COmputational INfrastructure for Operations Research (COIN-OR)

Ted Ralphs
Industrial and Systems Engineering
Lehigh University
http://www.lehigh.edu/~tkr2

and many others...

International Conference on Continuous Optimization, Tuesday August 3, 2004

Outline of Talk

- Introduction to COIN-OR
- Introduction to Open Source Software
- History of COIN-OR
- Components of COIN-OR
- The Future of COIN-OR
- How to Help
- Resources

The State of Computational Research

- The majority of the papers being written in OR today are computational in nature or have a computational component.
- Despite this, the pace of computational research is relatively slow and the transfer of knowledge to practitioners is even slower.
- Results of research are generally not reproducible.
- Research codes are buggy, narrowly focused, and lack robustness.
- There are few rewards for publishing software outside of archival journals.
- There is no peer review process for software and referees of computational papers have little to go on.
- Building on previous results is difficult and time-consuming.
- Interoperating with other software libraries (such as LP solvers) is difficult.
- The paradigm encouraged by archival journals does not work well for computational research.

Lifecycle of a Typical Software Package

1. Idea is developed (the easy part).
2. Software is developed containing numerous tweaks and hacks that make the idea work (the time-consuming part).
3. Because of space constraints, the tweaks and hacks are not described in the paper.
4. Afterward, the software is not documented or maintained because there is no incentive for doing so.
5. The only thing archived is what was written in the paper, which is not enough to reproduce the results.
6. If someone else wants to use the idea, they have to repeat 90% of the work again.

The Computational Interface for Operations Research

- To address some of these challenges, the Common Optimization Interface for Operations Research Initiative was launched by IBM at ISMP in 2000.
- Since then, the project has been run informally by a core team of volunteers, consisting of the maintainers of various projects.
- IBM seeded the repository with four initial projects, has hosted its Web site, and has provided funding.
- The goal was to develop the project and then hand it over to the community.
- The project has grown to be self-sustaining and was recently spun off as a nonprofit foundation.

What is COIN-OR?

- The COIN-OR Foundation
  - A recently formed non-profit foundation promoting the development and use of interoperable, open-source software for operations research.
  - A consortium of researchers in both industry and academia dedicated to improving the state of computational research in OR.
  - A venue for developing and maintaining standards.
  - A forum for discussion and interaction between practitioners and researchers.
- The COIN-OR Repository
  - A library of interoperable software tools for building optimization codes, as well as a few stand alone packages.
  - A venue for peer review of OR software tools.
  - A development platform for open source projects, including a CVS repository.
- See `www.coin-or.org` for more information.

Our Agenda

- Accelerate the pace of research in computational OR.
  - Reuse instead of reinvent.
  - Reduce development time and increase robustness.
  - Increase interoperability (standards and interfaces).
- Provide for software what the open literature provides for theory.
  - Peer review of software.
  - Free distribution of ideas.
  - Adherence to the principles of good scientific research.
- Define standards and interfaces that allow software components to interoperate.
- Increase synergy between various development projects.
- Provide robust, open-source tools for practitioners.

Growth of COIN-OR

- Aug00: COIN-OR announced to SRO crowd at ISMP 2000
  - 4 projects: BCP, OSI, VOL, CGL
  - 3 OSI Interfaces: XPRESS, OSL, Vol
  - 2 Cut Generators: Rounding, Lifted Knapsack Cover
- Aug00: DFO project, OsiCpx
- Aug01: OTS project, Odd-hole Cuts, CGL1, MaxCut, MKC, VolLp, and VolUfl Examples
- Oct01: Lift-and-Project (norm1) Cuts, CGL2 Example
- Dec01: Branch-and-Cut Example
- Mar02: Gomory and Probing Cuts
- May02: IPOPT, Osi-Dylp, Osi-SoPlex
- Jul02: CLP and SBB projects, OSI-CLP
- Nov02: OSI-GLPK
- Mar03: COIN-Data: CSP and Set Partitioning Instances

- Apr03: COIN-SMI approved
- May03: Multifario Project, BAC Example
- Jun03: NLPAPI Project
- Today:
  - 11 projects+
  - 8 OSI Interfaces
  - 6 Cut Generators

A Few (Ballpark) Statistics

- 15 core team members from academia, industry, and government.
- Close to half a million lines of code.
- 5 mailing lists with over 500 subscribers
- More than 500 CVS commits a month.
- More than 500 source downloads a month.
- Web site receives 10K hits per month.

