Advanced Decomposition Methods
Part I: all is one, one is all

Antonio Frangioni
Dipartimento di Informatica, Università di Pisa

COST/MINO Ph.D. School on Advanced Optimization Methods

Roma — June 8, 2016
Outline

1. Block-Structured (Mixed-Integer NonLinear) Programs
2. Dual decomposition (Dantzig-Wolfe/Lagrangian/Column Generation)
3. Primal decomposition (Benders’/Resource)
4. The Integer Case
5. Example Applications (the Reformulation before the Reformulation)
6. Conclusions (for now)
Many applications of Mixed-Integer NonLinear Programming are large-scale: millions/billions of variables/constraints

Good news: (almost) all large-scale problems are block-structured

Usually several nested forms of structure, but two main ones:

- **Block-diagonal**: complicating constraints
- **Staircase-structured**: complicating variables

Relaxing constraints / fixing variables yields independent subproblems

\[ \rightarrow \text{much easier because of size and/or structure (integrality, \ldots)} \]
Example I: Two-stage Stochastic (Linear) Programs

- Problems involving decisions over time and uncertainty
- First-stage (here-and-now) decisions $x$, constraints $E_0 x \leq b_0$
- Set $S$ of scenarios, realization known only after deciding $x$
- Recourse decisions $z_s$, different for each scenario $s \in S$, constraints $E_0^s x + E_s z_s \leq b_s$
- Minimize here-and-now cost plus average cost of reserve actions
  \[
  \min \left\{ c_0 x + \sum_{s \in S} \pi_s c_s z_s : E_0 x \leq b_0, \ E_0^s x + E_s z_s \leq b_s \quad s \in S \right\}
  \]
- Extends to multi-stage (structure repeats “fractally” into each $E_s$)
- Often other structures inside $E$, network a common one
- Extends to nonlinear risk measures (CVaR, . . . ), integer variables, . . .
- Many applications: energy\cite{1}, water, logistics, telecom, finance, . . .

\[1\] Tahanan, van Ackooij, F., Lacalandra “Large-scale Unit Commitment under uncertainty” 4OR 2015
Example II: (Linear) Multicommodity Network Design

- Graph $G = (N, A)$, multicommodity network design model

$$\min \sum_{k \in K} \sum_{(i,j) \in A} d^k c^k_{ij} x^k_{ij} + \sum_{(i,j) \in A} f_{ij} z_{ij}$$

(1)

$$\sum_{(i,j) \in A} x^k_{ij} - \sum_{(j,i) \in A} x^k_{ji} = \begin{cases} 1 & \text{if } i = s^k \\ 1 & \text{if } i = t^k \\ 0 & \text{otherwise} \end{cases} \quad i \in N, \ k \in K$$

(2)

$$\sum_{k \in K} d^k x^k_{ij} \leq u_{ij} z_{ij} \quad (i,j) \in A$$

(3)

$$x^k_{ij} \in [0, 1] \quad (i,j) \in A, \ k \in K$$

(4)

$$z_{ij} \in \{0, 1\} \quad (i,j) \in A$$

(5)

- $K \equiv \text{commodities} \equiv (s^k, t^k, d^k)$ (not completely generic)

- Pervasive structure in most of combinatorial optimization

- Many applications: logistic, transportation, telecom, energy, . . .

Dual decomposition, a.k.a.
Inner Approximation
Dantzig-Wolfe decomposition
Lagrangian Relaxation
Column Generation
Block-diagonal Convex (Linear) Program

- **Block-diagonal program:** convex $X$, $n$ "complicating" constraints

$$ (\Pi) \quad \max \{ \, cx : Ax = b \ , \ x \in X \, \} $$

e.g., $X = \{ \, x : Ex \leq d \, \} = \bigotimes_{k \in K} \left( X^k = \{ \, x^k : E^k x^k \leq d^k \, \} \right)$

($|K|$ large $\implies$ (\Pi) very large), $Ax = b$ linking constraints

- **We can efficiently optimize upon** $X$, for different reasons:
  - a bunch of (many, much) smaller problems instead of a large one
  - $X$ has (the $X^k$ have) structure (shortest path, knapsack, ...)

