Branch-and-Benders Cut for Stochastic Integer Programming

Jim Luedtke

Dept. of Industrial and Systems Engineering
Wisconsin Institute for Discovery
University of Wisconsin-Madison, USA
jim.luedtke@wisc.edu

Montreal Operations Research Student Chapter Keynote
May 27, 2022

Example: Generation/Transmission Capacity Expansion

Given candidate generators (sizes/locations) and transmission lines

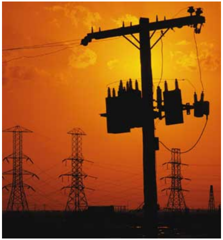
- Which to open to minimize capacity expansion and operating costs to meet demands?

Discrete decisions:

- Select generator/transmission line or not

Uncertainty:

- Future demands by location, weather/renewable yield



Example: Deterministic Model

Simplified generation expansion only model (ignoring transmission):

- Candidate generators: G, Demand locations: J
- Fixed cost f_i, capacity C_i for generator $i \in G$
- Load d_j: for $j \in J$
- x_i: Binary choice for generator opening decisions
- y_{ij}: Amount of demand at location j met from facility i
- z_j: Load shed at location j

$$
\begin{aligned}
\min \sum_{i \in G} f_i x_i + \sum_{i \in G} \sum_{j \in J} c_{ij} y_{ij} + \sum_{j \in J} p_j z_j \\
\text{s.t.} \quad \sum_{i \in G} y_{ij} + z_j = d_j \quad & \forall j \in J \\
\sum_{j \in J} y_{ij} \leq C_i x_i \quad & \forall i \in G \\
x_i \in \{0,1\}, y_{ij} \geq 0 \qquad & \forall i \in G,\ j \in J
\end{aligned}
$$

Two-stage Stochastic Optimization

What if capacity/demands are random?

Classic two-stage framework

1. Choose here-and-now decisions (aka first-stage)

$\Rightarrow$ Observe random variables

2. Make recourse decisions (in response to observed random variables, aka second-stage)

Goal: Choose current decisions to minimize immediate cost plus expected value of cost of "best response" decisions

- Can also consider risk-averse objectives, but we'll ignore that today

Must determine which decisions are here-and-now and which are recourse

Example: Stochastic model

- Load D_j: Random load for $j \in J$
- Capacity C_i: Random capacity for $i \in G$
- x_i: Binary choice for generator open decisions (here-and-now)
- y_{ij}: Amount of customer j demand met from facility i (recourse)
- z_j: Amount of customer j demand met that is not met (recourse)

$$\min_{x \in \{0,1\}^G} \sum_{i \in G} f_i x_i + \mathbb{E}[Q(x, D, C)]$$

$$\begin{aligned} Q(x, D, C) := \min_{y,z \geq 0} \; & \sum_{i \in G} \sum_{j \in J} c_{ij} y_{ij} + \sum_{j \in J} p_j z_j \\ \text{s.t.} \; & \sum_{i \in G} y_{ij} + z_j \geq D_j \quad \forall j \in J \\ & \sum_{j \in J} y_{ij} \leq C_i x_i \quad \forall i \in G \end{aligned}$$

General Model

Two-Stage Stochastic Mixed Integer Program (SMIP)

$$\begin{aligned} \min\ & c^\top x + \mathbb{E}[Q(x,\xi)] \\ \text{s.t.}\ & Ax \geq b \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1} \end{aligned}$$

where $\xi = (q, h, T, W)$ and

$$\begin{aligned} Q(x,\xi) = \min\ & q^\top y \\ \text{s.t.}\ & Wy = h - Tx \\ & y \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2} \end{aligned}$$

- x: first-stage decision variables
- y: second-stage decision variables
- Sometimes assume n_1, p_1, n_2, or p_2 are zero

What makes SMIP hard?

Stochastic integer programming combines challenges from Stochastic programming and Integer Programming

Stochastic programming challenges

- Evaluating expectation
- Huge size even with finite scenario approximation

Integer programming challenges

- Huge number of discrete options
- Weak relaxations can lead to huge enumeration trees

Tutorial Overview

- Sample Average Approximation (SAA) for approximating expected value (very brief!)
- How to solve the SAA approximation
  - Deterministic equivalent form
  - Integer programming methdology background
  - Benders cut based methods
- Recent developments

First Challenge: Evaluating Expected Value

Two-Stage Stochastic Mixed Integer Program (SMIP)

$$\begin{aligned} \min\ & c^\top x + \mathbb{E}[Q(x,\xi)] \\ \text{s.t. }\ & Ax \geq b \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1} \end{aligned}$$

High-dimensional ξ:

- Evaluating $\mathbb{E}[Q(x,\xi)]$ challenging even for a single fixed x
- Need approximation at many values of x

Simple idea: Sample average approximation

- Let ξ^s, $s = 1, \ldots, S$ be a Monte Carlo sample of ξ
- Use sample average to approximate expected value

$$\mathbb{E}[Q(x,\xi)] \approx \frac{1}{S} \sum_{s=1}^{S} Q(x, \xi^s)$$

Sample average approximation $\Rightarrow$ Deterministic, but very large-scale optimization model

Approximating Expected Value

Key question: How many scenarios required for "good approximation"?

