DECOMP: A Framework for Decomposition in Integer Programming

Matthew V. Galati
Ted K. Ralphs

SAS Institute - Analytical Solutions - Operations Research and Development, Cary, NC
Lehigh University - Department of Industrial and Systems Engineering, Bethlehem, PA

http://sagan.ie.lehigh.edu/mgalati
Outline

- Decomposition Methods
  - Cutting Plane Method
  - Dantzig-Wolfe Decomposition
  - Lagrangian Relaxation
- Integrated Decomposition Methods
  - Price and Cut
  - Relax and Cut
- Structured Separation and Motivation
- Decomp and Cut
- DECOMP Framework
Decomposition Methods
Preliminaries

Consider the following integer linear program (ILP):

\[ z_{IP} = \min_{x \in \mathbb{F}} \{ c^T x \} = \min_{x \in \mathbb{P}} \{ c^T x \} = \min_{x \in \mathbb{Z}^n} \{ c^T x : Ax \geq b \} \]

where

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

Denote \( \mathbb{P} = \text{conv}(\mathbb{F}) \) and \( \mathbb{P}' = \text{conv}(\mathbb{F}') \).

\( \text{OPT}(c, X) \): Subroutine returns \( x \in X \) that minimizes \( c^T x \).

\( \text{SEP}(x, X) \): Subroutine returns \( (a, \beta) \) which separates \( x \) from \( X \) (if exists).
Preliminaries

- **Assumption:**
  - $\text{OPT}(c, \mathcal{P})$ and $\text{SEP}(x, \mathcal{P})$ are "hard".
  - $\text{OPT}(c, \mathcal{P}')$ and $\text{SEP}(x, \mathcal{P}')$ are "easy".
  - $\mathcal{Q}''$ can be represented explicitly (description has polynomial size).
  - $\mathcal{P}'$ must be represented implicitly (description has exponential size).

- **Classical Example - Traveling Salesman Problem**

\[
\sum_{e \in \delta(u)} x_e &= 2 \quad \forall u \in V \\
\sum_{e \in \delta(S)} x_e &\geq 2 \quad \forall S \subset V, 2 \leq |S| \leq |V| - 1 \\
x_e \in \{0, 1\} \quad \forall e \in E
\]

- One classical decomposition of TSP is to look for a spanning subgraph with $|V|$ edges ($\mathcal{P}' = 1$-Tree) that satisfies the 2-degree constraints ($\mathcal{Q}''$).
Example - Polyhedra

\[
\begin{align*}
\text{min} & \quad x_1 \\
7x_1 - x_2 & \geq 13 \quad (1) \\
x_2 & \geq 1 \quad (2) \\
-x_1 + x_2 & \geq -3 \quad (3) \\
-x_2 & \geq -5 \quad (4) \\
0.2x_1 - x_2 & \geq -4 \quad (5) \\
-x_1 - x_2 & \geq -8 \quad (6) \\
-0.4x_1 + x_2 & \geq 0.3 \quad (7) \\
x_1 + x_2 & \geq 4.5 \quad (8) \\
3x_1 + x_2 & \geq 9.5 \quad (9) \\
0.25x_1 - x_2 & \geq -3 \quad (10) \\
x & \in \mathbb{Z}^2 \quad (11)
\end{align*}
\]

\[Q' = \{x \in \mathbb{R}^n \mid x \text{ satisfies (1) -- (5)}\}\]

\[Q'' = \{x \in \mathbb{R}^n \mid x \text{ satisfies (6) -- (10)}\}\]

\[\mathcal{P}' = \text{conv}(Q' \cap \mathbb{Z}^n)\]
Bounding

- **Goal:** Compute a lower bound on $z_{IP}$ by building an approximation to $\mathcal{P}$.

- The most straightforward approach is to use the continuous approximation

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

- Decomposition approaches attempt to improve on this bound by utilizing the fact that $OPT(c, \mathcal{P'})$ or $SEP(x, \mathcal{P'})$ is easy.

$$z_{D} = \min_{x \in \mathcal{P'}} \{ c^\top x | A'' x \geq b'' \} = \min_{x \in \mathcal{F'} \cap \mathcal{Q}''} \{ c^\top x \} = \min_{x \in \mathcal{P'} \cap \mathcal{Q}''} \{ c^\top x \} \geq z_{LP}$$

- $\mathcal{P'}$ is represented *implicitly* through solution of a subproblem.

