Extensions
to an Optimization Services
Instance Language

Robert Fourer, Jun Ma
Industrial Engineering & Management Sciences
McCormick School of Engineering and Applied Science
Northwestern University
{4er,maj}@iems.northwestern.edu

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

INFORMS Annual Meeting
Pittsburgh — Wednesday, November 8, 2006 — WC07.1
Abstract

- Optimization problems of interest today go beyond the traditional linear, integer, quadratic, and smooth nonlinear types. A language for problem instances must be extended accordingly. This presentation describes prospective extensions to OSiL, our proposed language standard, in such areas as combinatorial optimization and constraint programming, stochastic programming, and semidefinite and cone programming.
Why Extensions?

Better describe problem instances
- Describe broader variety
- Describe more concisely
- Make more natural and understandable

More readily transform models and problems
- Modeling system ↔ OSiL
- OSiL ↔ Solver
Outline

Stochastic programming

Cone & semidefinite programming

Piecewise-linear terms

Complementarity constraints

Logic & combinatorial constraints
Stochastic Programming

Core optimization problem

Stage information

Stochastic information
What about **SMPS Format**?

*Shares drawbacks of the MPS format for LPs, MIPs*

- Limited precision
- Limited name length
- Expensive to process
- Restricted to linear problems
- Not entirely standard

*Yet doesn't cover all problems of interest*

- See extensions at
  sba.management.dal.ca/profs/hgassmann/smps2.htm
OSiL Stochastic Extensions

**XML-based format**

- **Core** LP or MIP described using OSiL
- Multi-stage partitioning using `<stages>` element
- Stochastic characteristics using `<stochastic>` element
Example (Birge & Louveaux, *Intro to Stoch Prog*)

General information for core problem

```xml
<?xml version="1.0" encoding="UTF-8"?>
<osil xmlns="os.optimizationservices.org"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="os.optimizationservices.org
        http://www.optimizationservices.org/schemas/OSiL.xsd">
    <instanceHeader>
        <name>FinPlan</name>
        <source>Birge and Louveaux, Stochastic Programming</source>
        <description>
            This is the stochastic financial planning problem,
            as given in the book by Birge and Louveaux.
            It has four stages and eight scenarios.
            ...
        </description>
    </instanceHeader>
</osil>
```
Example (cont’d)

Variable & objective data for core problem

```xml
<instanceData>
  <variables numberOfVariables="8">
    <var name="Invest0Stocks" type="C" lb="0.0"/>
    <var name="Invest0Bonds"/>
    <var name="Invest1Stocks"/>
    <var name="Invest1Bonds"/>
    <var name="Invest2Stocks"/>
    <var name="Invest2Bonds"/>
    <var name="wealth"/>
    <var name="short"/>
  </variables>
  <objectives numberOfObjectives="1">
    <obj name="expectedWealth" maxOrMin="max" numberOfObjCoef="2">
      <coef idx="6">1.0</coef>
      <coef idx="7">-4.0</coef>
    </obj>
  </objectives>
</instanceData>
```
Example (cont’d)

Constraints and coefficient column-start positions for core problem

```xml
<constraints numberOfConstraints="4">
  <con name="Budget0" lb="55" ub="55"/>
  <con name="Budget1" lb="0" ub="0"/>
  <con name="Budget2" lb="0" ub="0"/>
  <con name="Budget3" lb="80" ub="80"/>
</constraints>
<linearConstraintCoefficients numberOfValues="14">
  <start>
    <el>0</el>  <el>2</el>  <el>4</el>
    <el>6</el>  <el>8</el>  <el>10</el>
    <el>12</el>  <el>13</el>  <el>14</el>
  </start>
</linearConstraintCoefficients>
```
Example (cont’d)