- Significant research into this
  [Mak et al., 1999, Shapiro and Homem-de-Mello, 2000,
  Ahmed and Shapiro, 2002, Kleywegt et al., 2002]
- Roughly: Achieving ϵ accurate solution requires $O(n_1/\epsilon^2)$ scenarios
- Good news: Required number grows "mildly" with number of decision variables and random variables
- Bad news: Poor dependence on ϵ

Conclusion

SAA enables solving SMIP problems to "modest accuracy"

Important SAA implementation details

- After obtaining a solution from SAA problem, must evaluate on *indepdendent* sample for valid objective estimate
- Multiple SAA problems need to be solved to obtain statistical estimate of optimality gap

SAA $\Rightarrow$ Finite Scenario SMIP

Scenarios: $\xi^s = (q_s, h_s, T_s, W_s)$, for $s = 1, \ldots, S$

$$\begin{aligned} \min\ & c^\top x + \frac{1}{S}\sum_{s=1}^{S} Q(x, \xi^s) \\ \text{s.t.}\ & Ax \geq b \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1} \end{aligned}$$

where

$$\begin{aligned} Q(x, \xi^s) = \min\ & q_s^\top y \\ \text{s.t.}\ & W_s y = h_s - T_s x \\ & y \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2} \end{aligned}$$

SMIP $\equiv$ Large-scale structured MIP

First option for solving a SMIP with finite scenarios

Extensive form (deterministic equivalent) of an SMIP

$$\begin{aligned} \min\ & c^\top x + \frac{1}{S}\sum_{s=1}^{S}(q_s)^\top y_s \\ \text{s.t.}\ & Ax \geq b \\ & T_s x + W_s y_s = h_s, \quad s = 1,\ldots,S \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1} \\ & y_s \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2}, \quad s = 1,\ldots,S \end{aligned}$$

Give this to your favorite MIP solver: Gurobi, CPLEX, Xpress, SCIP,...

Example: Stochastic generator expansion

$$
\begin{aligned}
\min_{x,y,z} \ & \sum_{i \in G} f_i x_i + \frac{1}{S} \sum_{s=1}^{S} \sum_{i \in G} \sum_{j \in J} c_{ij}^s y_{ijs} + \frac{1}{S} \sum_{s=1}^{S} \sum_{j \in J} q_j^s z_{js} \\
\text{s.t.} \ & \sum_{i \in G} y_{ijs} + z_{js} \geq d_{sj}, \quad j \in J, s = 1, \ldots, S \\
& \sum_{j \in J} y_{ijs} \leq C_{si} x_i, \quad i \in G, s = 1, \ldots, S \\
& x_i \in \{0, 1\}, \quad i \in I \\
& z_{js} \geq 0, y_{ijs} \geq 0, \quad i \in G, j \in J, s = 1, \ldots, S
\end{aligned}
$$

Solving via Deterministic Equivalent Form

Advantages

- Straightforward to implement
- State-of-the-art MIP solvers are able to solve impressive size problems
- This is the best approach for many problems!

Limitation

- Size can still become too large

Idea: Decomposition algorithms – solve sequence of smaller problems

- To understand these, we need to know a little integer programming methodology

Branch-and-bound

Basic idea behind most algorithms for solving integer programming problems

- Solve a *relaxation* of the problem
  - Some constraints are ignored or replaced with less stringent constraints
- Gives a lower bound on the true optimal value
- If the relaxation solution is feasible, it is optimal
- Otherwise, divide the feasible region (branch) and repeat

How long does branch-and-bound take?

Simple approximation:

$$\text{Total time} = (\text{Time to process a node}) \times (\text{Number of nodes})$$

Both can be very important:

- For very large instances (as in stochastic programming), solving a single relaxation can be too time-consuming
- Number of nodes can grow exponentially in number of decision variables if do not prune often enough

Keys to success

- Solve relaxations fast (enough)
- Obtain strong relaxations so that can prune high in tree

Valid inequalities

Let $X = \{x \in \mathbb{R}^n_+ : Ax \leq b, x_j \in \mathbb{Z}, j \in J\}$

Definition

An inequality $\pi x \leq \pi_0$ is a valid inequality for X if $\pi x \leq \pi_0$ for all $x \in X$. $(\pi \in \mathbb{R}^n, \pi_0 \in \mathbb{R})$

- Valid inequalities are also called "cutting planes" or "cuts"
- Goal of adding valid inequalities to a formulation: improve relaxation bound $\Rightarrow$ explore fewer branch-and-bound nodes

Key questions

- How to find valid inequalities?
- How to use them in a branch-and-bound algorithm?

Finding Valid Inequalities

Generator Expansion Example

- $x_i = 1$ if generator i is open, y_{ij} = demand at location j served from generator i
- Formulation we used earlier:

$$\sum_{j \in J} y_{ijs} \leq C_{si} x_i, \quad \forall i \in G$$

- Valid inequalities:

$$y_{ijs} \leq \min\{d_{sj}, C_{si}\} x_i, \quad \forall i \in I, j \in J$$

- Set of integer feasible points satisfying these are the same
- But many fractional points that satisfy original formulation do not satisfy the redundant constraints

Finding valid inequalities in general

- HUGE topic of research ⇒ Power of commercial MIP solvers

Branch-and-cut

At each node in branch-and-bound tree

1. Solve current LP relaxation $\Rightarrow \hat{x}$
2. Attempt to generate valid inequalities that cut off $\hat{x}$
3. If cuts found, add to LP relaxation and go to step 1

Why branch-and-cut?

