The Optimization Services Project on COIN-OR

Robert Fourer, Jun Ma
Industrial Engineering & Management Sciences
Northwestern University
[4er,maj]@iems.northwestern.edu

Kipp Martin
Graduate School of Business
University of Chicago
kmartin@gsb.uchicago.edu

10th INFORMS Computing Society Conference
Coral Gables, Florida — Wednesday, January 3, 2007 — WD02.1
“Optimization Services” (OS)

A framework for providing optimization tools
- XML-based
- Service-oriented
- Distributed
- Decentralized

A project for implementing such a framework
- Straightforward and ubiquitous access
- Powerful solvers

Using a robust service-oriented architecture
- Linking modeling languages, solvers, schedulers, data repositories
- Residing on different machines, in different locations, using different operating systems.
OS on the Internet

*Home site:* www.optimizationservices.org
- Overview, standards, publications, presentations, FAQs
- Contact information, downloads, licenses

*Developer site:* see.us.for.details
- Login, register, wiki, source repository, timeline, search

*Newsgroup:*
  groups.google.com/group/optimizationservices

*COIN mailing list:*
  list.coin-or.org/mailman/listinfo/os

...newsgroup and COIN mailing list are automatically cross-posted
OS Licenses, etc.

Written in multiple languages
- C/C++
- Java
- .NET

Released as open source code
- Under the Common Public License (“CPL”)

Available as a COIN-OR project
- Later this year
- Once complex build issues have been ironed out . . .
OS Builds: Platforms

Unix (fairly stable)
- Mac
- Linux

Windows (being tested)
- Windows (MS Visual Studio)
- Cygwin (gcc)
- MSYS (gcc, cl.exe)
OS Builds: Integration

Core (OSCommon library)

Modeler side
- AMPL/.
- LINGO, What’s Best (planned)
- MATLAB

Solver side
- COIN OSI
- AMPL/ASL
- Linear: CLP, CBC, CPLEX, Impact
- Nonlinear: IPOPT, LINDO, KNITRO (planned)
- CppAD (automatic differentiation)

... some still unstable
... looking for developers to provide others
OS Downloads

*OSxL XML schemas (OSRepresentation library)*
- in a zipped file or individually

*OSxL WSDL files (OSCommunication library)*
- in a zipped file or individually
OS Downloads (cont’d)

Sources and builds on common platforms

- C/C++
  - readers/writers
  - client agent for contacting remote services
  - interfaces to solvers and modeling systems
  - automatic differentiation, etc.

- Java (to be put up)
  - same features as C/C++, plus
    Web Services, server, distributed systems.

- .NET (C#) (to be put up)
  - similar to Java but not as complete
OS Repository

Linear (*netlib basic, infeasible, Kennington*)
- Individual XML (OSiL format) files available now
- Zip files to come

Mixed integer (*mainly from miplib 2003*)

Nonlinear
- CUTE now, more to come

Stochastic
- Thanks to Gus Gassmann

... all known documentation
(source, solution, description, type, etc.)
Standards

*OS framework provides standards in 3 areas*

- Optimization instance representation
- Optimization communication
  - accessing
  - interfacing
  - orchestration
- Optimization service registration and discovery
Standards

Optimization Services Protocol (OSxL)

- **Representation**
  - OSgL - general (schema)
  - OSiL - instance (schema)
  - OSIL - linear (reserved for LP-FML)
  - OSnL - nonlinear (schema)
  - OSrL - result (schema)
  - OSoL - option (schema)
  - OSA L - analysis (schema)
  - OSSL - simulation (schema)
  - OSTL - transformation (XSL)

- **Communication**
  - OShL - hook up (WSDL)
  - OSCL - call (WSDL)
  - OSfL - flow (BPEL*)

- **Registry**
  - **Representation**
    - OSqL - query (schema)
    - OSuL - uri (schema)
    - OSeL - entity (schema)
    - OSP L - process (schema)
    - OSbL - benchmark (schema)
    - OSRegistry - registry (schema)
  - **Communication**
    - OSdL - discover (WSDL)

