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

Columbia University, Thursday June 26, 2003

Outline

- Traditional decomposition methods
  - Lagrangian Relaxation
  - Dantzig-Wolfe Decomposition
  - Cutting Plane Algorithms
- A common framework
- Decomposition methods and dynamic cut generation
- Conclusions

Preliminaries

- We consider the following pure integer linear program:

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

where

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

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

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

- We are interested in $\mathcal{P} = conv(\mathcal{F})$ and $\mathcal{P}' = conv(\mathcal{F}')$.
- Assumptions
  - All input data are rational.
  - $\mathcal{P}$ is bounded.
  - Optimization/separation over $\mathcal{P}$ is "difficult."
  - Optimization/separation over $\mathcal{P}'$ is "easy."

Bounding

- Goal: Compute a lower bound on z_{IP}.
- The most straightforward approach is to solve the initial LP relaxation.

$$\min_{x \in \mathcal{Q}} \{c^\top x\}$$

- Decomposition approaches attempt to improve on this bound by utlizing our implicit knowledge of $\mathcal{P}'$.
- Decomposition algorithms
  - Dantzig-Wolfe decomposition
  - Lagrangian relaxation
  - Cutting plane method

Dantzig-Wolfe Decomposition

- Express the constraints of $\mathcal{Q}''$ *explicitly*.
- Express the constraints of $\mathcal{P}'$ *implicitly* through solution of a subproblem.
- The bound is obtained by solving the *Dantzig-Wolfe LP*:

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

- Solution method: Simplex algorithm with dynamic column generation.
- Subproblem: Optimization over $\mathcal{P}'$.
- Suppose $\hat{\lambda}$ is an optimal solution to (1).
- Then $z_{IP} \geq z_{DW} = c^\top \hat{x} \geq z_{LP}$, where

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

Lagrangian Relaxation

- Express the constraints of $\mathcal{Q}''$ *explicitly*.
- Express the constraints of $\mathcal{P}'$ *implicitly* through solution of a subproblem.
- The bound is obtained by solving the *Lagrangian dual*.

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

$$z_{LD} = \max_{u \in \mathbb{R}^{m''}_+} \{z_{LR}(u)\} \tag{4}$$

- Solution method: Subgradient optimization.
- Subproblem: Optimization over $\mathcal{P}'$.

Lagrangian Relaxation (cont.)

- Rewriting (4) as a linear program, we see it is dual to the Dantzig-Wolfe LP.

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

- So we have $z_{IP} \geq z_{LD} = z_{DW} \geq z_{LP}$.
- The vector $\hat{u} \in \mathbb{R}^{m''}_{+}$ such that $z_{LD} = z_{LR}(\hat{u})$ will be called a set of *optimal (dual) multipliers*.
- Note that these correspond exactly to the dual variables of the Dantzig-Wolfe LP.
- The member of $\mathcal{F}'$ that yields the optimal bound will be called the *optimal primal solution* to the Lagrangian dual.

Cutting Plane Method

- Express the constraints of $\mathcal{Q}''$ *explicitly*.
- Express the constraints of $\mathcal{P}'$ *implicitly* through solution of a subproblem.
- The bound is obtained by solving the initial LP relaxation augmented by facets of $\mathcal{P}'$.
- Solution method: Simplex with dynamic cut generation.
- Subproblem: Separation from $\mathcal{P}'$.
- We assumed that separation over $\mathcal{P}'$ was also "easy."
- This approach yields the bound

