Gecode

an open constraint solving library

Christian Schulte
KTH – Royal Institute of Technology, Sweden
Gecode

- Generic Constraint Development Environment
  - open source
  - C++ library
  - constraint propagation + complete (parallel) search
  - finite domain and finite set constraints
  - complete documentation (reference, tutorial, papers)
  - thousands of users
Overview

- History and facts
  - use cases

- Modeling (interfacing) & programming

- Openness
History

- 2002
  - development started
- 1.0.0
  - Dec 6, 2005
  - 43 kloc, 21 klod
- 2.0.0
  - Nov 14, 2007
  - 77 kloc, 41 klod
- 3.0.0
  - Mar 13, 2009
  - 81 kloc, 41 klod
- 3.7.3 (current)
  - Mar 23, 2012
  - 134 kloc, 56 klod

31 releases

- ... 4.0.0 at end of 2012
History

- 2002
  - development started
- 1.0.0
  - Dec 6, 2002
  - 43 kloc, 21 klob
- 2.0.0
  - Nov 14, 2007
  - 77 kloc, 41 klob
- 3.0.0
  - Mar 13, 2009
  - 81 kloc, 41 klob
- 3.7.3 (current)
  - Mar 23, 2012
  - 134 kloc, 56 klob
- … 4.0.0 at end of 2012

A decade of Gecode!
History

- **2002**
  - development started
- **1.0.0**
  - Dec 6, 2005
  - 43 kloc, 21 klod
- **2.0.0**
  - Nov 14, 2007
  - 77 kloc, 41 klod
- **3.0.0**
  - Mar 13, 2009
  - 81 kloc, 41 klod
- **3.7.3 (current)**
  - Mar 23, 2012
  - 134 kloc, 56 klod
- **... 4.0.0 at end of 2012**
History: Tutorial Documentation

- 2002
  - development started
- 1.0.0
  - Dec 6, 2005
    - 43 kloc, 21 klods
- 2.0.0
  - Nov 14, 2007
    - 77 kloc, 41 klods
- 3.0.0
  - Mar 13, 2009
    - 81 kloc, 41 klods
- 3.7.3 (current)
  - Mar 23, 2012
    - 134 kloc, 56 klods
- … 4.0.0 at end of 2012

Modeling with Gecode (98 pages)
Modeling & Programming with Gecode (448 pages)
Gecode 4.*

- Dynamic symmetry breaking (LDSB)
- Floating point variables and constraints
  - together with Vincent Barichard
- Activity-based search
  - [Michel, Van Hentenryck, CP AI OR 2012]
- Half-reification
  - [Feydy, Somogyi, Stuckey, CP 2011]
- Abstractions for LNS and Restarts
- Propagator groups
  - [Lagerkvist, Schulte, CP 2009]
- New and improved constraints
  - cumulative [Kameugne ea, CP 2011]
  - ...
- MiniZinc 2.0
- ...

...
People

Core team

- Christian Schulte  
  KTH – Royal Institute of Technology, Sweden
- Guido Tack  
  K.U. Leuven, Belgium
- Mikael Z. Lagerkvist

Code


Documentation

- Seyed Hosein Attarzadeh Niaki, Vincent Barichard, Felix Brandt, Markus Böhm, Roberto Castañeda, Gregory Crosswhite, Pierre Flener, Gustavo Gutierrez, Gabriel Hjort Blindell, Sverker Janson, Andreas Karlsson, Håkan Kjellerstrand, Chris Mears, Flutra Osmani, Dan Scott, Kish Shen.
Goals

- **Research**
  - architecture of constraint programming systems
  - propagation algorithms, search, modeling languages, ...

- **Efficiency**
  - competitive (winner MiniZinc challenges 2008-2011, all categories)
  - proving architecture right

- **Education**
  - state-of-the-art, free platform for teaching
Users

- **Research**
  - own papers
  - papers by others: experiments and comparison
  - Google scholar: some 650 references to Gecode

- **Education: teaching**
  - KTH, Uppsala U, U Freiburg, UC Louvain, Saarland U, American U Cairo, U Waterloo, U Javeriana-Cali, ...

- **Industry**
  - several companies have integrated Gecode into products (part of hybrid solvers)
Use Case: Education

Courses feasible that include
- modeling
- principles

but also
- programming search heuristics (branchers)
- programming constraints (propagators)

Essential for programming
- accessible documentation…
- …including many examples
Use Cases: Interfacing

- Quintiq integrates Gecode as CP component
  - in their modeling language
  - Quintiq: fast growing company in advanced planning and scheduling, example: truck scheduling for Walmart US

- Cologne: A Declarative Distributed Constraint Optimization Platform
  - U Penn, AT&T Labs, Raytheon
  - Datalog + constraints in distributed setup
  - [Liu ea, VLDB 2012]

- Whatever language: Java, Prolog (> 1), Lisp (> 1), Ruby, Python (> 1), Haskell, MiniZinc, …
Use Cases: Research

- Benchmarking platform for models
  - lots of people (majority?)

