Optimization Services: A Framework For Distributed Optimization

Kipp Martin
Booth School of Business
University of Chicago

November 7, 2010

Outline

What is Optimization Services (OS)?

Instance and Solver APIs

Using the Solver Service

Communication Protocols

Obtaining OS

Future Development

Background – Collaborators

- Robert Fourer – Northwestern University
- Horand (Gus) Gassmann – Dalhousie University
- Jun Ma – Northwester University
- Kipp Martin – University of Chicago
- Tim Middelkoop – University of Florida
- Imre Pólik – SAS Institute
- Wayne Shang – Northwestern University

Background – Motivation

Operations Research: focus is on standalone tools like modeling languages and solvers designed to work on a single machine.

IT Community: focus is on tools like Extensible Markup Language (XML), Service Oriented Architectures (SOA), and Web Services that facilitate distributed computing.

Motivation: The OR community could much more readily achieve its objective of the widespread use of optimization if optimization tools were built into technologies that the IT community is already using.

Background – Objective

Define standards for all activities necessary to support decentralized optimization on the Internet: representation of optimization instances, results, and solver options; communication between clients and solvers; and discovery and registration of optimization-related software using the concept of Web Services.

What Is Optimization Services (OS)?

1. A set of XML schemas optimization instances (OSiL), optimization results (OSrL), and solver options (OSoL).

2. Open source libraries that support and implement these standards/schemas. For each schema a corresponding in-memory object:

    1. OSiL – OSInstance

    2. OSrL – OSResult

    3. OSoL – OSOption

    Each in-memory object is an API with `get()` and `set()` methods. (We have OSxx readers and writers also.)

What Is Optimization Services (OS)?

3. A set of COIN-OR solver interfaces that implement the OS standards:
   - Bonmin
   - Clp (through Osi)
   - Cbc (through Osi)
   - Couenne
   - Dip (see Application Templates)
   - DyLP (through Osi)
   - Ipopt
   - SYMPHONY (through Osi)
   - Vol (through Osi)

What Is Optimization Services (OS)?

4. A set of modeling language interfaces that implement the OS standards:

   AMPL – **OSAmplClient –** inside AMPL this is like any other solver except that you can place calls to remote solver servers.

   ```
   model hs71.mod;
   option solver OSAmplClient;
   option OSAmplClient_options "serviceLocation yourURL ";
   solve;
   ```

What Is Optimization Services (OS)?

4. (continued) A set of modeling language interfaces that implement the OS standards:

   GAMS (using GAMSLinks) – GAMS (23.4 and above) ships with an "OS Solver". At the command line tell GAMS to use the OS solver.

   ```
   gams  eastborne.gms mip=os optfile=1
   ```

   The options file is:

   ```
   writeosrl result.osrl
   readosol solveroptions.osol
   writeosil eastborne.osil
   service  yourURL
   ```

What Is OS?

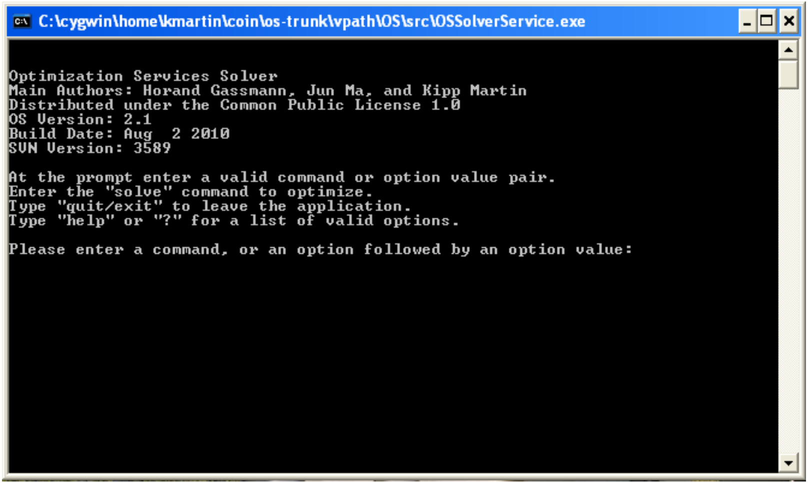
5. A command line executable `OSSolverService` for reading problem instances (in OSiL format, AMPL `nl` format, or MPS format) and calling a solver either locally or on a remote server.

   The `OSSolverService` has an interactive shell.

   You can just "double-click" on the executable and it will guide you through the process.