COIN-OR Activities

- User's meetings
- Conference sessions
- Workshops and tutorials
- Journal articles
- Conference talks
- Coding contest

Current Status

- We are currently in the process of spinning up the foundation.
- The foundation will be run two boards.
  - A strategic board to set overall direction
  - A technical board to advise on technical issues
- The boards will be composed of members from both industry and academia, as well as balanced across disciplines.
- Membership in the foundation will be available to both individuals and organizations.
- The foundation Web site will be moved to INFORMS servers this summer.
- We currently have a moratorium on accepting new projects, but this should be lifted soon.

What is Open Source?

- A coding paradigm in which development is done in a cooperative and distributed fashion.
- An economic model used by some "for-profit" software ventures.
- This model is followed by a number of well-known software projects.
  - Linux (Red Hat, etc.)
  - Netscape/Mozilla (AOL)
  - Star Office/Open Office (Sun)
  - Apache
- A type of software license (described on the next slide).
- To find out more, see `www.opensource.org` or the writings of Eric S. Raymond (*The Cathedral and the Bazaar*).

Open Source Licenses

- Strictly speaking, an open source license must satisfy the requirements of the *Open Source Definition*.
- A license cannot call itself "open source" until it is approved by the Open Source Initiative.
- Basic properties of an open source license
  - Access to source code.
  - The right to redistribute.
  - The right to modify.
- The license may require that modifications also be kept open.
- Most of the software in COIN-OR uses the CPL, which is a certified open-source license that is much less restrictive than the better-known GPL.
- License compatibility is an issue one has to be very careful about.

Why Open Source?

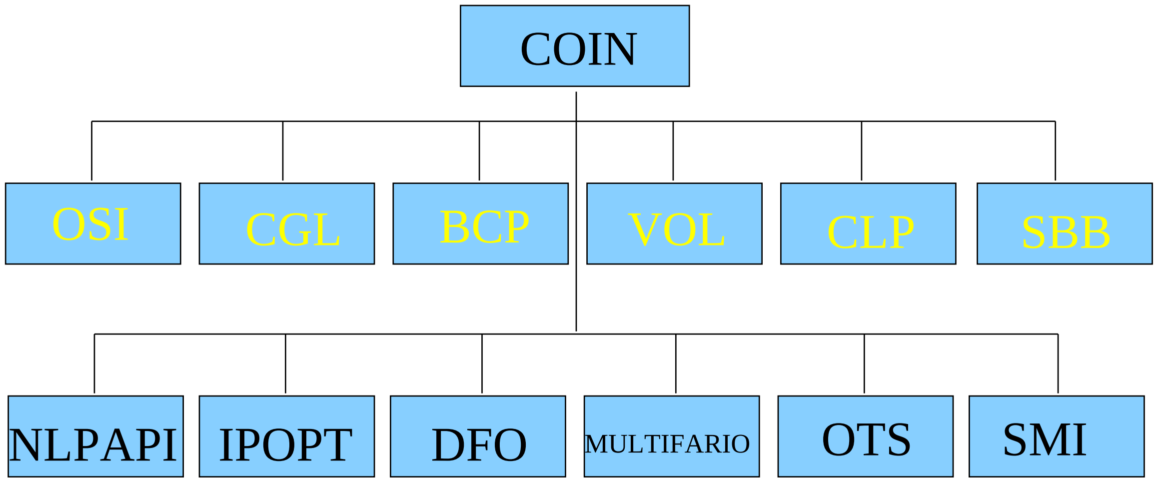
- Increases the pace of development.
- Produces more robust code.
- Introduces an inherent peer review process.
- Creates an informal reward structure.
- Creates an impetus for good documentation.
- Increases the use and distribution of code.
- Prevents obsolescence.
- Promotes reuse over reimplementation.
- Makes collaboration much easier!

