Optimization Services (OS)

-- A Framework for Optimization Software
-- A Computational Infrastructure
-- The Next Generation NEOS
-- The OR Internet

Jun Ma

Dissertation Defense

Industrial Engineering and Management Sciences
Northwestern University
05/06/2005
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Motivation

Future of Computing
Motivation

But how… with so many type of components

1. Modeling Language Environment (MLE)
   (AIMMS, AMPL, GAMS, LINGO, LPL, MOSEL, MPL, OPL, OSmL)

2. Solver
   (Too many)

3. Analyzer/Preprocessor
   (Analyzer, MProbe, Dr. AMPL)

4. Simulation
   (Software that does heavy computation, deterministic or stochastic)

5. Server/Registry
   (NEOS, BARON, HIRON, NIMBUS, LPL, AMPL, etc.)

6. Interface/Communication Agent
   (COIN-OSI, CPLEX-Concert, AMPL/GAMS-Kestrel, etc.)

7. Low Level Instance Representation
   (Next page)
## Motivation

But how… with so many optimization types and representation formats

<table>
<thead>
<tr>
<th>Type of Optimization</th>
<th>Formats/Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Programming</td>
<td>MPS, xMPS, LP, CPLEX, GMP, GLP, PuLP, LPFML, MLE instances</td>
</tr>
<tr>
<td>Quadratic Programming</td>
<td></td>
</tr>
<tr>
<td>Mixed Integer Linear Programming</td>
<td></td>
</tr>
<tr>
<td>Nonlinearly Constrained Optimization</td>
<td>MLE instances</td>
</tr>
<tr>
<td>Bounded Constrained Optimization</td>
<td>SIF (only for Lancelot solver)</td>
</tr>
<tr>
<td>Mixed Integer Nonlinearly Constrained Optimization</td>
<td></td>
</tr>
<tr>
<td>Complementarity Problems</td>
<td></td>
</tr>
<tr>
<td>Nondifferentiable Optimization</td>
<td></td>
</tr>
<tr>
<td>Global Optimization</td>
<td></td>
</tr>
<tr>
<td>Semidefinite &amp; Second Order Cone Programming</td>
<td>Sparse SDPA, SDPL, …</td>
</tr>
<tr>
<td>Linear Network Optimization</td>
<td>NETGEN, NETFLO, DIMACS, RELAX4</td>
</tr>
<tr>
<td>Stochastic Linear Programming</td>
<td>sMPS</td>
</tr>
<tr>
<td>Stochastic Nonlinear Programming</td>
<td>None</td>
</tr>
<tr>
<td>Combinatorial Optimization</td>
<td>None (except for TSP input, only intended for solving Traveling Sales Person problems.</td>
</tr>
<tr>
<td>Constraint and Logic Programming</td>
<td>None</td>
</tr>
<tr>
<td>Optimization with Distributed Data</td>
<td>None</td>
</tr>
<tr>
<td>Optimization via Simulation</td>
<td>None</td>
</tr>
</tbody>
</table>
Motivation
Look at the NEOS server Web site
Motivation
As if it’s not bad enough …

1. Tightly-coupled implementation (OOP? Why not!)

2. Various operating systems

3. Various communication/interfacing mechanisms

4. Various programming languages

5. Various benchmarking standards
Motivation
Now...

• The key issue is communication, not solution!
• ... and Optimization Services is intended to solve all the above issues.
OUTLINE

1. Motivations

2. Demonstration

3. Optimization Services and Optimization Services Protocol

4. Optimization System Background

5. Computing and Distributed Background

6. Optimization Services Protocol - Representation

7. Optimization Services Protocol - Communication

8. Optimization Services Protocol - Registry

9. Optimization Services modeling Language (OSmL)

10. Future and Derived Research
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Optimization Services (OS)
What is happening behind?

