Decomposition and Reformulation in Mixed-Integer Programming

IMA New Directions Short Course on Mathematical Optimization

Jim Luedtke

Department of Industrial and Systems Engineering
University of Wisconsin-Madison

August 11, 2016

What makes integer programs hard to solve?

1. Weak LP relaxation bounds
   - Pruning in branch-and-bound is rare – huge search trees
   - Possible solution: use a better formulation or add (strong) valid inequalities

What makes integer programs hard to solve?

1. Weak LP relaxation bounds
   - Pruning in branch-and-bound is rare – huge search trees
   - Possible solution: use a better formulation or add (strong) valid inequalities
2. Huge number of variables and/or constraints
   - Just solving LP relaxation is very time-consuming
   - Possible solution: cutting plane approach – but doesn't address huge number of variables

What makes integer programs hard to solve?

1. Weak LP relaxation bounds
   - Pruning in branch-and-bound is rare – huge search trees
   - Possible solution: use a better formulation or add (strong) valid inequalities

2. Huge number of variables and/or constraints
   - Just solving LP relaxation is very time-consuming
   - Possible solution: cutting plane approach – but doesn't address huge number of variables

Another possibility: Lagrangian relaxation or column generation

- May be able to help with either or both of these challenges
- Particularly useful if enables decomposition – splitting one large problem into many smaller ones

Outline

1. Lagrangian Relaxation
   - Example
   - Lagrangian Relaxation Bounds
   - Solving the Lagrangian Dual
2. Dantzig-Wolfe Reformulation and Column Generation
3. Branch-and-Price

Motivation

Consider this IP:

$$
\begin{aligned}
z^{IP} = \max \quad & \sum_{k=1}^{K} c_k^\top x_k \\
& Dx \leq d \\
& A_k x_k \leq b_k, \quad k = 1, \ldots, K \\
& x = (x_1, \ldots, x_K) \in \mathbb{Z}_+^{nK}
\end{aligned}
$$

Motivation

Consider this IP:

$$
\begin{aligned}
z^{IP} = \max \quad & \sum_{k=1}^{K} c_k^\top x_k \\
& Dx \leq d \\
& A_k x_k \leq b_k, \quad k = 1, \ldots, K \\
& x = (x_1, \ldots, x_K) \in \mathbb{Z}_+^{nK}
\end{aligned}
$$

If only we didn't have those pesky $Dx \leq d$ constraints ...

Motivation

Drop the pesky constraints:

$$
\begin{aligned}
z^R = \max \sum_{k=1}^{K} c_k^\top x_k \\
A_k x_k \leq b_k, \quad k = 1, \ldots, K \\
x = (x_1, \ldots, x_K) \in \mathbb{Z}_+^{nK}
\end{aligned}
$$

Motivation

Drop the pesky constraints:

$$
\begin{aligned}
z^R = \max \; & \sum_{k=1}^{K} c_k^\top x_k \\
& A_k x_k \le b_k, \quad k = 1, \ldots, K \\
& x = (x_1, \ldots, x_K) \in \mathbb{Z}_+^{nK}
\end{aligned}
$$

The problem decomposes: $z^R = \sum_{k=1}^{K} z_k$ where

$$z_k = \max\{c_k^\top x_k : A_k x_k \le b_k, x_k \in \mathbb{Z}_+^n\}$$

for $k = 1, \ldots, K$

Motivation

Drop the pesky constraints:

$$
\begin{aligned}
z^R = \max \sum_{k=1}^{K} c_k^\top x_k \\
A_k x_k \leq b_k, \quad k = 1, \ldots, K \\
x = (x_1, \ldots, x_K) \in \mathbb{Z}_+^{nK}
\end{aligned}
$$

The problem decomposes: $z^R = \sum_{k=1}^{K} z_k$ where

$$z_k = \max\{c_k^\top x_k : A_k x_k \leq b_k, x_k \in \mathbb{Z}_+^n\}$$

for $k = 1, \ldots, K$

$z^R \geq z^{IP}$, but dropping $Dx \leq d$ altogether is pretty severe

Lagrangian Relaxation

Simplify notation: Let A be the matrix combining all A_k submatrices, and $b = (b_1, \ldots, b_K)$, $c = (c_1, \ldots, c_K)$

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ & Dx \leq d \\ & Ax \leq b \\ & x \in \mathbb{Z}^n_+ \end{aligned}$$

Lagrangian Relaxation

Simplify notation: Let A be the matrix combining all A_k submatrices, and $b = (b_1, \ldots, b_K)$, $c = (c_1, \ldots, c_K)$

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ & Dx \leq d \\ & Ax \leq b \\ & x \in \mathbb{Z}^n_+ \end{aligned}$$

Lagrangian relaxation: relax the constraints $Dx \leq d$ by dualizing them – adding them to the objective with a penalty for violation

Lagrangian Relaxation

Simplify notation: Let A be the matrix combining all A_k submatrices, and $b = (b_1, \ldots, b_K)$, $c = (c_1, \ldots, c_K)$

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ & Dx \leq d \\ & Ax \leq b \\ & x \in \mathbb{Z}^n_+ \end{aligned}$$

Lagrangian relaxation: relax the constraints $Dx \leq d$ by dualizing them – adding them to the objective with a penalty for violation

- Problem with just constraints $Ax \leq b$ should be easier to solve
- We'll discuss how to choose the constraints to dualize later
- For simplicity we assume $x \in \mathbb{Z}^n_+$

Lagrangian Relaxation

First, re-write our IP, with $X := \{x \in \mathbb{Z}^n_+ : Ax \leq b\}$

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ \text{subject to } & Dx \leq d \\ & x \in X \end{aligned}$$

Let $u \in \mathbb{R}^m_+$, and define the following Lagrangian Relaxation problem:

$$IP(u): \quad z(u) = \max\{c^\top x + u^\top(d - Dx) : x \in X\}$$

Theorem

For any $u \in \mathbb{R}^m_+$, $z(u) \geq z^{IP}$.