$$z_{CP} = \min_{x \in \mathcal{P}'} \{cx : A''x \geq b''\} \tag{6}$$

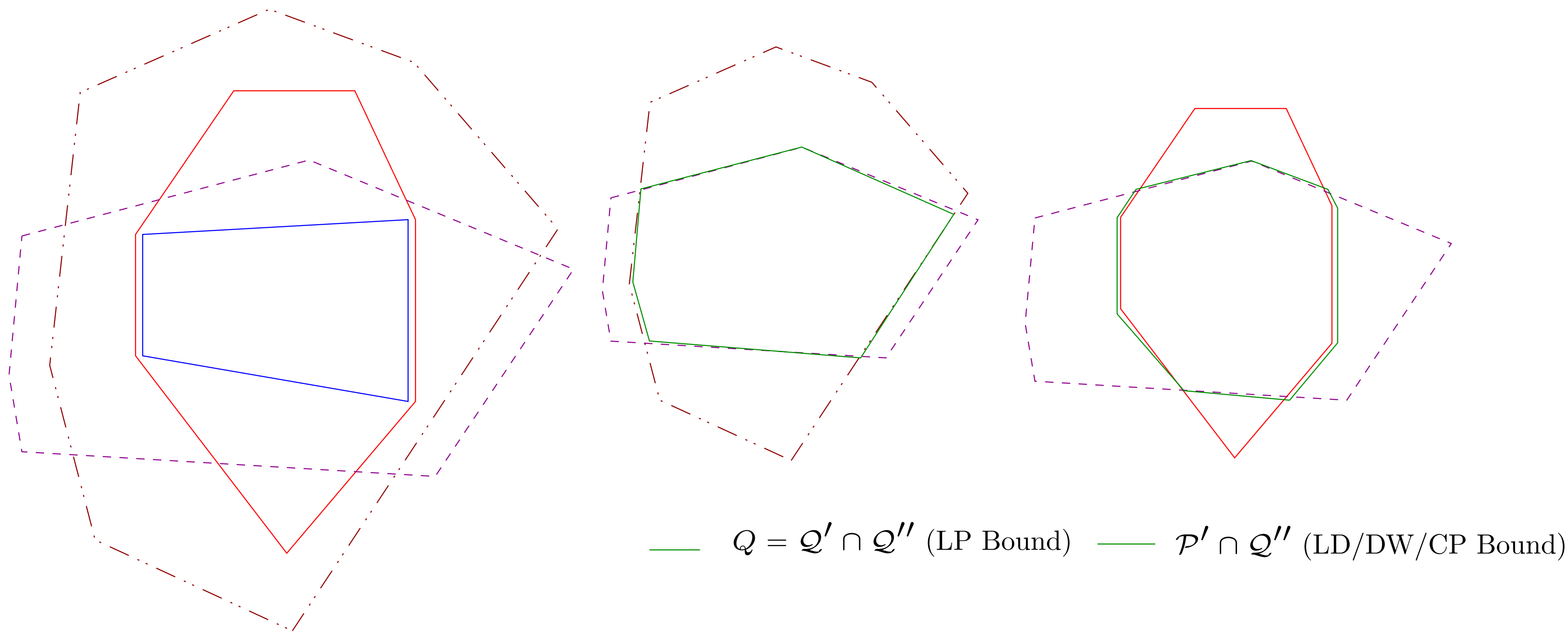
.

- Note that $\hat{x}$ from (2) is an optimal solution to (6), so we obtain the same bound again.

A Common Framework

- The three methods discussed are just three algorithms for computing the same quantity.
- 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 between the cutting plane method and the other two is how the implicit polyhedron is represented:
  - as the interesection of half-spaces (the *outer representation*), or
  - as the convex hull of a finite set (the *inner representation*).
- z_{DW} and z_{LR} are computed using inner representations of $\mathcal{P}'$, whereas z_{CP} uses an outer representation.