(much more so than solving the whole of (\Pi), anyway)

- **In other words we could efficiently solve** (\Pi) if linking constraints were removed: how to exploit it?
Dantzig-Wolfe reformulation

- Dantzig-Wolfe reformulation\cite{Dantzig60}: $X$ convex $\Rightarrow$ represent it by points

$$X = \{ x = \sum_{\bar{x} \in X} \bar{x} \theta_{\bar{x}} : \sum_{\bar{x} \in X} \theta_{\bar{x}} = 1 , \theta_{\bar{x}} \geq 0 \quad \bar{x} \in X \}$$

then reformulate $(\Pi)$ in terms of the convex multipliers $\theta$

$$\begin{align*}
(\Pi) \quad \begin{cases} 
\max & c \left( \sum_{\bar{x} \in X} \bar{x} \theta_{\bar{x}} \right) \\
A \left( \sum_{\bar{x} \in X} \bar{x} \theta_{\bar{x}} \right) & = b \\
\sum_{\bar{x} \in X} \theta_{\bar{x}} & = 1 , \theta_{\bar{x}} \geq 0 \quad \bar{x} \in X \end{cases}
\end{align*}$$

- only $n + 1$ rows (but how many columns?)
- note that “$\bar{x} \in X$” is an index, not a constraint ($\theta$ is the variable)

- A rather semi-infinite program, but “only” $\bar{x} \in \text{ext} X$ needed

- Not that this makes it any less infinite, unless $X$ is a polytope (compact polyhedron) $\Rightarrow$ finite set of vertices

Could this ever be a good idea? Actually, it could: polyhedra may have few faces and many vertices . . . or vice-versa

- **n-cube** \( |x_i| \leq 1 \quad \forall i \) \( 2n \) faces \( 2^n \) vertices
- **n-co-cube** \( \sum_i |x_i| \leq 1 \) \( 2^n \) faces \( 2n \) vertices

Except, most often the number of vertices is too large

A (linear) program with (exponentially/infinitely) many columns

But, efficiently optimize over \( X \) \( \rightleftharpoons \) generate vertices (\( \equiv \) columns)
Dantzig-Wolfe decomposition $\equiv$ Column Generation

- $\mathcal{B} \subset X$ (small), solve restriction of $(\Pi)$ with $X \rightarrow \mathcal{B}$, i.e.,

$$
(\Pi_{\mathcal{B}}) \begin{cases} 
\max & \sum_{\bar{x} \in \mathcal{B}} (c \bar{x}) \theta_{\bar{x}} \\
\sum_{\bar{x} \in \mathcal{B}} (A \bar{x}) \theta_{\bar{x}} & = b \\
\sum_{\bar{x} \in \mathcal{B}} \theta_{\bar{x}} & = 1, \quad \theta_{\bar{x}} \geq 0 \quad \bar{x} \in \mathcal{B}
\end{cases}
$$

- "master problem" ($\mathcal{B}$ small, not too costly)
- note how the parentheses have moved: linearity is needed (for now)

- If $\mathcal{B}$ contains the “right” columns, $x^* = \sum_{\bar{x} \in \mathcal{B}} \bar{x}\theta_{\bar{x}}^*$ optimal for $(\Pi)$

- How do I tell if $\mathcal{B}$ contains the “right” columns? Use duality

$$
(\Delta_{\mathcal{B}}) \min \{ \ yb + v : v \geq c\bar{x} - y(A\bar{x}) \quad \bar{x} \in \mathcal{B} \} 
$$

$$
= \min \{ \ f_{\mathcal{B}}(y) = \max \{ \ c\bar{x} + y(b - A\bar{x}) : \bar{x} \in \mathcal{B} \} \}
$$

one constraint for each $\bar{x} \in \mathcal{B}$
Dantzig-Wolfe decomposition ≡ Lagrangian relaxation

- Dual of (Π): \((\Delta) \equiv (\Delta_X)\) (many constraints)

- \(f_B = \) lower approximation of Lagrangian function

\[
(\Pi_y) \quad f(y) = \max \{ cx + y(b - Ax) : x \in X \}
\]

- Assumption: optimizing over \(X\) is “easy” for each objective \(\Rightarrow\) obtaining \(\bar{x}\) s.t. \(f(y) = c\bar{x} + y(b - A\bar{x})\) is “easy”

- Important: \((\Pi_y)\) Lagrangian relaxation\(^{[4]}\), \(f(y) \geq v(\Pi) = v(\Delta) \forall y\)
  provided \((\Pi_y)\) is solved exactly (or at least a \(\bar{f} \geq f(y)\) is used)

- Thus, \((\Delta_B)\) outer approximation of the Lagrangian dual

\[
(\Delta) \quad \min \{ f(y) = \max \{ cx + y(b - Ax) : x \in X \} \}
\]

\(^{[4]}\) Geoffrion “Lagrangean relaxation for integer programming” Mathematical Programming Study 1974
Lagrangian duality vs. Linear duality

- Note about the LP case \( X = \{ x : Ex \leq d \} \):

  \[
  (\Delta) \quad \min \left\{ yb + \max \left\{ (c - yA)x : Ex \leq d \right\} \right\} \\
  \equiv \min \left\{ yb + \min \left\{ wd : wE = c - yA, \, w \geq 0 \right\} \right\} \\
  \equiv \min \left\{ yb + wd : wE + yA = c, \, w \geq 0 \right\} \\
  \equiv \text{exactly the linear dual of } (\Pi)
  \]