- Reduce number of nodes to explore with improved relaxation bounds
- Add inequalities required to define feasible region (relevant for Benders for SMIP)

This approach is the heart of all modern MIP solvers

Case 1: SMIP with Continuous Recourse

$$
\begin{aligned}
\min\ & c^\top x + \sum_{s=1}^{S} p_s \theta_s \\
\text{s.t. }\ & Ax \geq b \\
& \theta_s \geq Q_s(x), \quad s = 1, \ldots, S \\
& x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}
\end{aligned}
$$

where for $s = 1, \ldots, S$

$$
\begin{aligned}
Q_s(x) = \min\ & q_s^\top y \\
\text{s.t. }\ & W_s y = h_s - T_s x \\
& y \in \mathbb{R}_+^{n_2} \times \mathbb{R}_+^{p_2}
\end{aligned}
$$

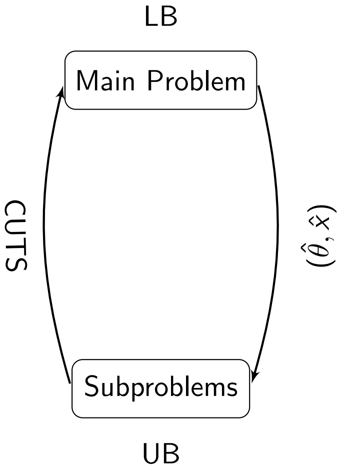
$Q_s(\cdot)$: Piecewise-linear convex function

- Supporting cuts via dual solution of the second-stage LP

Method 1: Basic Benders decomposition

$$
\begin{aligned}
(\mathrm{MP})_t^{LP} : \min_{\theta, x} \ & c^T x + \sum_{s=1}^{S} p_s \theta_s \\
\text{s.t. } & Ax \geq b, x \in \mathbb{R}_+^{n_1} \times {\color{red}\mathbb{Z}_+^{p_1}} \\
& e\theta_s \geq d_{s,t} + B_{s,t} x, \ s = 1, \ldots, S,
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{SP})^s : Q_s(\hat{x}) := \min_{y_s} \ & q_s^T y_s \\
\text{s.t. } & W_s y_s \geq h_s - T_s {\color{red}\hat{x}} \\
& y \in \mathbb{R}_+^{n_2}
\end{aligned}
$$



- Converges after finitely many iterations
- Main problem is a mixed-integer program

Deriving the Benders cuts

Under mild assumptions (so strong duality holds):

$$
\begin{aligned}
Q_s(\hat{x}) = \min\ & q_s^\top y \\
\text{s.t.}\ & W_s y \geq h_s - T_s\hat{x} \\
& y \in \mathbb{R}_+^{n_2} \times \mathbb{R}_+^{p_2} \\
= \max\ & \pi^\top (h_s - T_s\hat{x}) \\
\text{s.t.}\ & \pi^\top W_s \leq q_s \\
= \max\ & \{\bar{\pi}^\top h_s - \bar{\pi}^\top T_s\hat{x} : \bar{\pi} \in \mathrm{EXT}(\Pi_s)\}
\end{aligned}
$$

where $\mathrm{EXT}(\Pi_s)$ is the finite set of extreme points of scenario s dual feasible region

- If $\bar{\pi}$ is optimal dual solution at $\hat{x}$, add Benders cut:

$$\theta_s \geq \bar{\pi}^\top h_s - \bar{\pi}^\top T_s x$$

Basic Benders algorithm

Limitation

Solving the main MIP can become very time-consuming

- Tends to become more difficult as more cuts are added
- Unlike an LP, MIP main problem cannot be very effectively warm-started $\Rightarrow$ Significant "redundant" work

Alternative: Branch-and-Benders cut

Add Benders cuts as needed during a single branch-and-cut process.

Method 2: Branch-and-Benders cut

Initalize Benders main problem with Benders cuts

- E.g., solve the LP relaxation via Benders and keep cuts

Begin branch-and-cut algorithm. At each node in the search tree:

- Solve LP relaxation $\Rightarrow$ $(\hat{x}, \hat{\theta})$
- If LP bound exceeds known incumbent, prune.
- If $\hat{x}$ is integer feasible: $(\hat{x}, \hat{\theta})$ might not be feasible!
  - Solve scenario subproblems to generate Benders cuts
  - If $(\hat{\theta}_s, \hat{x})$ violates any Benders cut, add cut to LP relaxation and re-solve.
- If $\hat{x}$ not integer feasible:
  - Optional: Solve scenario subproblems and add Benders cuts if violated
  - Else: Branch to create new nodes

Cuts added when $\hat{x}$ is integer feasible are known as lazy cuts in MIP solvers (add via cut callback routine).

Example: Generator expansion

Example data:

- Three possible generator and four demand locations
- Fixed costs: $f = [120, 100, 90]$
- Capacity (deterministic): $C = [26, 25, 18]$
- Two equally likely scenarios: $d_1 = [12, 8, 6, 11]$, $d_2 = [8, 11, 7, 6]$

Example: After solving main LP relaxation

Main problem LP relaxation

$$
\begin{array}{ll}
\min & 120x_1 + 100x_2 + 90x_3 + 1/2(\theta_1 + \theta_2) \\
\text{s.t.} & \theta_1 \geq 1140 - 728x_1 - 675x_2 - 468x_3 \\
& \theta_1 \geq 179 - 52x_1 - 72x_3 \\
& \ldots \\
& \theta_2 \geq 990 - 728x_1 - 675x_2 - 468x_3 \\
& \theta_2 \geq 124 - 36x_3 \\
& \ldots \\
& 0 \leq x_i \leq 1, i = 1, 2, 3
\end{array}
$$

Optimal solution: $\hat{x} = (0.5, 0.76, 0.33)$, $\hat{\theta} = (129, 112)$

Optimal value (lower bound on SMIP): 286.5

- Subproblems yield no more violated Benders cuts
- Solution is optimal to the LP relaxation

