Stochastic Constraint Programming: a seamless modeling framework for decision making under uncertainty

Dr Roberto Rossi\(^1\)

\(^1\)LDI, Wageningen UR, the Netherlands

Mansholt Lecture
A Complete Overview
Theoretical Results

Modeling Frameworks

Applications


Decision Support Systems

Why going stochastic?

Decision Making Under Uncertainty
Why going stochastic?

Decision Making Under Uncertainty: A Pervasive Issue

- Land-Crop Allocation
- Sustainable Energy Production
- Food Quality Control
- Production Planning
- Financial Planning
- Inventory Control

Applications
Decision Support Systems
Modeling Frameworks
Theoretical Results
Uncertainty
Decision Making in a Deterministic Setting
Decision Making in a Deterministic Setting
Decision Making in a Deterministic Setting
Decision Making in a Deterministic Setting

- Applications
- Decision Support Systems
- Theoretical Results
- Modeling Frameworks
- Production Planning
- Transport Scheduling
- Inventory Control
- ERP
- CRM

Minimize: \( \sum_{j} c_j x_j \)
Subject to: \( \sum_{j} a_{ij} x_j \geq b_i \) \( \forall i \)
\( x_j \geq 0 \) \( \forall j \)
Decision Making under Uncertainty
Decision Making under Uncertainty

What is missing

Introduction
Modeling Framework
Ongoing Research
SCP and LDI
Conclusions
**0-1 Knapsack Problem**

**Problem:** we have $k$ kinds of items, 1 through $k$. Each kind of item $i$ has
- a value $r_i$
- a weight $w_i$.

We usually assume that all values and weights are nonnegative. The **maximum weight** that we can carry in the bag is $W$.

**Objective:** find a set of objects that provides the maximum value and that fits in the given capacity.
An example

Decision Making in a Deterministic Setting

0-1 KP: MIP Formulation

Objective:
\[ \text{max } \sum_{i=1}^{k} r_i x_i \]

Constraints:
\[ \sum_{i=1}^{k} w_i x_i \leq W \]

Decision variables:
\[ x_i \in \{0, 1\} \quad \forall i \in \{1, \ldots, k\} \]
## Decision Making under Uncertainty

### Static Stochastic Knapsack Problem

**Problem**: we have $k$ kinds of items and a knapsack of size $c$ into which to fit them.

- Each item $i$, if included in the knapsack, brings a deterministic profit $r_i$.
- The size $\mathcal{W}_i$ of each item is not known at the time the decision has to be made, but we assume that the decision maker knows the probability distribution of $\mathcal{W} = (\mathcal{W}_1, \ldots, \mathcal{W}_k)$.
- A per unit penalty cost $p$ has to be paid for exceeding the capacity of the knapsack.
- Furthermore, the probability of not exceeding the capacity of the knapsack should be greater or equal to a given threshold $\theta$.

**Objective**: find the knapsack that maximizes the expected profit.
An example

**Decision Making under Uncertainty**

### SSKP: Stochastic Programming Formulation

**Objective:**

\[
\max \left\{ \sum_{i=1}^{k} r_i X_i - \rho^E \left[ \sum_{i=1}^{k} \mathcal{W}_i X_i - c \right]^+ \right\}
\]

**Subject to:**

\[
\Pr \left\{ \sum_{i=1}^{k} \mathcal{W}_i X_i \leq c \right\} \geq \theta
\]

**Decision variables:**

\[ X_i \in \{0, 1\} \quad \forall i \in 1, \ldots, k \]

**Stochastic variables:**

\[ \mathcal{W}_i \rightarrow \text{item } i \text{ weight } \forall i \in 1, \ldots, k \]

**Stage structure:**

\[ V_1 = \{X_1, \ldots, X_k\} \]
\[ S_1 = \{\mathcal{W}_1, \ldots, \mathcal{W}_k\} \]
\[ L = [\langle V_1, S_1 \rangle] \]
Seamless stochastic optimization