Max $f(x) \quad \text{subject to} \quad \begin{align*}
\text{lb}_1 & \leq g_1(x) \leq \text{ub}_2 \\
\text{lb}_2 & \leq g_2(x) \leq \text{ub}_2
\end{align*}

$f(x)$ can be $\sin(x(1)) + x(x(2))$
$g_1(x)$ can be if$(x(1) > 0)$ then $x(2)$ else $\cos(x(2))$
$g_2(x)$ can be a metric from a finite element simulation
(non-closed form black box function evaluator)
Optimization Services
What is it? – A framework for optimization software
Optimization Services
What is it? – A computational infrastructure
Optimization Services
What is it? – The next generation NEOS

- The NEOS server and its connected solvers uses the OS framework.
- NEOS accepts the OSiL and other related OSP for problem submissions
- NEOS becomes an OS compatible meta-solver on the OS network
- NEOS hosts the OS registry
Optimization Services
What is it? – The OR Internet
Optimization Services Protocol (OSP)

- Application level networking protocol
- Interdisciplinary protocol between CS and OR
Optimization Services Protocol (OSP)
What does the protocol involve? – 20+ OSxL languages

*OSmL*: a modeling language and NOT an Optimization Services Protocol

*Letters not currently used*: w, z

OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Optimization System Background

What does an optimization system look like?

\[
\text{minimize } cx \\
\text{subject to } Ax = b \\
x \geq 0
\]
Optimization System Background

What is the difference between a model and an instance?

**Model**: high-level, user-friendly, symbolic, general, concise, understandable

**Instance**: Low-level, computer-friendly, explicit, specific, redundant, convenient

---

\[
\begin{align*}
\text{minimize } & \quad cx \\
\text{subject to } & \quad Ax = b \\
& \quad x \geq 0
\end{align*}
\]

\[
\begin{align*}
\text{minimize } & \quad -x_1 + 1/2(2x_1^2 - 3x_1x_3 + 4x_2^2 + 5x_3^2) \\
\text{subject to } & \quad 6x_1 + 7x_2 - 8x_3 \geq 9 \\
& \quad x_1 \geq 0, x_2 \geq 0, x_3 \geq 0
\end{align*}
\]
Optimization System Background
What’s the difference between local interfacing and communication agent
Optimization System Background

Why is analyzer important?
Optimization System Background
What’s the difference between a server and a registry

Thick Client
(modeling language environment)

Thin Client
(browser)

Client agent

Optimization Server

Data in HTML
Communication via HTTP

Data in instance generated by MLE
Communication via CORBA or XMLRPC

Solver

Solver

Solver

Solver

Solver

Solver

Solver

Solver

Solver

Solver

Solver
Optimization System Background

What’s a simulation?

\[
\begin{align*}
\text{minimize} & \quad x_1^2 + 2x_2^2 \\
\text{subject to} & \quad 2x_1 + 3x_2 \geq 9 \\
& \quad x_1 \geq 0, x_2 \geq 0
\end{align*}
\]

\[
\min_{x} \quad \text{mySimulation}(x_1, 2, x_2)
\]

\[
\text{subject to} \quad 2x_1 + 3x_2 \geq 9
\]
\[
x_1 \geq 0, x_2 \geq 0
\]

\[
\text{mySimulation}\{
\text{address} = \text{http://somesite.com/mySimulation}
\text{input}:
\text{a}
\text{b}
\text{c}
\text{output}:
\text{value} + \text{confidence} * 0
\}
\]

http://somesite.com/mySimulation
Optimization System Background
AMPL, NEOS and Kestrel

AMPL Model
diet.mod
diet.dat

AMPL Modeling Environment
model diet.mod;
data diet.dat;
option solver minos;

local modeling environment

Minos Solver

Problem Data

Solution

Expression Tree

AMPL-Solver Driver

AMPL: option optimizationservices on

Simulation for Optimization

Central Server

Optimization Client

Optimization Solver

Communication agent

local solving environment

networking

AMPL: solve;