Lagrangian Relaxation

First, re-write our IP, with $X := \{x \in \mathbb{Z}^n_+ : Ax \leq b\}$

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ \text{subject to } & Dx \leq d \\ & x \in X \end{aligned}$$

Let $u \in \mathbb{R}^m_+$, and define the following Lagrangian Relaxation problem:

$$IP(u): \quad z(u) = \max\{c^\top x + u^\top(d - Dx) : x \in X\}$$

Theorem

For any $u \in \mathbb{R}^m_+$, $z(u) \geq z^{IP}$.

Why? Let x^* be an optimal solution to z^{IP} ($z^{IP} = c^\top x^*$ and $d - Dx^* \geq 0$).

$$z(u) \geq c^\top x^* + u^\top(d - Dx^*) \geq z^{IP}$$

Lagrangian Dual

For $u \in \mathbb{R}^m_+$,

$$z(u) = \max\{c^\top x + u^\top (d - Dx) : x \in X\}$$

Definition

The problem:

$$w^{LD} = \min\{z(u) : u \in \mathbb{R}^m_+\}$$

is called a Lagrangian dual.

Properties

- $w^{LD} \geq z^{IP}$
- $w^{LD} \leq z(u)$ for all $u \in \mathbb{R}^m_+$

Lagrangian Dual

For $u \in \mathbb{R}^m_+$,

$$z(u) = \max\{c^\top x + u^\top (d - Dx) : x \in X\}$$

Definition

The problem:

$$w^{LD} = \min\{z(u) : u \in \mathbb{R}^m_+\}$$

is called a Lagrangian dual.

Properties

- $w^{LD} \geq z^{IP}$
- $w^{LD} \leq z(u)$ for all $u \in \mathbb{R}^m_+$

Modification with equality constraints $Dx = d$: variables u are free

Lagrangian Dual

$$IP(u): \quad z(u) = \max\{c^\top x + u^\top (d - Dx) : x \in X\}$$

Theorem

Let $u \in \mathbb{R}^m_+$ and $\hat{x}$ be an optimal solution to $IP(u)$. If,

(1) $D\hat{x} \leq d$, and

(2) $u^\top (d - D\hat{x}) = 0$,

then $\hat{x}$ is an optimal solution to IP.

Note: The second condition is necessary: $\hat{x}$ could be feasible to IP, but not optimal.

Lagrangian Dual

$$IP(u): \quad z(u) = \max\{c^\top x + u^\top(d - Dx) : x \in X\}$$

Theorem

Let $u \in \mathbb{R}^m_+$ and $\hat{x}$ be an optimal solution to $IP(u)$. If,

(1) $D\hat{x} \leq d$, and

(2) $u^\top(d - D\hat{x}) = 0$,

then $\hat{x}$ is an optimal solution to IP.

Note: The second condition is necessary: $\hat{x}$ could be feasible to IP, but not optimal.

- (1) $\Rightarrow \hat{x}$ is feasible
- (2) $\Rightarrow z^{IP} \leq z(u) = c^\top\hat{x} + u^\top(d - D\hat{x}) = c^\top\hat{x}$

Example: Stochastic Integer Programming

Extensive form of SIP

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

Example: Stochastic Integer Programming

Copy first-stage variables

$$
\begin{aligned}
z^{SMIP} = \min \sum_{s=1}^{S} p_s (c^\top x_s + q_s^\top y_s) & \\
A x_s \geq b & \qquad s = 1, \ldots, S \\
T_s x_s + W_s y_s = h_s & \qquad s = 1, \ldots, S \\
x_s = \sum_{s'=1}^{S} p_{s'} x_{s'} & \qquad s = 1, \ldots, S \\
x_s \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}, & \qquad s = 1, \ldots, S \\
y_s \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2}, & \qquad s = 1, \ldots, S
\end{aligned}
$$

Relax Nonanticipativity

- The constraints $x_s = \sum_{s'=1}^{S} p_{s'} x_{s'}$ are called *nonanticipativity constraints.*

Relax Nonanticipativity

- The constraints $x_s = \sum_{s'=1}^{S} p_{s'} x_{s'}$ are called *nonanticipativity constraints.*
- Relax these constraints using Lagrangian Relaxation with dual vectors $\lambda = (\lambda_1, \ldots, \lambda_S)$:

$$\mathcal{L}(\lambda) := \min \sum_{s=1}^{S} p_s (c^\top x_s + q_s^\top y_s) + \sum_{s=1}^{S} p_s \lambda_s^\top \left(x_s - \sum_{s'=1}^{S} p_{s'} x_{s'} \right)$$

$$\begin{aligned}
& Ax_s \geq b && s = 1, \ldots, S \\
& T_s x_s + W_s y_s = h_s && s = 1, \ldots, S \\
& x_s \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}, && s = 1, \ldots, S \\
& y_s \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2}, && s = 1, \ldots, S
\end{aligned}$$