"Given enough eyeballs, all bugs are shallow" –ESR

What Are the Downsides?

- Legal issues
- Initial effort is high
- Ongoing maintenance
- Funding issues
- Loss of control
- Loss of commercial opportunities
- ...

Components of the COIN-OR Library



Branch, cut, price toolbox

- OSI: Open Solver Interface
- CGL: Cut Generator Library
- BCP: Branch, Cut, and Price
- VOL: Volume Algorithm
- CLP: COIN-OR LP Solver
- SBB: Simple Branch and Bound
- COIN: COIN-OR Utilities

Stand-alone components

- IPOPT: Interior Point Optimization
- NLPAPI: Nonlinear Solver Interface
- DFO: Derivative Free Optimization
- MULTIFARIO: Solution Manifolds
- OTS: Open Tabu Search
- SMI: Stochastic Modeling Interface

OSI Overview

- Uniform API for accessing solvers.
- Currently, only linear and integer linear solvers are supported.
- Interface to commercial codes
  - CPLEX
  - OSL
  - XPRESS-MP
  - FortMP
  - MOSEK
- Interfaces to academic codes
  - CLP
  - Volume
  - dylp
  - SoPlex
  - GLPK
  - SYMPHONY

OSI Design (Current)

- The interface is defined by means of a C++ virtual class.
- For each solver, there is a corresponding derived class.
- Methods in each derived class translate the standard API into calls to the underlying solver's library.
- Code developed using the OSI should work with any supported solver just by relinking.

OSI Example

```
#include "OsiClpSolverInterface.hpp"

int main(void){
   OsiSolverInterface *si;
   si = new OsiClpSolverInterface;
   si->readMps("p0033");
   si->initialSolve();
   if ( si->isProvenOptimal() ) {
      std::cout << "Found optimal solution!" << std::endl;
      std::cout << "Objective value is " << si->getObjValue() << std::endl;
      int n = si->getNumCols();
      const double *solution;
      solution = si->getColSolution();
   } else {
      std::cout << "Didn't find optimal solution." << std::endl;
   }
   return 0;
}
```

OSI Design (Next Generation)

- Separate `OsiModel` and `OsiAlgorithm` base classes.
- Multiple derived model and algorithm classes.
- A model can be manipulated independently and is not bound to a solver until an algorithm is invoked.
- An `OsiModel` is a collection of `OsiVariable` and `OsiConstraint` objects.
- Each such object has an `OsiDomain`.
- An `OsiConstraint` consists of a method to produce its value as a function of `OsiVariable`s.
- We have a lot of technical issues to work out before this will be reality.

COIN Utility Library Overview

- Contains classes for
  - Storage and manipulation of sparse vectors and matrices.
  - Factorization of sparse matrices.
  - Storage of solver warm start information.
  - Message handling.
  - Reading/writing of MPS files.
  - Presolving LPs.
  - Other utilities (simultaneous sorting, timing, ...).
- These are the classes common to many of the other algorithms.

CGL Overview

- Methods to generate cuts valid for mixed integer programs.
- Uses the OSI to access an LP relaxation and return violated cuts.
- Cut classes currently in the CGL
  - Simple rounding
  - Gomory
  - Knapsack cover
  - Rudimentary lift-and-project
  - Odd hole
  - Probing
  - Clique
  - Flow covers coming soon.

BCP Overview

- A state of the art parallel framework for branch, cut, and price.
- Manages LP-based branch and bound with dynamic generation of cuts and variables.
- Interfaces with the LP solver through the OSI.
- Can also interface with the CGL.
- Subproblems are collections of cuts and variables.
- User derives C++ class corresponding to each class of cuts and/or variables.
- User can override default methods to completely customize the behavior of the solver.