A First Step Towards Seamless Stochastic Optimization

Seamless stochastic optimization

A First Step Towards Seamless Stochastic Optimization

**Advantages**
- **Seamless** Modeling under Uncertainty!
- **Stochastic OPL** not necessarily linked to CP

**Drawbacks**
- **Size** of the compiled model
- **Propagation** not fully supported
A Viable Approach for Seamless Stochastic Optimization


A Viable Approach for Seamless Stochastic Optimization

Stochastic Constraint Program

Objective:
\[
\max \left\{ \sum_{i=0}^{n} r_i x_i - p \left[ \sum_{i=0}^{n} W_i x_i - e \right]^2 \right\}
\]

Subject to:
\[
\frac{\sum_{i=0}^{n} W_i x_i}{e} \geq \theta
\]
Decision variables:
\[
X_i \in [0, 1], \quad V_i \in 1, \ldots, k
\]
Stage structure:
\[
V_i = \{X_{i1}, \ldots, X_{ik}\}
\]

Stochastic OPL Model

```
stoch myrand[onestage]=...;
int nbItems=...;
float c = ...;
float q = ...;
range Items 1..nbItems;
range onestage 1..1;
stoch W[Items,onestage]*myrand = ...;
stoch r[Items] = ...;
dvar float z;
dvar int x[Items] in 0..1;
maximize
sum[i in Items] x[i]*r[i] - expected(x*z)
subject to
  z \geq\ max[i in Items] W[i]*x[i] - q;
  prob(sum[i in Items] W[i]*x[i] <= q) \geq 0.6;
```

Constraint Programming Solver supporting Global Chance-Constraints

Filtering Algorithms for Global Chance-Constraints
In what follows we shall discuss:

- Constraint Programming
- Stochastic Constraint Programming
- Global Chance-Constraints
- Stochastic OPL.

During the discussion the SSKP will be employed as running example.
“Constraint programming (CP) represents one of the closest approaches computer science has yet made to the Holy Grail of programming: the user states the problem, the computer solves it”.

Eugene C. Freuder, Constraints, April 1997
A slightly formal definition

**A Constraint Satisfaction Problem** (CSP) is a triple $(V, D, C)$.

- $V = \{v_1, \ldots, v_n\}$ is a set of variables
- $D$ is a function mapping each variable $v_i$ to a domain $D(v_i)$ of values
- $C$ is a set of constraints.

**A Constraint Optimization Problem** (COP) consists of a CSP and objective function $f(\hat{V})$ defined on a subset $\hat{V}$ of the decision variables in $V$. The aim in a COP is to find a feasible solution that minimizes (maximizes) the objective function.
### Sample COP: 0-1 KP

- \( V = \{ x_1, \ldots, x_3 \} \)
- \( D(x_i) = \{ 0, 1 \} \quad \forall i \in \{ 1, \ldots, 3 \} \)
- \( C = \{ 8x_1 + 5x_2 + 4x_3 \leq 10 \} \)
- \( f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3 \)

The **optimal solution** for the COP has a value of 25 and corresponds to the assignment is \( x_2 = x_3 = 1 \) and \( x_1 = 0 \).
Constraint Programming

An Example

Sample COP: 0-1 KP

Objects:
- r=8, w=8
- r=15, w=5
- r=10, w=4

Knapsack:
Capacity: 10
Sample COP: 0-1 KP

Decision:

- r=8, w=8
- r=15, w=5
- r=10, w=4

Knapsack:

- Capacity: 10
- Required: 9
Constraint Programming

Solution Method

**Strategy**

- **Constraint Programming** proposes to solve CSPs/COPs by associating with each constraint a filtering algorithm.

- A filtering algorithm removes from decision variable domains values that cannot belong to any solution of the CSP/COP.

- **Constraint Propagation** is the process that repeatedly calls filtering algorithms until no new deduction can be made.