- \( y \) “partial” duals: duals \( w \) of \( Ex \leq d \) “hidden” in the subproblem

- There is only one duality

- Will repeatedly come in handy
Dantzig-Wolfe decomposition \equiv Dual row generation

- Primal/dual optimal solution \( x^*/(v^*, y^*) \) out of \( (\Pi_B)/(\Delta_B) \)
- \( x^* \) feasible to \( (\Pi) \), so optimal \( \iff (v^*, y^*) \) feasible to \( (\Delta) \)
  \[ \iff v^* \geq (c - y^*A)x \quad \forall x \in X \]
  \[ \iff v^* \geq \max \{ (c - y^*A)x : x \in X \} \]
- In fact: \( v^* \geq (c - y^*A)\bar{x} \equiv y^*b + v^* \geq f(y^*) \implies \\
  v(\Pi) \geq cx^* = y^*b + v^* \geq f(y^*) \geq v(\Delta) \geq v(\Pi) \implies \\
  x^*/(v^*, y^*) \) optimal
- Otherwise, \( B = B \cup \{ \bar{x} \} \): add new column to \( (\Pi_B) \) / row to \( (\Delta_B) \), rinse, repeat
- Clearly finite is \( ext X \) is, globally convergent anyway:
  the cutting plane algorithm for convex programs\(^5\) (applied to \( (\Delta) \))

v^* = f_B(y^*) lower bound on \( v(\Pi_B) \)
Geometry of the Lagrangian dual

- $v^* = f_B(y^*)$ lower bound on $v(\Pi_B)$
- Optimal solution $\bar{x}$ gives separator between $(v^*, y^*)$ and $\text{epi } f$
Geometry of the Lagrangian dual

- $v^* = f_B(y^*)$ lower bound on $v(\Pi_B)$
- Optimal solution $\bar{x}$ gives separator between $(v^*, y^*)$ and $epi \ f$
- $(c\bar{x}, A\bar{x}) = \text{new row in } (\Delta_B)$ (subgradient of $f$ at $y^*$)
Dantzig-Wolfe decomposition $\equiv$ Inner Approximation

- “Abstract” view of $(\Pi_B)$: $\text{conv}(B)$ inner approximation of $X$

$$\begin{align*}
(\Pi_B) \quad \max \{ \ cx : \ Ax = b , \ x \in \text{conv}(B) \ \} \\
\end{align*}$$

- $x^*$ solves the Lagrangian relaxation of $(\Pi_B)$ with $y^*$, i.e.,

$$x^* \in \arg\max \{ \ (c - y^*A)x : \ x \in \text{conv}(B) \ \}$$

$$\implies (c - y^*A)x \leq (c - y^*A)x^* \text{ for each } x \in \text{conv}(B) \subseteq X$$

- $(c - y^*A)\bar{x} = \max\{ (c - y^*A)x : x \in X \} \geq (c - y^*A)x^*$

- Column $\bar{x}$ has positive reduced cost

$$(c - y^*A)(\bar{x} - x^*) = (c - y^*A)\bar{x} - cx^* + y^*b = (c - y^*A)\bar{x} - v^* > 0$$

$$\implies \bar{x} \notin \text{conv}(B) \implies \text{makes sense to add } \bar{x} \text{ to } B$$
\[ \mathbf{Ax} = \mathbf{b} \]

\[ \mathbf{c - y}^* \mathbf{A} \]

\[ \text{conv}(\mathbf{B}) \cap \mathbf{Ax} = \mathbf{b} \]

\[ \text{from all } x \in X \text{ better than } x^* \]
c − y*A separates $conv(B) \cap Ax = b$ from all $x \in X$ better than $x^*$

Thus, optimizing it allows finding new points (if any)
Geometry of Dantzig-Wolfe/Column Generation

- $c - y^*A$ separates $\text{conv}(B) \cap Ax = b$ from all $x \in X$ better than $x^*$
- Thus, optimizing it allows finding new points (if any)
- Issue: $\text{conv}(B) \cap Ax = b$ must be nonempty
Extension I: the Unbounded Case