Coefficient row indexes and values for core problem

```xml
<rowIdx>
  <el>0</el> <el>1</el> <el>0</el>
  <el>1</el> <el>1</el> <el>2</el>
  <el>3</el> <el>2</el> <el>3</el>
  <el>3</el> <el>3</el>
</rowIdx>
<value>
  <el>-1</el> <el>1.25</el>
  <el>-1</el> <el>1.14</el>
  <el>-1</el> <el>1.25</el>
  <el>-1</el> <el>1.14</el>
  <el>-1</el> <el>1.25</el>
  <el>-1</el> <el>1.14</el>
  <el>-1</el> <el>1</el>
</value>
</linearConstraintCoefficients>
...
OSiL `<stages>` Element

*Partition variables & constraints by stage number*

- Required attribute gives number of stages

`<implicitOrder>`

- When already ordered by stage
- Just say where each stage begins

`<explicitOrder>`

- Specify a stage number for each variable and constraint
Example (cont’d)

Stage information: implicit

```xml
<stages numberOfStages="4">
  <stage name="stage 0">
    <variables numberOfVariables="2" startIdx="0" endIdx="1"/>
    <constraints numberOfConstraints="1" startIdx="0" endIdx="0"/>
  </stage>
  <stage name="stage 1">
    <variables numberOfVariables="2" startIdx="2" endIdx="3"/>
    <constraints numberOfConstraints="1" startIdx="1" endIdx="1"/>
  </stage>
</stages>
```
Example (cont’d)

Stage information: explicit

```xml
<stages>
  <stage name="stage 2">
    <variables numberOfVariables="2">
      <var idx="4"/>
      <var idx="5"/>
    </variables>
    <constraints numberOfConstraints="1">
      <con idx="2"/>
    </constraints>
  </stage>
  <stage name="stage 3">
    <variables numberOfVariables="2">
      <var idx="6"/>
      <var idx="7"/>
    </variables>
    <constraints numberOfConstraints="1">
      <con idx="3"/>
    </constraints>
  </stage>
</stages>
```
OSiL <stochasticInformation> Element

<decisionEventSequence>
  ➢ Whether decisions precede or follow events in a stage

<eventTree>
  ➢ <scenarioTree>: specify difference from “parent”
  ➢ <nodalTree>: describe node by node
  ➢ <stochasticImplicitTree>: describe via distributions

<softConstraints>
  ➢ <penalties>
  ➢ <probabilisticObjectives>
  ➢ <chanceConstraints>, <integratedChanceConstraints>
  ➢ <userDefinedRiskMeasures>
Alternatives for Explicit Scenarios

<scenarioTree>

- Every child represents a scenario as a *path*
  - from the root of the scenario tree
  - to one of its leaves
- First child is the *root scenario*
  - defined by the core problem
- Each subsequent child branches
  - directly from the root
  - or indirectly from some previous branch

*Every scenario has a parent*

- Only differences from parent are specified
Alternatives for Explicit Scenarios

<nodalTree>

- Every child represents a node of the scenario tree
  - by means of an <sNode> element
- First <sNode> corresponds to the root node
- Every <sNode> may have <sNode> children
  - defining branches of the tree

Every <sNode> specifies the problem at its stage

- By listing differences from its parent, or
- By specifying a single-stage <osi1> problem

... problem size may be stochastic
Example (cont’d)

Explicit scenarios by path

```xml
<stochasticInformation decisionEventSequence="decisionAfterEvent">
  <eventTree>
    <scenarioTree numberOfScenarios="8">
      <rootScenario prob="0.125"/>
      <scenario stage="3" prob="0.125" parent="0">
        <linearConstraintCoefficients numberOfValues="2">
          <el rowIdx="3" colIdx="4">1.06</el>
          <el rowIdx="3" colIdx="5">1.12</el>
        </linearConstraintCoefficients>
      </scenario>
      <scenario stage="2" prob="0.125" parent="0">
        <linearConstraintCoefficients numberOfValues="2">
          <el rowIdx="2" colIdx="2">1.06</el>
          <el rowIdx="2" colIdx="3">1.12</el>
        </linearConstraintCoefficients>
      </scenario>
      ...  
    </scenarioTree>
  </eventTree>
</stochasticInformation>
```
Alternatives for Implicit Scenarios

<stochasticImplicitTree>

- For independent random variables, or
- For random processes influenced by variables that are period-to-period independent

Specify distributions

- Predefined univariate & multivariate distributions
- Arbitary distributions via nonlinear functions

Associate distributions with problem parameters

- Stochastic elements
- Stochastic transformations
Example (cont’d)

Distributions for implicit scenarios