Relax Nonanticipativity

- The constraints $x_s = \sum_{s'=1}^{S} p_{s'} x_{s'}$ are called *nonanticipativity constraints.*
- Relax these constraints using Lagrangian Relaxation with dual vectors $\lambda = (\lambda_1, \ldots, \lambda_S)$:

$$\mathcal{L}(\lambda) := \min \sum_{s=1}^{S} p_s (c^\top x_s + q_s^\top y_s) + \sum_{s=1}^{S} p_s \lambda_s^\top \left(x_s - \sum_{s'=1}^{S} p_{s'} x_{s'} \right)$$

$$\begin{aligned}
& Ax_s \geq b && s = 1, \ldots, S \\
& T_s x_s + W_s y_s = h_s && s = 1, \ldots, S \\
& x_s \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}, && s = 1, \ldots, S \\
& y_s \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2}, && s = 1, \ldots, S
\end{aligned}$$

- Rewrite the objective ($\bar{\lambda} = \sum_{s=1}^{S} p_s \lambda_s$):

$$\sum_{s=1}^{S} p_s \left(\left(c + (\lambda_s - \bar{\lambda}) \right)^\top x_s + q_s^\top y_s \right)$$

Relax Nonanticipativity

- Rewritten objective ($\bar{\lambda} = \sum_{s=1}^{S} p_s \lambda_s$):

$$\sum_{s=1}^{S} p_s \Big(\big(c + (\lambda_s - \bar{\lambda})\big)^\top x_s + q_s^\top y_s \Big)$$

- Normalize λ_s so that $\bar{\lambda} = 0$
- Lagrangian relaxation problem decomposes: $\mathcal{L}(\lambda) = \sum_s p_s D_s(\lambda_s)$ where

$$\begin{aligned} D_s(\lambda_s) := \; & \min \; (c + \lambda_s)^\top x + q_s^\top y_s \\ & \text{s.t.} \; Ax \geq b, \; T_s x + W_s y = h_s \\ & x \in \mathbb{R}_+^{n_1} \times \mathbb{Z}_+^{p_1}, y \in \mathbb{R}_+^{n_2} \times \mathbb{Z}_+^{p_2} \end{aligned}$$

- Each subproblem is a deterministic mixed-integer program

Lagrangian Dual Problem

- For any $\lambda = (\lambda_1, \ldots, \lambda_S)$ with $\sum_s p_s \lambda_s = 0$,

$$\mathcal{L}(\lambda) \leq z^{SMIP}$$

Lagrangian dual

Find best lower bound:

$$w^{LD} := \max\left\{ \mathcal{L}(\lambda) : \sum_{s=1}^{S} p_s \lambda_s = 0 \right\}$$

How Good is the Bound From the Lagrangian dual?

The integer program we're trying to solve

$$z^{IP} = \max\{c^\top x : Dx \leq d, x \in X\} \tag{IP}$$

where $X = \{x \in \mathbb{Z}^n_+ : Ax \leq b\}$

How Good is the Bound From the Lagrangian dual?

The integer program we're trying to solve

$$z^{IP} = \max\{c^\top x : Dx \leq d, x \in X\} \tag{IP}$$

where $X = \{x \in \mathbb{Z}^n_+ : Ax \leq b\}$

Theorem

$$w^{LD} = \max\{c^\top x : Dx \leq d, x \in \text{conv}(X)\}$$

How Good is the Bound From the Lagrangian dual?

The integer program we're trying to solve

$$z^{IP} = \max\{c^\top x : Dx \leq d, x \in X\} \tag{IP}$$

where $X = \{x \in \mathbb{Z}^n_+ : Ax \leq b\}$

Theorem

$$w^{LD} = \max\{c^\top x : Dx \leq d, x \in \text{conv}(X)\}$$

Let $P = \{x \in \mathbb{R}^n_+ : Ax \leq b\}$

$$z^{LP} = \max\{c^\top x : Dx \leq d, x \in P\}$$

- $P \supseteq \text{conv}(X)$, so $w^{LD} \leq z^{LP}$
- If $P = \text{conv}(X)$, then $w^{LD} = z^{LP}$
- $w^{LD} < z^{LP}$ is only possible if $P \neq \text{conv}(X)$

Let's prove it for special case: X bounded, all integer vars

Strength of Lagrangian Dual of SMIP

Theorem

The Lagrangian dual bound satisfies

$$w^{LD} = \min \left\{ c^\top x + \sum_{s=1}^{S} p_s y_s : (x, y_s) \in \text{conv}(X_s), s = 1, \ldots, S \right\}$$

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

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

Strength of Lagrangian Dual of SMIP

Theorem

The Lagrangian dual bound satisfies

$$w^{LD} = \min \left\{ c^\top x + \sum_{s=1}^{S} p_s y_s : (x, y_s) \in \text{conv}(X_s), s = 1, \ldots, S \right\}$$

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

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

- In general $w^{LD} < z^{SMIP}$
- But $w^{LD} \geq z^{SLP}$ (the usual LP relaxation)

Strength of Lagrangian Dual of SMIP

Theorem

The Lagrangian dual bound satisfies

$$w^{LD} = \min \left\{ c^\top x + \sum_{s=1}^{S} p_s y_s : (x, y_s) \in \text{conv}(X_s), s = 1, \ldots, S \right\}$$

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

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

- In general $w^{LD} < z^{SMIP}$
- But $w^{LD} \geq z^{SLP}$ (the usual LP relaxation)
- w^{LD} at least as good as any bound obtained using cuts in single scenario subproblems
- In many test instances, w^{LD} is very close to z^{SMIP}

How to solve the Lagrangian dual?

