compute the Orbit of 1 under G’

$\exists x . x = 1G$ $\qquad$ $1G = 1G$

Formalisation of Queries

How to formalise ‘Compute the Orbit of 1 under G’

$\exists x.x = 1G$ $1G = 1G$

- ‘compute concrete set’ not expressible
- control of proof planner forces to instantiate concrete objects

Hierarchical Proof Planning

- Proof plans are composed of macro steps:
  methods = tactic + specification
- Execution of methods leads to logic level proof

- minor queries recur frequently in proofs of more complicated queries
- postpone the solution of these queries
- justify minor queries with critical methods
- plan subproofs when executing a critical method

Using Computer Algebra

1. in one generic control rule:
   - compute hints with GAP to instantiate meta-variables (e.g. generators, orbits, stabiliser sets etc.)
   - verified during subsequent planning process

Using Computer Algebra

1. in one generic control rule:
   - compute hints with GAP to instantiate meta-variables (e.g. generators, orbits, stabiliser sets etc.)
   - verified during subsequent planning process
2. in methods:
   - apply GAP to solve equations, apply or multiply permutations,
   - verified when proof plan is executed (including recursive calls to GAP)

Example — Membership

$M = \langle a_1, a_2 \rangle = \langle (1,10)(2,8)(3,11)(5,7), (1,4,7,6)(2,11,10,9) \rangle$

Show that $(1,9,2,8,11,3,10,4,7,5,6) \in M$ holds:

$L_{25} \quad \vdash \quad \{(1,9,2,8,11,3,10,4,7,5,6)\} \in \langle \{a_1, a_2\} \rangle$ InGroup

Example — Membership

$M = \langle a_1, a_2 \rangle = \langle (1,10)(2,8)(3,11)(5,7), (1,4,7,6)(2,11,10,9) \rangle$

Show that $(1,9,2,8,11,3,10,4,7,5,6) \in M$ holds:

$L_{29} \quad \vdash \quad (a_2 * a_1) \in \langle \{a_1, a_2\} \rangle$

$L_{28} \quad \vdash \quad \{(1,9,2,8,11,3,10,4,7,5,6)\} = (a_2 * a_1)$

$L_{25} \quad \vdash \quad \{(1,9,2,8,11,3,10,4,7,5,6)\} \in \langle \{a_1, a_2\} \rangle$ Re-Represent L_{28}, L_{29}

Example — Membership

$M = \langle a_1, a_2 \rangle = \langle (1,10)(2,8)(3,11)(5,7), (1,4,7,6)(2,11,10,9) \rangle$

Show that $(1,9,2,8,11,3,10,4,7,5,6) \in M$ holds:

$L_{29} \quad \vdash \quad (a_2 * a_1) \in \langle \{a_1, a_2\} \rangle$

$L_{28} \quad \vdash \quad \{(1,9,2,8,11,3,10,4,7,5,6)\} = (a_2 * a_1)$ Equal-With-GAP

$L_{25} \quad \vdash \quad \{(1,9,2,8,11,3,10,4,7,5,6)\} \in \langle \{a_1, a_2\} \rangle$ Re-Represent L_{28}, L_{29}

Example — Membership

$M = \langle a_1, a_2 \rangle = \langle (1,10)(2,8)(3,11)(5,7), (1,4,7,6)(2,11,10,9) \rangle$

Show that $(1,9,2,8,11,3,10,4,7,5,6) \in M$ holds:

L_{31}	$\vdash$	$a_1 \in \{a_1, a_2\}$	
L_{30}	$\vdash$	$a_2 \in \{a_1, a_2\}$	
L_{29}	$\vdash$	$(a_2 * a_1) \in \langle \{a_1, a_2\} \rangle$	Prod-Of-Gen L_{31}, L_{30}
L_{28}	$\vdash$	$\{(1,9,2,8,11,3,10,4,7,5,6)\} = (a_2 * a_1)$	Equal-With-GAP
L_{25}	$\vdash$	$\{(1,9,2,8,11,3,10,4,7,5,6)\} \in \langle \{a_1, a_2\} \rangle$	Re-Represent L_{28}, L_{29}

Example — Membership

$M = \langle a_1, a_2 \rangle = \langle (1,10)(2,8)(3,11)(5,7), (1,4,7,6)(2,11,10,9) \rangle$

Show that $(1,9,2,8,11,3,10,4,7,5,6) \in M$ holds:

L_{31}	$\vdash$	$a_1 \in \{a_1, a_2\}$	In-Set
L_{30}	$\vdash$	$a_2 \in \{a_1, a_2\}$	In-Set
L_{29}	$\vdash$	$(a_2 * a_1) \in \langle \{a_1, a_2\} \rangle$	Prod-Of-Gen L_{31}, L_{30}
L_{28}	$\vdash$	$\{(1,9,2,8,11,3,10,4,7,5,6)\} = (a_2 * a_1)$	Equal-With-GAP
L_{25}	$\vdash$	$\{(1,9,2,8,11,3,10,4,7,5,6)\} \in \langle \{a_1, a_2\} \rangle$	Re-Represent L_{28}, L_{29}

List of Methods

- 6 basic ND methods
- 5 methods from set theory
- 3 methods using GAP
- 5 methods from permutation group theory
- 6 domain specific methods (introducing lemmata etc.)
- 6 Critical methods

Experiments

1600 problems: randomly generated permutations in S_5 and S_8.

Generating set	Membership	Nonmembership Unexp.	Nonmembership Expanded	Average Order
2 Elem. of S_5	4.9	68.8	198.9	58.8
4 Elem. of S_5	6.1	88.9	360.5	112.6
2 Elem. of S_8	5.0	160.1	754.6	25217.2
4 Elem. of S_8	6.9	233.7	1313.0	37389.8

Future Work

- Extend our work to graph theory
- Show non-isomorphism of graphs
- . . .