Polyhedra, LP Bound, LD/DW/CP Bound



$\mathcal{P} = conv(\{x \in \mathbb{Z}^n : Ax \geq b\})$

$\mathcal{P}' = conv(\{x \in \mathbb{Z}^n : A'x \geq b'\})$

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

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

Improving the Bound

- So far, we have the bound $\min_{x \in \mathcal{P}'} \{cx : A''x \geq b''\}$.
- Can we improve on this?
- With the cutting plane method, the bound can be improved using heuristic cut generation.
- We can think of this as a dynamic tightening of the explicit polyhedron.
- Generic cutting plane method

  1. Construct the initial LP relaxation.
  2. Solve the current LP relaxation to obtain the solution $\hat{x}$.
  3. Try to separate $\hat{x}$ from $\mathcal{P}$, generating a set $\mathcal{L}$ of valid inequalities violated by $\hat{x}$.
  4. If $\mathcal{L} \neq \emptyset$, add the set $\mathcal{L}$ of valid inequalities to the LP relaxation and go to Step 2.
  5. If $\mathcal{L} = \emptyset$, then output $c\hat{x}$.

- Step 3 may generate facets of any number of polyhedra containing $\mathcal{P}$, including $\mathcal{P}$ itself.
- In principle, there are analogs of this for Dantzig-Wolfe decomposition and Lagrangian relaxation.

Dynamic Decomposition Methods

- Generic dynamic bounding method

1. Formulate an initial bounding problem using one of the three decomposition approaches by determining appropriate explicit and implicit polyhedra.
2. Solve the current bounding problem to obtain a valid lower bound.
3. Try to generate an inequality valid for $\mathcal{P}$ that increases the bound when added to the description of the explicit polyhedron.
4. If successful in Step 3, add the generated valid inequality to the description of the explicit polyhedron and formulate a new dual problem.
5. If unsuccessful in Step 3, then output the current lower bound.

- The difficulty is in performing Step 3.

Improving Inequalities

- We will call an inequality that improves the bound in Step 3, an *improving inequality*.
- An inequality is improving if and only if it is violated by all optimal primal solutions to (6).
- With the cutting plane method, we generate inequalities violated by *one* of the optimal primal solutions to (6) and hope for the best.
- We can do the same for Dantzig-Wolfe decomposition by computing $\hat{x}$ from (2), resulting in an implementation of *branch, cut, and price*.
- When employing the Lagrangian dual with subgradient optimization, we don't have access to primal solution information from (6).
- We can, however, get limited information by examining the optimal primal solution to the Lagrangian dual problem.

Dynamic Cut Generation in Lagrangian Relaxation

- An optimal primal solution to the Lagrangian dual is a member of $\mathcal{F}'$.
- If it is a member of $\mathcal{F}$, then it is optimal.
- Otherwise, we attempt to separate it from $\mathcal{P}'$ and add the newly generated inequality to the matrix $[A'', b'']$.
- When employed in a branch and bound framework, this technique is known as *relax and cut*.
- One advantage is that it is sometimes much easier to separate a member of $\mathcal{F}'$ from $\mathcal{P}$ than an arbitrary real vector.
- What are the chances of generating an improving inequality?
- What is the relationship of an optimal primal solution to the Lagrangian dual to an optimal solution of (6)?

Relating the Methods

- The set of alternative optimal primal solutions to the Lagrangian dual is $\mathcal{F}' \cap S$, where S is the face of $\mathcal{P}'$ defined as

$$\mathcal{S} = \{x \in \mathcal{P}' : (c^\top - \hat{u}^\top A'')x = (c^\top - \hat{u}^\top A'')\hat{s}\} \quad (7)$$

and $\hat{s}$ is any optimal primal solution to the Lagrangian dual.

- The face $\mathcal{S}$ contains $\hat{x}$, as defined in (2) and is a proper face if and only if $\hat{x}$ is not an inner point of $\mathcal{P}'$.
- If $\hat{x}$ is an inner point of $\mathcal{P}'$, then the convexity constraint in the Dantzig-Wolfe LP has a zero dual value and all members of $\mathcal{F}'$ are optimal for the Lagrangian dual.
- If $\hat{\lambda}$ is an optimal solution to the Dantzig-Wolfe LP, any $s \in \mathcal{F}'$ such that $\hat{\lambda}_s > 0$ is an optimal primal solution for the Lagrangian dual.