- **Decomposition Methods**
  - Cutting Plane Method (Outer Method)
  - Dantzig-Wolfe Decomposition / Lagrangian Relaxation (Inner Methods)
Example - Polyhedra

\[ c^T \]

\[ \begin{align*}
& \begin{array}{l}
(2,1) \\
\mathcal{P} \\
\mathcal{P}' \\
\mathcal{Q}' \\
\mathcal{Q}''
\end{array} \\
& \begin{array}{l}
(2,1) \\
\mathcal{P} \\
\mathcal{P}' \cap \mathcal{Q}''
\end{array} \\
& \begin{array}{l}
(2,1) \\
\mathcal{P} \\
\mathcal{P}' \cap \mathcal{Q}''
\end{array}
\end{align*} \]

\[ z_{LP} = 2.25 < z_D = 2.42 < z_{IP} = 3.0 \]
Cutting Plane Method

- **Cutting Plane Method** (CPM) gives an approximation of $\mathcal{P}$ by building an outer approximation of $\mathcal{P}'$ intersected with $\mathcal{Q}''$.

- Let $[D, d]$ denote the facets of $\mathcal{P}'$, so that

$$
\mathcal{P}' = \{ x \in \mathbb{R}^n : D x \geq d \}
$$

**Cutting Plane Method**

1. **Initialize**: Form outer approximation with $[D^0, d^0] = [A'', b'']$ and set $t \leftarrow 0$.

   $$
   \mathcal{P}^0_O = \{ x \in \mathbb{R}^n \mid D^0 x \geq d^0 \} \supseteq \mathcal{P}' \cap \mathcal{Q}''
   $$

2. **Master Problem**: Solve an LP to obtain an optimal *primal* solution $x^{t_{CP}}$.

   $$
   z_{CP}^t = \min_{x \in \mathbb{R}^n} \{ c^\top x \mid D^t x \geq d^t \}
   $$

3. **Subproblem**: Call $SEP(x^{t_{CP}}, \mathcal{P}')$ to generate *improving* v.i.s for $\mathcal{P}$, violated by $x^{t_{CP}}$.

4. **Update**: If found, form a new outer approximation, set $t \leftarrow t + 1$ and goto step 2.

   $$
   \mathcal{P}^{t+1}_O = \{ x \in \mathbb{R}^n \mid D^{t+1} x \leq d^{t+1} \} \supseteq \mathcal{P}
   $$

- The method converges to the bound

$$
z_{CP} = c^\top \hat{x}_{CP} = z_D
$$
Cutting Plane Method

\( P = \mathbb{Q} \cap \mathbb{Q]' \)

\( P^0_O = \mathbb{Q}' \cap \mathbb{Q}'' \)

\( x^0_{CP} = (2.25, 2.75) \)

\( P^1_O = P^0_O \cap \{ x \in \mathbb{R}^n \mid 3x_1 - x_2 \geq 5 \} \)

\( x^1_{CP} = (2.42, 2.25) \)
Dantzig-Wolfe Decomposition

- **Dantzig-Wolfe Decomposition** (DW) gives an approximation of $\mathcal{P}$ by building an *inner* description of $\mathcal{P}'$ intersected with $\mathcal{Q}''$.

- Let $\mathcal{E}$ denote the extreme points of $\mathcal{P}'$, so that
  \[ \mathcal{P}' = \{ x \in \mathbb{R}^n \mid x = \sum_{s \in \mathcal{E}} s \lambda_s, \sum_{s \in \mathcal{E}} \lambda_s = 1, \lambda_s \geq 0 \forall s \in \mathcal{E} \}. \]