Lagrangian dual

$$w^{LD} = \min\{z(u) : u \geq 0\}$$

How to find u^* that solves this optimization problem?

How to solve the Lagrangian dual?

Lagrangian dual

$$w^{LD} = \min\{z(u) : u \geq 0\}$$

How to find u^* that solves this optimization problem?

Key insight

$z(u)$ is a piecewise linear convex function of u.

We already have seen

$$z(u) = \max\{c^\top x^t + u^\top (d - Dx^t) : t = 1, \ldots, T\}$$

where $\{x^t : t = 1, \ldots, T\}$ are the finitely many points in $X = \{x \in \mathbb{Z}^n_+ : Ax \leq b\}$.

- Extends also to mixed-integer, unbounded sets (like Benders analysis)

Option 1: Subgradient

Lagrangian dual: A nonsmooth convex optimization problem

$$w^{LD} = \min\{z(u) : u \geq 0\}$$

- Assume for this algorithm that X is bounded so $\text{conv}(X)$ has no rays

Recall

Let $f : \mathbb{R}^m \to \mathbb{R}$ and $u \in \mathbb{R}^m$. A vector $\gamma(u)$ is called a subgradient of f at u if

$$f(v) \geq f(u) + \gamma(u)^\top (v - u) \quad \forall v \in \mathbb{R}^m$$

Option 1: Subgradient

Subgradient algorithm for generic convex problem: $\min\{f(u) : u \geq 0\}$

1. Initialize: $u = u^0$
2. Iteration $k \geq 0$:
   - Calculate $f(u^k)$ and find a subgradient $\gamma(u^k)$ of f at u^k
   - Step:

$$u^{k+1} = \max\{u^k - \mu_k \gamma(u^k), 0\}$$

Option 1: Subgradient

Subgradient algorithm for generic convex problem: $\min\{f(u) : u \geq 0\}$

1. Initialize: $u = u^0$
2. Iteration $k \geq 0$:
   - Calculate $f(u^k)$ and find a subgradient $\gamma(u^k)$ of f at u^k
   - Step:

$$u^{k+1} = \max\{u^k - \mu_k \gamma(u^k), 0\}$$

Notes:

- The values μ_k are step sizes that satisfy at a minimum: $\mu_k > 0$ and $\mu_k \to 0$ as $k \to \infty$

Option 1: Subgradient

Subgradient algorithm for Lagrangian dual problem:

$$\min\{z(u) : u \geq 0\}$$

1. Initialize: $u = u^0$
2. Iteration $k \geq 0$:
   - Calculate $z(u^k)$ and find a subgradient $\gamma(u^k)$ of z at u^k
   - Step:

$$u^{k+1} = \max\{u^k - \mu_k \gamma(u^k), 0\}$$

Key question: How to calculate $z(u^k)$ and $\gamma(u^k)$?

Option 1: Subgradient

Subgradient algorithm for Lagrangian dual problem:

$$\min\{z(u) : u \geq 0\}$$

How to calculate $z(u^k)$ and $\gamma(u^k)$?

Option 1: Subgradient

Subgradient algorithm for Lagrangian dual problem:

$$\min\{z(u) : u \geq 0\}$$

How to calculate $z(u^k)$ and $\gamma(u^k)$?

$$z(u^k) = \max\{c^\top x + (u^k)^\top (d - Dx) : x \in X\}$$

Option 1: Subgradient

Subgradient algorithm for Lagrangian dual problem:

$$\min\{z(u) : u \geq 0\}$$

How to calculate $z(u^k)$ and $\gamma(u^k)$?

$$z(u^k) = \max\{c^\top x + (u^k)^\top (d - Dx) : x \in X\}$$

Subgradients of z

Let $x^k \in X$ be an optimal solution to calculation of $z(u^k)$. Then

$$\gamma(u^k) = d - Dx^k$$

is a subgradient of z at u^k.

Option 1: Subgradient

Subgradients of z

Let $x^k \in X$ be an optimal solution to calculation of $z(u^k)$. Then

$$\gamma(u^k) = d - Dx^k$$

is a subgradient of z at u^k.

Proof: Let $v \in \mathbb{R}^m$.

$$z(v) = \max\{c\top x + v^\top(d - Dx) : x \in X\}$$

Option 1: Subgradient

Subgradients of z

Let $x^k \in X$ be an optimal solution to calculation of $z(u^k)$. Then

$$\gamma(u^k) = d - Dx^k$$

is a subgradient of z at u^k.

Proof: Let $v \in \mathbb{R}^m$.

$$\begin{aligned} z(v) &= \max\{c\top x + v^\top(d - Dx) : x \in X\} \\ &\geq c^\top x^k + v^\top(d - Dx^k) \end{aligned}$$

Option 1: Subgradient

Subgradients of z

Let $x^k \in X$ be an optimal solution to calculation of $z(u^k)$. Then

$$\gamma(u^k) = d - Dx^k$$

is a subgradient of z at u^k.