- **Constraint Solving** interleaves filtering algorithms and a search procedure (for instance a backtracking algorithm).
An Example

Sample COP: 0-1 KP

- $V = \{x_1, \ldots, x_3\}$
- $D(x_i) = \{0, 1\} \ \forall i \in \{1, \ldots, 3\}$
- $C = \{8x_1 + 5x_2 + 4x_3 \leq 10\}$
- $f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3$

Constraint Propagation initially does not produce any effect.
Sample COP: 0-1 KP

- \( V = \{x_1, \ldots, x_3\} \)
- \( D(x_i) = \{0, 1\} \quad \forall i \in \{1, \ldots, 3\} \)
- \( C = \{8x_1 + 5x_2 + 4x_3 \leq 10\} \)
- \( f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3 \)

To build a search tree, we apply the **lexicographic variable and value ordering heuristics** and use **labeling as domain splitting procedure**.
An Example

Sample COP: 0-1 KP

\[ V = \{x_1, \ldots, x_3\} \]
\[ D(x_i) = \{0, 1\} \quad \forall i \in \{1, \ldots, 3\} \]
\[ C = \{8x_1 + 5x_2 + 4x_3 \leq 10\} \]
\[ f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3 \]

Filtered domains at \( P_0 \)

\[
\begin{align*}
D(x_1) &= \{0, 1\} \\
D(x_2) &= \{0, 1\} \\
D(x_3) &= \{0, 1\} \\
D(z) &= \{0, \ldots, 33\}
\end{align*}
\]
Sample COP: 0-1 KP

\[ V = \{x_1, \ldots, x_3\} \]
\[ D(x_i) = \{0, 1\} \quad \forall i \in \{1, \ldots, 3\} \]
\[ C = \{8x_1 + 5x_2 + 4x_3 \leq 10\} \]
\[ f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3 \]

Filtered domains at \( P_1 \)

\[ D(x_1) = \{0\} \quad D(x_2) = \{0, 1\} \quad D(x_3) = \{0, 1\} \]
\[ D(z) = \{0, \ldots, 25\} \]
**An Example**

Sample COP: 0-1 KP

\[ V = \{ x_1, \ldots, x_3 \} \]
\[ D(x_i) = \{ 0, 1 \} \quad \forall i \in \{ 1, \ldots, 3 \} \]
\[ C = \{ 8x_1 + 5x_2 + 4x_3 \leq 10 \} \]
\[ f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3 \]

Optimal solution associated with \( P_1 \)

\[ D(x_1) = \{ 0 \} \quad D(x_2) = \{ 1 \} \quad D(x_3) = \{ 1 \} \quad D(z) = \{ 25 \} \]
An Example

Sample COP: 0-1 KP

\[ V = \{x_1, \ldots, x_3\} \]
\[ D(x_i) = \{0, 1\} \quad \forall i \in \{1, \ldots, 3\} \]
\[ C = \{8x_1 + 5x_2 + 4x_3 \leq 10\} \]
\[ f(x_1, \ldots, x_3) = 8x_1 + 15x_2 + 10x_3 \]

Filtered domains at \(P_2\)

\[ D(x_1) = \{1\} \quad D(x_2) = \{0\} \quad D(x_3) = \{0\} \quad D(z) = \{8\} \]

We backtrack to \(P_0\) and we return the optimal solution with value 25 found in the subtree associated with node \(P_1\).
### Efficiency

Filtering algorithms
- **detect inconsistencies** in a proactive fashion
- **speed up** the search

provided that the time spent in filtering is less than the time saved in terms of search efforts.

A challenging research topic is the **design** of efficient filtering strategies.
Global Constraints

Not only binary relations

In constraint programming is common to find constraints over a non-predefined number of variables

- alldifferent
- element
- cumulative
- ...