Dantzig-Wolfe Decomposition

1. **Initialize**: Form inner approximation with $\mathcal{E}^0 \subset \mathcal{E}$ and set $t \leftarrow 0$.
   \[ \mathcal{P}_I^0 = \{ x \in \mathbb{R}^n \mid x = \sum_{s \in \mathcal{E}^0} s \lambda_s, \sum_{s \in \mathcal{E}^0} \lambda_s = 1, \lambda_s \geq 0 \forall s \in \mathcal{E}^0 \} \subseteq \mathcal{P}' \]

2. **Master Problem**: Solve the DW-LP to obtain optimal *dual* solution $(u^t_{DW}, \alpha^t_{DW})$.
   \[ \bar{z}^t_{DW} = \min_{\lambda \in \mathbb{R}^{|\mathcal{E}|}_+} \{ c^\top (\sum_{s \in \mathcal{E}^t} s \lambda_s) \mid A'' (\sum_{s \in \mathcal{E}^t} s \lambda_s) \geq b'', \sum_{s \in \mathcal{E}^t} \lambda_s = 1 \} \]

3. **Subproblem**: Call $OPT(c^\top - (u^t_{DW})^\top A'')$, $\mathcal{P}'$, to generate *improving* e.p.s with reduced cost $rc(s) = (c^\top - (u^t_{DW})^\top A'') s - \alpha^t_{DW} < 0$.

4. **Update**: If found, form a new inner approximation, set $t \leftarrow t + 1$ and goto Step 2.
   \[ \mathcal{P}_I^{t+1} = \{ x \in \mathbb{R}^n \mid x = \sum_{s \in \mathcal{E}^{t+1}} s \lambda_s, \sum_{s \in \mathcal{E}^{t+1}} \lambda_s = 1, \lambda_s \geq 0 \forall s \in \mathcal{E}^{t+1} \} \subseteq \mathcal{P}' \]

- The method converges to the bound
  \[ z_{DW} = c^\top (\sum_{s \in \mathcal{E}} s \hat{\lambda}_s) = c^\top \hat{x}_{DW} = z_D \]
Dantzig-Wolfe Decomposition

\( c^T \)

\( c^T - \hat{u}^T A'' \)

\( P \)

\( P' \)

\( P_0 \) = \text{conv}(E_0)

\( Q'' \)

\( x_{DW}^0 = (4.25, 2) \)

\( \hat{s} = (2, 1) \)

(a)

\( P \)

\( P' \)

\( P_1 \) = \text{conv}(E_1)

\( Q'' \)

\( x_{DW}^1 = (2.64, 1.86) \)

\( \hat{s} = (3, 4) \)

(b)

\( P \)

\( P' \)

\( P_2 \) = \text{conv}(E_2)

\( Q'' \)

\( x_{DW}^2 = (2.42, 2) \)

\( \hat{s} = (2, 1) \)

(c)
Lagrangian Relaxation

- **Lagrangian Relaxation** (LD) formulates a relaxation to the original ILP as finding the minimal extreme point of $\mathcal{P}'$ with respect to a cost which is penalized if the point lies outside of $\mathcal{Q}''$.

- The Lagrangian Dual is a piecewise-linear concave function

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

- Rewriting LD as an LP gives the dual of the DW-LP.

  $$z_{LD} = \max_{\alpha \in \mathbb{R}, u \in \mathbb{R}^{m''}_+} \{ \alpha + b''^\top u \mid \alpha \leq (c^\top - u^\top A'')s \ \forall s \in \mathcal{E} \}.$$ 

- So, $z_{LD} = z_{DW}$ and Lagrangian Relaxation also achieves the bound $z_D$.

**Lagrangian Relaxation**

1. **Initialize**: Define $s^0 \in \mathcal{E}$, initialize dual multipliers $u^0_{LD}$ for $[A'', b'']$ and set $t \leftarrow 0$.
2. **Master Problem**: Update the dual multipliers using directional information from $s^t$.
3. **Subproblem**: Call the subroutine $OPT(c - (u^t_{LD})^\top A'', \mathcal{P}')$, to obtain a new direction $s^{t+1} \in \mathcal{E}$. If the stopping criterion is not met, go to Step 2.
The continuous approximation of $\mathcal{P}$ is formed as the intersection of two explicitly defined polyhedra (both with a small description).