Proof: Let $v \in \mathbb{R}^m$.

$$\begin{aligned} z(v) &= \max\{c\top x + v^\top(d - Dx) : x \in X\} \\ &\geq c^\top x^k + v^\top(d - Dx^k) \\ &= c^\top x^k + (u^k)^\top(d - Dx^k) + v^\top(d - Dx^k) - (u^k)^\top(d - Dx^k) \end{aligned}$$

Option 1: Subgradient

Subgradients of z

Let $x^k \in X$ be an optimal solution to calculation of $z(u^k)$. Then

$$\gamma(u^k) = d - Dx^k$$

is a subgradient of z at u^k.

Proof: Let $v \in \mathbb{R}^m$.

$$\begin{aligned} z(v) &= \max\{c\top x + v^\top(d - Dx) : x \in X\} \\ &\geq c^\top x^k + v^\top(d - Dx^k) \\ &= c^\top x^k + (u^k)^\top(d - Dx^k) + v^\top(d - Dx^k) - (u^k)^\top(d - Dx^k) \\ &= z(u^k) + (v - u^k)^\top(d - Dx^k) \end{aligned}$$

Option 1: Subgradient

Subgradients of z

Let $x^k \in X$ be an optimal solution to calculation of $z(u^k)$. Then

$$\gamma(u^k) = d - Dx^k$$

is a subgradient of z at u^k.

Proof: Let $v \in \mathbb{R}^m$.

$$\begin{aligned} z(v) &= \max\{c\top x + v^\top(d - Dx) : x \in X\} \\ &\geq c^\top x^k + v^\top(d - Dx^k) \\ &= c^\top x^k + (u^k)^\top(d - Dx^k) + v^\top(d - Dx^k) - (u^k)^\top(d - Dx^k) \\ &= z(u^k) + (v - u^k)^\top(d - Dx^k) \\ &= z(u^k) + (v - u^k)^\top\gamma(u^k) \end{aligned}$$

Option 1: Subgradient

- Advantages: easy to implement, can potentially get good improvement in few iterations
- Disadvantage: convergence can be slow because it ignores history of information (past solution values and subgradients)

Solving the Lagrangian Dual: Option 2

We have already seen:

$$\begin{aligned} w^{LD} &= \min\{z(u) : u \geq 0\} \\ &= \min\ \eta + d^\top u \\ &\quad \text{s.t. } \eta + u^\top (Dx^t) \geq c^\top x^t, \quad t = 1, \ldots, T, \\ &\quad\quad u \geq 0 \end{aligned}$$

This last problem is a linear program with a huge number of constraints

- Solve by a cutting plane algorithm – Just like Benders decomposition!

Solving the Lagrangian Dual: Other Options

Lagrangian dual

$$\min\{z(u) : u \geq 0\}$$

Bundle methods: Nonlinear programming methods for nonsmooth optimization

- Use all (or much of) past subgradients as in cutting plane algorithm

Solving the Lagrangian Dual: Other Options

Lagrangian dual

$$\min\{z(u) : u \geq 0\}$$

Bundle methods: Nonlinear programming methods for nonsmooth optimization

- Use all (or much of) past subgradients as in cutting plane algorithm
- Use a stabilization technique to limit "bouncing around"
  - Trust region: restrict $\| u - u^k \| \leq \alpha_k$ in master LP

Solving the Lagrangian Dual: Other Options

Lagrangian dual

$$\min\{z(u) : u \geq 0\}$$

Bundle methods: Nonlinear programming methods for nonsmooth optimization

- Use all (or much of) past subgradients as in cutting plane algorithm
- Use a stabilization technique to limit "bouncing around"
  - Trust region: restrict $\| u - u^k \| \leq \alpha_k$ in master LP
  - Augmented objective: add $\lambda_k \| u - u^k \|$ to master LP objective

Solving the Lagrangian Dual: Other Options

Lagrangian dual

$$\min\{z(u) : u \geq 0\}$$

Bundle methods: Nonlinear programming methods for nonsmooth optimization

- Use all (or much of) past subgradients as in cutting plane algorithm
- Use a stabilization technique to limit "bouncing around"
  - Trust region: restrict $\| u - u^k \| \leq \alpha_k$ in master LP
  - Augmented objective: add $\lambda_k \| u - u^k \|$ to master LP objective
  - Bundle-level: solve unrestricted LP to obtain L, then solve a second problem finding closest u to u^k that obtains a minimum improvement in the objective

Choosing the Lagrangian Dual

Trade-offs in choosing constraints to dualize

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ & Dx \leq d,\ Ax \leq b \\ & x \in \mathbb{Z}^n_+ \end{aligned}$$

- Dualize "hard" constraints leaving something easy (convex hull described by remaining inequalities)
  - Bound is only as strong as the LP bound
  - Subproblems (for fixed u) will be easy to solve
  - Might be more efficient than LP

Choosing the Lagrangian Dual

Trade-offs in choosing constraints to dualize

$$\begin{aligned} z^{IP} = \max\ & c^\top x \\ & Dx \leq d,\ Ax \leq b \\ & x \in \mathbb{Z}^n_+ \end{aligned}$$

- Dualize "hard" constraints leaving something easy (convex hull described by remaining inequalities)
  - Bound is only as strong as the LP bound
  - Subproblems (for fixed u) will be easy to solve
  - Might be more efficient than LP
- Leave some "hard" constraints undualized (convex hull not described by the inequalities)
  - Bound can be significantly better than LP bound
  - Solving the subproblems can be more difficult
  - If enables decomposition, the "hard" subproblems may not be too bad in practice – e.g. knapsack

What to do After Solving the Lagrangian Dual?