These constraints are called **global constraints**

- they can be used in a variety of situations
- they are associated with powerful **filtering strategies**
- new **custom** global constraints can be defined
A Stochastic Constraint Satisfaction Problem (SCSP) is a 7-tuple:

\[ \langle V, S, D, P, C, \theta, L \rangle. \]

- \( V = \{v_1, \ldots, v_n\} \) is a set of decision variables
- \( S = \{s_1, \ldots, s_n\} \) is a set of stochastic variables
- \( D \) is a function mapping each variable to a domain of potential values
- \( P \) is a function mapping each variable in \( S \) to a probability distribution for its associated domain
- \( C \) is a set of (chance)-constraints, possibly involving stochastic variables
- \( \theta_h \) is a threshold probability associated to chance-constraint \( h \)
- \( L = [\langle V_1, S_1 \rangle, \ldots, \langle V_i, S_i \rangle, \ldots, \langle V_m, S_m \rangle] \) is a list of decision stages.

By considering an objective function \( f(\hat{V}, \hat{S}) \) we obtain a SCOP.
**Sample SCOP: SSKP**

- $V = \{x_1, \ldots, x_3\}$
- $D(x_i) = \{0, 1\}$ $\forall i \in \{1, \ldots, 3\}$
- $S = \{w_1, \ldots, w_3\}$
- $D(w_1) = \{5(0.5), 8(0.5)\}$, $D(w_2) = \{3(0.5), 9(0.5)\}$, $D(w_3) = \{15(0.5), 4(0.5)\}$
- $C = \{Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.2\}$
- $L = \langle V, S \rangle$
- $f(x_1, \ldots, x_3) =$

\[
8x_1 + 15x_2 + 10x_3 - 2\mathbb{E} \text{max} \left[ 0, \sum_{i=1}^{3} w_i x_i - 10 \right]
\]
Stochastic Constraint Programming

Sample SCOP: SSKP

**Objects:**

<table>
<thead>
<tr>
<th>r=8</th>
<th>r=15</th>
<th>r=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>w=5 OR 8</td>
<td>w=3 OR 9</td>
<td>w=15 OR 4</td>
</tr>
</tbody>
</table>

**Knapsack:**

Capacity: 10
Stochastic Constraint Programming

Sample SCOP: SSKP

Decision:
- \( r=8 \) \( w=5 \) OR \( 8 \)
- \( r=15 \) \( w=3 \) OR \( 9 \)
- \( r=10 \) \( w=15 \) OR \( 4 \)

Observation:
- \( r=8 \) \( w=8 \)
- \( r=15 \) \( w=3 \)
- \( r=10 \) \( w=15 \)

Knapsack:
- Capacity: 10
- Required: 11
Sample SCOP: SSKP

Obj: 17
A Sample SCOP: DSKP

- $V = \{x_1, \ldots, x_3\}$
- $D(x_i) = \{0, 1\} \quad \forall i \in \{1, \ldots, 3\}$
- $S = \{w_1, \ldots, w_3\}$
- $D(w_1) = \{5(0.5), 8(0.5)\}$, $D(w_2) = \{3(0.5), 9(0.5)\}$, $D(w_3) = \{15(0.5), 4(0.5)\}$
- $C = \{Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.2\}$
- $L = [\langle\{x_1\}, \{w_1\}\rangle, \langle\{x_2\}, \{w_2\}\rangle, \langle\{x_3\}, \{w_3\}\rangle]$
- $f(x_1, \ldots, x_3) =$
  \[E[8x_1 + 15x_2 + 10x_3] - 2E\max\left[0, \sum_{i=1}^{3} w_ix_i - 10\right]\]
**Stochastic Constraint Programming**

## Sample SCOP: DSKP

<table>
<thead>
<tr>
<th>Objects:</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$r=8$</td>
<td>$r=15$</td>
<td>$r=10$</td>
</tr>
<tr>
<td>$w=5$ OR $8$</td>
<td>$w=3$ OR $9$</td>
<td>$w=15$ OR $4$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Knapsack:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Capacity:</strong></td>
<td>10</td>
</tr>
</tbody>
</table>
Stochastic Constraint Programming

Sample SCOP: DSKP

Decision stage 1:

Decision stage 1:

\[ r=8 \]
\[ w=5 \text{ OR } 8 \]

\[ r=15 \]
\[ w=3 \text{ OR } 9 \]

\[ r=10 \]
\[ w=15 \text{ OR } 4 \]

Observation:

\[ r=8 \]
\[ w=8 \]

Knapsack:

Capacity: 10
Required: 8
Stochastic Constraint Programming

Sample SCOP: DSKP

Decision stage 2:

- \( r=8 \), \( w=8 \)
- \( r=15 \), \( w=3 \) OR \( w=9 \)
- \( r=10 \), \( w=15 \) OR \( w=4 \)

Observation:

- \( r=15 \), \( w=3 \)

Knapsack:

- Capacity: 10
- Required: 11
Sample SCOP: DSKP

Decision stage 3:

- \( r=8 \) \( w=8 \)
- \( r=15 \) \( w=3 \)
- \( r=10 \) \( w=15 \) OR 4

Y

Observation:

- \( r=10 \) \( w=15 \)

Knapsack:

- Capacity: 10
- Required: 26
Stochastic Constraint Programming

Sample SCOP: DSKP

Obj: 27.9687

\[
\begin{align*}
&x_1 = 1, \\
&w_1 = 5, \\
&w_1 = 8, \\
&x_2 = 1, \\
&w_2 = 3, \\
&w_2 = 9, \\
&x_3 = 0, \\
&w_3 = 4, \\
&w_3 = 15, \\
&w_3 = 4, \\
&x_3 = 1, \\
&w_3 = 15, \\
&w_3 = 4, \\
&w_3 = 15, \\
&x_3 = 1, \\
&w_3 = 15, \\
&w_3 = 4, \\
&w_3 = 15, \\
&w_3 = 4, \\
&0, \\
&0, \\
&19, \\
&8, \\
&16, \\
&5, \\
&22, \\
&11
\end{align*}
\]
By using the approach discussed in

S. A. Tarim, S. Manandhar and T. Walsh,
*Stochastic Constraint Programming: A Scenario-Based Approach*,
Constraints, Vol.11, pp.53-80, 2006

it is possible to compile any SCSP/SCOP down to a deterministic equivalent CSP.
SSKP: Compiled Deterministic Equivalent CSP

```c
int nbWorlds=8;
range Worlds 1..nbWorlds;
int nbItems=3;
range Items 1..nbItems;
float c = 2;
float W[Worlds,Items] =[[5,3,15],
[5,3,4],
[5,9,15],
[5,9,4],
[8,3,15],
[8,3,4],
[8,9,15],
[8,9,4]];
float Pr[Worlds]=
[0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125];
float r[Items] = [8,15,10];
float q = 10;

var float+ z[Worlds];
var int+ x[Items] in 0..1;

maximize ((sum(i in Items)x[i]*r[i])−c*(sum(j in Worlds)Pr[j]*z[j]))

subject to{
    forall(j in Worlds) z[j]>(sum(i in Items)W[j,i]*x[i])−q;
    sum(j in Worlds) Pr[j]*(sum(i in Items)W[j,i]*x[i] <= q) >= 0.2;
};
```
SSKP: Compiled Deterministic Equivalent CSP with Global Chance-Constraints

```plaintext
int nbWorlds=8;
range Worlds 1..nb Worlds;
int nbItems=3;
range Items 1..nb Items;
float c = 2;
float W[Worlds, Items] =
    {{5.3, 15},
     {5.3, 4},
     {5.9, 15},
     {5.9, 4},
     [8.3, 15],
     [8.9, 4],
     [8.9, 15],
     [8.9, 4]};

float Pr[Worlds] =
    [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125];

float r[Items] = [8.15, 10];
float q = 10;

var float+ z;
var int+ x[Items] in 0..1;

maximize ((\sum (i in Items) x[i]*r[i]) - c*(\max(0, z-q)));

subject to{
    stochLinIneq(x, W, Pr, q, 0.2);
    expectedLinEq(x, W, Pr, z);
};
```
Also in Stochastic Constraint Programming (SCP) we have
- constraints
- filtering algorithms