Example: Branch-and-cut phase

Current main problem

$$\boxed{\begin{array}{ll} \min & 120x_1 + 100x_2 + 90x_3 + 1/2(\theta_1 + \theta_2) \\ \text{s.t.} & \theta_1 \geq 1140 - 728x_1 - 675x_2 - 468x_3 \\ & \theta_1 \geq 179 - 52x_1 - 72x_3 \\ & \dots \\ & \theta_2 \geq 990 - 728x_1 - 675x_2 - 468x_3 \\ & \theta_2 \geq 124 - 36x_3 \\ & \dots \\ & \cancel{0 \leq x_i \leq 1, i = 1, 2, 3} \\ & x_i \in \{0, 1\}, i = 1, 2, 3 \end{array}}$$

Load this (partial) formulation to the MIP solver and start solution process

- Let's first suppose MIP solver adds no cuts of its own
- It will be ready to branch (LP relaxation already solved)

Example: After some branches

$$\begin{array}{ll} \min & 120x_1 + 100x_2 + 90x_3 + 1/2(\theta_1 + \theta_2) \\ \text{s.t.} & \theta_1 \geq 1140 - 728x_1 - 675x_2 - 468x_3 \\ & \theta_1 \geq 179 - 52x_1 - 72x_3 \\ & \dots \\ & \theta_2 \geq 990 - 728x_1 - 675x_2 - 468x_3 \\ & \theta_2 \geq 124 - 36x_3 \\ & \dots \\ & 0 \leq x_i \leq 1, i = 1, 2, 3 \end{array}$$

Node 3: Fix $x_1 = 1$, $x_2 = 0$

Optimal solution: $\hat{x} = (1, 0, 0.42)$, $\hat{z} = 355.2$

Node 4: Fix $x_1 = 1$, $x_2 = 1$

Optimal solution: $\hat{x} = (1, 1, 0)$, $\hat{z} = 345.5$

Node 4 yields integer feasible solution!

- But $(\hat{x}, \hat{\theta})$ is not necessarily feasible! (if $\hat{\theta}_s < Q_s(\hat{x})$ for some s)
- We MUST check if there are any violated Benders cuts

Scenario subproblems at Node 4

Subproblems with $\hat{x} = (1, 1, 0)$:

$$\begin{array}{ll} \min & \sum_{i=1}^{3} \sum_{j=1}^{4} c_{ij} y_{ij} + \sum_{j=1}^{4} 30 z_j \\ \text{s.t.} & \sum_{i=1}^{4} y_{ij} + z_j = d_{1j}, \ \ \forall j \\ & \sum_{j=1}^{4} y_{ij} \leq 26 \cdot 1 \\ & \sum_{j=1}^{4} y_{ij} \leq 25 \cdot 1 \\ & \sum_{j=1}^{4} y_{ij} \leq 18 \cdot 0 \\ & y_{ij} \geq 0, z_j \geq 0 \end{array}$$

$$\begin{array}{ll} \min & \sum_{i=1}^{3} \sum_{j=1}^{4} c_{ij} y_{ij} + \sum_{j=1}^{4} 30 z_j \\ \text{s.t.} & \sum_{i=1}^{4} y_{ij} + z_j = d_{2j}, \ \ \forall j \\ & \sum_{j=1}^{4} y_{ij} \leq 26 \cdot 1 \\ & \sum_{j=1}^{4} y_{ij} \leq 25 \cdot 1 \\ & \sum_{j=1}^{4} y_{ij} \leq 18 \cdot 0 \\ & y_{ij} \geq 0, z_j \geq 0 \end{array}$$

Yields violated Benders cut:

$$\theta_1 \geq 141 - 36x_3$$

Does not yield a violated Benders cut

Upper bound (because $\hat{x}$ is integer feasible!):
$\sum_i f_i \hat{x}_i + \sum_s p_s Q_s(\hat{x}) = 220 + 1/2(141 + 124) = 352.5$

Branch-and-cut results

Algorithm eventually terminates after 8 nodes

- Not very efficient for a 3-variable binary problem!

What went wrong?

- Poor LP relaxations!

What to do?

- Add Benders cuts at fractional LP solutions
- Use integrality to add stronger cuts (valid inequalities not implied by LP relaxation)

Two options for using integrality to add stronger cuts

- Generate cuts directly in the main problem
- Generate cuts in the subproblems

Main problem cuts

Idea

Derive valid inequalities for the mixed-integer set:

$$\begin{aligned}\{(x,\theta) : \; & Ax \geq b, \\ & e\theta_s \geq d_{s,t} + B_{s,t}x, \;\; s = 1, \ldots, S, \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}, \theta \in \mathbb{R}^S\}\end{aligned}$$

where the constraints in second row are some Benders cuts

- E.g., split cuts, mixed-integer rounding, Gomory mixed-integer cuts,. . . ,
- Ideally, would have all Benders cuts defining $\{(x,\theta) : \theta_s \geq Q_s(x)\}$ but in general too many to enumerate

Main problem cuts: Help the MIP solver help you

Main Problem Cuts

Derive valid inequalities for the mixed-integer set:

$$\begin{aligned}\{(x,\theta) : \; & Ax \geq b, \\ & e\theta_s \geq d_{s,t} + B_{s,t}x, \;\; s = 1,\dots,S, \\ & x \in \mathbb{R}^{n_1}_{+} \times \mathbb{Z}^{p_1}_{+}, \theta \in \mathbb{R}^{S}\}\end{aligned}$$

where the constraints in second row are some Benders cuts

- MIP solvers will (try to) do this for you if Benders cuts are given to the solver as part of the formulation!
- Facility location example: Gurobi improves root node relaxation from 286.5 to 326.8 (compared to 352 opt)

Many MIP solvers do not derive cuts based on cuts you add in a callback

Main problem cuts: Help the MIP solver help you

Main Problem Cuts

Derive valid inequalities for the mixed-integer set:

$$\begin{aligned}\{(x,\theta) : \; & Ax \geq b, \\ & e\theta_s \geq d_{s,t} + B_{s,t}x, \;\; s = 1,\dots,S, \\ & x \in \mathbb{R}^{n_1}_{+} \times \mathbb{Z}^{p_1}_{+}, \theta \in \mathbb{R}^S\}\end{aligned}$$

where the constraints in second row are some Benders cuts

Takeaway

Phase 0 (solve LP relaxation with Benders, include cuts in formulation) can be very important for effective branch-and-cut implementation

- Don't just add cuts in a callback

Cuts in the subproblems: Help yourself!

