A Gentle Introduction to COIN-OR's Optimization Solver Interface (OSI)

Brady Hunsaker

John Forrest

Lou Hafer

Robin Lougee-Heimer

Ted Ralphs

Matthew Saltzman

hunsaker@engr.pitt.edu

Outline

- COIN-OR and OSI
- **Using OSI in your code**
- **Examples and possibilities**
- Accessing documentation
- Downloading, configuring, and compiling OSI
- Asking for help

What is COIN-OR?

COmputational INfrastructure for Operations Research.

- **A consortium** of researchers and practitioners dedicated to improving the state of computational research in OR.
- **An initiative** promoting the development and use of interoperable, open-source software for operations research.
- **A repository** of open-source software for OR.
- Incorporated as the **COIN-OR Foundation, Inc.**, in March, 2004. Nonprofit application pending.

The COIN-OR Repository

- A library of interoperable software tools for building optimization codes, as well as several stand-alone packages.
- A venue for peer review of OR software tools.
- A development platform for open-source projects, including a CVS repository.
- Currently hosted by IBM, in process of moving to INFORMS.

Some COIN-OR Components

OSI *an open solver interface layer*

COIN COIN-OR utility library

BCP a parallel branch-cut-price framework

CGL a cut generation library

SBB Simple Branch and Bound, a branch and cut code

CLP COIN LP, a native simplex solver

VOL the Volume Algorithm

Optimization Solver Interface (OSI)

Uniform interface to LP/IP solvers:

- CLP (COIN-OR)
- CPLEX (ILOG)
- dylp (dynamic LP; BonsaiG LP Solver)
- GLPK (GNU LP Kit)
- OSL (IBM)
- SoPlex (Konrad-Zuse-Zentrum für Informationstechnik Berlin)
- Volume (COIN-OR)
- XPRESS (Dash Optimization)

Reasons to use COIN-OR OSI

- Learn one API for many solvers
- Perform development with ‘white box’ open source solvers.
- Switch easily from one solver to another

Steps to use OSI

1. Download source code
2. Configure based on available solvers
3. Compile
4. Create a makefile for your project (optional)
5. **Use OSI in your code**

C++ basics

- Related data and functions (*methods*) are grouped together into *objects*
- Usually, data in objects is accessed through functions
- In OSI, the main objects come from the class `OsiSolverInterface`
- Function calls are referenced similar to structures in C. Say the object is `OsiSolverInterface *si`
  - `si->getObjValue()`
  - `si.getObjValue()` if `si` is not a pointer

Online C++ references

- C++ Annotations by Frank B. Brokken; intended for people who know C and want to learn C++.

`http://www.icce.rug.nl/documents/cplusplus/cplusplus.html`

- C/C++ Reference, `http://www.cppreference.com/`

Using OSI

Solver dependent parts:

- Include the header files for solver(s) you want to use.
- Create an `OsiXxxSolverInterface` object.

Solver independent:

- Call functions to load/create a problem.
- Call functions to solve the problem.
- Call functions to report on the solution, modify the problem and re-solve, or do something else

A simple example: basic.cpp

Read MPS file and solve.

- `si->readMps("p0033")`
- `si->initialSolve()`
- `si->isProvenOptimal()`
- `si->getObjValue()`
- `si->getNumCols()`
- `si->getColSolution()`

Changing solvers is easy: `basic2.cpp`

- Change the include file
- Change the instantiation of the object

Querying the interface: `query.cpp`

- `si->getNumRows()`
- `si->getNumCols()`
- `si->getNumElements()`
- `si->getColUpper()`
- `si->getIterationCount()`
- `si->isProvenPrimalInfeasible()`
- `si->isProvenDualInfeasible()`
- `si->isIterationLimitReached()`
- There are many more.

Setting some parameters: `parameters.cpp`

- `si->setIntParam( OsiMaxNumIteration, 10)`
- `si->setDblParam( OsiPrimalTolerance, 0.001)`
- `si->getStrParam( OsiSolverName, solver)`

Building an instance: `build.cpp`

Uses the COIN utility library to work with sparse vectors and sparse matrices.

- Must include needed header files
- Two new classes: `CoinPackedVector` and `CoinPackedMatrix`
- Each has its own methods
  - `row1.insert(0, 1.0);`
  - `matrix->setDimensions(0, n_cols);`
  - `matrix->appendRow(row1);`
- Documentation also available for these classes.
- `si->loadProblem(*matrix, col_lb, ... )`

Solver-specific functions: `specific.cpp`

This depends on the specific solver interface.

- `clpPointer = (dynamic_cast<OsiClpSolverInterface *>(si))->getModelPtr();`
- `clpPointer->setLogLevel(0)`

In CPLEX, for example, you need to get the model pointer and environment pointer—there is a method to retrieve each.

Other features of OSI

- Several methods for loading problems
- Re-solve after modifying problem
- Integer programs
- “Hints” for presolving, scaling, using dual simplex
- Warm starts and hot starts
- Simplex-level controls for basis, pivots, etc. (currently only implemented for CLP, I think)

Accessing documentation

- Most documentation is extracted from the code itself using `doxygen`.
- `make doc` will generate documentation locally (on your computer) in HTML format. You can easily add documentation for your modifications and additions.
- Some tutorial examples and links to the documentation available at `http://sagan.ie.lehigh.edu/coin/` (maintained by Matt Galati)
- Also available online at COIN-OR website: `http://www.coin-or.org/`

Downloading, Configuring, Compiling

- Download tarball from www.coin-or.org.
- Repository can also be accessed with CVS.
- Configuration in the Makefiles directory
  - Edit Makefile.location to tell COIN-OR which solvers are available and where they are
  - Edit Makefile.<platform> (e.g. Makefile.Linux, Makefile.SunOS) if you want to control the compiler, linker, etc. The default settings are probably OK.

Compiling, Makefiles

- Compile with the command `make` in the directory Coin and then Osi. May need to do `make` in subdirectories of Osi as well, such as OsiGlpk and OsiDylp, depending on the solvers available.
- Create a Makefile for your project that indicates the location of OSI headers and libraries.

Asking for help

We want to help make your use of OSI successful!

- First review the appropriate documentation—the answer may be there.
- Send email to `coin-discuss@www-124.ibm.com`. **This address is likely to change soon–check www.coin-or.org before sending.**
- In your email, give as much detail as you can:
  - Operating system
  - COIN-OR modules (OSI, CLP, etc.)
  - Solvers
  - Error messages