In contrast to CP, in SCP constraints divide into
- hard constraints
- chance-constraints

Global Chance-Constraints
Perhaps the most interesting aspect of SCP is that the concept of *global constraint* can be also adopted in a stochastic environment, thus leading to
- Global Chance-Constraints (Rossi et al., 2008)
Global Chance-Constraints

Filtering in SCSPs

Stochastic Constraint Programming

Global Chance-Constraints

- represent relations among a non-predefined number of \textbf{decision} and \textbf{random} variables
- implement dedicated filtering algorithms based on
  - \textbf{feasibility} reasoning
  - \textbf{optimality} reasoning

Global Chance-Constraints performing optimality reasoning are called \textbf{Optimization-Oriented Global Chance-Constraints} (Rossi et al., 2008).
Filtering in SCSPs

“Synthesizing Filtering Algorithms for Global Chance-Constraints” (Hnich et al., 2009)
Filtering Algorithms for GCCs

Stochastic Programming Model

\[ \Pr \left\{ \sum_{i=1}^{k} \mathcal{W}_i X_i \leq c \right\} \geq \theta \]

Stochastic Constraint Programming Model

\[ C = \{ \Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.2 \} \]

Global Chance-Constraint

\texttt{stochLinIneq(x,W,Pr,q,0.2);}
Filtering Algorithms for GCCs

\[ \Pr (w_1x_1 + w_2x_2 + w_3x_3 \leq 10) > 0.5 \]

\[ w_1 = 5 \]
\[ w_2 = 3 \]
\[ w_3 = 4 \]

\[ w_2 = 9 \]
\[ w_3 = 15 \]

\[ w_3 = 4 \]

\[ x_1 = \{0, 1\} \]
\[ x_2 = \{0, 1\} \]
\[ x_3 = \{0, 1\} \]
Global Chance-Constraints

Filtering Algorithms for GCCs

\[ P_r(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) > 0.5 \]

- \( w_3 = 15 \)
- \( w_2 = 3 \)
- \( w_3 = 4 \)
- \( w_2 = 9 \)
- \( w_3 = 15 \)
- \( w_1 = 5 \)
- \( w_3 = 4 \)
- \( w_1 = 8 \)
- \( w_2 = 3 \)
- \( w_3 = 4 \)
- \( w_2 = 9 \)
- \( w_3 = 15 \)
- \( w_2 = 9 \)
- \( w_3 = 4 \)
Filtering Algorithms for GCCs

$\Pr(w_1x_1+w_2x_2+w_3x_3 \leq 10) > 0.5$

$x_1 = \{0, 1\}$
$x_2 = \{0, 1\}$
$x_3 = \{0, 1\}$
Filtering Algorithms for GCCs

\[ \Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.5 \]
Filtering Algorithms for GCCs

\[ \Pr (w_1 x_1 + w_2 x_2 + w_3 x_3 \leq 10) \geq 0.5 \]
Global Chance-Constraints

Filtering Algorithms for GCCs

\[ \Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.5 \]

- \( x_1 = \{0, 1\} \)
  - \( w_1 = 5 \) \( x_2 = 1 \)
    - \( w_2 = 3 \) \( w_3 = 4 \) \( x_3 = 1 \) \( w_3 = 15 \) \( 8 \)
    - \( w_2 = 9 \) \( x_3 = \{0, 1\} \) \( w_3 = 15 \) \( 14 \)
  - \( w_1 = 8 \) \( x_2 = 1 \)
    - \( w_2 = 3 \) \( w_3 = 4 \) \( x_3 = 1 \) \( w_3 = 15 \) \( 8 \)
    - \( w_2 = 9 \) \( x_3 = 1 \) \( w_3 = 15 \) \( 14 \)

