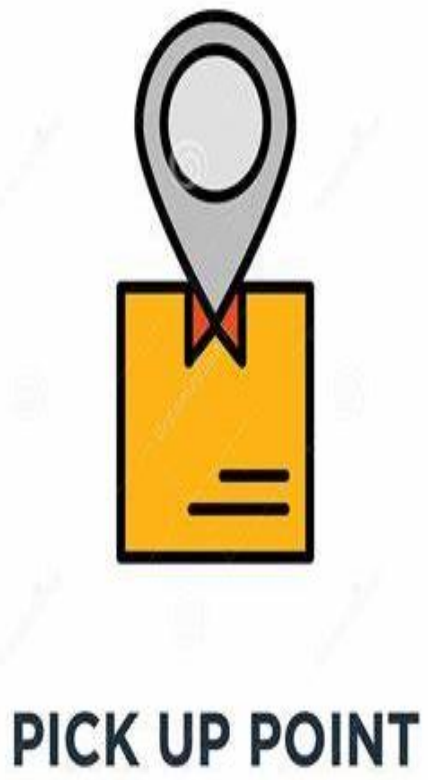
PAPER REVIEW PRESENTATION

FAMU-FSU College of Engineering

Student name: L. D. RICHARD CLINTON (LR21P)
Subject: Integer Programming

Solving a Practical Pickup and Delivery Problem

Objective: To minimize the total cost of covering the orders.



- There are three models that have been studied most widely in the literature:

  - Capacitated vehicle routing problem (CVRP)
  - Vehicle routing problem with time windows (VRPTW)
  - Practical pick-up and delivery problem (PPDP)

What is a practical pickup and delivery problem?

- **A practical pickup and delivery problem:** In this problem, there are multiple carriers and multiple vehicle types available to cover a set of pickup and delivery orders, each of which has multiple pickup time windows and multiple delivery time windows.

- Conditions: The associated complications are as follows.

- Multiple time windows
- Compatibility constraints
- Nested precedence
- DOT rules
- Complex cost structures.

Problem description:

The PPDP can be precisely described as follows.

- A shipper has a set of N orders N={1,….,N} to be shipped by K outside TL carriers K={1,….,K} with H vehicle types H={1,….,H}.
- Each order j ∈ N consists of a pickup at some location j+ and a delivery at some other location j− in the underlying transportation network.
- Each order j ∈N is a specific product (e.g., machinery, food), has a weight Wj, and requires a loading time Lj and an unloading time Uj.
- Each carrier k ∈ K has V_{kh} type-h vehicles available, for h ∈ H, for the shipper. Each type-h vehicle has a capacity of Q_h.

Goal:

To determine how many vehicles with each vehicle type from each carrier to use and construct a feasible trip for each vehicle to be used so that all the orders are shipped at a minimum cost.

The General Framework

To formulate the PPDP mathematically, we define the parameters S_{kh}, S, f_s, and e_{sj}, and binary integer variables x_s as follows:

S_{kh} = set of all feasible trips by a single vehicle with vehicle type $h \in \boldsymbol{H}$ and carrier $k \in \boldsymbol{K}$;

$S = \bigcup_{k,h} S_{kh}$ is the set of all feasible single-vehicle trips;

f_s = cost of trip $s \in S$;

$$e_{sj} = \begin{cases} 1 & \text{if order } j \in \boldsymbol{N} \text{ is covered by trip } s \in S \\ 0 & \text{otherwise;} \end{cases}$$

$$x_s = \begin{cases} 1 & \text{if trip } s \in S \text{ is adopted in the solution} \\ 0 & \text{otherwise.} \end{cases}$$

The PPDP can be formulated as the following set partitioning type problem:

$$\text{(SP)} \quad \text{Minimize} \quad \sum_{s \in S} f_s x_s, \tag{1}$$

$$\text{subject to} \quad \sum_{s \in S} e_{sj} x_s = 1, \quad \text{for } j \in \boldsymbol{N} \tag{2}$$

$$\sum_{s \in S_{kh}} x_s \le V_{kh}, \quad \text{for } k \in \boldsymbol{K} \text{ and } h \in \boldsymbol{H} \tag{3}$$

$$x_s \in \{0, 1\}, \quad \text{for } s \in S. \tag{4}$$

In this formulation:

(1) - The objective is to minimize the total cost of covering the orders.