Jun Ma, Optimization Services, May 06, 2005
Optimization System Background
Motorola Optimization System
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
   5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Computing and Distributed Background

What we used in our implementation

1. Java, Open Source Libraries, Object-oriented Programming (OS library)

2. Networking Protocols: HTTP, SOAP, OSP
   (OS server: Tomcat, Axis, OS library)

3. Eclipse IDE for JAVA development

4. XML Spy for XML Schema design
Computing and Distributed Background

XML and XML Dialect (e.g. MathML, OSiL)

MathML

$(2X_1 + 3X_2)^2$

OSiL

$(2X_1 + 3X_2)^2$
Computing and Distributed Background

XML Schema

\[
\begin{align*}
\text{minimize} & \quad 100(x_1 - x_0^2)^2 + (1 - x_0)^2 + 7x_1 \\
\text{subject to} & \quad x_0 + 7x_1 \leq 10 \\
& \quad \ln(x_0x_1) + 7x_0 + 5x_1 \leq 10 \\
& \quad x_0, x_1 \geq 0
\end{align*}
\]

```xml
<variables>
  <var lb="0" name="x0" type="C"/>
  <var lb="0" name="x1" type="C"$objCoef="7.0"/>
</variables>

<x:simpleType>
  <xs:restriction base="xs:string">
    <xs:enumeration value="C"/>
    <xs:enumeration value="B"/>
    <xs:enumeration value="L"/>
    <xs:enumeration value="S"/>
  </xs:restriction>
</xs:simpleType>
```

```xml
<x:s:complexType name="var">
  <xs:attribute name="name" type="xs:string" use="optional"/>
  <xs:attribute name="init" type="xs:string" use="optional"/>
  <xs:attribute name="type" type="xs:string" use="optional" default="C">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="C"/>
        <xs:enumeration value="B"/>
        <xs:enumeration value="L"/>
        <xs:enumeration value="S"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="lb" type="xs:double" use="optional" default="0"/>
  <xs:attribute name="ub" type="xs:double" use="optional" default="INF"/>
  <xs:attribute name="objCoef" type="xs:double" use="optional" default="0.0"/>
  <xs:attribute name="multiplier" type="xs:integer" use="optional" default="1"/>
</xs:s:complexType>
```
Computing and Distributed Background

Other XML Technologies

1. **Parsing**: SAX and DOM models

2. **Transformation**: XSL style sheet

3. **Lookup**: XPath and XQuery

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
  <xsl:template match="/">
    <html>
      <body>
        <hr/>
        <h1>Stocks</h1>
        <p>
          <xsl:for-each select="stock">
            <p>stock: <xsl:value-of select="@name"/>
          </xsl:for-each>
        </p>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```
Computing and Distributed Background

Web services

- **Platform and implementation independent** components
- **Described** using a service description language (WSDL)
- **Published** to a registry of services (UDDI, OS Registry)
- **Discovered** through a standard mechanism (UDDI, OS Registry)
- **Invoked** through a declared API (SOAP)
- **Composed** with other services (SOAP)
Computing and Distributed Background

Web services and SOAP

Architecture View

Protocol View

SOAP

HTTP

TCP/IP

SOAP

HTTP

Web Service

Java application

ANY client!

Java-Structure

SOAP Server

SOAP

SOAP Message

VB application

VB-Structure

SOAP client

SOAP Envelope

SOAP Header

SOAP Body

Payload Document(s)

SOAP Fault

Contains call and response information

POST/services/VersionRequest HTTP/1.0
Content-Length: 123
Host: http://user:ims.nwu.edu/
Content-type: text/xml; charset=utf-8
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope ...>
  <soap:Body>
    <m:SolverVersionRequestMsg
      xmlns:m="http://www.optimizationservices.org/soap-methods">
      <question xsi:type="xsd:string">
        What is the version of the IMPACT MINLP solver?
      </question>
    </m:SolverVersionRequestMsg>
  </soap:Body>
</soap:Envelope>
## Computing and Distributed Background

