Decomposition and Dynamic Cut Generation in Integer Programming

Ted K. Ralphs
Matthew V. Galati

Department of Industrial and Systems Engineering
Lehigh University, Bethlehem, PA

http://www.lehigh.edu/~tkr2

Institute for Operations Research and Management Science Annual Conference, Atlanta, GA, Monday October 20, 2003
Outline

• Preliminaries, Traditional Decomposition Methods
  – Dantzig-Wolfe Decomposition
  – Lagrangian Relaxation
  – Cutting Plane Method

• Dynamic Decomposition Methods
  – Price and Cut
  – Relax and Cut
  – Decompose and Cut

• Applications/Examples

• DECOMP Framework
Preliminaries

• We consider the following pure integer linear program:

$$z_{IP} = \min_{x \in F} \{c^\top x\} = \min_{x \in P} \{c^\top x\}$$

where

$$F = \{x \in \mathbb{Z}^n : A'x \geq b', A''x \geq b''\} \quad Q = \{x \in \mathbb{R}^n : A'x \geq b', A''x \geq b''\}$$

$$F' = \{x \in \mathbb{Z}^n : A'x \geq b'\} \quad Q' = \{x \in \mathbb{R}^n : A'x \geq b'\}$$

$$Q'' = \{x \in \mathbb{R}^n : A''x \geq b''\}$$

• We will consider $P = \text{conv}(F)$ and $P' = \text{conv}(F')$.

• Assumptions

  – All input data are rational.
  – $P$ is bounded.
  – Optimization/separation over $P$ is “difficult.”
  – Optimization/separation over $P'$ is “easy.”
Decomposition Methods for IP

Polyhedra

\[ P = \text{conv}(\{ x \in \mathbb{Z}^n : Ax \geq b \}) \]

\[ P' = \text{conv}(\{ x \in \mathbb{Z}^n : A'x \geq b' \}) \]

\[ Q = \{ x \in \mathbb{R}^n : A'x \geq b' \} \]

\[ Q' = \{ x \in \mathbb{R}^n : A''x \geq b'' \} \]
Bounding

• **Goal**: Compute a lower bound on $z_{IP}$ by solving a *bounding problem*.

• The most commonly used bounding problem is the initial LP relaxation.

$$\min_{x \in Q} \{ c^T x \}$$

• Decomposition approaches attempt to improve on this bound by utilizing implicit knowledge of $P'$.
  
  – Enforce membership in $Q''$ *explicitly*.
  – Enforce membership in $P'$ *implicitly* through solution of a subproblem.

• Decomposition algorithms
  
  – Dantzig-Wolfe decomposition
  – Lagrangian relaxation
  – Cutting plane method
Dantzig-Wolfe Decomposition (DW)

- The bounding problem is the *Dantzig-Wolfe LP*:

\[
z_{DW} = \min \{ c(\sum_{s \in F'} s \lambda_s) : A''(\sum_{s \in F'} s \lambda_s) \geq b'', \sum_{s \in F'} \lambda_s = 1, \lambda_s \geq 0 \ \forall s \in F' \} \tag{1}
\]

- **Solution method**: Simplex algorithm with dynamic column generation.

- **Subproblem**: Optimization over \( P' \).

- Let \( \hat{\lambda} \) be an optimal solution to (1) (the *optimal decomposition*) and

\[
\hat{x} = \sum_{s \in F'} s \hat{\lambda}_s \in P' \tag{2}
\]

Then, \( z_{IP} \geq z_{DW} = c^\top \hat{x} \geq z_{LP} \).
Lagrangian Relaxation (LD)

• The bounding problem is the *Lagrangian dual*:

\[
z_{LR}(u) = \min_{s \in \mathcal{F}'} \{(c^\top - u^\top A'')s + u^\top b''\}
\] (3)

\[
z_{LD} = \max_{u \in \mathbb{R}_+^m} \{z_{LR}(u)\}
\] (4)

• **Solution method**: Subgradient optimization.

• **Subproblem**: Optimization over \(\mathcal{P}'\).