- $X$ unbounded $\iff$ $\text{rec } X \supset \{0\} \implies f(y) = v(\Pi_y) = \infty$ happens
- $X = \text{conv}(\text{ext } X = X_0) + \text{cone}(\text{ext } \text{rec } X = X_\infty)$
- $B = (B_0 \subset X_0) \cup (B_\infty \subset X_\infty) = \{\text{points } \bar{x}\} \cup \{\text{rays } \bar{\chi}\} \implies$
  $$
  (\Pi_B) \begin{cases}
  \max & c \left( \sum_{\bar{x} \in B_0} \bar{x} \theta_{\bar{x}} + \sum_{\bar{\chi} \in B_\infty} \bar{\chi} \theta_{\bar{\chi}} \right) \\
  A \left( \sum_{\bar{x} \in B_0} \bar{x} \theta_{\bar{x}} + \sum_{\bar{\chi} \in B_\infty} \bar{\chi} \theta_{\bar{\chi}} \right) = b \\
  \sum_{\bar{x} \in B_0} \theta_{\bar{x}} = 1 \\
  \theta_{\bar{x}} \geq 0 \quad \bar{x} \in B_0, \quad \theta_{\bar{\chi}} \geq 0 \quad \bar{\chi} \in B_\infty
  \end{cases}
  $$
- In $(\Delta_B)$, constraints $y(A\bar{\chi}) \geq c\bar{\chi}$ (a.k.a. “feasibility cuts”)
- $(\Pi_{y^*})$ unbounded $\iff$ $(c - y^* A)\bar{\chi} > 0$ for some $\bar{\chi} \in \text{rec } X$
  (violated constraint) $\implies B_\infty = B_\infty \cup \{\bar{\chi}\}$
- $(\Delta) = \min \{ f(y) : y \in Y \}$, $(\Pi_{y^*})$ provides either subgradients of $f$
  (a.k.a. “optimality cuts”), or violated valid inequalities for $Y$
Extension II: the Nonlinear Case

- Nonlinear case: \( c(\cdot) \) concave, \( A(\cdot) \) component-wise convex
  
  \((\Pi)\) \( \max \ \{ c(x) : A(x) \leq b , \ x \in X \} \)

  \((\Delta)\) \( \max \ \{ f(y) = yb + \max \ \{ c(x) - yA(x) : x \in X \} : y \geq 0 \} \)

- Any \( \bar{x} \in X \) still gives \( f(y) \geq c(\bar{x}) + y(b - A(\bar{x})) \), same \((\Delta_B) / (\Pi_B)\)

- \( c(\sum_{\bar{x} \in B} \bar{x} \theta_{\bar{x}}) \geq \sum_{\bar{x} \in B} c(\bar{x}) \theta_{\bar{x}} \) \((c(\cdot) \) concave),
  
  \( A(\sum_{\bar{x} \in B} \bar{x} \theta_{\bar{x}}) \leq \sum_{\bar{x} \in B} A(\bar{x}) \theta_{\bar{x}} \leq b \) \((A(\cdot) \) convex) \implies

  \((\Pi_B)\) safe inner approximation \((\nu(\Pi_B) \leq \nu(\Pi))\)

- Basically everything keeps working, but you may need constraint qualification\(^6\) (usually easy to get)

---

Primal decomposition, a.k.a.
Outer Approximation
Benders’ decomposition
Resource decomposition
Staircase-structured Convex (Linear) Program

- Staircase-structured program: convex $X$, “complicating” variables

$$\max \{ cx + ez : Dx + Ez \leq d , \ x \in X \}$$

e.g, $Dx + Ez \leq d \equiv D_k x + E_k z_k \leq d_k \ k \in K \ (|K| \text{ large}) \implies$

$$Z(x) = \{ z : Ez \leq d - Dx \}$$

$$= \bigotimes_{k \in K} (Z_k(x) = \{ z_k : E_k z_k \leq d_k - D_k x \})$$

- We can efficiently optimize upon $Z(x)$, for different reasons:
  - a bunch of (many, much) smaller problems instead of a large one
  - $Z(x)$ has (the $Z_k(x)$ have) structure (shortest path, knapsack, . . . )

(much more so than solving the whole of $(\Pi)$, anyway)

- In other words we could efficiently solve $(\Pi)$ if linking variables were fixed: how to exploit it?
Benders’ reformulation

- Benders’ reformulation: define the **convex value function**

  \[ (B) \quad \max \left\{ \, cx + v(x) = \max \{ \, ez : Ez \leq d - Dx \, \} : x \in X \, \right\} \]

  (note: clearly \( v(x) = -\infty \) happens)

- Clever trick\(^7\): **use duality** to reformulate the inner problem

  \[ v(x) = \min \left\{ \, w(d - Dx) : w \in W = \{ \, w : wE = e , w \geq 0 \, \} \, \right\} \]

  so that \( W \) does not depend on \( x \)

- As usual, \( W = \text{conv}( \text{ext} \, W = W_0 ) + \text{cone}( \text{ext rec} \, W = W_\infty ) \implies \)

  \[ (B) \quad \max \, cx + v \]

  \[ \begin{align*}
  v & \leq \bar{w}(d - Dx) & \bar{w} & \in W_0 \\
  0 & \leq \bar{w}(d - Dx) & \bar{w} & \in W_\infty \\
  x & \in X 
  \end{align*} \]

  still very large, but we can generate \( \bar{w} / \bar{w} \) by computing \( v(x) \)