### Web services and WSDL

<table>
<thead>
<tr>
<th>Operations</th>
<th>OSiL</th>
<th>OSiL, OSoL</th>
</tr>
</thead>
<tbody>
<tr>
<td>String getJobID()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String solve(String, String)</td>
<td>OSiL</td>
<td>OSoL</td>
</tr>
<tr>
<td>String solve(String)</td>
<td>OSiL</td>
<td>OSiL</td>
</tr>
<tr>
<td>String retrieveResult(String)</td>
<td>OSiL</td>
<td>JobID</td>
</tr>
<tr>
<td>String analyze(String)</td>
<td>OSaL</td>
<td>OSiL</td>
</tr>
</tbody>
</table>

See next figure...
Computing and Distributed Background

Web services and WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
  <wsdl:service name="OptimizationSolverService">
    <!-- All port locations, i.e. service URI addresses, are to be found dynamically in the OS registry. They should NOT to be hard coded below in <wsdl:port><wsdl:port> The following is just a hard coded example for reference. Do not use-->
    <wsdl:port name="OptimizationSolverService" binding="os:OptimizationSolverServiceSoapBinding">
      <wsdlsoap:address location="http://www.optimizationservices.org/os/SampleSolverServiceJws"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

"solve" operation is wrapped in a soap envelope over the http protocol and using rpc style

The element should be empty.
Read the comments in <!-- comments -->
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Optimization Services Representation

Who else did this before?

• Many “standards”
• All limited to problem input
• Highly fragmented and no general format
• Fourer, Lopes, and Martin’s LPFML (OSIL)
• Kristjánsson’s OptML
• Bradley’s NaGML
• We are the first in designing
  - Systematic representation of major optimization types
  - All major instance types (result, analysis, input, query, etc.)
  - Web services (SOAP) based communication standards
  - Optimization registry
  - A universal framework
Optimization Services Representation

Optimization Services general Language (OSgL)

General data structures; Included in other schemas

Optimization Services instance Language (OSiL)

- Linear
- Mixed integer
- Bound constrained optimization
- General quadratic optimization
- Nonlinear unconstrained/constrained
- General mixed integer nonlinear
- General nonlinear with user-defined functions
- Global optimization
- General nonlinear with simulations (black-box functions)
- Optimization over simulation/nondifferentiable optimization
- General nonlinear with xml data (either within OSiL or remotely located)
- General nonlinear with data look up (XPath)
- Network and graph definition
- Network programming
- Constraint programming
- Semidefinite programming
- Semi-infinite programming
- Cone programming
- Complementarity problems
- Stochastic linear/nonlinear (distribution based recourse problem, scenario based recourse problem, chance constrained)
- Combinatorial optimization/Heuristic Optimization (TSP, MST, SP, MF, MCF, VRP, Set Covering, Coloring etc. etc.)
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Optimization Services Communication
Optimization Services hookup Language (OShL)

Hookup to solvers, and analyzers
1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Optimization Services Registry

Optimization Services Registry (OSyL) is a native XML database that contains a sequence of service information, each consisting of a triplet (OSeL, OSpL, OSbL).

- **OSyL**: OS registry database; native XML; A sequence of [OSeL, OSpL, OSbL]
- **OSeL**: Optimization Services entity Language
- **OSpL**: Optimization Services process Language
- **OSbL**: Optimization Services benchmark Language
- **OSjL**: Optimization Services join Language (communication)
- **OSkL**: Optimization Services knock Language (communication)

**Operation**

**String knock ()**

- **OSpL**: Optimization Services process Language
- **OSbL**: Optimization Services benchmark Language

XML Example:

```xml
  <description>
    OS registry is a native XML database. It contains a sequence of service, each consisting of a triplet (OSeL, OSpL, OSbL).
  </description>
  <news>
    <el date="2005-04-05">Impact Generalized Mixed Integer Solver joins the OS registry</el>
    <el date="2005-03-23">Ziena Knitro Service joins the OS registry</el>
    <el date="2005-02-27">Lindo MINLP Service joins the OS registry</el>
  </news>
</OSyL>
```
Optimization Services Registry

Optimization Services Registry (OSuL) - A sequence of URI (URL) addresses for service locations with degree of fitness.