$$z_{LP} = \min_{x \in \mathbb{R}^n} \{c^T x \mid x \in Q' \cap Q''\}$$

Decomposition Methods form an approximation as the intersection of one explicitly defined polyhedron (with a small description) and one implicitly defined polyhedron (with a large description).

$$z_D = \min_{x \in \mathbb{R}^n} \{c^T x \mid x \in \mathcal{P}' \cap Q''\} \geq z_{LP}$$

Each of the traditional decomposition methods contain two primary steps

- **Master Problem**: Update the primal or dual solution information.
- **Subproblem**: Update the approximation of $\mathcal{P}$: $SEP(x, \mathcal{P}')$ or $OPT(c, \mathcal{P}')$.

Integrated Decomposition Methods form an approximation as the intersection of two implicitly defined polyhedra (both with a large description).

So, we improve on the bound $z_D$ by building both an inner approximation $\mathcal{P}_I$ of $\mathcal{P}'$ intersected with some outer approximation $\mathcal{P}_O \subset Q''$. 
Integrated Decomposition Methods
Price and Cut

- Price and Cut (PC) gives an approximation of \( P \) by building an *inner* description of \( P' \) (as in DW) intersected with an *outer* approximation of \( P \).

**Price and Cut**

1. **Initialize**: Form inner approximation with \( E^0 \subset E \), an outer approximation with \([D^0, d^0] = [A'', b'']\) and set \( t \leftarrow 0 \).
   
   \[
   P^0_I = \{ x \in \mathbb{R}^n | x = \sum_{s \in E^0} s \lambda_s, \sum_{s \in E^0} \lambda_s = 1, \lambda_s \geq 0 \ \forall s \in E^0 \} \subseteq P'
   \]
   
   \[
   P^0_O = \{ x \in \mathbb{R}^n | D^0 x \geq d^0 \} \supseteq P
   \]

2. **Master Problem**: Solve the DW-LP to obtain the optimal *dual* solution \((u^t_{PC}, \alpha^t_{PC})\) and the optimal decomposition \( \lambda^t_{PC} \in \mathbb{R}^E \), which yields the optimal *primal* solution \( x^t_{PC} \).
   
   \[
   z^t_{PC} = \min_{\lambda \in \mathbb{R}_+^E} \{ c^T (\sum_{s \in E^t} s \lambda_s) | D^t (\sum_{s \in E^t} s \lambda_s) \geq d^t, \sum_{s \in E^t} \lambda_s = 1 \}
   \]

3. Do either (a) or (b).

   (a) **Pricing Subproblem and Update**: Call \( OPT(c^T - (u^t_{PC})^T D^t, P') \), to generate *improving* e.p.s with \( rc(s) < 0 \). If found, form a new inner approximation \( P^t+1_I \), set \( t \leftarrow t + 1 \) and go to Step 2.

   (b) **Cutting Subproblem and Update**: Call \( SEP(x^t_{PC}, P) \) to generate *improving* v.i.s. If found, form a new outer approximation \( P^t+1_O \), set \( t \leftarrow t + 1 \) and go to Step 2.
Relax and Cut (RC) improves on the bound $z_D$ using LD and augmenting the multiplier space with valid inequalities that are violated by the solution to the Lagrangian subproblem.

Relax and Cut
1. **Initialize**: Define $s^0 \in \mathcal{E}$, $[D^0, d^0] = [A'', b'']$, initialize dual multipliers $u^0_{LD}$ for $[D^0, d^0]$ and set $t \leftarrow 0$.
2. **Master Problem**: Update the dual multipliers using directional information from $s^t$.
3. Do either (a) or (b).
   (a) **Pricing Subproblem**: Call the subroutine $OPT(c - (u^t_{LD})^T D^t, P')$, to obtain a new direction $s^{t+1} \in \mathcal{E}$. If the stopping criterion is not met, go to Step 2.
   (b) **Cutting Subproblem**: Call the subroutine $SEP(s^t, \mathcal{P})$ to generate improving v.i.s. found, add them to $[D^t, d^t]$ along with new dual multipliers, and go to Step 2.
Structured Separation

In general, the complexity of $OPT(c, X) = SEP(x, X)$.