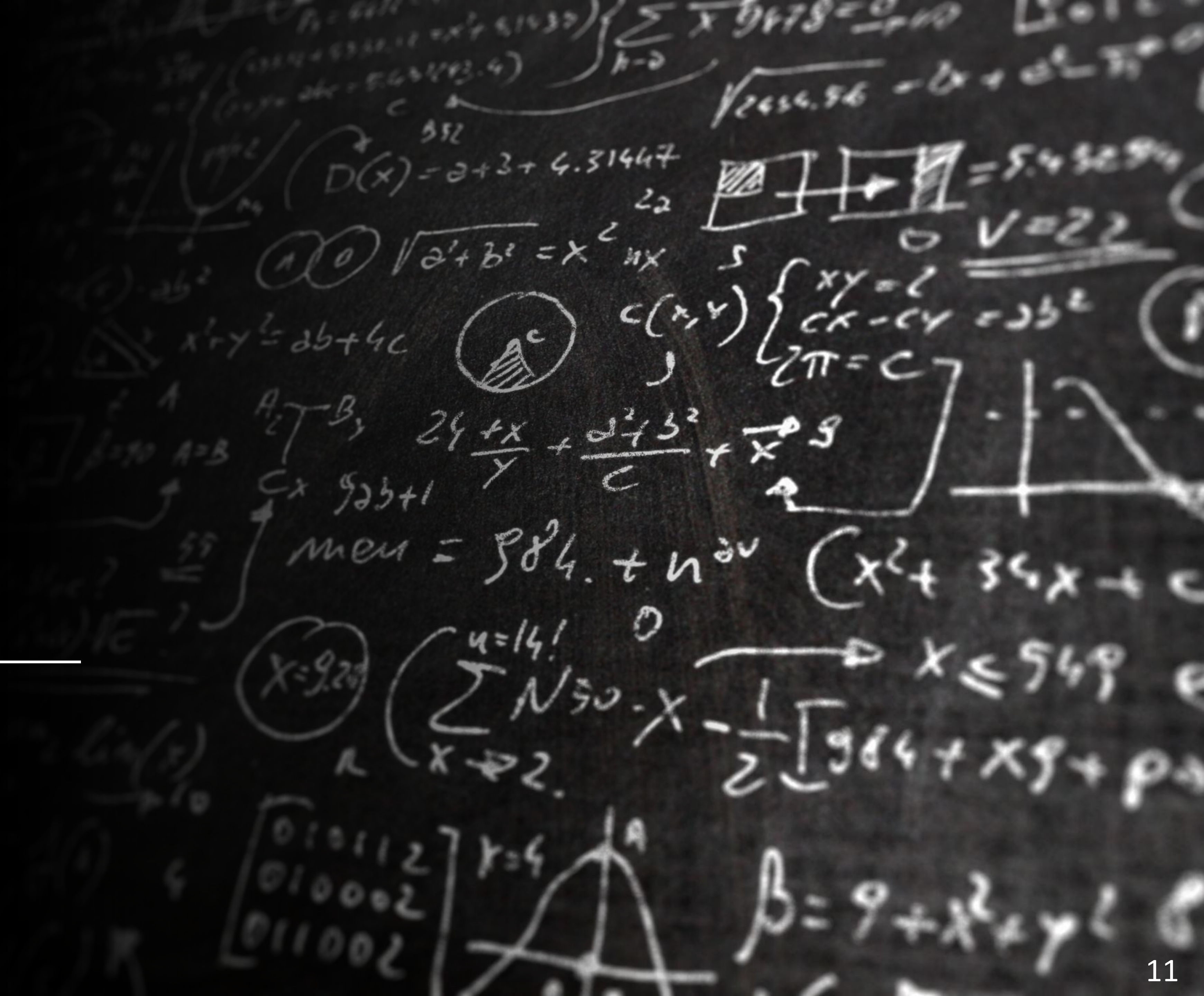
(2) - Constraint ensures that each order is covered.