```
<distributions numberOfDistributions="3">
  <distr stage="1">
    <multivariate>
      <multivariateDiscrete>
        <scenario prob="0.5">
          <el>1.25</el>
          <el>1.14</el>
        </scenario>
        <scenario prob="0.5">
          <el>1.06</el>
          <el>1.12</el>
        </scenario>
      </multivariateDiscrete>
    </multivariate>
  </distr>
  ...
  <distr stage="3">
    <univariate>
      <binomial N="1" p="0.5"/>
    </univariate>
  </distr>
</distributions>
```
Example (cont’d)

Elements for implicit scenarios

\[
\begin{pmatrix}
    r_{11} \\
    r_{12} \\
    r_{21} \\
    r_{22} \\
    r_{31} \\
    r_{32}
\end{pmatrix}
= 
\begin{bmatrix}
    1 & 0 & 0 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 & 0 & 0 \\
    0 & 0 & 1 & 0 & 0 & 0 \\
    0 & 0 & 0 & 1 & 0 & 0 \\
    0 & 0 & 0 & 0 & -0.19 & 0 \\
    0 & 0 & 0 & 0 & -0.02 & 0
\end{bmatrix}
\begin{pmatrix}
    \xi_{11} \\
    \xi_{12} \\
    \xi_{21} \\
    \xi_{22} \\
    \xi_{31} \\
    \xi_{32}
\end{pmatrix}
+ 
\begin{pmatrix}
    0 \\
    0 \\
    0 \\
    0 \\
    1.25 \\
    1.14
\end{pmatrix}
\]

\[
<\text{stochasticElements numberOfElements="6"}>
    <\text{el rowIdx="1" colIdx="0"}/>
    <\text{el rowIdx="1" colIdx="1"}/>
    <\text{el rowIdx="1" colIdx="2"}/>
    <\text{el rowIdx="2" colIdx="2"}/>
    <\text{el rowIdx="2" colIdx="3"}/>
    <\text{el rowIdx="3" colIdx="4" baseValue="1.25"}/>
    <\text{el rowIdx="3" colIdx="5" baseValue="1.14"}/>
</\text{stochasticElements}>
Example (cont’d)

Transformations for implicit scenarios

```
<stochasticTransformations>
  <linearTransformation>
    <matrixCoefficients numberOfValues="6">
      <start>
        <el>0</el>  <el>1</el>  <el>2</el>  
        <el>3</el>  <el>4</el>  <el>5</el>  <el>6</el>
      </start>
      <rowIdx>
        <el>0</el>  <el>1</el>  <el>2</el>  
        <el>3</el>  <el>4</el>  <el>5</el>
      </rowIdx>
      <value>
        <el>1.0</el>  <el>1.0</el>  
        <el>1.0</el>  <el>1.0</el>  
        <el>-0.19</el>  <el>-0.02</el>
      </value>
    </matrixCoefficients>
  </linearTransformation>
</stochasticTransformations>
```
Alternatives for Soft Constraints

\(<\text{penalties}>\)

- Specifies penalization for violating constraints

**Various ways to specify**

- \(<\text{simpleRecourse}>\) : linear shortage and surplus penalties
- \(<\text{robustOptimization}>\) : quadratic penalties
- \(<\text{piecewiseLinearQuadratic}>\)
- \(<\text{userDefinedPenalty}>\) : shortage and surplus specified like other user-defined functions

\ldots \text{separate for each constraint}
Alternatives for Soft Constraints

\(<\text{chanceConstraints}>\)

- \(<\text{simpleChanceConstraint}>\)
- \(<\text{jointChanceConstraint}>\)
- \(<\text{probabilisticObjective}>\)

*One (simple) or more (joint) rowIdx attributes*

- rowIdx \(\geq 0\) implies chance constraint
  * probability that the constraint is satisfied
- rowIdx \(< 0\) implies probabilistic objective
  * minimize or maximize the probability that the objective is \(\geq\) or \(\leq\) a constant
Example (cont’d)

Simple recourse penalties