- \( x_1 = \{0, 1\} \)
  - \( w_1 = 5 \) \( x_2 = 1 \)
    - \( w_2 = 3 \) \( w_3 = 4 \) \( x_3 = 1 \) \( w_3 = 15 \) \( 8 \)
    - \( w_2 = 9 \) \( x_3 = \{0, 1\} \) \( w_3 = 15 \) \( 14 \)
  - \( w_1 = 8 \) \( x_2 = 1 \)
    - \( w_2 = 3 \) \( w_3 = 4 \) \( x_3 = 1 \) \( w_3 = 15 \) \( 8 \)
    - \( w_2 = 9 \) \( x_3 = 1 \) \( w_3 = 15 \) \( 14 \)
A language specifically introduced by Tarim et al. (Tarim et al., 2006) for modeling decision problems under uncertainty. It captures several high level concepts that facilitate the process of modeling uncertainty:

- stochastic variables (independent or conditional distributions)
- several probabilistic measures for the objective function (expectation, variance, etc.)
- chance-constraints
- decision stages
- ...
Stochastic OPL

int N = 3;
int c = 10;
int p = 2;
float \( \theta = 0.2 \)
range Object [1..3];
int value[Object] = [8,15,10];
stoch int weight[Object] = [<5(0.5),8(0.5)>,
                          <3(0.5),9(0.5)>,<15(0.5),4(0.5)>];

var int+ X[Object] in 0..1;
stages = [<X,weight>];
var int+ z;

maximize sum(i in Object) X[i]*value[i] - p*z
subject to{
z = max(0,expected(sum(i in Object) X[i]*weight[i] - c));
prob(sum(i in Object) X[i]*weight[i] - c \leq 0) \geq \theta;
}
Stochastic OPL

DSKP

```plaintext
int N = 3;
int c = 10;
int p = 2;
float θ = 0.2
range Object [1..3];
int value[Object] = [8,15,10];
stoch int weight[Object] = [<5(0.5),8(0.5)>,
                           <3(0.5),9(0.5)>,<15(0.5),4(0.5)>];
var int+ X[Object] in 0..1;
var int+ z;

maximize sum(i in Object) X[i] * value[i] - p * z
subject to{
z = max(0,expected(sum(i in Object) X[i] * weight[i] - c));
prob(sum(i in Object) X[i] * weight[i] - c ≤ 0) ≥ θ;
}
```
Overview of the Framework

A Viable Approach for Seamless Stochastic Optimization

Stochastic Constraint Program

Stochastic OPL Model

```
stoch myrand[onstage]=...;
int nItems=...;
float q = ...;
range Items 1..nItems;
range onstage 1..1;
float W[Items, onstage]=myrand = ...;
float r[Items] = ...;
dvar float z;
dvar int x[Items] in 0..1;
maximize sum(i in Items) x[i]*r[i] - expected(z);
subject to:
  z >= sum(i in Items) W[i]*x[i] < q;
  prob(sum(i in Items) W[i]*x[i] <= q) >= 0.6;
```

Solution

Constraint Programming Solver supporting Global Chance-Constraints

Filtering Algorithms for Global Chance-Constraints

**Contribution**

A generic approach for **constraint reasoning under uncertainty**.
*Works with any existing propagation algorithm!*

**Drawback**

Only implemented for linear inequalities/equalities:  
\[
\text{stochLinIneq}(x, W, Pr, q, 0.2); \\
i.e. \text{SSKP} \rightarrow \Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.2
\]

**Future work**

Considering more **global constraints**:

- allDifferent()
- NValue()
- Cumulative()
- ...
S. A. Tarim, S. Manandhar and T. Walsh,
*Stochastic Constraint Programming: A Scenario-Based Approach*,
Constraints, Vol.11, pp.53-80, 2006
Scenario Reduction

Inspired by:

A. J. Kleywegt, A. Shapiro, T. Homem-De-Mello
The Sample Average Approximation Method for Stochastic Discrete Optimization
Scenario Reduction

\[ \Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.5 \]