---

Benders’ decomposition

- Select (small) $\mathcal{B} = (\mathcal{B}_0 \subset \mathcal{W}_0) \cup (\mathcal{B}_\infty \subset \mathcal{W}_\infty)$, solve master problem
  
  $\max \{ cx + \nu \}$

  $\nu \leq \bar{w}(d - Dx)$ \quad $\bar{w} \in \mathcal{B}_0$

  $0 \leq \bar{\omega}(d - Dx)$ \quad $\bar{\omega} \in \mathcal{B}_\infty$

  $x \in X$

  $= \max \{ cx + \nu_{\mathcal{B}}(x) : x \in X \cap \mathcal{V}_{\mathcal{B}} \}$, where

  $\nu_{\mathcal{B}}(x) = \min \{ \bar{w}(d - Dx) : \bar{w} \in \mathcal{B}_0 \} \geq \nu(x)$, $\mathcal{V}_{\mathcal{B}} \supseteq \text{dom } \nu$

- Find (primal) optimal solution $x^*$, compute $\nu(x^*)$, get either $\bar{w}$ or $\bar{\omega}$, update either $\mathcal{B}_0$ or $\mathcal{B}_\infty$, rinse & repeat

- Benders’ decomposition $\equiv$ Cutting Plane approach to $\{B\}^{[5]}$

- Spookily similar to the Lagrangian dual, ain’t it?

- Except, constraints are now attached to dual objects $\bar{w}$ / $\bar{\omega}$
Benders is Lagrange . . .

- Block-diagonal case

\[(\Pi) \quad \max \{ \ cx : \ Ax = b, \ Ex \leq d \} \]
\[(\Delta) \quad \min \{ \ yb + wd : \ wE + yA = c, \ w \geq 0 \} \]

Think of \( y \) as complicating variables in \((\Delta)\), you get

\[(\Pi) \quad \max \{ \ cx : \ Ax = b, \ Ey \leq d \} \]
\[(\Delta) \quad \min \{ \ yb + \min\{ \ wd : \ wE = c - yA, \ w \geq 0 \} \} \]
\[= \min \{ \ yb + \max\{ \ (c - yA)x : \ Ex \leq d \} \} \]

i.e., the Lagrangian dual of \(\Pi\)

- The value function of \((\Delta)\) is the Lagrangian function of \((\Pi)\)
...Lagrange is Benders...

- Dual of $(\Pi)$ (linear case $X = \{ x : Ax = b \}$)

  $(\Pi)$ \( \max \{ cx + ez : Dx + Ez \leq d, Ax = b \} \)

  $(\Delta)$ \( \min \{ yb + wd : yA + wD = c, wE = e, w \geq 0 \} \)

  Lagrangian dual of the dual constraints $yA + wD = c$ (multiplier $x$):

  $(\Delta)$ \( \max \{ \min \{ yb + wd + (c - yA + wD)x : wE = e, w \geq 0 \} \} \)

  \[ = \max \{ cx + \min \{ y(b - Ax) + w(d - Dx) : wE = e, w \geq 0 \} \} \]

  \[ = \max \{ cx + \min \{ y(b - Ax) \} + \]

  \[ \min \{ w(d - Dx) : wE = e, w \geq 0 \} \} \]

  \[ = \max \{ cx + \min \{ ez : Dx + Ez \leq e \} : Ax = b \} \]

  i.e., Benders’ reformulation of $(\Pi)$

- The Lagrangian function of $(\Delta)$ is the value function of $(\Pi)$
Both Lagrange and Benders boil down to

\[
\min \left\{ \phi(\lambda) : \lambda \in \Lambda \right\}
\]

with \(\Lambda\) and \(\phi\) convex, nondifferentiable, both only implicitly known by means of a (potentially costly) oracle that, given \(\bar{\lambda}\), provides:

- either \(\phi(\bar{\lambda}) < \infty\) and \(\bar{g} \in \partial \phi(\bar{\lambda}) \equiv \phi(\lambda) \geq \phi(\bar{\lambda}) + \bar{g}(\lambda - \bar{\lambda})\)
- or \(\phi(\bar{\lambda}) = \infty\) and a valid cut for \(\Lambda\) violated by \(\bar{\lambda}\)

“Natural” algorithm: the Cutting Plane method\(^5\) \(\equiv\) revised simplex method with mechanized pricing in the discrete case

Many other variants/algorithms possible (cf. Part II)
The Nonlinear Case

- Each $f(x, \cdot)$ and $G(x, \cdot)$ concave, $Z$ convex:
  