1. Use it as a basis for heuristics
   - Problem-specific
   - E.g., fix some variables based on Lagrangian subproblem and solve a smaller problem
   - Often, relatively easy to restore feasibility
     - E.g., in stochastic IP, fix first-stage vars, solve second-stage problems

What to do After Solving the Lagrangian Dual?

1. Use it as a basis for heuristics
   - Problem-specific
   - E.g., fix some variables based on Lagrangian subproblem and solve a smaller problem
   - Often, relatively easy to restore feasibility
     - E.g., in stochastic IP, fix first-stage vars, solve second-stage problems
2. Use it as a relaxation within a branch-and-bound search
   - Instead of solving LP relaxations!
   - Branching is problem-specific (we'll see this later)

Outline

1 Lagrangian Relaxation

2 Dantzig-Wolfe Reformulation and Column Generation
- Dantzig-Wolfe Reformulation
- Column Generation

3 Branch-and-Price

Motivation

Consider this IP:

$$
\begin{aligned}
z^{IP} = \max \quad & \sum_{k=1}^{K} c_k^\top x_k \\
& Dx \leq d, \\
& x_k \in X_k, \quad k = 1, \ldots, K
\end{aligned}
$$

Where $X_k = \{x \in \mathbb{Z}_+^n : A_k x \leq b_k\}$

- Could relax the constraints $Dx \leq d$, as in Lagrangian relaxation
- Different option: Dantzig-Wolfe Reformulation

Dantzig-Wolfe Reformulation

Assume X_k is bounded, so it has finitely many points:

- I.e., $X_k = \{x^{k,t}\}_{t=1}^{T_k}$

Then, $x_k \in X_k$ if and only if there exists $\lambda_{k,t} \in \{0,1\}$, $t = 1, \ldots, T_k$ such that:

$$x_k = \sum_{t=1}^{T_k} \lambda_{k,t} x^{k,t}, \quad \sum_{t=1}^{T_k} \lambda_{k,t} = 1$$

Dantzig-Wolfe Reformulation (2)

Replace the constraints $x_k \in X_k$ with λ formulation yields:

$$
\begin{aligned}
z^{IP} = \max \quad & \sum_{k=1}^{K} c_k^\top x_k \\
& \sum_{k=1}^{K} D_k x_k \leq d, \\
& x_k = \sum_{t=1}^{T_k} \lambda_{k,t} x^{k,t}, \quad \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad k = 1, \ldots, K \\
& \lambda_{k,t} \in \{0,1\}, \quad t = 1, \ldots, T_k, \; k = 1, \ldots, K
\end{aligned}
$$

Dantzig-Wolfe Reformulation (2)

Replace the constraints $x_k \in X_k$ with λ formulation yields:

$$
\begin{aligned}
z^{IP} = \max \; & \sum_{k=1}^{K} c_k^\top x_k \\
& \sum_{k=1}^{K} D_k x_k \leq d, \\
& x_k = \sum_{t=1}^{T_k} \lambda_{k,t} x^{k,t}, \quad \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad k = 1, \ldots, K \\
& \lambda_{k,t} \in \{0,1\}, \quad t = 1, \ldots, T_k, \; k = 1, \ldots, K
\end{aligned}
$$

Now, substitute out the x_k variables...

Dantzig-Wolfe Reformulation (3)

Obtain the Dantzig-Wolfe Reformulation

$$
\begin{aligned}
\max \quad & \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(c_k^\top x^{k,t}\right) \lambda_{k,t} \\
& \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(D_k x^{k,t}\right) \lambda_{k,t} \leq d \\
& \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad k = 1, \ldots, K \\
& \lambda_{k,t} \in \{0,1\}, \quad t = 1, \ldots, T_k, \ k = 1, \ldots, K
\end{aligned}
$$

Questions about Dantzig-Wolfe formulation

1. How can we solve the linear programming relaxation? (Since it may have exponentially many variables)

Questions about Dantzig-Wolfe formulation

1. How can we solve the linear programming relaxation? (Since it may have exponentially many variables)
2. What is the strength of the LP relaxation, compared to the original formulation?

Questions about Dantzig-Wolfe formulation

1. How can we solve the linear programming relaxation? (Since it may have exponentially many variables)
2. What is the strength of the LP relaxation, compared to the original formulation?
3. If we can solve the LP relaxation, then how should we use it in branch-and-bound?

Questions about Dantzig-Wolfe formulation

1. How can we solve the linear programming relaxation? (Since it may have exponentially many variables)
2. What is the strength of the LP relaxation, compared to the original formulation?
3. If we can solve the LP relaxation, then how should we use it in branch-and-bound?

We've already seen the answer to the first two questions!

