Facility Location-Routing-Scheduling Problem: LRS

Zeliha Akca
Lehigh University
zea2@lehigh.edu

Rosemary Berger
Lehigh University
rtb3@lehigh.edu

What We Have Done
What We Will Do

1 Definition of LRS
2 Branch and Price Algorithm
3 Mater and Pricing Problem
4 Solution of Pricing Problem
5 What We Have Done
6 What We Will Do
LRS

Location Routing and Scheduling Problem:

3 dependent problems:
① locate facilities
② construct routes for vehicles
③ assign routes to vehicles

=> capacitated facilities
=> capacitated vehicles
=> time restriction for the vehicles
Location-Routing and Scheduling Problem:

→ In literature, heuristic solution for LRS problem (no IP formulation)
→ Exact solutions for RS and LR
→ We choose, Branch and Price Algorithm:
  ● IP formulation includes many constraints (s.t. sub tour elimination constraints)
  ● Can be written in set partitioning problem easily
  ● Easy to think routes in terms of columns
  ● With set partitioning formulation, many possible columns
→ Other methods to solve:
  ● Lagrangian Relaxation
  ● Branch and bound and cut
  ● Heuristic design
  ● ?
Problems in Literature

- Facility Location - too many
- Vehicle Routing - too many
- Routing Scheduling
- Location Routing
- Location Routing and Scheduling

⇒ Location routing: one-to-one relation btw routes and vehicles.
⇒ Location scheduling: not necessarily one-to-one relation assignment of one vehicle to many paths.
IP FORMULATION

Objective: Minimize total cost.

\[ \text{TotalCost} = \text{Fixed cost of Facility and Vehicle} + \]
\[ \text{Operating cost of Vehicles} \]

Constraints:

1. Each demand node should be served once
2. # of a vehicle entering a node must be equal to # of the vehicle leaves this node
3. Capacity restriction for facility
4. Capacity restriction for vehicles
5. Flow balance equations (to satisfy demand and eliminate the subtours)
6. Time restriction to the routes
Set Partitioning Model:

- **Pairing** Concept:
  Set of routes assigned to a vehicle and can be served within the given time limit.
ALTERNATE FORMULATION

Set Partitioning Model:

- **Variables** for set partitioning based on pairing concept:

\[
Z_{jp} = \begin{cases} 
1 & \text{if pairing } p \text{ is chosen for facility } j, \forall p \in P_j, \forall j \in M \\
0 & \text{otherwise}
\end{cases}
\]

- \( P_j \): set of feasible pairs of facility \( j \)

\[
T_j = \begin{cases} 
1 & \text{if facility } j \text{ is open}, \forall j \in M \\
0 & \text{otherwise}
\end{cases}
\]

- \( N \): set of customers; \( M \): set of facilities; \( I = N \cup M \)
Set Partitioning Model:

\[
\begin{align*}
\text{Min} & \quad \sum_{j \in M} T_j \cdot \text{FixCost} + \sum_{j \in M} \sum_{p \in P_j} C_{jp} \cdot Z_{jp} \\
\text{s.t.} & \quad \sum_{j \in M} \sum_{p \in P_j} a_{ipj} \cdot Z_{jp} = 1 \quad \forall i \in N \\
& \quad \sum_{i \in N} \sum_{p \in P_j} a_{ipj} \cdot \text{Demand}_i \cdot Z_{jp} \leq \text{Cap}_j \cdot T_j \quad \forall j \in M \\
& \quad Z_{jp} \leq T_j \quad \forall j \in M, p \in P_j \\
& \quad Z_{jp}, T_j \in \{0, 1\}, \forall j \in M, p \in P_j
\end{align*}
\]

\(a_{ipj} = 1\) if node \(i\) is in pairing \(p\) of facility \(j\).
Branch and Price Algorithm

1. **Restricted Master Problem (RMP)**
   - Initial Columns
   - Solve LP
   - LP–LOWER BD
   - IP– UPPER BD.