Using the OSSolverService

Double clicking on `OSSolverService.exe` gives:



C:\cygwin\home\kmartin\coin\os-trunk\vpath\OS\src\OSSolverService.exe

```
Optimization Services Solver
Main Authors: Horand Gassmann, Jun Ma, and Kipp Martin
Distributed under the Common Public License 1.0
OS Version: 2.1
Build Date: Aug  2 2010
SVN Version: 3589

At the prompt enter a valid command or option value pair.
Enter the "solve" command to optimize.
Type "quit/exit" to leave the application.
Type "help" or "?" for a list of valid options.

Please enter a command, or an option followed by an option value:
```

What Is OS?

6. A library for converting MPS files and AMPL `nl` files into the OSiL XML-based format.

7. Standards that facilitate the communication between clients and optimization solvers using Web Services and libraries that support these standards.

8. Java server software that works with Apache Tomcat and Apache Axis. This software uses Web Services technology and acts as middleware between the client that creates the instance, and solver on the server that optimizes the instance and returns the result.

Instance and Solver APIs

The C++ code "mimics" the XML schema.

Every interface (OSInstance, OSResult, OSOption) adheres to these mapping rules!

Instance and Solver APIs

Schema:

```
<xs:complexType name="Constraints">
<xs:sequence>
<xs:element name="con" type="Constraint"/>
</xs:sequence>
<xs:attribute name="number" type="xs:nonNegativeInteger"/>
</xs:complexType>
```

CPP Code:

```
class Constraints{
public:
     Constraints();
     ~Constraints();
     int numberOfConstraints;
     Constraint **con;
}; //class Constraints
```

Instance and Solver APIs

Mapping Rules:

- Each XML schema complexType corresponds to a class in `OSInstance`. Elements in the actual XML file then correspond to objects in the `OSInstance` class.

- An attribute or element used in the definition of a `complexType` is a member of the corresponding in-memory class; moreover the type of the attribute or element matches the type of the member.

- A schema sequence corresponds to an array. For example, the complexType `Constraints` has a sequence of `<con>` elements that are of type `Constraint`.

Instance and Solver APIs

You can use the OSInstance API to build an optimization (linear, nonlinear) instance.

Either work with pointers to the correct objects or use the **convenience** methods.

```
osinstance->setVariableNumber( 2);
osinstance->addVariable(0, "x0", -100, 100, ’C’);
osinstance->addVariable(1, "x1", 0, 1, ’B’);
```

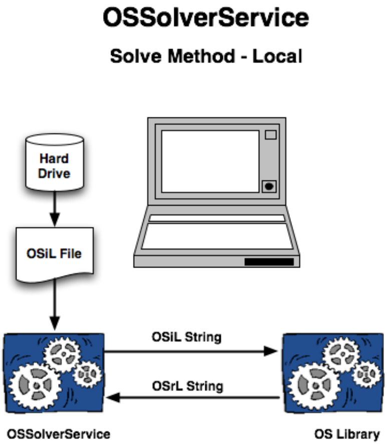
Instance and Solver APIs

Using the solver interface:

```
DefaultSolver *solver  = NULL;

solver = new CouenneSolver();

solver->osinstance = osinstance;

solver->osoption = osoption;

solver->buildSolverInstance(); //a pure virtual function

solver->setSolverOptions(); //a pure virtual function

solver->solve(); //a pure virtual function
```

Using the OSSolverService

The OS build includes the **OSSolverService** executable. This executable can be called locally, or on a remote server.
A local call:



Using the OSSolverService

Here is the local call

```
OSSolverService -config
                ../data/configFiles/testlocal.config
```

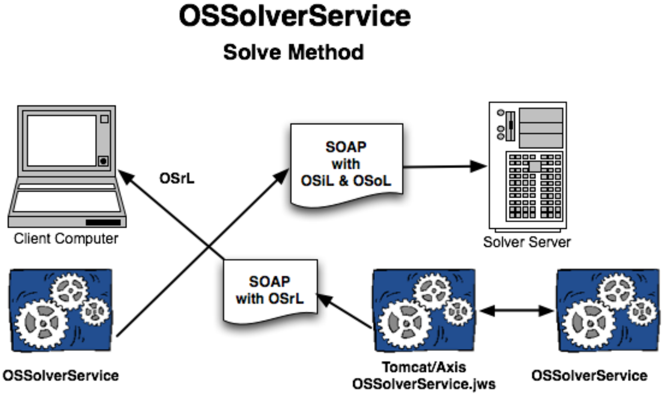
where **testlocal.config** is

```
-osil ../data/osilFiles/parincLinear.osil
-solver ipopt
-serviceMethod solve
```

