Optimization Services (OS)

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

Jun Ma

Industrial Engineering and Management Sciences
Northwestern University

T.J. Watson Lab, IBM, 06/23/2005
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

[Diagram of Future of Computing and related technologies]
Motivation

But how... with so many type of components

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

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>Optimization Type</th>
<th>Example Formats</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, SDPLR</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

M X N drivers

M + N drivers
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.
Demonstration

```
(-- First the xml files :) )
let $foodcostsroot := doc("/xml/foodcosts.xml")
let $foodnutrientsroot := doc("/xml/foodnutrients.xml")
let $nutrientsroot := doc("/xml/nutrients.xml")
(-- Now the sets. Each element in the set corresponds
    to a record or constraints in the relational database :) )
let $foodcosts := $foodcostsroot/dataroot/foodcosts
let $foodnutrients := $foodnutrientsroot/dataroot/foodnutrients
let $nutrients := $nutrientsroot/dataroot/nutrients

return
<mathProgram source="AMPL book by Fournier, Gay, and Kernighan">
    <!-- ILLUSTRATE VARIABLE UPPER BOUNDS -->
    <variables>
        <var name="Buy({$i/food})" lb="{$i/f_min}" ub="{$i/f_max}"/>
    </variables>
    <!-- OBJECTIVE FUNCTION -->
    <obj maxOrMin="min" name="Cost_of_Diet"> SUM(for $j in $foodcosts return
        {$j/cost}^*Buy({$j/food})) </obj>
    <!-- NUTRIENTS CONSTRAINTS This illustrates a JOIN -->
    <con name="nutrient_{${i/nutrient}}"> {$i/n_min} <= SUM(for $j in $foodnutrients where $j/nutrient = $i/nutrient
        return {$j/amt}^*Buy({$j/food})) <= {$i/n_max} </con>
</mathProgram>
```
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?

XML-based standard

Modeler

Model/Data
Parse to OSiL
OSmL

Max $f(x)$

s.t. $lb_1 <= g_1(x) <= ub_2$

$lb_2 <= g_2(x) <= ub_2$

$f(x)$ can be $\sin(x(1)) + x(x(2))$

$g_1(x)$ can be $\text{if}(x(1)>0) \text{then } x(2) \text{ else cost}(x(2))$

$g_2(x)$ 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)

What is it? – Application level networking protocol
– Interdisciplinary protocol between CS and OR

The 7-layer OSI Model
The 4-layer Internet model
Optimization Services Protocol (OSP)
What does the protocol involve? – 20+ OSxL languages

Representation
- OSGl - general (schema)
- OSIL - instance (schema)
- OSIL - linear (reserved for LP-FML)
- OSnL - nonlinear (schema)
- OSrL - result (schema)
- OSoL - option (schema)
- OSaL - 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)
- OSPL - process (schema)
- OSbL - benchmark (schema)
- OSyL - yellow-page (schema)

Communication
- OSdL - discover (WSDL)
- OSjL - join (WSDL)
- OSkL - knock (WSDL)
- OSvL - validate (WSDL)

*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?

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

What is the difference between a model and an instance?

**model** high-level, user-friendly symbolic, general, concise, understandable

**compile**

**instance** Low-level, computer-friendly explicit, specific, redundant, convenient

---

\[
\begin{align*}
\text{minimize} & \quad \mathbf{c}^T \mathbf{x} \\
\text{subject to} & \quad A \mathbf{x} = \mathbf{b} \\
& \quad x \geq 0
\end{align*}
\]

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

More on local interface

9. “COIN OSI Solver” with GLPK Solvers (OSI is in fact a parser)

10 What we suggest for COIN OSI2 with any general solver, say “LANCELOT” or a new COIN Solver

Jun Ma, Optimization Services, June 23, 2005
Optimization System Background

Why is analyzer important?
Optimization System Background

What’s the difference between a server and a registry
Optimization System Background

What's a simulation?

minimize \( x_1^2 + 2x_2^2 \)
subject to \( 2x_1 + 3x_2 \geq 9 \)
\( x_1 \geq 0, x_2 \geq 0 \)

\[
\text{mySimulation} = \begin{cases} 
\text{address} = \text{http://somesite.com/mySimulation} \\
in : \text{http://somesite.com/mySimulation} \\
a \\
b \\
c \\
\text{value} = a^2 + b \cdot c^2 \\
\text{confidence} = 0 \\
\end{cases}
\]

\[
\text{mySimulation}(x_1, 2, x_2) \]
subject to \( 2x_1 + 3x_2 \geq 9 \)
\( x_1 \geq 0, x_2 \geq 0 \)
Optimization System Background
AMPL, NEOS and Kestrel

AMPL Model
model diet.mod;
data diet.dat;
option solver minos;

AMPL Model
model diet.mod;
data diet.dat;
option solver kestrel;

local modeling environment

local solving environment

AMPL: model diet.mod;
AMPL: data diet.dat;
AMPL: option solver minos;
AMPL: solve;