Key Idea

Use valid inequalities to obtain stronger LP relaxation of each scenario mixed-integer set:

$$X_s := \{(x, y) : Ax \geq b,\ \ T_s x + W_s y = h_s$$
$$x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}, y \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2}\}$$

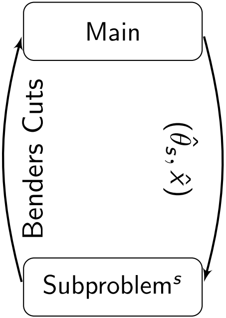
- NB: So far, we have only seen a convergent algorithm for the case y is continuous
- But subproblem approach for generating cuts is valid and useful for y mixed-integer
- See: [Sen and Higle, 2005, Sen and Sherali, 2006, Gade et al., 2014, Zhang and Küçükyavuz, 2014, Ntaimo, 2013]

Cuts generated for a single scenario $\Rightarrow$ Can still apply Benders decomposition

Cuts in subproblem

$$
\begin{aligned}
(\mathrm{MP})_t^{LP} : \min_{\theta, x} \; & c^T x + \sum_{s=1}^{S} p_s \theta_s \\
\text{s.t. } \; & Ax \geq b, x \in \mathbb{R}_+^{n_1} \times {\color{red}\mathbb{R}_+^{p_1}} \\
& e\theta_s \geq d_{s,t} + B_{s,t} x, \; \forall s, \\
& \theta \in \mathbb{R}^S
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{SP})^s : Q_s(\hat{x}) := \min_{y_s} \; & q_s^T y_s \\
\text{s.t. } \; & W_s y_s \geq h_s - T_s {\color{red}\hat{x}} \\
& y \in \mathbb{R}_+^{n_2} \times {\color{red}\mathbb{R}_+^{p_2}}
\end{aligned}
$$

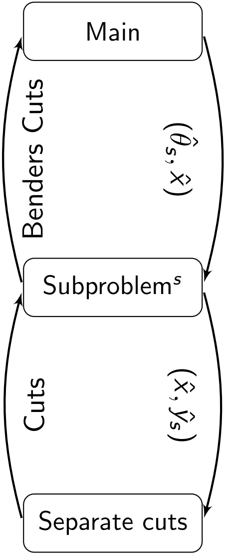


Cuts in subproblem

$$
\begin{aligned}
(\mathrm{MP})_t^{LP}: \min_{\theta, x}\ & c^T x + \sum_{s=1}^{S} p_s \theta_s \\
\text{s.t. }\ & Ax \geq b, x \in \mathbb{R}_+^{n_1} \times {\color{red}\mathbb{R}_+^{p_1}} \\
& e\theta_s \geq d_{s,t} + B_{s,t} x,\ \forall s, \\
& \theta \in \mathbb{R}^S
\end{aligned}
$$

$$
\begin{aligned}
(\mathrm{SP})^s: Q_s(\hat{x}) := \min_{y_s}\ & q_s^T y_s \\
\text{s.t. }\ & W_s y_s \geq h_s - T_s {\color{red}\hat{x}} \\
& \boxed{C_s y_s \geq g_s - D_s {\color{red}\hat{x}}} \\
& y \in \mathbb{R}_+^{n_2} \times {\color{red}\mathbb{R}_+^{p_2}}
\end{aligned}
$$

Add cuts to $(\mathrm{SP})^s$, even if it's originally an LP ($p_2 = 0$)



Cuts in the subproblems: Help yourself!

Question

How to generate valid inequalities for each scenario mixed-integer set?

$$X_s := \{(x, y) : Ax \geq b,\ T_s x + W_s y = h_s \\ x \in \mathbb{R}^{n_1}_+ \times \mathbb{Z}^{p_1}_+, y \in \mathbb{R}^{n_2}_+ \times \mathbb{Z}^{p_2}_+\}$$

Generating such cuts might require expertise in general integer programming cuts

- Split cuts, Gomory mixed-integer cuts, Chvátal-Gomory cuts,...

But it also might not...