(3) - Constraint represents that there are a total of V_{kh} vehicles available with a vehicle type h and carrier k.

- Let us note that the real-world complications involved in the PPDP (i.e., multiple time windows, DOT rules, compatibility constraints, nested precedence constraints) are not explicitly shown in (SP). Instead, all these complications are embedded in the columns of the formulation, which correspond to single-vehicle trips. The advantage of this is that all the complexity can be handled locally at the level of individual vehicles.

- Since the number of columns in (SP) is equal to the number of feasible single-vehicle trips, i.e., |S|, which is exponentially large, it is impractical to solve this formulation directly. Instead, we solve a restricted version of this problem, denoted as (SP′), which only contains a small subset of columns. This subset of columns is generated by solving the linear relaxation of (SP), denoted as (LSP), using the standard column generation procedure designed for large-scale linear programs.

What are the steps involved to solve the PPDP problem?



Approach to solve PPDP: 10 steps

Steps required to solve PPDP:

Step 1. Solve the linear relaxation (LSP) by the column generation procedure. Let S′ denote the set of all the columns (i.e., feasible trips) generated in this procedure.

Step 2. If the solution to (LSP) obtained in Step 1 is integral, then stop, and this solution is used as the solution to the problem. Otherwise, solve the restricted set partitioning type problem (SP′) which is the same as (SP) except that the set S in (SP) is replaced by the subset S′.

The computational experiments show that the solution for (LSP) obtained in Step 1 is integral in many cases, and in most other cases, few variables in the solution have fractional values. Therefore, we directly apply the IP solver of CPLEX to solve (SP′) in Step 2. This usually takes little time to get an integer solution even when there are more than 10,000 columns in (SP′).

In Step 1, to solve the linear relaxation problem (LSP), the column generation procedure decomposes (LSP) into a master problem and KH subproblems, one for each combination of carrier and vehicle type. The following is the framework of this procedure.

Column Generation for Solving (LSP).

Step 3. Generate an initial set of columns. Set up an initial restricted master problem of (LSP) with the initial set of columns.

Step 4. Solve the current master problem. Get the optimal dual variable values.

Solve the current master problem. Get the optimal dual variable values. Let π_j and σ_{kh} be the dual variable values corresponding to index j of the constraint (2), and index kh of the constraint (3) in (SP). The reduced cost of a column x_s corresponding to trip $s \in S_{kh}$ is given by the following formula:

$$r_s = f_s - \sum_{j \in s} \pi_j - \sigma_{kh}. \quad (5)$$

Step 5: Solve the subproblem for each carrier $k \in K$ and vehicle type $h \in H$. If no trip $s \in S$ is found with a negative reduced cost, stop; the problem (LSP) is solved already.

Otherwise, add the columns corresponding to the trips generated with the most negatively reduced cost to the master problem, go to Step 2.

Note: The master problem involved in the above procedure is a linear program and is solved by the LP solver of CPLEX. However, since the subproblems contain all the complications of the PPDP problem and are NP-hard, it is unlikely that any exact solution algorithm is capable of solving these subproblems of a large size efficiently.

Step 6: To solve these subproblems two fast heuristics called MERGE and TWO-PHASE are proposed. We will also describe how the initial set of columns is generated there.