2. **Pricing Problem**
   - Find columns with (−) reduced cost
   - Until no new column
   - Dual Variables info

3. **If LP is not integral**
   - Branching
   - Modified RPM
     - Cut is added
     - Solve LP
   - modified Pricing Problem

ROOT NODE

Modified RPM
Cut is added
Restricted Master Problem:

- Initial pairs are formed

- Each pair represent a column in set partitioning formulation
- Restricted-since includes set of columns, not all columns
Pricing problem:

- Create ‘pair’: a column for $Y_{jp}$

- If 3rd const changed to:

  $$\sum_{j \in P_j} a_{ip} Z_{jp} \leq T_j \ \forall j \in M \text{ and } i \in N \quad (6)$$

  We have: $\pi_i, \mu_j, \gamma_{ji}$ dual variables

- **Reduced Cost** for $Y_{jp}$

  $$\hat{C}_{jp} = C_{jp} - \sum_{i \in N} a_{ipj} \pi_i + \sum_{i \in N} a_{ipj} Demand_i \mu_j + \sum_{i \in N} a_{ipj} \sigma_{ji} \quad (7)$$

  $C_{jp}$ = Operating cost of the vehicle ($\propto$ travel time) + Fixed Cost of a vehicle

  $\Rightarrow$ **Independent** pricing problem for each facility
**Elementary Shortest Path with Resource Constraint**

⇒ Pricing Problem = ESPRC If:
  1. Set up a network, including all customers and a source and sink nodes
  2. Arc costs:

\[ c_{kl} = \text{OperCost}.d_{kl} - \pi_l + \text{Demand}_i\mu_j + \sigma_{jl} \]  

(8)

⇒ find minimum cost path to the sink
⇒ in our case allow visits more than once to sink
⇒ If Total cost of path + Vehicle fixed Cost \( \leq 0 \),
   add the column to restricted master problem
⇒ stop when the shorthest path does not give negative cost column
**Elementary Shortest Path with resource constraint**

- **What** is an elementary path?
  Each node can be visited at most once.

- **Why** elementary instead of walks?
  Trade off between more difficult pricing problem and more depth in branch and bound tree

- In our case: # of visits to sink $\geq 1$

- In each visit to sink, current truck load is set to zero

- Adapt the *Labelling Algorithm for ESPRC* by Feillet, Dejax, Gendreau, Geuguen.
ESPRC

- Problem: too many feasible paths
- Keep resource consumptions, visited nodes, and cost
- Keep unreachable nodes for each label
- A node may be unreachable from other if not enough resource or is already visited.
- Eliminate dominated labels with respect to resource consumption and unreachable nodes.
Currently

What we have done

- Design Master Problem and Pricing Problem
- Adapted ESPRC algorithm to solve Pricing Problem
- Do the column generation
- Solve the root node
MINTO

→ **MINTO:** Mixed INTeger Optimizer
→ MINTO uses LP solver and do branch and bound algorithm
→ MINTO can do many applications such as preprocessing, constraint generation, primal heuristics
→ MINTO allows user to write own algorithm (for column generation, constraint generation, heuristics, ..) specific to the problem

Prof. Linderoth supports MINTO in our University

**WHAT ELSE?**

→ **COIN-BCP:** (Common Optimization INterface) and **SYMPHONY**
  - Open source
  - allows parallelization in branch and bound tree
  - supported by Prof. Ralphs
What we will do

- More implementation
  - Test problems
  - Determine the right number of columns to be generated in each time
  - Different LP algorithms, to find better reduced costs
  - See how well the root node solution
  - Create column pool
  - Branching strategies
  - parallelization
- Alternate solution: 2-sub problems approach
- IP formulation
- Focus on the pricing problem
**2-SUB PROBLEM APPROACH**

- Master problem includes 3 set of variables:
  - Location variables
  
  \[ Z_{jp} = \begin{cases} 
  1 & \text{if pairing } p \text{ is chosen for facility } j, \forall p \in P_j, \forall j \in M \\
  0 & \text{otherwise} 
\end{cases} \]

  \[ X_{jk} = \begin{cases} 
  1 & \text{if path } k \text{ is chosen for facility } j, \forall k \in S_j, \forall j \in M \\
  0 & \text{otherwise} 
\end{cases} \]

- 2 nested sub problems: generating paths, and combining these paths as pairs.

- **SP1: Generating paths:** vehicle routing problem or elementary shortest path with 2 resources

- **SP2: Combining paths:** knapsack problem
2-SUB PROBLEM ALGORITHM

1. Solve LP–RMP

2. Get dual variables for each customer and facility

3. Solve SP1: New Paths with (−) cost

   a. With new paths and old paths

   b. Solve SP2: generate pairs with (−) cost

4. Form pairs heuristically add to MP

5. Solve Master Problem

6. Get dual variables
THANKS...

Any Questions?