A Dual Decomposition Algorithm for Solving Chance Constrained Binary Programs

Yan Deng¹, Jon Lee¹, Sigian Shen¹
¹ Department of Industrial and Operations Engineering, University of Michigan.

Introduction

Chance constrained binary program:

\[
\begin{align*}
\min & \quad f(x) \\
\text{s.t.} & \quad g_j(x) + \psi_j(y_j) \leq b_j, \quad j = 1, \ldots, m \\
& \quad A_k x + b_k \leq c_k, \quad k = 1, \ldots, K
\end{align*}
\]

- \( \psi_j(y_j) \): random variables with distribution \( P(Y) \) and \( P(Y) \) known.
- \( A_k \) is a technology matrix for scenario \( \sigma_k \).
- \( \psi_j(y_j) \) is piece-wise linear in \( y_j \).
- \( \lambda, \rho \) are nonanticipativity constraints.

Algorithm Overview:

- **LB**: Lagrangian relaxation, decomposable into parallel subproblem scenarios (Craig and Schultz 1999).
- **UB**: objective of feasible solution.
- **Proposed bounds and eventually close duality gap by shrinking feasible region (Ahmed 2013).**
- **Dual decomposition algorithm** and this simplifies the subgradient method.
- **Avoid big-\( \Omega \) constant which frequently appears in solving chance constrained programs based on finite scenarios.
- **Use “cropping” bound (Lee 2003).**

Dual Decomposition

In light of Ahmed (2013), we recover a decomposable Lagrangian relaxation and verify it as a conditional LB, given a subset \( S \) of vertices excluded from consideration.

1. For each scenario \( k \), introduce variable \( z_k \in \{0, 1\} \) to activate/deactivate the inner constraints and create a copy of \( x \) as \( x^{(k)} \). Then \( \text{Problem } 1 \) is equivalently reformulated as:

\[
\begin{align*}
\text{min} \quad & \sum_{k=1}^{K} f(x^{(k)}) \\
\text{s.t.} \quad & g_j(x^{(k)}) + \psi_j(y^{(k)}) \leq b_j, \quad j = 1, \ldots, m \\
& A_k x^{(k)} + b_k \leq c_k, \quad k = 1, \ldots, K
\end{align*}
\]

Proposition 2. If \( x^{(k)} \) is feasible for \( \text{Problem } 1 \), then it is a tighter than \( L_3(\lambda, \rho) \):

\[
\begin{align*}
V_3(\lambda, \rho) = -\lambda x^{(k)} + \sum_{k=1}^{K} \lambda_k z_k + \sum_{k=1}^{K} h_k^{(k)}(z_k) + \sum_{k=1}^{K} g_k^{(k)}(z_k)
\end{align*}
\]

Proposition 3. The algorithm terminates in finite stenations, and returns an \( \text{optimal solution to } (1) \).

Proof: (i) Feasible region shrinks by \( S \neq \emptyset \) each iteration; (ii) \( x \) feasible for \( (3a) \); (iii) use \( U \) UB to avoid UB without updating UB.

Cutting Planes (Step 3i):

- No-good cut to exclude some \( z_k \).
- Aggregate \( \text{(NGC)} \) to form cutting inequality:

\[
\sum_{k=1}^{K} (1 - z_k) \leq 1 \quad (3d) \quad \text{NGC}
\]

- Objective cut: \( UB \leq f(x) \leq UB \quad [f(x) \notin Z, V_y] \quad \text{(OBC)} \)

- Feasible solution with objective better than UB will NOT be excluded without updating UB.

Computation

Machine: CPU 3.20 GHz with 8GB memory; CPLEX 12.5.1.

Instances: \( D = 20 \), \( K = 200 \) and \( \epsilon = 0.05 \), constructed based on probprob on SLPILP (Ahmed et al. 2013).

Solution time (sec):

<table>
<thead>
<tr>
<th>Instance</th>
<th>Dual decomposition algorithm</th>
<th>L2C</th>
<th>OBC</th>
<th>V2C</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>486</td>
<td>995</td>
<td>353</td>
<td>10.0%</td>
</tr>
<tr>
<td>s2</td>
<td>680</td>
<td>771</td>
<td>543</td>
<td>11.3%</td>
</tr>
<tr>
<td>s3</td>
<td>2569</td>
<td>917</td>
<td>569</td>
<td>11.2%</td>
</tr>
<tr>
<td>s4</td>
<td>1025</td>
<td>843</td>
<td>587</td>
<td>11.5%</td>
</tr>
<tr>
<td>s5</td>
<td>608</td>
<td>679</td>
<td>472</td>
<td>11.6%</td>
</tr>
</tbody>
</table>

- Report optimality gap at termination if not solved in 1hr.
- Differences in the three schemes:
  - L2C: Use \( L_3(\lambda, \rho) \) and update \( \lambda \) and \( \rho \) in the subgradient method.
  - Use cuts (NGC) and (OBC) only.
- V2C: Change to use \( V_3(\lambda, \rho) \) and update only.
- Aggregate (NGC) if possible to form (CRC).

References


http://www2.isye.gatech.edu/~sandee/slpilp/, 2013.