Let's revisit our IP

$$
\begin{aligned}
z^{IP} = \max \quad & \sum_{k=1}^{K} c_k^\top x_k \\
& \sum_{k=1}^{K} D_k x_k \leq d, \\
& x_k \in X_k, k = 1, \ldots, K
\end{aligned}
$$

For $u \in \mathbb{R}^m_+$, Lagrangian subproblem is:

$$
z(u) = \max \Big\{ \sum_{k=1}^{K} c_k^\top x_k + u^\top \big(d - \sum_{k=1}^{K} D_k x_k\big) : x_k \in X_k, k = 1, \ldots, K \Big\}
$$

Let's revisit our IP

$$
\begin{aligned}
z^{IP} = \max \ & \sum_{k=1}^{K} c_k^\top x_k \\
& \sum_{k=1}^{K} D_k x_k \leq d, \\
& x_k \in X_k, k = 1, \ldots, K
\end{aligned}
$$

For $u \in \mathbb{R}^m_+$, Lagrangian subproblem is:

$$
\begin{aligned}
z(u) &= \max \Big\{ \sum_{k=1}^{K} c_k^\top x_k + u^\top \big(d - \sum_{k=1}^{K} D_k x_k\big) : x_k \in X_k, k = 1, \ldots, K \Big\} \\
&= u^\top d + \sum_{k=1}^{K} \max \Big\{ (c_k^\top - u^\top D_k) x : x \in X_k \Big\}
\end{aligned}
$$

Lagrangian Relaxation, cont'd

$$
\begin{aligned}
w^{LD} &= \min_{u \geq 0} \; z(u) \\
&= \min_{u \geq 0} \; u^\top d + \sum_{k=1}^{K} \max\Big\{ (c_k^\top - u^\top D_k) x : x \in X_k \Big\} \\
&= \min \; u^\top d + \sum_{k=1}^{K} \eta_k \\
&\qquad \eta_k + u^\top D_k x^{k,t} \geq c_k^\top x^{k,t}, \quad t = 1, \ldots, T_k, \;\; k = 1, \ldots, K \\
&\qquad u \geq 0
\end{aligned}
$$

where $\left\{ x^{k,t} : t = 1, \ldots, T_k \right\}$ are the extreme points of $\mathrm{conv}(X_k)$ for all k

Lagrangian Relaxation: Take the Dual

$$
\begin{aligned}
w^{LD} = \min \; & u^\top d + \sum_{k=1}^{K} \eta_k \\
& \eta_k + u^\top D_k x^{k,t} \geq c_k^\top x^{k,t}, \quad t = 1, \ldots, T_k, \; k = 1, \ldots, K \\
= \max \; & \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(c_k^\top x^{k,t} \right) \lambda_{k,t} \\
& \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(D_k x^{k,t} \right) \lambda_{k,t} \leq d, \\
& \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad k = 1, \ldots, K, \\
& \lambda_{k,t} \geq 0, \quad t = 1, \ldots, T_k, \; k = 1, \ldots, K
\end{aligned}
$$

Dual of the LP Formulation

$$
\begin{aligned}
w^{LD} = \max \; & \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(c_k^\top x^{k,t} \right) \lambda_{k,t} \\
& \sum_{k=1}^{K} \sum_{t=1}^{T_k} \left(D_k x^{k,t} \right) \lambda_{k,t} \leq d, \\
& \sum_{t=1}^{T_k} \lambda_{k,t} = 1, \quad k = 1, \ldots, K, \\
& \lambda_{k,t} \geq 0, \quad t = 1, \ldots, T_k, \; k = 1, \ldots, K
\end{aligned}
$$

Does this look familiar?

Questions About Dantzig-Wolfe Formulation

1. How can we solve the linear programming relaxation?
   - Column generation: Adding a variable corresponds to adding a constraint in Lagrangian Dual problem
   - "Pricing problem" will be identical to Lagrangian relaxation subproblem

Questions About Dantzig-Wolfe Formulation

1. How can we solve the linear programming relaxation?
   - Column generation: Adding a variable corresponds to adding a constraint in Lagrangian Dual problem
   - "Pricing problem" will be identical to Lagrangian relaxation subproblem
2. What is the strength of the LP relaxation, compared to the original formulation?
   - Exactly equal to the Lagrangian relaxation bound
   - Always at least as good as original LP relaxation
   - Exactly equal to LP relaxation if $\text{conv}(X_k) = \{x \in \mathbb{R}^{n_k} : A_k x \leq b_k\}$ for all k

Formulating With a Huge Numbers of Variables

- DW reformulation $\Rightarrow$ Formulation with a huge number of variables as a *reformulation* of a given problem
- In many applications, it is conceptually easier to start with a formulation having a huge number of variables
- Can also be useful in avoiding symmetry in alternative formulations

Example: Cutting Stock Problem

A steel company makes a set of products I

- Product $i \in I$ has width $w_i > 0$
- Product $i \in I$ has demand $b_i > 0$

Products are made by cutting them from a roll of length L

- Multiple products can be cut from a roll
- E.g., $L = 10$, $w_1 = 4$, $w_2 = 3$
- One roll can make one product 1 and two of product 2 $(4 + 2 * 3 = 10)$

Let's formulate an IP to minimize number of rolls used

Example: Cutting Stock Problem

A steel company makes a set of products I

- Product $i \in I$ has width $w_i > 0$
- Product $i \in I$ has demand $b_i > 0$

New set: P - Set of all possible "cutting patterns"