```xml
<softConstraints>
  <penalties>
    <row idx="3">
      <simpleRecourse surplusPenalty="1" shortagePenalty="-4"/>
    </row>
  </penalties>
</softConstraints>
```
Cone and Semidefinite Programming

Design considerations

- Generalizations of $x \geq 0$ to $x \in C$, a convex cone
- Small number of cone types

Tentative extensions

- An element for each cone type
- Child elements and/or attributes indicate the variables involved
**Cone Types**

*Second-order*

- Quadratic cone
  \[ x_1^2 \geq \sum_{j=2}^{n} x_j^2 \]

- Rotated quadratic cone
  \[ 2x_1x_2 \geq \sum_{j=3}^{n} x_j^2 \]

*Semidefinite*

- Symmetric matrix $X$ of variables is positive semi-definite
Example

Constraint & variable data for core problem

```xml
<cones>
  <quadraticCone>
    <el>1</el>
    <el>3</el>
  </quadraticCone>
  <quadraticCone>
    <el>2</el>
    <el mult="3" incr="1">4</el>
  </quadraticCone>
  <rotatedQuadraticCone startIndex="7" endIndex="9"/>
</cones>
```
Piecewise-Linear Terms

Design considerations

- Univariate function of a numerical expression
- Defined by alternating breakpoints and slopes
  - Start and end with slopes
  - Value at zero is zero unless overridden
- Ordering of pieces must be unambiguous

Tentative extensions

- Series of <piece> elements
  - one slope and one breakpoint attribute
  - last <piece> has no breakpoint
  - sorted by breakpoint value
- Optional <level> element gives value at zero
- A final element specifies the operand as an expression tree
Example

... should we do something more general?