In the merge procedure, Consider two trips in the basis of the solution of the current master problem, u and v, corresponding to the same carrier k and same vehicle type h, i.e., u, $v \in S_{kh}$. Since they are in the basis, their reduced costs are both 0, thus,

$$\sum_{j \in u} \pi_j = f_u - \sigma_{kh} \quad \text{and} \quad \sum_{j \in v} \pi_j = f_v - \sigma_{kh}.$$

If the trips u and v contain distinct orders, then the reduced cost of w is provided below:

$$\begin{aligned} r_w &= f_w - \sum_{j \in w} \pi_j - \sigma_{kh} = f_w - \sum_{j \in u} \pi_j - \sum_{j \in v} \pi_j - \sigma_{kh} \\ &= (f_w - f_u - f_v) + \sigma_{kh}. \end{aligned}$$

Step 7: After merging we now follow a procedure called “Generate”.

In order to generate a trip for the orders in R, one needs to specify a sequence for visiting the pickup and delivery locations of the orders involved and a time for visiting each location.

GENERATE consists of two parts: (i) routing—to find a sequence for visiting the pickup and delivery locations of the orders in R; (ii) scheduling—to determine when to visit each location once the visiting sequence is fixed.

Since different carriers and vehicles may have different cost rates, a trip generated this way may have a different reduced cost than the original trip. If it has a negative reduced cost, then it is added to the master problem.

Step 8: In this heuristic, whenever two trips are merged into a new single trip, to save computational time, only the vehicle types and carriers involved in the two original trips are considered for the new trip.

Step 9: Finally, we use the heuristic TWO-PHASE.

Step 10: A dynamic programming (DP) algorithm for solving the subproblems to optimality. The objective value of (LSP) obtained by applying this DP algorithm to the subproblems is thus, a lower bound of the PPDP.

Computational results for the medium problem:

Table 2 **Computational Results for Medium Problems**

Test Problem			MERGE					TWO-PHASE				
# of Orders	Square Size	DP Avg. CPU	Avg. Gap %	Max. Gap %	Avg. CPU	Avg. % of CPU on B&B	Avg. # of Columns in (SP′)	Avg. Gap %	Max. Gap %	Avg. CPU	Avg. % of CPU on B&B	Avg. # of Columns in (SP′)
50	800 × 800	104.96	1.84	4.19	8.79	0.2	449	0.75	1.39	49.44	0.3	694
70		599.49	2.96	5.62	19.33	0.5	673	1.28	3.32	96.20	0.3	1,147
90		3,196.66	3.43	6.36	28.36	0.5	1,025	1.57	3.79	189.30	0.2	1,744
110		6,901.67	3.35	5.41	46.36	0.7	1,397	1.28	2.12	279.20	0.5	2,493
50	1200 × 1200	21.04	1.20	2.54	6.05	0.5	405	0.93	1.94	16.86	0.4	535
70		89.74	2.65	6.37	15.00	0.7	605	1.13	3.28	33.53	0.3	823
90		411.43	2.45	3.97	24.87	0.8	936	1.01	2.40	74.82	0.4	1,272
110		1,219.76	2.63	5.89	35.52	0.8	1,253	1.21	2.36	128.24	0.5	1,734
130		4,198.44	3.04	6.95	56.44	0.9	2,097	1.67	4.62	183.30	0.8	2,778
150		6,340.64	3.74	9.72	105.5	1.4	2,721	2.00	6.06	309.31	1.0	3,595
50	1600 × 1600	8.27	1.48	3.22	3.96	0.4	372	0.24	0.39	8.41	0.3	439
70		29.06	1.22	2.71	10.26	0.5	545	0.96	2.78	18.03	0.4	650
90		114.56	1.89	5.57	19.79	0.9	886	1.13	4.42	37.60	0.5	1,056
110		290.86	1.99	3.66	32.25	0.9	1,169	0.91	3.33	61.34	0.6	1,403
130		969.58	2.19	3.98	54.83	1.3	1,988	1.28	3.00	117.18	0.8	2,333
150		2,099.02	2.15	3.20	68.46	3.0	2,560	1.26	2.00	139.84	1.9	3,028
170		3,961.14	2.31	3.52	64.79	1.2	2,934	1.46	1.76	188.29	0.9	3,568
190		6,154.41	2.70	3.75	78.02	1.3	3,410	1.51	2.62	263.95	0.9	4,189
210		14,301.57	3.70	5.54	100.9	1.3	3,744	1.69	2.38	293.13	1.5	4,661