  $(\Pi)$ $\max \{ f(x, z) : G(x, z) \geq 0 , x \in X , z \in Z \}$
  
  $(B)$ $\max \{ v(x) : x \in X \}$

  where $v(x) = \max \{ f(x, z) : G(x, z) \geq 0 , z \in Z \}$

  $(B) \equiv (\Pi)$ without assumptions on $f(\cdot, z), G(\cdot, z)$ and $X$ (hard)

- Which duality would you use? Lagrangian\[8\], of course

  $v(x) = \min \{ \max \{ f(x, z) + \lambda G(x, z) : z \in Z \} : \lambda \geq 0 \}$

- Under appropriate constraint qualification, two cases occur:
  
  - either $\exists \bar{\lambda} \geq 0 , \bar{z} \in Z$ s.t. $v(x^*) = f(x^*, \bar{z}) + \bar{\lambda} G(x^*, \bar{z}) > -\infty$
  
  - or $v(x^*) = -\infty \implies \{ z \in Z : G(x^*, z) \geq 0 \} = \emptyset \implies \exists \bar{\nu} \geq 0 , \bar{z} \in Z$

  s.t. $\max \{ \bar{\nu} G(x^*, z) : z \in Z \} = \bar{\nu} G(x^*, \bar{z}) < 0$

---

The Nonlinear Case (cont.d)

- General form of the master problem
  \[(B) \quad \max v\]
  \[v \leq \max \{ f(x, z) + \bar{\lambda}G(x, z) : z \in Z \} \quad \bar{\lambda} \in \Lambda_0\]
  \[0 \leq \max \{ \bar{\nu}G(x^*, z) : z \in Z \} \quad \bar{\nu} \in \Lambda_\infty\]
  \[x \in X\]

- Er ... how on Earth do you manage those nasty “max”?

- Must be that the “max” can be done independently of \(x\)!

- Example: \(f(z_i)\) concave, univariate
  \[\max \{ \sum_i x_i f(z_i) : \sum_i x_i z_i \leq c, \quad z_i \geq 0, \quad A x \leq b, \quad x \geq 0 \}\]
  \[v(x) = \min_{\lambda} \sum_i \max \{ x_i (f(z_i) - \lambda z_i) : z_i \geq 0 \} + \lambda c\]
  \[v(x) \leq \sum_i x_i \max \{ (f(z_i) - \bar{\lambda} z_i) : z_i \geq 0 \} + \bar{\lambda} c\]

  can optimize on the \(z\) independently from the \(x\) \(\Rightarrow\)
  “normal” linear cuts
The Integer Case
Block-structured Integer Programs

- What if \( X \) combinatorial (e.g., \( X = \{ x \in \mathbb{Z}^n : Ex \leq d \} \))?

\[
(\Pi) \quad \max \{ cx : Ax = b, \ x \in X \}
\]

- The Lagrangian dual is

\[
(\Delta) \quad \min \{ yb + \max \{ (c - yA)x : x \in X \} \}
\]

nothing changes if we can still efficiently optimize over \( X \), e.g. due to size (decomposition) and/or structure (integrality)

- ... except we are solving a different problem:

\[
(\bar{\Pi}) \quad \left\{ \begin{array}{l}
\max \ c \left( \sum_{\bar{x} \in X} \bar{x} \theta_{\bar{x}} \right) \\
A \left( \sum_{\bar{x} \in X} \bar{x} \theta_{\bar{x}} \right) = b \\
\sum_{\bar{x} \in X} \theta_{\bar{x}} = 1 \quad , \quad \theta_{\bar{x}} \geq 0 \quad \bar{x} \in X
\end{array} \right.
\]

\[\equiv \max \{ cx : Ax = b, \ x \in \text{conv}(X) \}\]

i.e., a (potentially good) relaxation of \((\Pi)\)
Good news: \((\bar{\Pi})\) better (not worse) than continuous relaxation
\((\text{conv}(X) \subseteq \{ x \in \mathbb{R}^n : Ex \leq d \})\)

Bad news: if \((\Pi_y)\) “too easy” \((\text{conv}(X) = \{ x \in \mathbb{R}^n : Ex \leq d \})\), a.k.a. integrality property), then \((\bar{\Pi})\) same as continuous relaxation

Trade-off: \((\Pi_y)\) must be easy, but not too easy (no free lunch)

Anyway, at best gives good bounds \(\Rightarrow\)
Branch & Bound with DW/Lagrangian/CG \(\equiv\) Branch & Price

Branching nontrivial: may destroy subproblem structure
\(\Rightarrow\) branch on \(x\) (but \((\Pi_B)\) is on \(\theta\))