• Rewriting (4) as a linear program, we see it is dual to the DW LP.

\[
z_{LD} = \max_{\eta \in \mathbb{R}, u \in \mathbb{R}_+^m} \{\eta : \eta \leq (c - uA'')s + ub'' \ \forall s \in \mathcal{F}'\}
\] (5)

• So we have \(z_{IP} \geq z_{LD} = z_{DW} \geq z_{LP}\).

• We denote by \(\hat{u}\) an optimal (dual) solution to (4).
Cutting Plane Method (CP)

- The bounding problem is the initial LP relaxation augmented with facet-defining inequalities from $\mathcal{P}'$:

$$z_{CP} = \min_{x \in \mathcal{P}'} \left\{ cx : A''x \geq b'' \right\} \quad (6)$$

- **Solution method**: Simplex with dynamic cut generation.

- **Subproblem**: Separation from $\mathcal{P}'$.

- We assumed that separation over $\mathcal{P}'$ was also “easy.”

- Note that $\hat{x}$ from (2) is an optimal solution to (6), so $z_{IP} \geq z_{CP} = z_{DW} \geq z_{LP}$. 

A Common Framework

• The three methods compute the same bound [Geoffrion74].

• The basic ingredients are the same:
  – the original polyhedron \( \mathcal{P} \),
  – an implicit polyhedron \( \mathcal{P}' \), and
  – an explicit polyhedron \( \mathcal{Q}'' \).

• The essential difference is how the implicit polyhedron is represented:
  – CP: as the intersection of half-spaces (the outer representation), or
  – DW/LD: as the convex hull of a finite set (the inner representation).
\[ P = \text{conv}\left\{ x \in \mathbb{Z}^n : Ax \geq b \right\} \]

\[ P' = \text{conv}\left\{ x \in \mathbb{Z}^n : A'x \geq b' \right\} \]

\[ Q' = \left\{ x \in \mathbb{R}^n : A'x \geq b' \right\} \]

\[ Q'' = \left\{ x \in \mathbb{R}^n : A''x \geq b'' \right\} \]

\[ Q = Q' \cap Q'' \text{ (LP Bound)} \]

\[ P' \cap Q'' \text{ (LD/DW/CP Bound)} \]
Outline

• Preliminaries, Traditional Decomposition Methods
  – Dantzig-Wolfe Decomposition
  – Lagrangian Relaxation
  – Cutting Plane Method

• Dynamic Decomposition Methods
  – Price and Cut
  – Relax and Cut
  – Decompose and Cut

• Applications/Examples

• DECOMP Framework
Improving the Bound

• With traditional methods, we achieve the bound \( \min_{x \in \mathcal{P}'} \{ cx : A''x \geq b'' \} \).

• With the cutting plane method, this bound can be improved.

**Cutting Plane Method**

1. Construct the initial LP relaxation \( \text{LP}^0 \) and set \( i \leftarrow 0 \).

\[
z_{LP} = \min_{x \in \mathbb{R}^n} \{ c^\top x : A'x \geq b', A''x \geq b'' \}
\]

2. Solve \( \text{LP}^i \) to obtain an optimal solution \( \hat{x}^i \) and valid lower bound \( z^i = c^\top \hat{x}^i \).

3. Attempt to separate \( \hat{x}^i \) from \( \mathcal{P} \), generating a set \([D^i, d^i]\) of valid inequalities violated by \( \hat{x}^i \).

4. If valid inequalities were found in **Step 3**, form the augmented LP relaxation \( \text{LP}^{i+1} \) by setting \([A'', b''] \leftarrow \begin{bmatrix} A'' & b'' \\ D^i & d^i \end{bmatrix} \). Then, set \( i \leftarrow i + 1 \) and go to **Step 2**.

5. If no valid inequalities were found in **Step 3**, then output \( z^i \).
Improving Inequalities

• The challenge is in performing Step 3.

• An inequality found in Step 3 that improves the current bound when added to the bounding problem is an *improving inequality*.