**Observation:** Restrictions on the input or output of these subroutines can change their complexity.

**Template Paradigm**, restricts the output of $SEP(x, X)$ to valid inequalities $(a, \beta)$ that conform to a certain structure. This class of inequalities forms a polyhedron $C \supset X$.

For example, let $\mathcal{P}$ be the convex hull of solutions to the TSP.

- $SEP(x, \mathcal{P})$ is $NP$-Complete.
- $SEP(x, C)$ is polynomially solvable, for $C \supset \mathcal{P}$ the Subtour Polytope (Min-Cut) or Blossom Polytope (Padberg-Rao).

**Structured Separation**, restricts the input of $SEP(x, X)$, such that $x$ conforms to some structure. For example, if $x$ is restricted to solutions to a combinatorial problem, then separation often becomes much easier.
Example - TSP

- **Traveling Salesman Problem Formulation:**

  \[
  x(\delta(u)) = 2 \quad \forall u \in V \\
  x(\delta(S)) \geq 2 \quad \forall S \subset V, 2 \leq |S| \leq |V| - 1 \\
  x_e \in \{0, 1\} \quad \forall e \in E
  \]

- \(P' = 1\)-Tree Relaxation: \(OPT(c, 1 - Tree)\) in \(O(m \log m)\)

  \[
  x(E) = |V| \\
  x(\delta(S)) \geq 1 \quad \forall S \subset V, 2 \leq |S| \leq |V| - 1 \\
  x_e \in \{0, 1\} \quad \forall e \in E
  \]

- \(P' = 2\)-Matching Relaxation: \(OPT(c, 2 - Match)\) in polynomial time

  \[
  x(\delta(u)) = 2 \quad \forall u \in V \\
  x_e \in \{0, 1\} \quad \forall e \in E
  \]
Example - TSP

- Separation of Subtour Inequalities:
  \[ x(\delta(S)) \geq 2 \]

- \textsc{Sep}(x, \text{Subtour})\), for \(x \in \mathbb{R}^n\) can be solved in \(O(|V|^4)\) (Min-Cut)

- \textsc{Sep}(s, \text{Subtour})\), for \(s\) a 2-matching, can be solved in \(O(|V|)\)
  - Simply determine the connected components \(C_i\), and set \(S = C_i\) for each component (each gives a violation of 2).
Example - TSP

- Separation of Blossom Inequalities:

\[ x(E(H)) + \sum_{i=1}^{k} x(E(T_i)) \leq |H| + \sum_{i=1}^{k} (|T_i| - 1) - \lfloor k/2 \rfloor \]

- \( SEP(x, Blossoms) \), for \( x \in \mathbb{R}^n \) can be solved in \( O(|V|^5) \) (Padberg-Rao)

- \( SEP(s, Blossoms) \), for \( s \) a 1-Tree, can be solved in \( O(|V|) \)

  - Simply determine the cycle \( C \), and set \( H = C \) and \( T_i \) to be chains originating at nodes in \( C \) (gives a violation of \( \lfloor k/2 \rfloor \)).
Motivation

- In Relax and Cut, the solutions to the Lagrangian subproblem \( s \in \mathcal{E} \) typically have some *nice* combinatorial structure. So, the cutting step in Relax and Cut \( SEP(s, \mathcal{P}) \), can be relatively easy as opposed to general separation.

- **Question:** Can we take advantage of this in other contexts?

- LP theory tells us that in order to improve the bound, it is *necessary and sufficient* to cut off the entire face of optimal solutions \( \mathcal{F} \).

- This condition is difficult to verify, so we typically use the *necessary condition* that the generated inequality be violated by some member of that face, \( x \in \mathcal{F} \).
  
  - In the Cutting Plane Method, we search for inequalities that violate \( x_{CP}^t \in \mathcal{F}^t \), where \( \mathcal{F}^t \) is optimal face over \( \mathcal{P}_O^t \cap \mathcal{Q}'' \).
  