```xml
<piecewiseLinear>
  <piece slope="-2" breakpoint="-3"/>
  <piece slope="1" breakpoint="-2"/>
  <piece slope="0.333333" breakpoint="1"/>
  <piece slope="-1" breakpoint="2"/>
  <piece slope="1.25"/>
  <level value="10"/>
  <var idx="7"/>
</piecewiseLinear>
```
Complementarity

Definition
- Two inequalities must hold . . .
- At least one of them with equality

Applications
- Equilibrium problems in economics and engineering
- Optimality conditions for nonlinear programs, bi-level linear programs, etc.

Forms
- “Square” systems of complementarity conditions
  * # of variables =
    # of complementarity constraints + # of equality constraints
- Mathematical programs
  with complementarity constraints (MPCCs)
Classical Complementarity

LP with nonnegative variables

- Complementary slackness conditions

PrimalConstr \{i in I\}:
  \[ \sum \{j in J\} a[i,j] * X[j] = b[i] \]

PrimalBounds \{j in J\}: \[ X[j] >= 0 \]

DualConstr \{j in J\}:
  \[ \sum \{i in I\} Y[i] * a[i,j] + Z[j] = c[j] \]

DualBounds \{i in I\}: \[ Z[i] >= 0 \]

Complementarity \{j in J\}:
  \[ X[j] = 0 \text{ or } Z[j] = 0 \]

- Multiplicative alternative

Complementarity \{j in J\}:
  \[ X[j] * Z[j] = 0 \]
Mixed Complementarity

*LP with bounded variables*

- Complementary slackness conditions

  \[
  \text{PrimalConstr } \{i \in I\}: \sum_{j \in J} a_{i,j} \cdot X[j] = b[i];
  \]

  \[
  \text{PrimalBounds } \{j \in J\}: l[j] \leq X[j] \leq u[j];
  \]

  \[
  \text{DualConstr } \{j \in J\}:
  \sum_{i \in I} Y[i] \cdot a_{i,j} + Z[j] = c[j];
  \]

  \[
  \text{Complementarity } \{j \in J\}:
  X[j] = l[j] \text{ implies } Z[j] \geq 0 \quad \text{and}
  X[j] = u[j] \text{ implies } Z[j] \leq 0 \quad \text{and}
  l[j] < X[j] < u[j] \text{ implies } Z[j] = 0;
  \]

- Variational inequality alternative

  \[
  \text{Complementarity } \{j \in J\}:
  \forall \{Y[j] \in \text{interval}[l[j],u[j]]\}
  (Y[j] - X[j]) \cdot Z[j] \geq 0;
  \]
New complements Operator

**LP with nonnegative variables**

PrimalConstr \{i in I\}:
\[
\sum_{j in J} a[i,j] \cdot X[j] = b[i];
\]

DualConstr \{j in J\}:
\[
\sum_{i in I} Y[i] \cdot a[i,j] + Z[j] = c[j];
\]

Complementarity \{j in J\}:
\[
X[j] \geq 0 \text{ complements } Z[j] \geq 0;
\]

**LP with bounded variables**

PrimalConstr \{i in I\}:
\[
\sum_{j in J} a[i,j] \cdot X[j] = b[i];
\]

DualConstr \{j in J\}:
\[
\sum_{i in I} Y[i] \cdot a[i,j] + Z[j] = c[j];
\]

Complementarity \{j in J\}:
\[
l[j] \leq X[j] \leq u[j] \text{ complements } Z[j];
\]
without Auxiliary Z-Variables

LP with nonnegative variables

PrimalConstr {i in I}:
    sum {j in J} a[i,j] * X[j] = b[i];

Complementarity {j in J}:
    X[j] >= 0 \text{ complements}
    sum {i in I} Y[i] * a[i,j] <= c[j];

LP with bounded variables

PrimalConstr {i in I}:
    sum {j in J} a[i,j] * X[j] = b[i];

Complementarity {j in J}:
    l[j] <= X[j] <= u[j] \text{ complements}
    c[j] - sum {i in I} Y[i] * a[i,j];
Nonlinear

Price-dependent demands

\[
\begin{align*}
\text{var Price \{i in PROD\};} \\
\text{var Level \{j in ACT\};} \\
\text{subject to Pri_Compl \{i in PROD\}:} \\
\quad \text{Price[i] >= 0 complements} \\
\quad \quad \text{sum \{j in ACT\} io[i,j] * Level[j]} \\
\quad \quad \quad >= \text{demzero[i]} - \text{demrate[i] * Price[i];} \\
\text{subject to Lev_Compl \{j in ACT\}:} \\
\quad \text{Level[j] >= 0 complements} \\
\quad \quad \text{sum \{i in PROD\} Price[i] * io[i,j] <= cost[j];}
\end{align*}
\]

... not obviously an optimality condition for an optimization problem
From Applications

Prices of coal shipments

subject to delct \{cr in creg, u in users\}:
\[
0 \leq ct[cr,u] \text{ complements} \\
ctcost[cr,u] + cv[cr] \geq p["C",u];
\]

Height of membrane

subject to dv \{i in 1..M, j in 1..N\}:
\[
1b[i,j] \leq v[i,j] \leq ub[i,j] \text{ complements} \\
(dy/dx) \times (2v[i,j] - v[i+1,j] - v[i-1,j]) \\
+ (dx/dy) \times (2v[i,j] - v[i,j+1] - v[i,j-1]) \\
- c \times dx \times dy;
\]

... more at Complementarity Problem Net
http://www.cs.wisc.edu/cpnet/
Operands: Always Two Inequalities

Two single inequalities

- single-ineq1 complements single-ineq2
  * Both inequalities must hold,
  * at least one at equality

One double inequality

- double-ineq complements expr
  expr complements double-ineq
  * The double-inequality must hold, and
  * if at lower limit then \( expr \geq 0 \),
    if at upper limit then \( expr \leq 0 \),
    if between limits then \( expr = 0 \)
Complementarity Extensions to OSiL

Design

- Introduce new `<complements>` element to expression tree
- Require two child nodes

Implementation

- Check for “two inequalities” requirement  
  after the validation phase
Example

- \( p[1] \geq 0 \) complements
  
  \[
  400h[0]^3p[1]/\exp(1.416*p[1]) - \\
  121.14h[1] - 121.14h[0] \geq 0;
  \]