- \( w_1 = 5 \)
  - \( w_2 = 9 \)
    - \( w_3 = 15 \)
    - \( w_3 = 4 \)
  - \( w_3 = 4 \)
  - \( w_3 = 15 \)
- \( w_1 = 8 \)
  - \( w_2 = 3 \)
    - \( w_3 = 15 \)
    - \( w_3 = 4 \)
  - \( w_3 = 4 \)
- \( w_2 = 3 \)
  - \( w_3 = 4 \)
  - \( w_3 = 15 \)
- \( w_2 = 9 \)
  - \( w_3 = 4 \)
  - \( w_3 = 15 \)
- \( w_3 = 4 \)
  - \( w_3 = 15 \)
Scenario Reduction

\[ \Pr (w_1 x_1 + w_2 x_2 + w_3 x_3 \leq 10) \geq 0.5 \]

\( w_3 = 15 \)

\( w_2 = 3 \)

\( w_1 = 5 \)

\( w_1 = 8 \)

\( w_3 = 15 \)

\( w_2 = 3 \)

\( x_1 = \{0, 1\} \)

\( x_2 = \{0, 1\} \)

\( x_3 = \{0, 1\} \)
**Scenario Reduction**

\[ \Pr (w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.5 \]

- \( w_1 = 5 \)
- \( w_2 = 3 \)
- \( x_1 = \{0, 1\} \)
- \( x_2 = \{0, 1\} \)
- \( x_3 = \{0, 1\} \)
- \( w_3 = 15 \)
- \( w_3 = 15 \)
Scenario Reduction

\[ \Pr(w_1x_1 + w_2x_2 + w_3x_3 \leq 10) \geq 0.5 \]

- \( w_3 = 15 \)
- \( w_2 = 3 \)
- \( w_1 = 5 \)
- \( x_1 = \{0, 1\} \)
- \( x_2 = \{0, 1\} \)
- \( x_3 = \{0, 1\} \)
Neuro-evolutionary Stochastic Constraint Programming

S. Prestwich, S. A. Tarim, R. Rossi, B. Hnich,
Evolving Parameterised Policies for Stochastic Constraint Programming,
submitted for possible publication to the 15th International Conference on Principles
and Practice of Constraint Programming (CP-09) Lisbon, Portugal, September 21-24,
2009
Food Supply Chain Networks
- Farm - Slaughterhouse allocation (with W. Rijpkema)
- Inventory Control for Perishable Items (with K. Pauls-Worms)

Agrifood Domain
- Land Allocation
- Pest Control

Environmental Domain
- ...

Education
- Decision Science II ???
The LDI research framework

A. Analysis and Design of Innovative Logistics, Information & Knowledge Management Concepts

B. Design of Decision Support Models and Tools
- Knowledge Models and Ontologies
- OR Models and Algorithms, Simulation Models
- Software tools (Simulation Games, Knowledge Bases and Digital Learning Materials)

C. Design of Efficient Algorithms (needed to solve the complex applied DS models)
The LDI research framework

A. Analysis and Design of Innovative Logistics, Information & Knowledge Management Concepts
   - Agribusiness and Food Supply Chains
   - Environment and Natural Resources

B. Design of Decision Support Models and Tools
   - Knowledge Models and Ontologies
   - OR Models and Algorithms, Simulation Models
   - Software tools (Simulation Games, Knowledge Bases and Digital Learning Materials)

C. Design of Efficient Algorithms
   (needed to solve the complex applied DS models)

Problem characteristics: Complexity, Dynamics, Uncertainty

Multi-disciplinary methods, models and views

Stochastic Constraint Programming
Filtering Algorithms

IDE & Compiler
We discussed a **Framework** for **Modeling Decision Problems under Uncertainty**
- Stochastic Constraint Programming
- Global Chance-constraints
- Stochastic OPL

We presented some current and possible future **application areas**

We **positioned** our Framework within the **LDI research framework**
Questions