- Use problem-specific cuts, or a better formulation
- E.g., facility location problem

Facility location: Subproblem cuts

Feasible region for a scenario s:

$$\{(x,y,z) : \begin{array}{ll} \sum_{i \in G} y_{ij} + z_j \geq d_{sj}, & j \in J \\ \sum_{j \in J} y_{ij} \leq C_{is} x_i, & i \in G \\ y_{ij} \geq 0, z_j \geq 0, & i \in G, j \in J \\ x_i \in \{0,1\}, & i \in G \end{array} \}$$

Recall: Valid inequalities

$$y_{ij} \leq \min\{d_{sj}, C_{si}\} x_i, \quad i \in G, j \in J$$

Two options for using them (because there is a "small" number of them)

- Directly add them to the scenario subproblem formulations ✓
- Add them as cuts when solving scenario subproblems

Branch-and-cut again

Initialization phase: Solve new LP relaxation via Benders

Iteration 1: Main linear problem (no θ variable yet)

$$\begin{aligned} \min\ & 120x_1 + 100x_2 + 90x_3 \\ \text{s.t. }\ & 0 \leq x_i \leq 1, i = 1, 2, 3 \end{aligned}$$

Optimal solution: $\hat{x} = (0, 0, 0)$
Optimal value (lower bound on SMIP): 0

Example: Iteration 1

Subproblems with $\hat{x} = (0,0,0)$:

$$\begin{aligned}
\min\ & \textstyle\sum_{i=1}^{3}\sum_{j=1}^{4} c_{ij}y_{ij} + \sum_{j=1}^{4} 30z_j \\
\text{s.t. } & \textstyle\sum_{i=1}^{4} y_{ij} + z_j = d_{1j},\ \forall j \\
& \textstyle\sum_{j=1}^{4} y_{ij} \leq 26 \cdot 0 \\
& \textstyle\sum_{j=1}^{4} y_{ij} \leq 25 \cdot 0 \\
& \textstyle\sum_{j=1}^{4} y_{ij} \leq 18 \cdot 0 \\
& y_{11} \leq 12 \cdot 0 \\
& \dots \\
& y_{34} \leq 11 \cdot 0 \\
& y_{ij} \geq 0, z_j \geq 0
\end{aligned}$$

$$\begin{aligned}
\min\ & \textstyle\sum_{i=1}^{3}\sum_{j=1}^{4} c_{ij}y_{ij} + \sum_{j=1}^{4} 30z_j \\
\text{s.t. } & \textstyle\sum_{i=1}^{4} y_{ij} + z_j = d_{2j},\ \forall j \\
& \textstyle\sum_{j=1}^{4} y_{ij} \leq 26 \cdot 0 \\
& \textstyle\sum_{j=1}^{4} y_{ij} \leq 25 \cdot 0 \\
& \textstyle\sum_{j=1}^{4} y_{ij} \leq 18 \cdot 0 \\
& y_{11} \leq 8 \cdot 0 \\
& \dots \\
& y_{34} \leq 6 \cdot 0 \\
& y_{ij} \geq 0, z_j \geq 0
\end{aligned}$$

Valid inequalities in Benders subproblem $\Rightarrow$ Better cuts in main problem. . .

Example after root LP solved

Final main problem LP relaxation

$$\boxed{\begin{array}{ll} \min & 120x_1 + 100x_2 + 90x_3 + 1/2(\theta_1 + \theta_2) \\ \text{s.t.} & \theta_1 \geq 1140 - 923x_1 - 922x_2 - 864x_3 \\ & \ldots \\ & \theta_2 \geq 990 - 794x_1 - 812x_2 - 758x_3 \\ & \ldots \\ & 0 \leq x_i \leq 1, i = 1, 2, 3 \end{array}}$$

Optimal solution: $\hat{x} = (0.56, 0.93, 0)$, $\hat{\theta} = (190.3, 152.4)$
Recall: Original LP relaxation bound $= 286.5$

- Bound using this formulation: 332.4 (recall opt $= 352$)
- After Gurobi cuts on this: 350.5

Recap: SMIP with continuous recourse

Two general approaches:

1. Direct Benders: Sequence of MIPs
2. Branch-and-cut adding Benders cuts (and others) in tree

Which is better?

1. Sequence of MIPs

- Easy to implement
- Tends to solve fewer scenario subproblems
- Main MIP problems may become bottleneck
- Takes full advantage of MIP solver

2. Branch-and-cut

- More difficult to implement
- Single tree eliminates redundant work
- Allows exploiting subproblem cuts, e.g., based on problem structure

My advice: Try simpler option first!

Mixed-integer recourse

What goes wrong with Benders approach with mixed-integer recourse?

Stochastic MIP

$$\begin{aligned} \min\ & c^\top x + \sum_{s=1}^{S} p_s \theta_s \\ \text{s.t. }\ & Ax \geq b \\ & \theta_s \geq Q_s(x), \quad s = 1, \dots, S \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1} \end{aligned}$$

Where for $s = 1, \dots, S$

$$\begin{aligned} Q_s(x) = \min\ & q_s^\top y \\ \text{s.t. }\ & W_s y = h_s - T_s x \\ & y \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2} \end{aligned}$$

- $Q_s(x)$: Value function of a mixed-integer program
- Benders cuts (including strengthened with subproblem cuts) still valid
- But main problem constraints $\theta_s \geq Q_s(x)$ cannot be enforced with Benders cuts alone!

Special methods for many cases

Key ingredient in each case

Use cuts/branching to enforce constraint $\theta_s \geq Q_s(x)$ for x feasible to first-stage problem

- *All* can be improved using main problem cuts and subproblem cuts

Pure binary first stage:

- Integer L-shaped cuts: [Laporte and Louveaux, 1993]
- Split cuts, transfer from one scenario to another: [Sen and Higle, 2005]
- Gomory cuts: [Gade et al., 2014]
- Fenchel cuts: [Ntaimo, 2013]
- Coordination branching: [Alonso-Ayuso et al., 2003]
- Lagrangian cuts: [Zou et al., 2019]

Pure integer first and second-stage:

- Gomory cuts [Zhang and Küçükyavuz, 2014]

Other cases (cont'd)

Mixed binary in first and second-stage:

- Lift-and-project (split) cuts: [Carøe, 1998, Tanner and Ntaimo, 2008]
- Reformulation linearization technique: [Sherali and Zhu, 2007]
- Disjunctive cuts from branch-and-cut tree: [Sen and Sherali, 2006]