Optimization Services Query Language (OSqL) - Like SQL for relational databases, can use XQuery, OSA (analysis), predefined.

Optimization Services Discovery Language (OSdL) - Communicates the query to the OS registry to discover services.

Optimization Services URI Language (OSuL) - Represents a sequence of URI (URL) addresses for service locations with degree of fitness.

Optimization Services Validate Language (OSvL) - A validation service provided by the OS registry that validates all OSxL instances.
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Optimization Services modeling Language (OSmL)
A derived research

Open source and general purpose
Standard based (XQuery input; OSiL output)
Suitable for distributed optimization
XML data is ubiquitous
#set, parameter, and variable constructions

param T;
set PROD;
set LINKS = {PROD, 1..T};
param HC {PROD};
param CAP {1..T};
param DEM {LINKS};
param PCOST {PROD, 1..T};

#VARIBLE DECLARATION
var x {PROD, 1..T} >= 0;
var I {PROD, 0..T} >=0;
var y {PROD, 1..T}binary;

#OBJECTIVE CONSTRUCTION
minimize Total_Cost:
    sum {i in PROD} I[i, 0] + sum {i in PROD, t in 1..T} (PCOST[i, t]*x[i, t] + HC[i]*I[i, t] + FXC[i]*y[i, t]);

# INITIAL INVENTORY CONSTRAINTS
subject to Init_Inv {i in PROD}:
    I[i, 0] = 0.0;

# DEMAND CONSTRAINTS
subject to Balance {i in PROD, t in 1..T}:
    x[i, t] + I[i, t-1] - I[i, t] = DEM[i, t];

# FIXED CHARGE CONSTRAINTS
subject to Fixed_Charge {i in PROD, t in 1..T}:
    x[i, t] <= CAP[t]*y[i, t];

# CAPACITY CONSTRAINTS
subject to Capacity {t in 1..T}:
    sum {i in PROD} x[i, t] <= CAP[t];
Optimization Services modeling Language

4 ways of combining XML with optimization

1. Use XML to represent the instance of a mathematical program
2. Develop an XML modeling language dialect
3. Enhance modeling languages with XML features such as XPath
4. Use XML technologies to transform XML data into a problem instance
OUTLINE

1. Motivations
2. Demonstration
3. Optimization Services and Optimization Services Protocol
4. Optimization System Background
5. Computing and Distributed Background
6. Optimization Services Protocol - Representation
7. Optimization Services Protocol - Communication
8. Optimization Services Protocol - Registry
9. Optimization Services modeling Language (OSmL)
10. Future and Derived Research
Future and Derived Research

- The Optimization Services project
- Standardization
- Problem repository building
- OS server software, library enhancement
- Derived research in distributed systems (coordination, scheduling and congestion control)
- Derived research in decentralized systems (registration, discovery, analysis, control)
- Derived research in local systems (OSI? OSiI, OSrI, OSoI?)
- Derived research in optimization servers (NEOS)
- Derived research in computational software (AMPL, Knitro, Lindo/Lingo, IMPACT, OSmL, MProbe, Dr. AMPL, etc.)
- Derived research in computational algorithm
  - Parallel computing

Library developers, registry/server developers, and other auxiliary developers

Computing on demand and “result on demand”
Acknowledgement

- Robert Fourer
- Kipp Martin
- Tom Tirpak and my colleagues at Motorola Advanced Technology Center
- My committee
- Professor Mehrotra’s group
- My wife and my family

http://www.optimizationservices.org (.net)