*OSmL: a modeling language and NOT an Optimization Services Protocol
*Letters not currently used: w, z
OSiL: Optimization Problem Instances

**Design goals**
- Simple, clean, extensible, object-oriented

**Standard problem types supported**
- Linear
- Quadratic
- General nonlinear
- Mixed integer for any of above
- Multiple objective for any of above
- Complementarity
OSiL (cont’d)

Extensions (stable or near-stable)

- User-defined functions
- XML data (within the OSiL or remotely located)
- Data lookup (via XPath)
- Logical/combinatorial expressions and constraints
- Simulations (black-box functions)

Prototypes

- Cone & semidefinite programming
- Stochastic
  - recourse, penalty-based, scenario (implicit or explicit)
  - risk measure/chance constrained
  - major univariate, multivariate, user-defined distributions
  - general linear transformation and ARMA processes
OSrL: Optimization Problem Results

Counterpart to OSiL for solver output

- General results such as serviceURI, serviceName, instanceName, jobID, time
- Results related to the solution such as status (unbounded, globallyOptimal, etc.), substatus, message
- Results related to variables (activities), objectives (optimal levels), constraints (dual values)
- Service statistics such as currentState, availableDiskspace, availableMemory, currentJobCount, totalJobsSoFar, timeLastJobEnded, etc.
- Results related to individual jobs including state (waiting, running, killed, finished), userName, submitTime, startTime, endTime, duration, dependencies, scheduledStartTime, requiredDirectoriesAndFiles.
OSrL (cont’d)

Additional solution support

- Support for non-numeric solutions such as those returned from combinatorial or constraint programming solvers
- Support for multiple objectives
- Support for multiple solutions
- Integration of analysis results collected by the solver
OSoL: Optimization Options

Counterpart to OSiL for solver instructions

- General options including serviceURI, serviceName, instanceName, instanceLocation, jobID, license, userName, password, contact
- System options including minDiskSpace, minMemorySize, minCPUSpeed
- Service options including service type
- Job options including scheduledStartTime, dependencies. requiredDirectoriesAndFiles, directoriesToMake, directoriesToDelete, filesToCreate, filesToDelete, processesToKill, inputFilesToCopyFrom, inputFilesToCopyTo, etc.

Limited standardization of algorithmic options

- Currently only initial values
OSoL (cont’d)

Including support for:

- Various networking communication mechanisms
- Asynchronous communication (such as specifying an email address for notification at completion)
- Stateful communication (achieved mainly through the built-in mechanism of associating a network request with a unique jobID)
- Security such as authentication and licensing
- Retrieving separately uploaded information (when passing a large file as a string argument is inefficient)
- Extended or customized solver-specific or algorithm-specific options
Other XML Schema-Based Standards

Kept by the OS registry

- OSeL (entity, experimental): static information on optimization services (such as type, developer)
- OSpL (process, near stable): dynamic information on optimization services (such as jobs being solved)
- OSbL (benchmark, experimental): benchmark information on optimization services

For use by the discovery process

- OSqL (query, experimental): specification of the query format used to discover the optimization services in the OS registry
- OSuL (uri/url, experimental): specification of the discovery result (in uri or url) sent back by the OS registry
Other Schema-Based Standards (cont’d)

Formats and definitions

- OSsL (simulation, stable): format for input and output used by simulation services invoked via the Optimization Services to obtain function values

- OSgL (general, near stable): definitions of general elements and data types used by other OSxL schemas. Usually included in the beginning of another OSxL schema through the statement:
  `<xs:include schemaLocation="OSgL.xsd"/>

- OSnL (nonlinear, stable): definitions (operators, operands, etc.) of the nonlinear, combinatorial, and other nodes used in other OSxL’s, mainly OSiL
Other WSDL-Based Standards

**WSDL**
- Web Service Definition Language

**WSDLs for OS (stable)**
- OShL (hook): for invoking solver/analyzer services
- OSdL (discover): for invoking optimization registry services to register and discover services
- OScL (call) for invoking simulation services, usually to obtain function values.
www.optimizationservices.org . . .

. . . Questions?