Pure integer second-stage:

- Reformulation, integer subproblems, specialized branching: [Ahmed et al., 2004]

General mixed-integer both stages:

- Scaled cuts (Benders main problem cuts included in scenario subproblems): [van der Laan and Romeijnders, 2020]

Multi-cut vs. Single-cut Benders

Benders algorithm described: *Multi-cut*

- Objective function approximation: $\sum_{s \in S} p_s \theta_s$
- Benders cuts generated added for each θ_s separately

$$\theta_s \geq \bar{\pi}_s^\top h_s - \bar{\pi}_s^\top T_s x$$

Alternative: *Single-cut*

- Objective function approximation: Θ
- Benders cuts obtained by *aggregating* Benders cuts across all scenarios

$$\Theta \geq \sum_{s \in S} p_s(\bar{\pi}_s^\top h_s - \bar{\pi}_s^\top T_s x)$$

Comparison

- Multi-cut tends to converge in fewer iterations
- Single-cut keeps size of main problem small
- Single-cut may be faster if solving scenario subproblems is fast and main problem is large

Hybrids are also possible by working with a partition of scenarios

Solving the "Phase 1" LP relaxation problem

What to do if convergence of Phase 1 is slow?

- Phase 1: Add Benders cuts until LP relaxation is solved

Use the *level* algorithm or a regularized algorithm (keeps iterates from moving "too far" in consecutive iterations)

- Can be particularly helpful if using single-cut Benders algorithm

Use an alternative cut-generating problem

- Better choice of cuts to add can lead to faster convergence
- Example: [Fischetti et al., 2010]
- Especially useful for problems requiring feasibility cuts
- Restricted to multi-cut version

Accelerated generation of Lagrangian cuts

Lagrangian cuts: valid for convex hull of single-scenario projected problem

$$E^s = \{(x, \theta_s) \in X \times \mathbb{R} : Ax \geq b, \theta_s \geq Q_s(x)\}$$

Definition:

$$\begin{aligned} Q_s^*(\pi, \pi_0) &= \min_x \left\{ \pi^\top x + \pi_0 Q_s(x) : Ax \geq b, x \in X \right\} \\ &= \min_{x,y} \left\{ \pi^\top x + \pi_0 (q^s)^\top y : (x, y) \in K^s \right\} \end{aligned}$$

Valid inequality for conv(E^s) for any $\pi \in \mathbb{R}^{n_1}$, $\pi_0 \in \mathbb{R}$:

$$\pi^\top x + \pi_0 \theta_s \geq Q_s^*(\pi, \pi_0)$$

Given relaxation solution $(\hat{x}, \hat{\theta}_s)$, a Lagrangian cut is found by solving

$$\max_{\pi,\pi_0} \{ Q_s^*(\pi, \pi_0) - \pi^\top \hat{x} - \pi_0^\top \hat{\theta}_s : (\pi, \pi_0) \in \Pi_s^* \}$$

where Π_s^* is a suitably normalized set of possible cut coefficients

Lagrangian cuts (cont'd)

How to solve the cut separation problem?

$$\max_{\pi,\pi_0}\{Q_s^*(\pi,\pi_0) - \pi^\top \hat{x} - \pi_0^\top \hat{\theta}_s : (\pi,\pi_0) \in \Pi_s^*\}$$

Cutting plane algorithm!

- Replace $Q_s^*(\pi,\pi_0)$ with piecewise-linear concave over-estimation:

$$\hat{Q}_s^*(\pi,\pi_0) = \min\{\pi^\top \bar{x} + \pi_0 Q_s(\bar{x}) : \bar{x} \in \bar{K}^s\}$$

- Solve relaxed cut gen problem $\to (\hat{\pi},\hat{\pi}_0)$ + Upper bound on max cut violation
- Solve subproblem to evaluate $Q_s^*(\hat{\pi},\hat{\pi}_0)$:

$$\min_{x,y}\left\{\hat{\pi}^\top x + \hat{\pi}_0 (q^s)^\top y : (x,y) \in K^s\right\}$$

- Add optimal solution $\hat{x}$ to $\bar{K}^s$ to update $\hat{Q}_s^*$
- Repeat until an "approximately most violated" cut is found

Lagrangian cuts (cont'd)

Cutting plane algorithm challenge

- Convergence can be slow (solve many IP subproblems to generate a single cut)

Idea: Heavily restrict cut coefficient space

- [Chen and Luedtke, 2022a]: require π to be a linear combination of a small number of previously generated Benders cut coefficients
- Low-dimensional space $\Rightarrow$ Significantly faster convergence (empirically)
- With good choice of restriction, loss in cut quality not too drastic

Sparse multi-term disjunctive cuts

Binary special case:

$$E^s = \{(x, \theta_s) \in \{0,1\}^{n_1} \times \mathbb{R} : Ax \geq b, \theta_s \geq Q_s(x)\}$$

For $I \subseteq \{1, \ldots, n_1\}$, consider multi-term following relaxation:

$$E^s \subseteq \bigcup_{\chi \subseteq \{0,1\}^I} \{(x, \theta_s) \in [0,1]^{n_1} \times \mathbb{R} : Ax \geq b, \theta_s \geq Q_s(x), x_I = \chi\}$$

- Union of $2^{|I|}$ polyhedra
- [Balas, 1979]: Valid inequalities for union of polyhedra
- Multi-term disjunctions can yield stronger cuts than standard (e.g., split) approaches
- Complexity of cut generation grows significantly with $|I|$: in practice usually $|I| = 2$
- [Perregaard and Balas, 2001]: Extend computationally but still $|I| \leq 4$

Sparse cuts

"Definition":

- A cut $\theta \geq \alpha^\top x + \beta$ is sparse if the number of nonzeros in the coefficient vector α is small

Advantages of sparse cuts

- Impoves speed of re-solving of linear programs after adding cuts
- [Dey et al., 2018]: Provide conditions when sparse cuts approximate relaxation obtainable by dense cuts

Can we generate sparse multi-term disjunctive cuts efficiently?