AMPL: model diet.mod;
AMPL: data diet.dat;
AMPL: option optimizationservices on
AMPL: solve;
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**

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

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

```
<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" use="optional" default="C"> <xs:s:simpleType>
    <xs:s:restriction base="xs:string">
      <xs:s:enumeration value="C"/>
      <xs:s:enumeration value="B"/>
      <xs:s:enumeration value="L"/>
      <xs:s:enumeration value="S"/>
    </xs:s:restriction>
  </xs:s:simpleType>
  <xs:s:attribute>
    <xs:s:attribute name="lb" type="xs:double" use="optional" default="0"/>
    <xs:s:attribute name="ub" type="xs:double" use="optional" default="INF"/>
    <xs:s:attribute name="objCoef" type="xs:double" use="optional" default="0.0"/>
    <xs:s:attribute name="multiplier" type="xs:positiveInteger" use="optional" default="1"/>
  </xs:s:attribute>
</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

4. Communication: SOAP, WSDL, UDDI
Computing and Distributed Background

Web services and SOAP

Architecture View

Protocol View

VB application

VB-Structure

SOAP client

Java-Structure

SOAP Server

Java application

ANY client!

SOAP

HTTP

TCP/IP

Web Service

Envelope

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>
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

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

Operations
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 Registry

Optimization Services Registry is a native XML database. It contains a sequence of services, each consisting of a triplet (OSeL, OSpL, OSbL).

Example XML:

```xml
  <description>
    OS registry is a native XML database.
    It contains a sequence of services, 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="2005-03-29">Ziena Knitro Service joins the OS registry</el>
    <el date="2005-02-27">Lindo MINLP Service joins the OS registry</el>
  </news>
</Osyl>
```

Example operation:

```
<Ospl>
  <OSeL ... />
  <OSbL ... />
</Ospl>
```

Example service:

```
<service>
  <OSeL ... />
  <OSpL ... />
  <OSbL ... />
</OSeL>
```

Example grand operation:

```
<Osyl>
  <service ... />
  <service ... />
  <service ... />
</Osyl>
```
Optimization Services Registry

Optimization Services Query Language (OSqL, representation)
Like SQL for relational database; can use XQuery, OSaL (analysis), predefined

Optimization Services Discovery Language (OSdL, communication)
Send the query to the OS registry to discover services

Optimization Services URI Language (OSuL, representation)
A sequence of URI (URL) addresses for service locations with degree of fitness

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

<xml version="1.0" encoding="UTF-8">
<OSuL xmlns="os.optimizationservices.org" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="os.optimizationservices.org http://www.optimizationservices.org/schemas/OSuL.xsd">
<uri>http://www.abc.com/lpsolver.jws</uri>
<uri match="exact">http://www.pdf.notbookservice.url</uri>
<uri match="more" match="approx">
<uri match="guess">
</OSuL>

Operation

String validate (String)
error Message

OSxL

<nonLinear>
<numberConstraints>
<numberVariables num="12">
<continuous num="3" />
<integer num="9" />
<binary num="0" />
</numberVariables>
</programDescription>
</programDataAnalysis>...<programDataAnalysis>
</OSaL>
</standard>
<entity>
<service>
<keyWords>key=interior point method=Key=convex programming</keyWords>
</service>
<optimizationType>
<variableType>mixedInteger</variableType>
</optimizationType>
</entity>
</standard>
</OSqL>
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
Optimization Services modeling Language

```xml
<ampl>
  param T; set PROD; set LINKS = {PROD, 1..T};
  param HC {PROD}; param PCOST {PROD};
  param CAP {1..T}; param DEM {LINKS};
  param PCOST {PROD, 1..T};

  variable x {PROD, 1..T} >= 0;
  variable I {PROD, 0..T} >= 0;
  variable y {PROD, 1..T} binary;

  objective Total_Cost minimize:
  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 CONSTRAINT
  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];
</ampl>
```
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? OSIL, OSRI, OSOL?)
- 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
  - Effective Optimization Services process orchestration
  - Modeling and compilation
  - Efficient OSxL instance parsing and preprocessing algorithms.
- Promote areas where lack of progress are partly due to lack of representation schemes
- Modeling language developers, solver developers, and analyzer developers
- Library developers, registry/server developers, and other auxiliary developers
- Computing on demand and “result on demand”
Acknowledgement

• Robert Fourer (advisor)
• Kipp Martin (co-advisor)
• Tom Tirpak, Motorola Advanced Technology Center
• Professor Mehrotra’s group
• NEOS/OTC team at Argonne National Lab
• Professor John Birge
• Professor Horand Gassmann
• Jorge Nocedal & Richard Waltz, Knitro/Ziena
• Professors at Northwestern IEMS (Mark Daskin, Wally Hopp, Barry Nelson)
• Professor Gordon Bradley
• Bjarni Kristjansson, MPL
• Linus Schrage, Lindo

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