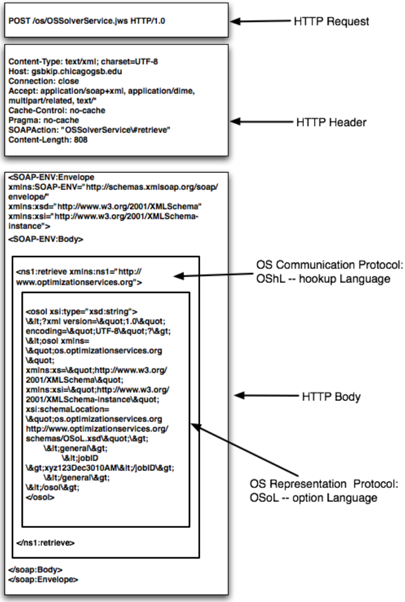
Options at command line override options in the configure file.

Using the OSSolverService

A remote call:



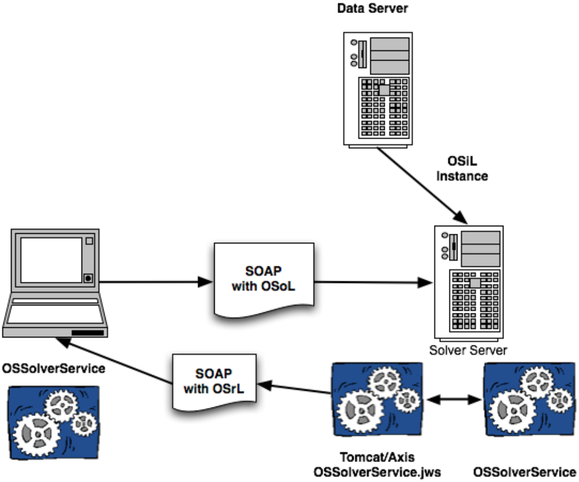
Using the OSSolverService



```
POST /os/OSSolverService.jws HTTP/1.0

Content-Type: text/xml; charset=UTF-8
Host: gsbkip.chicagogsb.edu
Connection: close
Accept: application/soap+xml, application/dime,
multipart/related, text/*
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: "OSSolverService\#retrieve"
Content-Length: 808

<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/
envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
<SOAP-ENV:Body>

  <ns1:retrieve xmlns:ns1="http://
  www.optimizationservices.org">

    <osol xsi:type="xsd:string">
    \&lt;?xml version=\&quot;1.0\&quot;
    encoding=\&quot;UTF-8\&quot;?\&gt;
    \&lt;osol xmlns=
    \&quot;os.optimizationservices.org
    \&quot;
    xmlns:xs=\&quot;http://www.w3.org/
    2001/XMLSchema\&quot;
    xmlns:xsi=\&quot;http://www.w3.org/
    2001/XMLSchema-instance\&quot;
    xsi:schemaLocation=
    \&quot;os.optimizationservices.org
    http://www.optimizationservices.org/
    schemas/OSoL.xsd\&quot;\&gt;
          \&lt;general\&gt;
                \&lt;jobID
    \&gt;xyz123Dec3010AM\&lt;/jobID\&gt;
          \&lt;/general\&gt;
    \&lt;/osol\&gt;
    </osol>

  </ns1:retrieve>

</soap:Body>
</soap:Envelope>
```

Using the OSSolverService

A remote call with data solver server and data server:



Using the OSSolverService

Tell the remote solver were to look for the data. Give the `OSSolverService` a service location and an option file.

The option file specifies the location of the model instance and which solver to invoke.