Some Consequences

- Any improving inequality must be violated by *some* optimal solution to the Lagrangian dual.
- In fact, any improving inequality must be violated by some optimal solution s to the Lagrangian dual such that $\hat{\lambda}_s > 0$.
- Hence, if we have access to the optimal decomposition, we will be much more likely to be able to generate an improving inequality.
- This leads us back to the idea of using Dantzig-Wolfe decomposition with dynamic cut generation.
- What if instead of separating the fractional solution $\hat{x}$, we separate each $s \in \mathcal{F}'$ such that $\hat{\lambda}_s > 0$?
  - Any improving inequality must be violated by some $s \in \mathcal{F}'$ such that $\hat{\lambda}_s > 0$.
  - It might be easier to separate members of $\mathcal{F}'$ than $\hat{x}$.

Separating the Members of $\mathcal{F}'$

- Consider a class of valid inequalities for $\mathcal{P}$ with which it is
  - difficult to separate an arbitrary fractional solution, but
  - easy to separate members of $\mathcal{F}'$.
- Does this occur in practice? Yes.
- How can we take advantage of this situation?

The Vehicle Routing Problem

The VRP is a combinatorial problem whose *ground set* is the edges of a graph $G(V, E)$. Notation:

- V is the set of customers and the depot (0).
- d is a vector of the customer demands.
- k is the number of routes.
- C is the capacity of a truck.

A feasible solution is composed of:

- a partition $\{R_1, \ldots, R_k\}$ of V such that $\sum_{j \in R_i} d_j \leq C,\ 1 \leq i \leq k$;
- a permutation σ_i of $R_i \cup \{0\}$ specifying the order of the customers on route i.

Classical Formulation for the VRP

IP Formulation:

$$\begin{array}{rcll} \sum_{j=1}^{n} x_{0j} & = & 2k & \\ \sum_{j=1}^{n} x_{ij} & = & 2 & \forall i \in V \setminus \{0\} \\ \sum_{\substack{i \in S \\ j \notin S}} x_{ij} & \geq & 2b(S) & \forall S \subset V \setminus \{0\},\ |S| > 1. \end{array}$$

$b(S)$ = lower bound on the number of trucks required to service S (normally $\lceil (\sum_{i \in S} d_i)/C \rceil$).

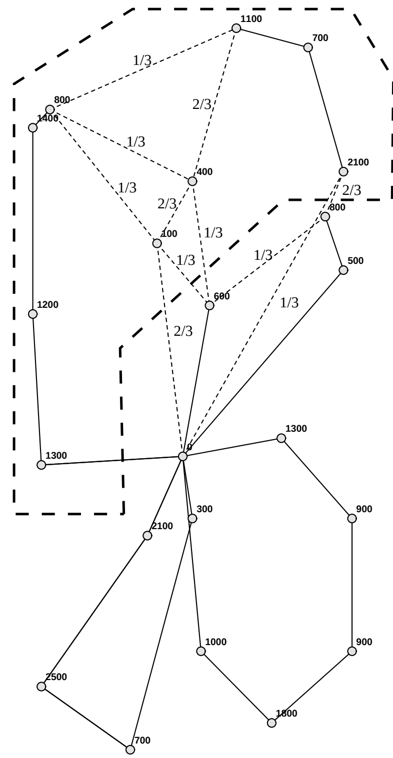
- If $C = \sum_{i \in S} d_i$, then we have an instance of the Multiple Traveling Salesman Problem.
- Separation for the inequalities in this formulation is NP-hard.
- Given the incidence vector of an MTSP, we can easily determine whether it satisfies all of these inequalities.