- $a_{ip} =$ number of product i made by pattern $p \in P$

Example: Cutting Stock Problem

A steel company makes a set of products I

- Product $i \in I$ has width $w_i > 0$
- Product $i \in I$ has demand $b_i > 0$

New set: P - Set of all possible "cutting patterns"

- $a_{ip} =$ number of product i made by pattern $p \in P$

Let $x_p =$ number of pattern p to cut

$$\begin{aligned} \min \quad & \sum_{p \in P} x_p \\ \text{s.t.} \quad & \sum_{p \in P} a_{ip} x_p \geq b_i, \quad i \in I \\ & x_p \in \mathbb{Z}_+, \quad p \in P \end{aligned}$$

Column Generation for Cutting Stock

Let P' be a given subset of cutting patterns

- E.g., include all patterns producing just one product to ensure feasibilty

Restricted Master LP

$$\min \sum_{p \in P'} x_p$$

$$\text{s.t.} \sum_{p \in P'} a_{ip} x_p \geq b_i, \quad i \in I$$

$$x_p \geq 0, \quad p \in P'$$

Dual

$$\max \sum_{i \in I} b_i \pi_i$$

$$\text{s.t.} \sum_{i \in I} a_{ip} \pi_i \leq 1, \quad p \in P'$$

$$x_p \geq 0, \quad p \in P'$$

Let $(\hat{x}, \hat{\pi})$ be an optimal primal/dual solution. $\hat{x}$ optimal $\Leftrightarrow$ $\hat{\pi}$ feasible

Outline

1 Lagrangian Relaxation

2 Dantzig-Wolfe Reformulation and Column Generation

3 Branch-and-Price

Branch-and-Price

What to do after solving the LP relaxation by column generation?

Possibilities:

- Heuristics: E.g., start MIP solve with this subset of columns
  - If find a feasible solution close to column generation LP bound, you may be happy
- If you need an optimal solution (or better than you have found) ...
  - Branch-and-price: Do column generation at every node!

Branch-and-Price

Difficulty in branching

- Fix $\lambda_{k,t} = 1$, we choose $x^{k,t}$ completely – very restrictive
- Fix $\lambda_{k,t} = 0$, exclude just the one solution $x^{k,t}$!

Fixing $\lambda_{k,t} = 0$ also destroys subproblem structure:

- Must exclude the solution x^k:

$$\max\{(c_k - u^\top D_k)x : x \in X_k, x \neq x^k\}$$

Branch-and-Price: Alternative Branching

Branch on the "original" problem variables x_k

- E.g. fix $x_{ki} = 0$
- In LP Master Problem: Delete columns $x^{k,t}$ with $x_i^{k,t} \neq 0$
  - Fix $\lambda^{k,t} = 0$ for such columns if already generated

Branch-and-Price: Alternative Branching

Branch on the "original" problem variables x_k

- E.g. fix $x_{ki} = 0$
- In LP Master Problem: Delete columns $x^{k,t}$ with $x_i^{k,t} \neq 0$
  - Fix $\lambda^{k,t} = 0$ for such columns if already generated
- Avoid generating new columns with $x_i^{k,t} \neq 0$: In subproblem k, fix $x_i = 0$

$$\max\left\{(c_k - u^\top D_k)^\top x : x \in X_k, x_i = 0\right\}$$

  - Often preserves subproblem structure

Branch-and-Price: Alternative Branching

Branch on the "original" problem variables x_k

- E.g. fix $x_{ki} = 0$
- In LP Master Problem: Delete columns $x^{k,t}$ with $x_i^{k,t} \neq 0$
  - Fix $\lambda^{k,t} = 0$ for such columns if already generated
- Avoid generating new columns with $x_i^{k,t} \neq 0$: In subproblem k, fix $x_i = 0$

$$\max\left\{(c_k - u^\top D_k)^\top x : x \in X_k, x_i = 0\right\}$$

  - Often preserves subproblem structure
- Similarly for alternative branch $x_{ki} = 1$

Branching in Cutting Stock

Branching is more difficult in cutting stock formulation

- Multiple patterns being selected
- Cannot simply say a pattern must make $\leq k$ or $\geq k+1$ of a product
- Would enforce that for all patterns

Must branch by adding constraints (not just bounds on variables)

- LP relaxations get larger
- Pricing problems become more complicated

We'll skip these details

How to Implement Branch-and-Price?

- Solving the LP master problem is not too hard to implement
  - Good news: in many applications it often yields a provably near-optimal solution!

How to Implement Branch-and-Price?

- Solving the LP master problem is not too hard to implement
  - Good news: in many applications it often yields a provably near-optimal solution!
- Commercial solvers don't allow you to add variables in the branch-and-bound tree

How to Implement Branch-and-Price?

- Solving the LP master problem is not too hard to implement
  - Good news: in many applications it often yields a provably near-optimal solution!
- Commercial solvers don't allow you to add variables in the branch-and-bound tree
- SCIP (`scip.zib.de` - free for academics) supports branch-and-price
- Generic Column Generation (`www.or.rwth-aachen.de/gcg/`) - based on SCIP
- Some open-source frameworks are available at `www.coin-or.org`
  - BCP: Branch-cut-price
  - CHiPPS and DIP – also supports Lagrangian relaxation
  - ABACUS