  - In the Price and Cut Method, we search for inequalities that violate \( x_{PC}^t \in \mathcal{F}^t \), where \( \mathcal{F}^t \) is optimal face over \( \mathcal{P}_I^t \cap \mathcal{P}_O^t \).
Motivation

- Now, consider the following set

\[ S(u, \alpha) = \{ s \in E \mid (c^T - u^T A'')s = \alpha \}, \]

- Then, \( S(u_{DW}^t, \alpha_{DW}^t) \) is the set e.p.s with \( rc(s) = 0 \) in the DW-LP master or the set of alternative optimal solutions to the Lagrangian subproblem.

**Theorem 1** \( F^t \subseteq conv(S(u_{DW}^t, \alpha_{DW}^t)) \)

- Therefore, separation of \( S(u_{DW}^t, \alpha_{DW}^t) \) gives an alternative necessary and sufficient condition for an inequality to be improving.

- By convexity, it is clear that every improving inequality must violate at least one extreme point in the optimal decomposition.

**Theorem 2** If \( (a, \beta) \in \mathbb{R}^{(n+1)} \) is an improving then there must exist an \( s \in D = \{ s \in E \mid \lambda_s^t > 0 \} \) such that \( a^T s < \beta \)

**Theorem 3** \( D = \{ s \in E \mid \lambda_s^t > 0 \} \subseteq S(u_{PC}^t, \alpha_{PC}^t) \)

- Theorems 1-3, along with the observation that structured separation can be relatively easy, motivates the following revised PC method.
Theorems 1-3 give us an alternative *necessary condition* for finding improving inequalities. PC gives us the optimal decomposition $D = \{ s \in \mathcal{E} \mid \lambda_s > 0 \}$.

**Key Idea:** In the cutting subproblem, rather than (or in addition to) separating $x^t_{PC}$, separate each $s \in D$.

The violated subtour found by separating the 2-Matching *also* violates the fractional point, but was found at little cost.
Decomp and Cut

- In the context of the traditional CPM, we can construct (inverse DW) the decomposition $\lambda$ from the current fractional solution $x_{CP}$ by solving the following LP

$$\max \{0^T \lambda : \sum_{s \in \mathcal{E}} s\lambda_s = x_{CP}, \sum_{s \in \mathcal{E}} \lambda_s = 1\}$$

- If we find a decomposition $\mathcal{D}$, then we separate each $s \in \mathcal{D}$, as in revised PC.
- If we fail, then the LP proof of infeasibility (Farkas Cut) gives us a separating hyperplane which can be used to cut off the current fractional point.
DECOMP Framework
DECOMP Framework

- **DECOMP** provides a flexible software framework for testing and extending the theoretical framework presented thus far, with the primary goal of *minimal user responsibility*.

- DECOMP was built around data structures and interfaces provided by COIN-OR: **COmputational INfrastructure for Operations Research**.

- **BCP** provides a framework for parallel implementation of PC in a branch and bound framework with *LP-Based Bounding*.

- A generalization of BCP currently under development:
  - **ALPs**: Abstract Library for Parallel Search
  - **BiCePS**: Branch, Constrain and Price [*Generic Bounding*]
  - **BLIS**: BiCePS Linear Integer Solver = BCP

- DECOMP could provide an implementation of the **BiCePS** layer.
DECOMP Framework

- The framework, written in C++, is accessed through two user interfaces:
  - **Applications Interface**: `DecompApp`
  - **Algorithms Interface**: `DecompAlgo`

- One important feature of DECOMP is that the user only needs to provide methods for their application in the original space ($x$-space), rather than in the space of a particular reformulation.

- This allows for users to consider cuts and variables in their most *intuitive* form and greatly simplifies the process of expansion into rows and columns.

- Features:
  - **Automatic reformulation** - row and column expansion in DW master, dualization and multiplier updates in RC, etc...
  - One interface to all default algorithms: CPM/DC, DW, LD, PC, RC.
  - Built on top of the COIN/OSI interface, so easily interchange LP solvers.
  - Active LP compression, variable and cut pool management.
  - Easily switch between relaxations (choice of $\mathcal{P}'$).
Applications Interface

- In order to develop an application, the user must derive the following methods/objects. All other methods have appropriate defaults but are \textbf{virtual} and may be overridden.