```
<complements>
  <geq>
    <var idx="25"/>
    <number value="0"/>
  </geq>
  <geq>
    <sum>
      <times>
        <number value="400"/>
        <times>
          <power>
            <var idx="47"/>
            <number value="3"/>
          </power> ...
        </times>
      </times>
      <number value="0"/>
    </geq>
</complements>
```
Example (more complete)

```
<complements>
  <geq>
    <var idx="25"/>
    <number value="0"/>
  </geq>
  <geq>
    <sum>
      <times>
        <number value="400"/>
        <times>
          <power>
            <var idx="47"/>
            <number value="3"/>
          </power>
        ...
      </times>
    </times>
  </sum>
  ...
  <var idx="47" coef="-121.14"/>
</geq>
</complements>
```
Logic and Combinatorial Constraints

Design considerations
- Expression types
- Constraint types
- New operators

Examples of tentative extensions
- Logic operators
- Counting operator
- “All different” operator
- Variable indexed by a variable
Expression Types

Numerical

- Value is a number
  \[
  \text{var Trans \{ORIG, DEST\} >= 0;}
  \]

Logical

- Value is “true” or “false”

Object

- Value is a member of some set
  \[
  \text{var JobForSlot \{SLOTS\} in JOBS;}
  \]

Set

- Value is a set of numbers or objects:
  \[
  \text{var MEMBERS \{PROJECTS\} within VOLUNTEERS;}
  \]
Constraint Types

Range constraints

- $lowerBound \leq numExpr \leq upperBound$
- For one-sided constraint, $lowerBound = -\infty$ or $upperBound = +\infty$
- For equality, $lowerBound = upperBound$

Logic constraints

- $logicExpr$
- Logical
  - $(Mk[i] = 0 \text{ and } Mk[i] = 0) \text{ or } Mk[i] + Mk[i] \geq lbd$
- Counting
  - atmost $mxsrv \{j \text{ in } D\} \left( \sum \{p \text{ in } PRD\} Tr[i,j,p] \geq 10 \right)$
- Special-structure
  - $\text{alldiff } \{j \text{ in } Jobs\} \left( \text{MachineForJob}[j] \right)$
New Operators

**Numerical-valued on constraints**

- Counting
  \[
  \text{count} \{j \in D\} \left( \sum \{p \in PRD\} \ Tr[i,j,p] \geq 10 \right)
  \]

**Logic-valued on constraints**

- Logical
  \[
  (Mk[1] = 0 \text{ and } Mk[2] = 0) \text{ or } (Mk[1] + Mk[2] \geq 100)
  \]

- Counting
  \[
  \text{atmost mxsrv} \{j \in D\} \left( \sum \{p \in PRD\} \ Tr[i,j,p] \geq 10 \right)
  \]

**Special-structure ("global")**

- All-different
  \[
  \text{alldiff} \{j \in Jobs\} (\text{MachineForJob}[j])
  \]

- Distribution
  \[
  \text{numberof} 3 \text{ in} \left( \{j \in 1..nJobs\} \text{ MachineForJob}[j] \right)
  \]
New Operators (cont’d)

Indexing

- Variables in subscripts of parameters or variables