• A **necessary and sufficient condition** for an inequality to be improving is that it is violated by all optimal primal solutions to (6).

• This condition is difficult to verify.

• As a surrogate, we use the **necessary condition** that the generated inequality be violated by $\hat{x}$.

• This process can be thought of as a dynamic tightening of the **explicit polyhedron**.

• In principle, there are analogs of the cutting plane method for Dantzig-Wolfe decomposition and Lagrangian relaxation.

• We call these **dynamic decomposition methods**.
Dynamic Decomposition Methods

**Dynamic Decomposition Method**

1. Construct the initial bounding problem $P^0$ and set $i \leftarrow 0$.

   $z_{CP} = \min_{x \in \mathcal{P}'} \{c^\top x : A'' x \geq b''\}$

   $z_{LD} = \max_{u \in \mathbb{R}_+^n} \min_{x \in \mathcal{P}'} \{(c^\top - u^\top A'')x + u^\top b''\}$

   $z_{DW} = \min_{\lambda \in \mathbb{R}_+^\mathcal{F}'} \{c^\top (\sum_{s \in \mathcal{F}'} s \lambda_s) : A'' (\sum_{s \in \mathcal{F}'} s \lambda_s) \geq b'', \sum_{s \in \mathcal{F}'} \lambda_s = 1\}$

2. Solve $P^i$ to obtain a valid lower bound $z^i$.

3. Try to generate a set of improving inequalities $[D^i, d^i]$ valid for $P$.

4. If valid inequalities were found in Step 3, form the bounding problem $P^{i+1}$ by setting $[A'', b''] \leftarrow \begin{bmatrix} A'' & b'' \\ D^i & d^i \end{bmatrix}$. Then, set $i \leftarrow i + 1$ and go to Step 2.

5. If no valid inequalities were found in Step 3, then output $z^i$. 
Price and Cut (PC)

**Price and Cut:** Use $\text{DW}$ as the bounding problem.

$$
\begin{align*}
    z_{\text{DW}} &= \min_{\lambda \in \mathbb{R}_{+}^{F'}} \left\{ c^{\top} \left( \sum_{s \in F'} s \lambda_s \right) : A'' \left( \sum_{s \in F'} s \lambda_s \right) \geq b'', \sum_{s \in F'} \lambda_s = 1 \right\}
\end{align*}
$$

and attempt to separate $\hat{x} = \sum_{s \in F'} s \hat{\lambda}_s$.

- Cut generation takes place in original space, maintaining the structure of the column generation subproblem.
- Both PC and CP try to separate $\hat{x}$ from $\mathcal{P}$.
- With PC, however, we get additional information, i.e., the optimal decomposition $\hat{\lambda}$.
- **Question:** Can we take advantage of this information?
Relax and Cut (RC)

**Relax and Cut**: Use LD as the bounding problem.

\[ z_{LD} = \max_{u \in \mathbb{R}^n_+} \min_{s \in F'} \{ (c^T - u^T A'') s + u^T b'' \} \]

and attempt to separate \( \hat{s} \in F' \), a solution to \( z_{LR}(\hat{u}) \).

- It is often much easier to separate a member of \( F' \) from \( P \) than an arbitrary real vector, such as \( \hat{x} \).
- However, there is no way to know whether the generated inequalities are improving or are violated by \( \hat{x} \).
- **Questions**:
  - Can we improve our chances of generating an improving inequality?
  - Can we characterize the relationship between \( \hat{s} \) and \( \hat{x} \)?
Improving Inequalities (cont.)

Observation 1. The set of alternative optimal solutions to $z_{LR}(\hat{u})$ is

$$S = \{ s \in \mathcal{F}' : (c^\top - \hat{u}^\top A'') s = (c^\top - u^\top A'') \hat{s} \}.$$ 

Theorem 1. The convex hull of $S$ is a face of $\mathcal{P}'$ and the optimal face $F$ of $\min_{x \in \mathcal{P}'} \{ c^\top x : A'' x \geq b'' \}$ is contained in $\text{conv}(S)$.