Computational results for the large problem:

Table 3 Computational Results for Large Problems

Test Problem		TWO-PHASE			MERGE				
# of Orders	Square Size	Avg. CPU	Avg. % of CPU on B&B	Avg. # of Columns in (SP′)	Avg. Gap %	Max. Gap %	Avg. CPU	Avg. % of CPU on B&B	Avg. # of Columns in (SP′)
300	800 × 800	7,416.48	3.2	17,105	4.18	5.43	636.02	1.6	10,860
350		11,766.24	6.5	24,242	4.53	6.04	872.69	1.4	13,069
400		22,203.28	5.1	35,194	4.99	6.44	1,294.6	1.6	15,149
450		22,230.75	2.8	38,665	4.68	5.55	1,750.6	1.7	17,409
500		25,909.69	2.4	48,521	4.28	5.03	2,696.1	1.4	19,758
300	1200 × 1200	2,381.36	2.2	12,608	3.27	5.31	475.77	1.6	10,229
350		4,628.85	1.7	17,608	2.88	3.63	706.76	1.5	12,284
400		5,430.97	2.5	24,295	3.05	4.45	1,022.2	1.7	14,165
450		9,900.59	1.8	26,484	3.48	4.57	1,460.1	1.5	16,173
500		12,085.91	1.6	44,649	4.01	4.74	2,499.9	1.7	18,410
300	1600 × 1600	1,204.97	1.3	9,768	1.98	4.50	368.15	2.1	9,916
350		1,957.45	1.7	15,353	1.91	3.50	516.46	2.6	11,932
400		2,843.89	2.1	21,673	2.09	4.00	814.00	2.7	13,762
450		4,142.78	0.7	23,568	2.10	3.23	1,001.2	2.5	15,779
500		6,165.79	2.0	35,605	2.70	3.42	1,669.7	1.3	17,255

Conclusion:

- In this paper, we have presented a pickup and delivery problem which involves a set of practical issues such as multiple time windows, heterogeneous vehicle types, DOT rules, nested precedence constraints, and compatibility constraints.

- We have designed hybrid approaches that combine the well-known column generation methodology with fast heuristics for the subproblems resulting in the column generation procedure.

- The heuristics MERGE and TWO-PHASE are shown to be capable of finding near-optimal solutions quickly for problems with up to 210 orders. Furthermore, it is shown that our algorithms can handle problems with up to 500 orders within acceptable computational time.

References:

1. Barnhart, C., E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, P. H. Vance. 1998. Branch-and-price: Column generation for solving huge integer programs. Oper. Res. 46 316–329.
2. Bodin, L. D. 1990. Twenty years of routing and scheduling. Oper. Res. 38 571–579.
3. Bramel, J., D. Simchi-Levi. 1997. The logic of Logistics: Theory, Algorithms, and Applications for Logistics Management. SpringerVerlag, New York.
4. Chen, Z.-L., W. B. Powell. 1999. Solving parallel machine scheduling problems by column generation. INFORMS J. Comput. 1178–94.
5. Crainic, T. G., G. Laporte. 1998. Fleet Management and Logistics. Kluwer Academic Publishers, Boston, MA.
6. Dantzig, G. B., P. Wolfe. 1960. Decomposition principle for linear programs. Oper. Res. 8 101–111.

Any questions?

THANK YOU!

THE END