Lamentably little support from off-the-shelf tools: master problem gives a valid bound only at termination, although subproblem always gives one (but not associated to continuous feasible solution)
Digression: How to Choose your Lagrangian relaxation

- There may be many choices
  \( (\Pi) \max \{ \ cx : \ Ax = b \ , \ Ex \leq d \ , \ x \in \mathbb{Z}^n \} \)
  \( (\Pi'_y) \max \{ \ cx + y(b - Ax) : \ x \in X' = \{ x \in \mathbb{Z}^n : \ Ex \leq d \} \} \)
  \( (\Pi''_w) \max \{ \ cx + w(d - Ex) : \ x \in X'' = \{ x \in \mathbb{Z}^n : \ Ax = b \} \} \)

- The best between \( (\Delta') \) and \( (\Delta'') \) depends on integrality of \( X', X'' \):
  - If both have it, both \( (\Delta') \) and \( (\Delta'') \) \( \equiv \) continuous relaxation
  - If only one has it, the one that does not, but if both don't have it?

- Here comes Lagrangian decomposition\(^9\) (scale by 1/2)
  \( (\Pi) \equiv \max \{ \ cx' + cx'' : \ x' \in X' , \ x'' \in X'' , \ x' = x'' \} \)
  \( (\Pi_\lambda) \max \{ (c + \lambda)x' : x' \in X' \} + \max \{ (c - \lambda)x'' : x'' \in X'' \} \)
  \( (\bar{\Delta}) \equiv (\bar{\Pi}) \max \{ \ cx : \ x \in \text{conv}(X') \cap \text{conv}(X'') \} \)
  better than both (but need to solve two hard subproblems)

---

Geometry of Lagrangian Decomposition

Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation

Lagrangian relaxation of blue constraints shrinks the red (= grey) part
Lagrangian relaxation of red constraints shrinks the blue (= grey) part
Lagrangian decomposition (both red and blue constraints) shrinks both $\Rightarrow$ the grey part more

But the intersection of convex hulls is larger (bad) than the convex hull of the intersection

Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation
Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation

Lagrangian relaxation of blue constraints shrinks the red (= grey) part

Lagrangian relaxation of red constraints shrinks the blue (= grey) part

Lagrangian decomposition (both red and blue constraints) shrinks both $\Rightarrow$ grey part more

But the intersection of convex hulls is larger (bad) than the convex hull of the intersection

Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation

Lagrangian relaxation of blue constraints
Geometry of Lagrangian Decomposition

- Intersection between red and blue ≡ grey ≡ continuous relaxation
- Lagrangian relaxation of blue constraints shrinks the red (⇒ grey) part

A. Frangioni (DI — UniPi) Advanced Decomposition Methods I Roma 2016 32 / 42
Geometry of Lagrangian Decomposition

- Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation
- Lagrangian relaxation of blue constraints shrinks the red ($\rightarrow$ grey) part
- Lagrangian relaxation of red constraints
Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation

- Lagrangian relaxation of blue constraints shrinks the red ($\Rightarrow$ grey) part
- Lagrangian relaxation of red constraints shrinks the blue ($\Rightarrow$ grey) part
Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation

Lagrangian relaxation of blue constraints shrinks the red ($\implies$ grey) part

Lagrangian relaxation of red constraints shrinks the blue ($\implies$ grey) part

Lagrangian decomposition (both red and blue constraints)
Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation

Lagrangian relaxation of blue constraints shrinks the red ($\implies$ grey) part

Lagrangian relaxation of red constraints shrinks the blue ($\implies$ grey) part

Lagrangian decomposition (both red and blue constraints) shrinks both $\implies$ the grey part more
Geometry of Lagrangian Decomposition

- Intersection between red and blue $\equiv$ grey $\equiv$ continuous relaxation
- Lagrangian relaxation of blue constraints shrinks the red ($\rightarrow$ grey) part
- Lagrangian relaxation of red constraints shrinks the blue ($\rightarrow$ grey) part
- Lagrangian decomposition (both red and blue constraints) shrinks both $\rightarrow$ the grey part more
- But the intersection of convex hulls is larger (bad) than the convex hull of the intersection
Digression: Alternative Good Formulations for $conv(X)$

- (Under mild assumptions) $conv(X)$ is a polyhedron $\implies$
  
  $$conv(X) = \{ x \in \mathbb{R}^n : \tilde{E}x \leq \tilde{d} \}$$

- There are good formulations for the problem

- Except, practically all good formulations are too large

  $$Ax = b \quad Ex \leq d \quad \implies \quad Ax = b \quad \tilde{E}x \leq \tilde{d}$$

- Very few exceptions (integrality property $\approx$ networks)

- Good part: working in the natural variable space

- But a few more variables do as a lot more constraints:
The good news is: rows can be generated incrementally
Row generation/polyhedral approaches