- Benchmarking platform for implementations
  - lots of people
  - requires open source (improve what Gecode implements itself)

- Gecode models as reference

- Base system for extensions
  - Qecode: quantified constraints (Benedetti, Lalouet, Vautard)
  - Gelato: hybrid of propagation and local search (Cipriano, Di Gaspero, Dovier)
  - Gecode interfaces powerful enough: no extension required
Deployment & Distribution

- Open source ≠ Linux only
  - Gecode is native citizen of: Linux, Mac, Windows

- High-quality
  - extensive test infrastructure (around 16% of code base)
  - you have just one shot!

- Downloads from Gecode webpage
  - software: between 25 to 125 per day
  - documentation: between 50 to 300 per day

- Included in
  - Debian, Ubuntu, FreeBSD, …
Modeling & Programming
Architecture

model / interface

search  integer  set

domain agnostic kernel
Architecture

- propagation loop
- backtracking for search
- memory management
Architecture

- search engines
  - depth-first (DFS) and branch-and-bound (BAB)
  - parallel search
  - whatever you fancy: program yourself
search engines
  depth-first (DFS) and branch-and-bound (BAB)
  parallel search

search tool: Gist (millions of nodes)
Architecture

Ingredients:
- variables
- propagators (constraints)
- branchers (search heuristics)

Whatever you fancy: program yourself!
Architecture

- direct C++ modeling or interfacing
- language interfaces: MiniZinc, Java, JavaScript, Lisp, Ruby, Eclipse Prolog, …
Modeling (interfacing)

- Use modeling layer in C++
  - matrices, operators for arithmetical and logical expressions, …

- Use predefined
  - constraints
  - search heuristics and engines

- Documentation
  - getting started 30 pages
  - concepts and functionality 96 pages
  - case studies 76 pages
Modeling (interfacing)

- Constraint families
  - arithmetics, Boolean, ordering, …
  - alldifferent, count (global cardinality, …), element, scheduling, table and regular, sorted, sequence, circuit, channel, bin-packing, lex, geometrical packing, nvalue, lex, value precedence, …

- Families
  - different variants and different propagation strength

- “All” global constraints from MiniZinc have native implementation in Gecode
Gecode ⇔ Global Constraint Catalogue

- 74 constraints implemented:
  - abs_value, all_equal, alldifferent, alldifferent_cst, among,
  - among_seq, among_var, and, arith, atleast, atmost, bin_packing,
  - bin_packing_capa, circuit, clause_and, clause_or, count, counts,
  - cumulative, cumulatives, decreasing, diffn, disjunctive, domain,
  - domain_constraint, elem, element, element_matrix, eq, eq_set,
  - equivalent, exactly, geq, global_cardinality, gt, imply, in,
  - in_interval, in_intervals, in_relation, in_set, increasing,
  - int_value_precede, int_value_precede_chain, inverse,
  - inverse_offset, leq, lex, lex_greater, lex_greatereq, lex_less,
  - lex_lesseq, link_set_to_booleans, lt, maximum, minimum, nand,
  - neq, nor, not_all_equal, not_in, nvalue, nvalues, or, roots,
  - scalar_product, set_value_precede, sort, sort_permutation,
  - strictly_decreasing, strictly_increasing, sum_ctr, sum_set, xor
Programming

- Interfaces for programming
  - propagators (for constraints)
  - branchers (for search heuristics)
  - variables
  - search engines

- Documentation
  - propagators 40 pages 58 pages
  - branchers 22 pages
  - variables 44 pages
  - search engines 12 pages 26 pages
Openness
Open Source

- MIT license
  - permits commercial, closed-source use
  - disclaims all liabilities (as far as possible)

- License motivation
  - public funding
  - focus on research

- Not a reason
  - attitude, politics, dogmatism
Open Architecture

- More than a license
  - *license* restricts what users *may do*
  - *code and documentation* restrict what users *can do*

- Modular, structured, documented, readable
  - complete tutorial and reference documentation
  - ideas based on scientific publications

- Equal rights: clients are first-class citizens
  - you can do what we can do: APIs
  - you can know what we know: documentation
  - on every level of abstraction
Open Development

- We encourage contributions
  - direct, small contributions
    - we take over maintenance and distribution
  - larger modules on top of Gecode
    - you maintain the code, we distribute it

- Prerequisites
  - MIT license
  - compiles and runs on platforms we support
Summary

- Open source libraries require open architecture
  - users need good code to build on

- Open architecture promotes equality
  - client code is first-class citizen
  - encourages code contributions

- Open development fosters research
  - collaboration
  - experiments are reproducible