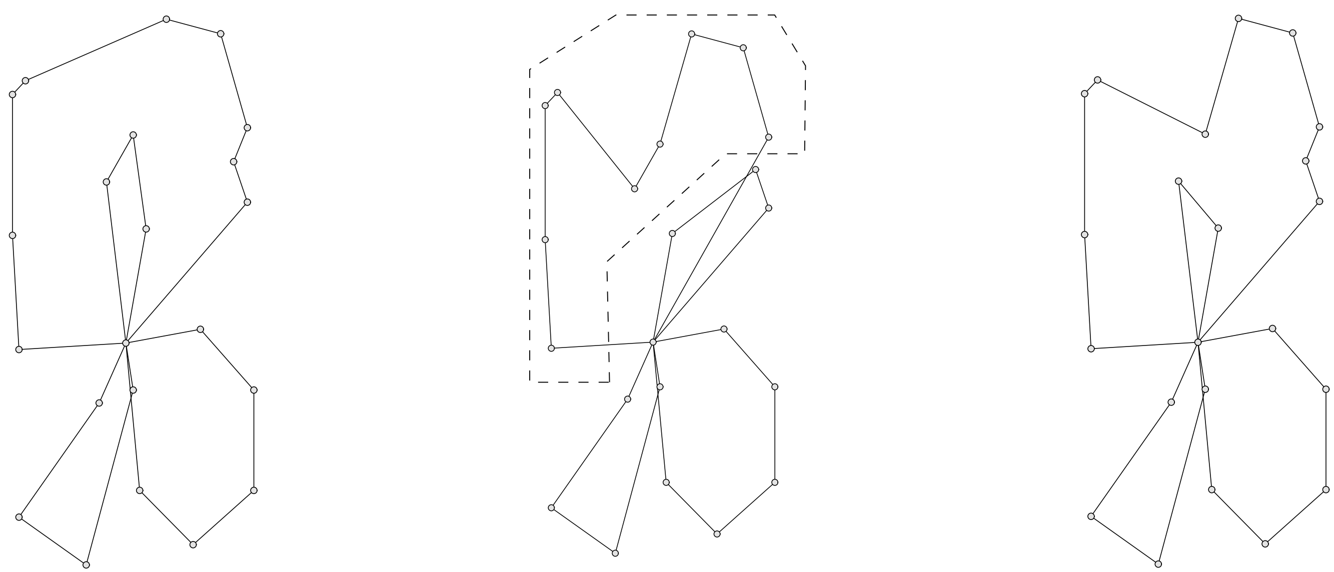
Using Decomposition to Separate

- Instead of computing a Dantzig-Wolfe decomposition at each iteration, only compute it *when needed*.

  1. Use a standard LP-based branch and cut framework.
  2. Try to separate each fractional point using standard procedures.
  3. If the usual procedures fail, try to decompose the current fractional point into a convex combination of members of $\mathcal{F}'$.
  4. If successful, separate over members of the decomposition.

- This allows us to take advantage of our ability to separate members of $\mathcal{F}'$ from $\mathcal{P}$, but can be more efficient and easier to implement than D-W.


1100
700
1/3
2/3
800
1400
1/3
400
2100
1/3
2/3
2/3
800
100
1/3
1/3
1/3
500
600
1/3
1200
2/3
1300
0
1300
300
900
2100
1000
900
2500
1800
700



A Decomposition-based Separation Algorithm

- We now state the basic algorithm more formally and review some of the implementational details.

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

$$\min\{\mathbf{0}^T\lambda \mid \sum_{s\in\mathcal{F}'} \lambda_s s = \hat{x}, \sum_{s\in\mathcal{F}'} \lambda_s = 1, \lambda_s \geq 0\}. \tag{8}$$

  2. If the LP has a feasible solution $\hat{\lambda}$, separate $s \in \mathcal{F}'$ such that $\hat{\lambda}_s > 0$ from $\mathcal{P}$ to obtain a list of inequalities potentially violated by $\hat{x}$.
  3. Return any generated inequalities that are violated by $\hat{x}$.