Theorem 2. $D = \{ s \in \mathcal{F}' : \hat{\lambda}_s > 0 \} \subseteq S$

Theorem 3. If $(a, \beta) \in \mathbb{R}^{(n+1)}$ is an improving inequality, then there must exist an $s \in D$ such that $a^\top s < \beta$.

• Hence, any improving inequality must be violated by
  – $\hat{x}$,
  – at least one alternative optimal solution to $z_{LR}(\hat{u})$, and
  – at least one $s \in \mathcal{F}'$ such that $\hat{\lambda}_s > 0$. 

Price and Cut (revisited)

• **Idea**: Use the optimal decomposition to help generate improving inequalities.

• Rather than (or in addition to) separating $\hat{x}$, separate each $s \in D$.

• As with RC, it is often much easier to separate a member of $\mathcal{F}'$ from $\mathcal{P}$ than an arbitrary real vector, such as $\hat{x}$.

• RC only gives us one member of $S$ to separate, while PC gives us a set $D \subseteq \mathcal{S}$, one of which must be violated.
Decomposition Methods for IP

(a) (b) (c)

\[ Q' = c^\top - \hat{u}^\top A'' \]

\[ c^\top = c^\top - \hat{u}^\top A'' \]

\[ z_{DW} = z_{LD} = z_{LP} \]

\[ z_{IP} = z_{DW} = z_{LD} > z_{LP} \]

\[ z_{DW} = z_{LD} > z_{LP} \]

\[ \text{conv}(S) = \mathcal{P}' \]

\[ \mathcal{P}' \supset \text{conv}(S) \supset F \]

\[ x_{LP} = x \]

\[ x_{LP} = \hat{x} \]

\[ x_{LP} = \hat{x} \]

\[ s \in \mathcal{F}' : \hat{\lambda}_s > 0 \]
Decompose and Cut (DC)

**Decompose and Cut:** Use CP as the bounding problem.

\[
z_{CP} = \min_{x \in P'} \{ c^\top x : A''x \geq b'' \} \]

Compute the decomposition \( \hat{\lambda} \) of \( \hat{x} \), then separate each \( s \in D \), as in PC.

- Both DC and PC separate the members of a decomposition of \( \hat{x} \).
- DC may be more efficient than PC, since we only need to compute the decomposition when standard separation fails.
Decompose and Cut (details)

- **Separation in Decompose and Cut**

  1. **Attempt to decompose** $\hat{x}$ **into a convex combination of members of** $\mathcal{F}'$ **by solving the LP:**

     $$\max_{\lambda \in \mathbb{R}^{\mathcal{F}'}_+} \{0^\top \lambda : \sum_{s \in \mathcal{F}'} s \lambda_s = \hat{x}, \sum_{s \in \mathcal{F}'} \lambda_s = 1\},$$

     (7)

  2.1 **If** (7) **is feasible, set** $D = \{s \in \mathcal{F}' : \hat{\lambda}_s > 0\}$

  2.2 **Else, return a Farkas Cut** $(a, \beta)$ **valid for** $\mathcal{P}' \subseteq \mathcal{P}$ **which violates** $\hat{x}$.

  3. **Separate each** $s \in D$ **and return any cuts that also violate** $\hat{x}$.

- **Column Generation in Decompose and Cut**

  1.0 **Generate an initial subset** $\mathcal{G}$ **of** $\mathcal{F}'$.

  1.1 **Solve** (7) **over** $\mathcal{G}$ **using the dual simplex algorithm.**

  1.2a **If** (7) **is feasible, return** $D = \{s \in \mathcal{F}' : \hat{\lambda}_s > 0\}$.