The solver and model instance may be on different machines.

```
<general>
   <instanceLocation locationType="http">
      http://www.coin-or.org/OS/parincLinear.osil
   </instanceLocation>
   <solverToInvoke>clp</solverToInvoke>
</general>
```

Communication Protocols

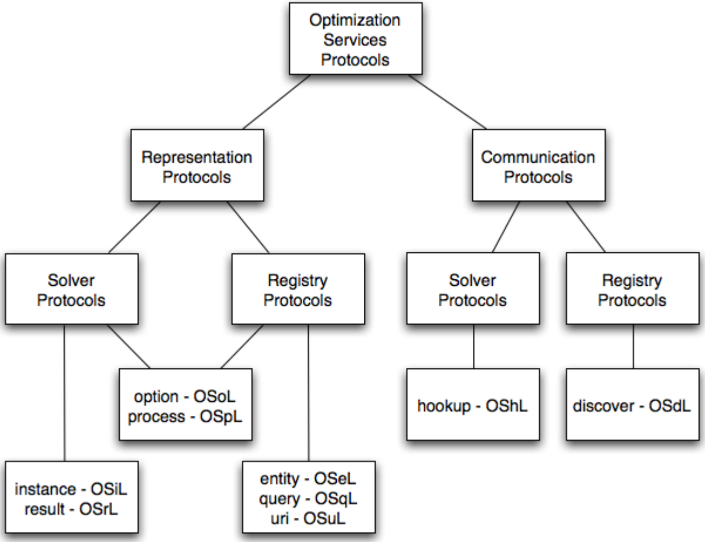


Figure: A Summary of OS Protocols.

Communication Protocols

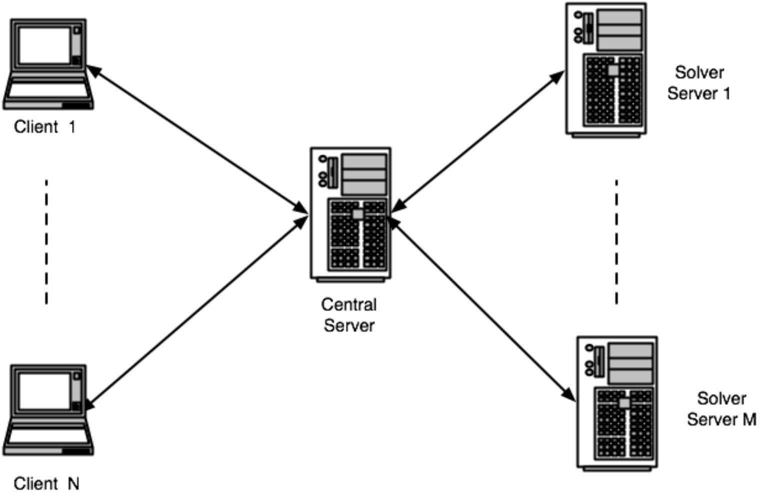


Figure: Centralized Distributed Computing Architecture.

Communication Protocols

Key Trend: An important trend in industry is the move to services oriented architectures and Web services. **All** of the major players such as IBM, Microsoft, Oracle, Sun, etc are talking about service oriented architectures and bringing out products.

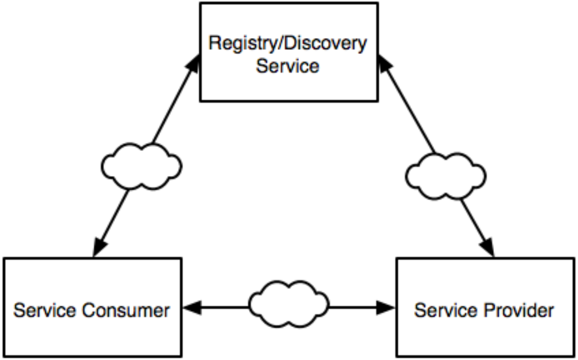


Figure: Services Oriented Architecture (SAO) Paradigm.

Communication Protocols

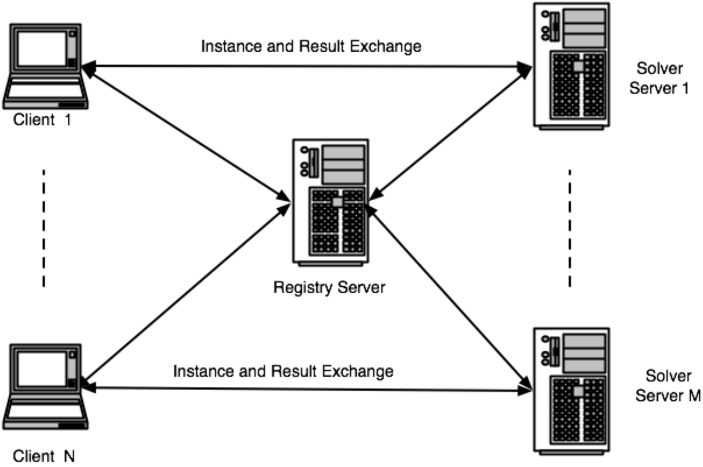


Figure: Centralized Distributed Computing Architecture.