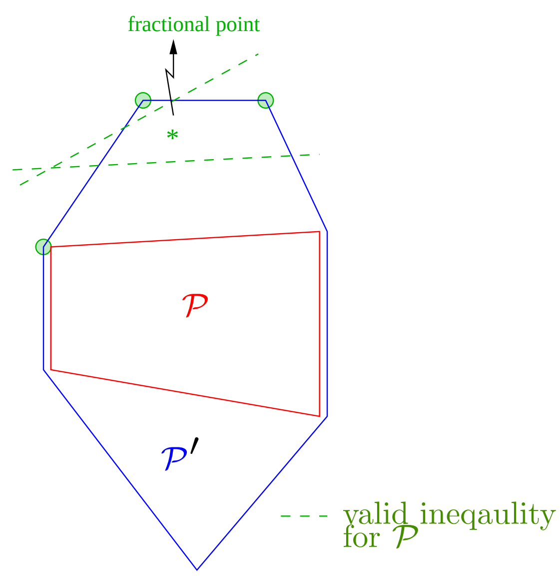
- Note that if the fractional solution is inside $\mathcal{P}'$, then the LP in Step 1 always has a feasible solution.

- How do we solve the LP in Step 1?

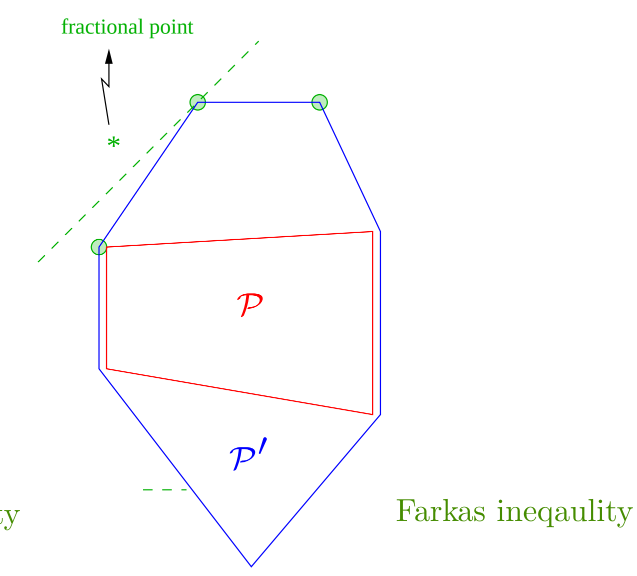
Solving the Decomposition LP

- We can solve this LP by column generation, just as we do with D-W.
- The column generation subproblem is an instance of IP'.

1. Start with a small subset of columns (members of $\mathcal{F}'$).
2. If decomposition fails, optimize over $\mathcal{P}'$ using resulting Farkas inequality (row of B^{-1}).
3. Add new column (member of $\mathcal{F}'$), repeat.



(a) $\hat{x} \in \mathcal{P}'$



(b) $\hat{x} \notin \mathcal{P}'$

Applications Under Development

- Vehicle Routing Problem
  - k-Traveling Salesman Problem : GSECs
  - k-Tree : GSECs, Combs, Multistars
- Steiner Problem in Graphs
  - Minimum Spanning Tree : Lifted SECs
- Knapsack Constrained Circuit Problem
  - Knapsack Problem : Maximal-Set Inequalities
- Edge-Weighted Clique Problem
  - Tree Relaxation : Trees, Cliques
- Axial Assignment Problem
  - Assignment Problem : Q-Facets
- Traveling Salesman Problem [Labonte/Boyd]
  - Fractional 2-Factor Problem : SECs

Conclusions

- Just as there are many ways of computing the bound $z_{DW} = z_{LP} = z_{CP}$, there are many ways of incorporating dynamic cut generation into decomposition.
- Very little is known computationally about many of the variants.
- We are currently in the process of developing a software framework to implement some of these methods.
- This framework will end up in COIN.