  1.2b **Else, optimize over** $\mathcal{P}'$ **using the resulting Farkas inequality (row of** $B^{-1}$). **If the result has negative reduced cost, add it to** $\mathcal{G}$ **and go to Step 1.1, else return the Farkas inequality.
Outline

- Preliminaries, Traditional Decomposition Methods
  - Dantzig-Wolfe Decomposition
  - Lagrangian Relaxation
  - Cutting Plane Method
- Dynamic Decomposition Methods
  - Price and Cut
  - Relax and Cut
  - Decompose and Cut
- Applications/Examples
- DECOMP Framework
Separating the Members of $\mathcal{F}'$

- All three dynamic decomposition methods rely on the existence of a class of valid inequalities for $\mathcal{P}$ for which it is
  - difficult to separate an arbitrary fractional solution, but
  - easy to separate members of $\mathcal{F}'$.

- Does this occur in practice? Yes.
The Vehicle Routing Problem

ILP Formulation:

\[ \sum_{e \in \delta(0)} x_e = 2k \]  \hspace{1cm} (1)
\[ \sum_{e \in \delta(i)} x_e = 2 \quad \forall i \in V \setminus \{0\} \]  \hspace{1cm} (2)
\[ \sum_{e \in \delta(S)} x_e \geq 2b(S) \quad \forall S \subset V \setminus \{0\}, |S| > 1 \]  \hspace{1cm} (3)

\[ b(S) = \text{lower bound on the number of trucks required to service } S \]
\[ = \left\lceil \left( \sum_{i \in S} d_i \right) / C \right\rceil \text{ (normally)} \]

• Relaxations:
  - Multiple Traveling Salesman Problem: Set \( C = \sum_{i \in S} d_i \).
  - k-Tree: Set \( C = \sum_{i \in S} d_i \). Relax (2) but leave \( \sum_{e \in E} x_e = n + k \).

• Facets of VRP (under certain conditions): GSECs (3), Combs, Multistars

• **Decompose and Cut** - VRP/kTSP for GSECs [Ralphs, et al. *On the Capacitated Vehicle Routing Problem*, Mathematical Programming 03]

• **Relax and Cut** - VRP/kTree for GSECs, Combs, Multistars [Martinhon, Lucena, Maculan, *Stronger K-Tree Relaxations for the VRP*, unpublished 01]
\(\hat{\lambda}_1 = \frac{1}{3}\)

\(\hat{\lambda}_2 = \frac{1}{3}\)

\(\hat{\lambda}_3 = \frac{1}{3}\)
Decomposition Methods for IP

(a) $\hat{x}$

(b) $\hat{\lambda}^1 = \frac{1}{2}$

(c) $\hat{\lambda} = \frac{1}{2}$
Axial Assignment Problem

ILP Formulation:

\[
\begin{align*}
\text{min} & \quad \sum_{(i,j,k) \in I \times J \times K} c_{ijk} x_{ijk} \\
\text{s.t.} & \quad \sum_{(j,k) \in J \times K} x_{ijk} = 1 \quad \forall i \in I \\
& \quad \sum_{(i,k) \in I \times K} x_{ijk} = 1 \quad \forall j \in J \\
& \quad \sum_{(i,j) \in I \times J} x_{ijk} = 1 \quad \forall k \in K \\
& \quad x_{ijk} \in \{0, 1\} \quad \forall (i, j, k) \in I \times J \times K
\end{align*}
\]

- Relaxation: Assignment Problem: Relax first set of constraints.
- Facets of AAP: \( Q_1(u) \) and \( P_1(u, v) \) - cliques of the intersection graph \( K_{n,n,n} \)
- Let \( C(u) = \{w \in T : |u \cap w| = 2\} \), \( C(u, v) = \{w \in T : |u \cap w| = 1, |w \cap v| = 2\} \)

\[
\begin{align*}
x_u + \sum_{w \in C(u)} x_w & \leq 1 \quad \forall u \in T \\
x_u + \sum_{w \in C(u,v)} x_w & \leq 1 \quad \forall u, v \in T, u \cap v = \emptyset
\end{align*}
\]