- The good news is: rows can be generated incrementally

 Relevant concept: separator
Row generation/polyhedral approaches

- The good news is: rows can be generated incrementally

\[ Ax = b \]

- Relevant concept: separator
Branch & Cut

- \( \mathcal{R} = \) (small) subset of row indices, \( E_R x \leq d_R \) reduced set

- Solve outer approximation to \( \bar{\Pi} \)
  \[
  \bar{\Pi}_R \quad \text{max} \{ \, c x : A x = b \, , \, E_R x \leq d_R \, \} 
  \]
  feed the separator with primal optimal solution \( x^* \)

- Separator for (several sub-families of) facets of \( \text{conv}(X) \)

- Several general approaches, countless specialized ones

- Most often separators are hard combinatorial problems themselves
  (though using general-purpose MIP codes is an option\(^{[10]}\))

- May tail off, branching useful far before having solved \( \bar{\Pi}_X \)

---

\(^{[10]}\) Fischetti, Lodi, Salvagnin “Just MIP it!” MATHEURISTICS, Ann. Inf. Syst., 2009
Which is best?

- Row generation naturally allows multiple separators
- Very well integrated in general-purpose solvers
  (but harder to exploit “complex” structures)
- Column generation naturally allows very unstructured separators
- Simpler to exploit “complex” structures
  (but much less developed software tools)
- Column generation is row generation in the dual
- Then, of course, Branch & Cut & Price
  (nice, but software issues remain and possibly worsen)
Staircase-structured Integer Programs

- What if $X = \{ x \in \mathbb{Z}^n : E x \leq d \}$ combinatorial?
  
  \[ (\bar{\Pi}) \quad \max \left\{ cx + ez : Ax + Bz \leq b, \; x \in X \right\} \]

- Nothing changes . . . except $(B_B)$ now is combinatorial $\implies$ hard

- However $(B_W)$ now is equivalent to $(\bar{\Pi})$ $\implies$ no branching needed
  (unless for solving $(B_B)$) $\implies$ no Branch & Benders’

- Conversely, everything breaks down if $z \in \mathbb{Z}^m$: there is no (workable, exact) dual of an Integer Program

- Can do with “approximated” duals (strong formulations, RLT$^{[11]}$, . . .) but equivalence lost $\implies$ branching again

---

Example Applications, a.k.a. the Reformulation before the Reformulation
(Very) Classical decomposition approaches for (2SILP)

- Here-and-now decisions are naturally complicating variables
- The (expected) value function decomposes by scenario
  \[ v(x) = c_0 x + \sum_{s \in S} \pi_s \min \left\{ c_s z_s : E_s z_s \leq b_s - E_0^s x \right\} \]
- Alternative approach: split variables, introduce copy constraints
  \[
  \min c_0 x + \sum_{s \in S} \pi_s c_s z_s \\
  E_0 x \leq b_0 \\
  E_0^s x_s + E_s z_s \leq b_s , \quad x_s = x \quad s \in S
  \]
  relax them in a Lagrangian fashion
- Lagrangian approach chooses all variables for all scenarios (no unfeasibility), tries to make here-and-now agree by changing prices
- Difference more pronounced in multi-stage programs
Classical decomposition approaches for (MCND)

- Design ($z$) variables are “naturally” linking / complicating
  - What remains is flow/paths: convex even if integer
  - Benders’ cuts are metric inequalities\(^{[12]}\) defining the multiflow feasibility

- Resource decomposition\(^{[13]}\): add artificial linking variables
  \[
  d^k x^k_{ij} \leq u^k_{ij}, \quad \sum_{k \in K} u^k_{ij} \leq u_{ij}
  \]

- Different possible linking constraints:
  - (3): $\implies$ flow (shortest path) relaxation (integrality property $\equiv$ “easy”)
  - (2): $\implies$ knapsack relaxation (only one integer variable per problem)
  - different efficiency (algorithm-dependent\(^{[14,15]}\)), others possible

---


Conclusions (for now)
Conclusions (part I)

- Structured (Integer) Programs are challenging, but structure can be exploited: main tools are reformulation + duality

- Two different approaches, “primal” and “dual”

- Different twists, different conditions to work:
  - who is complicating (constraints vs. variables), but tricks (≡ other reformulations) can be used to create the desired structure
  - who is reformulated (subproblem vs. master problem)
  - where integer/nonconvexity can be (subproblem vs. master problem)
  - where branching/cutting is done (subproblem vs. master problem)
  - where/which nonlinearities can be easily dealt with

- (For linear programs) Lagrange is Benders’ in the dual, and vice-versa

- Both boil down to the 50+-years old Cutting Plane algorithm\[5\]

- Has it aged well? We’ll see tomorrow