Communication Protocols

An SOA is a **philosophy** for how a distributed component architecture should work it is not a specific technology

Web Services is a technology that implements this philosophy.

Definition: Web Services is SOAP over a transport protocol such HTTP, SMTP, FTP, etc.

HTTP is the most common protocol and is the protocol we use HTTP in our implementations.

Communication Protocols

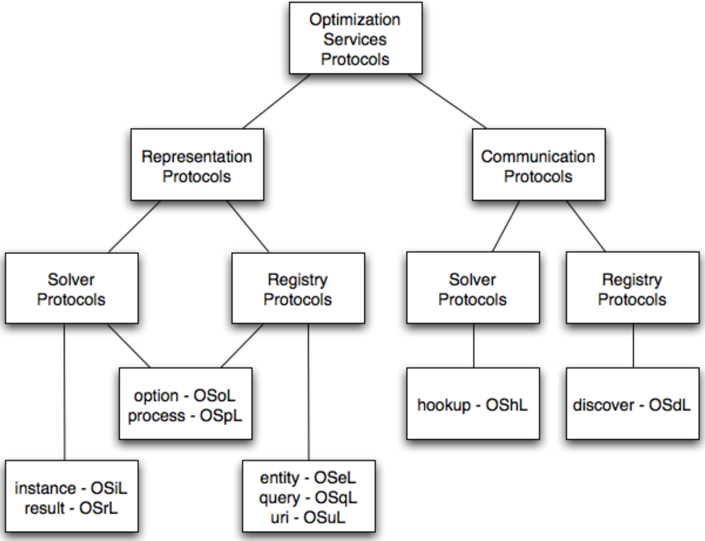
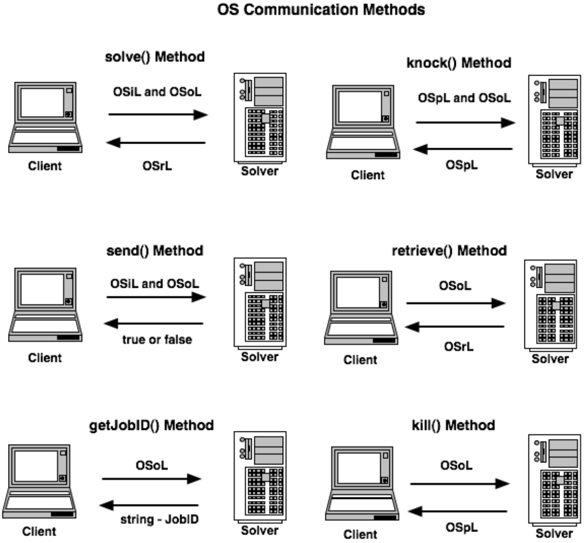


Figure: A Summary of OS Protocols.

Communication Protocols



Obtaining OS

Summary: OS has been built successfully on:

- Various flavors of GNU/Linux
- Windows using Microsoft Visual Studio (both Express and full version – project files available)
- Windows using MSYS and Microsoft cl
- Windows using MINGW and gcc
- Windows using Cygwin and gcc
- Windows using Cygwin and cl
- Mac OS X (both Intel and Power PC)

`https://projects.coin-or.org/TestTools/wiki/NightlyBuildInAction`

Obtaining OS

Binary Format Available for:

- Windows with Microsoft Visual Studio cl compiler (32 bit)
- GNU/Linux 32 and 64 bit
- Mac OS X (Intel)

Future Development

1. **Modification:** Right now we have implemented instance representation, option representation, and result representation. We need OSmL – Optimization Services modification Language for modifying problems. Adding rows, columns, etc.
2. **Python Modeling Languages:** integrate OS with the COIN-OR modeling languages **PuLP** and **Pyomo**.
3. **Decomposition:** Integrate OS with algorithms, in particular DIP, so that block optimization problems can be solved in parallel over the Web.
4. **Broader Instance Representations:** Extend the type of optimization problems handled by OSiL. Leading candidates are stochastic, disjunctive, and cone.
5. **Implement Registry Services:** Extend the communication and representation protocols to registry services.

Related Sessions

1. **TD40:** COIN-OR Under the Hood – I will discuss **CoinEasy** and **ApplicationTemplates**. A discussion of application templates that illustrate using OS with various solvers.

2. **WA40:** Solver APIs II – I will discuss in more detail the OS solver API for COIN-OR solvers.