- Relax and Cut - AP3/AP for \( Q_1 \) [Balas and Saltzman, *An Algorithm for the Three-Index Assignment Problem* Operations Research 91]
(a) $\hat{x}$

\[
\begin{align*}
(0, 0, 3) & \quad 1/3 & (0, 3, 1) & \quad 2/3 \\
(1, 0, 1) & \quad 1/3 & (1, 1, 2) & \quad 2/3 \\
(2, 1, 0) & \quad 1/3 & (2, 2, 0) & \quad 1/3 \\
(2, 3, 2) & \quad 1/3 & (3, 0, 0) & \quad 1/3 \\
(3, 2, 3) & \quad 2/3
\end{align*}
\]

(b) $\hat{\lambda}_1 = \frac{1}{3}$

(c) $\hat{\lambda}_2 = \frac{1}{3}$

(d) $\hat{\lambda}_3 = \frac{1}{3}$

\[
\sum_{w \in C(0,0,1)} \hat{x}w = 1 \ 1/3 > 1
\]

(e) $Q_1(0, 0, 1)$

\[
\sum_{w \in C((0,0,3),(1,3,1))} \hat{x}w = 1 \ 1/3 > 1
\]

(f) $P_1((0, 0, 3), (1, 3, 1))$
Applications Under Development

- **Vehicle Routing Problem**
  - k-Traveling Salesman Problem: GSECs
  - k-Tree: GSECs, Combs, Multistars

- **Axial Assignment Problem**
  - Assignment Problem: Clique-Facets

- **Steiner Tree Problem**
  - Minimum Spanning Tree: Lifted SECs, Partition Inequalities

- **Knapsack Constrained Circuit Problem**
  - Knapsack Problem: Cycle Cover, Maximal-Set Inequalities

- **Edge-Weighted Clique Problem**
  - Tree Relaxation: Trees, Cliques

- **Subtour Elimination Problem** [G. Benoit / S. Boyd]
  - Fractional 2-Factor Problem: SECs
Outline

• Preliminaries, Traditional Decomposition Methods
  – Dantzig-Wolfe Decomposition
  – Lagrangian Relaxation
  – Cutting Plane Method

• Dynamic Decomposition Methods
  – Price and Cut
  – Relax and Cut
  – Decompose and Cut

• Applications/Examples

• DECOMP Framework
DECOMP Framework

- **Goal**: Framework to allow for direct comparison of all three dynamic decomposition methods.

- **COIN-OR**: **CO**mputational **IN**frastructure for **O**perations **R**esearch

- **BCP**: Parallel Branch, Price and Cut (LP-based Bounding)

- **ALPs**: Abstract Library for Parallel Search
  - **BiCePS**: Branch, Constrain and Price Software (Generic Bounding)
  - **BLIS**: BiCePS Linear Integer Solver

- **DECOMP** will provide
  - CGL-like full implementation of *Decompose and Cut*
  - **BiCePS** *plug-and-play* for *Price and Cut* and *Relax and Cut*

- **DECOMP** user implements two methods:
  - `solve_relaxed_problem` (includes several built-in solvers)
  - `separate_relaxed_point`
Future Work

- There are many, many computational questions to be answered to make these methods practical.
  - How do the cuts generated by separating members of $F'$ compare to those generated by separating $\hat{x}$?
  - Which members of $F'$ should we separate if we have a choice?
  - Of the cuts that are violated by a given member of $F'$, which are the most likely to be improving?
  - Can we “warmstart” the process of finding a decomposition in DC?
  - Can we use a global pool of members of $F'$ similar to a cut pool?

- One can also envision numerous extensions to this basic framework.
  - Explore uses outside of integer programming.
  - Consider tightening the implicit polyhedron instead of the explicit.
  - Try to generate more Farkas inequalities and possibly tighten them.
  - Restrict column generation when finding the decomposition in DC.

- More theoretical issues also need to be thought about, such as complexity and convergence.
Conclusions

• There are many variants on this basic theme.
• Very little is known about these methods computationally.
• There is a large gap between theory and practice.
• We are currently in the process of implementing these methods and working through the computational issues.
• Our very preliminary results indicate that these methods have potential, but there are many tradeoffs.
• Our software framework will end up in the COIN-OR repository, so that others may use it.
• Stay tuned, a computational paper will be coming soon.