  - \texttt{DecompApp::createCore()}. Define $[A'', b'']$.
  - \texttt{DecompVar}. Define a variable $s \in \mathcal{F}'$ in terms of $x$-space.
  - \texttt{DecompCut}. Define a cut $(a, \beta)$ in terms of $x$-space.
  - \texttt{DecompApp::solveRelaxedProblem()}. Provide a subroutine for $\text{OPT}(c, \mathcal{P}')$, given a cost vector $c$, that returns a set of solutions as \texttt{DecompVar} objects $\in \mathcal{F}'$.
  - \texttt{DecompApp::generateCuts(s)}. Provide a subroutine $\text{SEP}(s, \mathcal{P})$, given a \texttt{DecompVar} $\in \mathcal{F}'$, that returns a set of \texttt{DecompCut} objects.

- If the user wishes to do traditional CPM or PC, they must also provide

  - \texttt{DecompApp::generateCuts(x)}. Provide a subroutine $\text{SEP}(x, \mathcal{P})$, given a arbitrary real vector, that returns a set of \texttt{DecompCut} objects.
Applications Interface

- By default, `DecompVar` is a virtual object defined as a sparse vector of index/value assignments in $x$-space.
  - For some applications, it is possible to more *compactly* represent a variable (many combinatorial problems). In this case, the user can derive `APPDecompVar`, which defines the assignment in $x$-space.

- By default, `DecompCut` is a virtual object defined as a sparse vector if index/value assignments in $x$-space, and a right-hand side, $a^T x \geq \beta$.
  - For template cuts, it is often possible to more *compactly* represent a cut. In this case, the user can derive `APPDecompCut`, which defines the expansion of a cut in $x$-space.
The base class `DecompAlgo` provides the shell (master / subproblem) for integrated decomposition methods.

Each of the methods described have derived default implementations `DecompAlgoX : public DecompAlgo`.

New, hybrid or extended methods can be easily derived by overriding the various subroutines which are called from the base class. For example,

- Alternative methods for solving the master LP in DW, such as **interior point methods** or ACCPM.
- The user might choose to add a stabilizing factor to the dual updates in LD, as in **bundle methods**.
- The user might choose the **Volume algorithm** for solving the LD, which provides an approximation primal solution for which cuts can be generated.
Applications Under Development

- **Steiner Tree Problem**
  - Minimum Spanning Tree: Lifted SECs, Partition - RC* [Lucena 92]

- **Traveling Salesman Problem**
  - One-Tree: Blossoms, Combs
  - Matching: SECs

- **Vehicle Routing Problem**
  - k-Traveling Salesman Problem: GSECs - DC [Ralphs, et al. 03]
  - k-Tree: GSECs, Combs, Multistars - RC* [Marthinhon, et al. 01]

- **Axial Assignment Problem**
  - Assignment Problem: Clique-Facets - RC [Balas, Saltzman 91]

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

- **Edge-Weighted Clique Problem**
  - Tree Relaxation: Trees, Cliques - RC [Hunting, et al. 01]

- **Subtour Elimination Problem [G. Benoit / S. Boyd]**
  - Fractional 2-Factor Problem: SECs - DC / LP Context [Benoit, Boyd 03]
Summary

- Decomposition Methods approximate $P$ as $P' \cap Q''$, where $P'$ may have a \textit{large} description.
- Integrated Decomposition Methods optimize over $P_I \cap P_O$, where $P_I \subset P'$ and $P_O \supset P$. Both polyhedra may have a \textit{large} description.
- Structured separation can be much easier than general separation.
- We gave some motivation for two new techniques: \textit{revised-PC} and \textit{DC}.
  - The question remains: Empirically, how \textit{good} are the cuts generated by separation of $s \in D$?
  - However, for some facet classes, it doesn’t matter - \textit{we simply don’t know how to separate} $x \in \mathbb{R}^n$. These ideas provide a starting point.
- \textbf{DECOMP} provides an easy-to-use framework for comparing and developing various decomposition-based methods.
- The code is open-source, currently released under CPL and will eventually be available through the \textit{COIN-OR} project repository \url{www.coin-or.org}. 

Matthew Galati, December 23, 2004