```plaintext
param mCLI integer > 0;
param nLOC integer > 0;
param srvCost {1..mCLI, 1..nLOC} > 0;
param bdgCost > 0;
var Serve {1..mCLI} integer >= 1, <= nLOC;
var Open {1..nLOC} integer >= 0, <= 1;
minimize TotalCost:
    sum {i in 1..mCLI} srvCost[i,Serve[i]] +
    bdgCost * sum {j in 1..nLOC} Open[j];
subject to OpenDefn {i in 1..mCLI}:
    Open[Serve[i]] = 1;
```
New Operators (cont’d)

Indexing

- Variables constrained by subscripts

```
set ABLE within {1..mCLI, 1..nLOC};
param srvCost {ABLE} > 0;

......
minimize TotalCost:
    sum {i in 1..mCLI} srvCost[i,Serve[i]] + ...
```

...(i,Serve[i]) must be in ABLE

With set operands

- Set valued: union, intersection, difference
- Numerical valued: cardinality
- Logic valued: membership, containment
- Special-structure: all-disjoint
Logic Extensions to OSiL

**Design**
- Use same “nonlinear” expression tree
- Define new nodes to represent new operators

**Implementation**
- Extend API to give solvers access to constraint expressions
Example: Logic Operators

(Mk[i] = 0 and Mk[i] = 0) or Mk[i] + Mk[i] >= lbd

<or>
  <and>
    <eq>
      <var idx="23"/>
      <number value="0"/>
    </eq>
    <eq>
      <var idx="103"/>
      <number value="0"/>
    </eq>
    <and>
      <geq>
        <plus>
          <var idx="23"/>
          <var idx="103"/>
        </plus>
        <number value="150"/>
      </geq>
    </and>
  </and>
</or>
Example: “at most” operator

\[ \text{atmost mxsrv } \{ j \in D \} \ (\sum \{ p \in \text{PRD} \} \ Tr[i,j,p] \geq \lim[j]) \]

```xml
<atMost>
  <number value="2"/>
  <geq>
    <sum>
      <var idx="20"/>
      <var idx="21"/>
      <var idx="22"/>
    </sum>
    <number value="10"/>
  </geq>
  <geq>
    <sum>
      <var idx="30"/>
      <var idx="31"/>
      <var idx="32"/>
    </sum>
    <number value="27"/>
  </geq>
  ...
</atMost>
```
Examples (cont'd)

alldiff \{ j \text{ in } \text{Jobs} \} \left( \text{MachineForJob}[j] \right)

\[
\begin{aligned}
&<\text{alldiff}>
&\quad <\text{var} \text{ idx}=\text{"27"} /> \\
&\quad <\text{var} \text{ idx}=\text{"37"} /> \\
&\quad <\text{var} \text{ idx}=\text{"47"} /> \\
&</\text{alldiff}>
\end{aligned}
\]

\[
\begin{aligned}
\text{Open}[\text{Serve}[7]] \text{ where } m\text{CLI} = 40, \text{nLOC} = 15 \\
&\rightarrow \text{Serve corresponds to Var}[0], \ldots, \text{Var}[39] \\
&\rightarrow \text{Open corresponds to Var}[40], \ldots, \text{Var}[54]
\end{aligned}
\]

\[
\begin{aligned}
&<\text{var}>
&\quad <\text{plus}>
&\quad \quad <\text{number value}=\text{"39"} /> \\
&\quad \quad <\text{var} \text{ idx}=\text{"6"} /> \\
&\quad </\text{plus}>
&</\text{var}>
\end{aligned}
\]