Sparse mulit-term disjunctive cuts

Summary of results from [Chen and Luedtke, 2022b]

- Goal: generate valid inequalities for multi-term disjunction defined by I

Idea

Restrict *support* of the generated inequalities to variables in $I \Rightarrow$ Sparse

Key result:

- Generating such a valid inequality can be accomplished by solving $2^{|I|}$ linear programs *once*, then solving a "cut generating linear program" with $2^{|I|}$ constraints
- Makes it feasible to scale to $|I| \approx 10 - 12$
- Promising numerical results for problems with "natural sparsity"

Parting thoughts

Branch-and-Benders cuts

- SMIP is an important application
- But many large-scale structured MIP problems have similar structure

Work to do

- SMIP still not "routine", even in simplest case of continuous recourse
- Lack of available software that integrates state-of-the-art methods

Questions?

- jim.luedtke@wisc.edu

Ahmed, S. and Shapiro, A. (2002).
The sample average approximation method for stochastic programs with integer recourse.
Preprint available at www.optimization-online.org.

Ahmed, S., Tawarmalani, M., and Sahinidis, N. (2004).
A finite branch-and-bound algorithm for two-stage stochastic integer programs.
Mathematical Programming, 100(2):355–377.

Alonso-Ayuso, A., Escudero, L. F., and no, M. T. O. (2003).
Bfc, a branch-and-fix coordination algorithmic framework for solving some types of stochastic pure and mixed 0–1 programs.
European Journal of Operational Research, 151(3):503 – 519.

Balas, E. (1979).
Disjunctive programming.
In *Annals of Discrete Mathematics*, volume 5, pages 3–51. Elsevier.

Carøe, C. C. (1998).

Decomposition in Stochastic Integer Programming.
PhD thesis, Department of Operations Research, University of Copenhagen, Denmark.

Chen, R. and Luedtke, J. R. (2022a).
On generating lagrangian cuts for two-stage stochastic integer programs.
INFORMS Journal on Computing.
https://arxiv.org/abs/2106.04023.

Chen, R. and Luedtke, J. R. (2022b).
Sparse multi-term disjunctive cuts for the epigraph of a function of binary variables.
In *IPCO 2022*, Lecture Notes in Computer Science, Berlin. Springer-Verlag.

Dey, S. S., Molinaro, M., and Wang, Q. (2018).
Analysis of sparse cutting planes for sparse MILPs with applications to stochastic MILPs.
Mathematics of Operations Research, 43(1):304–332.

Fischetti, M., Salvagnin, D., and Zanette, A. (2010).
A note on selection of benders' cuts.
Mathematical Programming, 124:175–182.

Gade, D., Küçükyavuz, S., and Sen, S. (2014).
Decomposition algorithms with parametric Gomory cuts for two-stage stochastic integer programs.
Mathematical Programming, 144(1-2):39–64.

Kleywegt, A. J., Shapiro, A., and Homem-de Mello, T. (2002).
The sample average approximation method for stochastic discrete optimization.
SIAM Journal on Optimization, 12(2):479–502.

Laporte, G. and Louveaux, F. (1993).
The integer L-shaped method for stochastic integer programs with complete recourse.
Operations Research Letters, 13(3):133–142.

Mak, W.-K., Morton, D., and Wood, R. (1999).

Monte Carlo bounding techniques for determining solution quality in stochastic programs.
Operations Research Letters, 24:47–56.

Ntaimo, L. (2013).
Fenchel decomposition for stochastic mixed-integer programming.
Journal of Global Optimization, 55:141–163.

Perregaard, M. and Balas, E. (2001).
Generating cuts from multiple-term disjunctions.
In *International Conference on Integer Programming and Combinatorial Optimization*, pages 348–360. Springer.

Sen, S. and Higle, J. L. (2005).
The C^3 theorem and a D^2 algorithm for large scale stochastic mixed-integer programming: set convexification.
Mathematical Programming, 104:1–20.

Sen, S. and Sherali, H. (2006).

Decomposition with branch-and-cut approaches for two-stage stochastic mixed-integer programming.
Mathematical Programming, 106:203–223.

Shapiro, A. and Homem-de-Mello, T. (2000).
On the rate of convergence of optimal solutions of Monte Carlo approximations of stochastic programs.
SIAM Journal on Optimization, 11:70–86.

Sherali, H. and Zhu, X. (2007).
On solving discrete two-stage stochastic programs having mixed-integer first- and second-stage variables.
Mathematical Programming, 108:597–616.

Tanner, M. and Ntaimo, L. (2008).
Computations with disjunctive cuts for two-stage stochastic mixed 0-1 integer programs.
Journal of Global Optimization, 58:365–384.

van der Laan, N. and Romeijnders, W. (2020).

A converging benders' decomposition algorithm for two-stage mixed-integer recourse models.

Zhang, M. and Küçükyavuz, S. (2014).
Finitely convergent decomposition algorithms for two-stage stochastic pure integer programs.
SIAM Journal on Optimization, 24:1933–1951.

Zou, J., Ahmed, S., and Sun, X. A. (2019).
Stochastic dual dynamic integer programming.
Mathematical Programming, 175:461–502.