VOL Overview

- A subgradient method that produces both primal and dual solutions.
- Can be used to obtain approximate primal solutions to linear programs quickly.
- Has an OSI interface.
- Can be warm started, just as with the simplex algorithm.
- Can be used within an LP-based branch and bound framework.
- Effective for combinatorial problems where the LP relaxations are especially difficult to solve.
- Can also be used in a Lagrangian relaxation setting.

CLP Overview

- A full-featured, open source LP solver.
- Has interfaces for primal, dual, and network simplex.
- Can be accessed through the OSI.
- Reasonably robust and fast.

SBB Overview

- A lightweight generic MILP solver.
- Uses OSI to solve the LP relaxations.
- Uses CGL to generate cuts.
- Optimized for CLP.

IPOPT Overview

- IPOPT implements a primal-dual interior point algorithm for large-scale nonlinear optimization.
- It is a general purpose solver that does not focus on problems with a particular structure.
- It is capable of handling very large-scale instances ($O(10^6)$ variables).
- Has both AMPL and CUTEr interfaces.

NLPAPI Overview

- NLPAPI is a subroutine library for building nonlinear programming problems.
- The general form is an objective with a set of simple bounds, equality and inequality constraints.
- It is built around the *group partially separable* structure that LANCELOT defines.
- However, constraints and objective may also be defined as functions of the problem variables.

DFO Overview

- DFO is used for solving general nonlinear optimization problems that have these characteristics:
  - the model is relatively small (less than 100 variables),
  - the objective function is relatively expensive to compute,
  - derivatives are not available and cannot be estimated efficiently.
- There may also be some noise in the function evaluation procedures.
- Such models arise in engineering design, where the objective function evaluation is a simulation package treated as a black box.

Multifario Overview

- An implicitly defined manifold is a set of points that satisfy $F(u) = 0$, for some smooth $F : \mathbf{R}^n \to \mathbf{R}^{n-k}$.
- This set of points consists of pieces of k-dimensional manifolds that meet at $(k-1)$-dimensional manifolds of singular points.
- Multifario is a subroutine library that implements a continuation method for computing such manifolds.
- Such manifolds arise in dynamical systems.
- Multifario takes as input a set of initial points on the manifold, and computes the manifold of points connected to the initial points.

OTS Overview

- An object-oriented tabu search framework.
- User defines the following.
  - Solution structure
  - Objective function
  - Tabu list
  - Move
  - Move manager
- OTS takes care of the rest.

SMI Overview

- SMI is a solver interface for stochastic programs.
- SMI is intended to be like OSI in the sense that an `SmiXX` object is an implementation derived from a base class that takes care of
  - handling probability distributions,
  - managing problem generation,
  - interacting with solvers to obtain solution information, etc.
- SMI uses OSI to access an underlying solver, so any OSI-enabled solver can be used.

In the Pipeline

- Next generation of OSI.
- Advanced Library for Parallel Search (ALPS)
  - Framework for implementing (parallel) algorithms based on tree search.
  - Generalizes notions from BCP.
  - Base layer makes no assumptions about bounding method (could be nonlinear).
  - Additional layers will implement specific algorithm classes, such as BCP.
- SYMPHONY is a customizable callable library for solving mixed-integer linear programs, sequentially or in parallel.
- SYMPHONY has an OSI interface.

Getting and Installing the Code

- Code can be dowloaded either directly from CVS or in daily tarballs at

  `www.coin-or.org`

- There are ports for Windows and many flavors of Unix.
- Instructions for installation are available in the FAQs.

Goal for the Future

- Formalize procedures.
- Make it easier to contribute code.
- Improve documentation.
- Improve ease of use.
- Formal versions and releases.
- Broaden user base.
- Find sources of funding!

How You Can Help

- Contribute your software.
- Use the software that's in the repository already.
  - Help document.
  - Contribute bug fixes.
- Help with development
  - New cut classes for CGL.
  - New interface classes for OSI.
- Visit `www.coin-or.org` for more information.