Column Generation

Teo Chung-Piaw (NUS)

25^{th} February 2003, Singapore

1 Lecture

1.1 Outline

SLIDE 1

- Cutting Stock Problem
- Classical Integer Programming Formulation
- Set Covering Formulation
- Column Generation Approach
- Connection with Lagrangian Relaxation
- Computational issues

2 Cutting Stock Problem

2.1 Introduction

2.1.1 Example

SLIDE 2

- A paper company has a supply of large rolls of paper, each of width W.

SLIDE 3

- Customers demand n_i rolls of width w_i (i=1, ..., m). ($w_i \leq W$)

  Example:

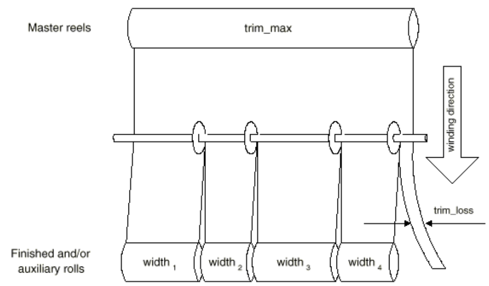
Quantity Ordered n_i	Order Width (inches) w_i
97	45
610	36
395	31
211	14

SLIDE 4

- The demand can be met by slicing a large roll in a certain way, called a **pattern**.
- For example, a large roll of width 100 can be cut into

– 4 rolls each of width 25, or

– 2 rolls each of width 35, with a waste of 30.



3 Solution Approach I

3.1 L. V. Kantorovich

3.1.1 Formulation

Slide 5

(1939 Russian, 1960 English) "Mathematical Methods of Planning and Organising Production" *Management Science*, **6**, 366-422.

- $\mathcal{K}$: Set of available rolls.
- y^k: 1 if roll k is cut, 0 otherwise.
- x_i^k: number of times item i is cut on roll k.

Objective: To minimize the number of rolls used to meet all the demand

$$\min \sum_{k \in \mathcal{K}} y^k$$

Slide 6

Constraints

- Total number of times item i is cut is not less than the demand.

$$\sum_{k \in \mathcal{K}} x_i^k \geq n_i$$

- The width of a roll is at most W

$$\sum_i w_i x_i^k \leq W y^k$$

SLIDE 7

$$
\begin{array}{ll}
\min & \sum_{k\in\mathcal{K}} y^k \\
s.t. & \sum_{k\in\mathcal{K}} x_i^k \geq n_i \quad \text{for a fixed item} \quad i, \\
& \sum_i w_i x_i^k \leq W y^k \quad \text{for fixed roll} \quad k, \\
& x_i^k \geq 0, 0 \leq y^k \leq 1 \\
& \text{Integrality constraints on all variabes}
\end{array}
$$

3.1.2 Quality of Solution

SLIDE 8

Scenario I: $\boxed{\text{N1}}$

- n_i : uniform, between 1 and 100 (rand(100)+1);
- w_i : uniform, between 1 and 30 (rand(30)+1);
- Width of Roll, $W = 3000$;

Rolls	Items	constr	variables	CPU (s)
30	60	90	1830	2.8
50	100	150	5050	14.33
100	200	300	20100	179
200	400	600	80200	3048

Note 1 ***Code***

The OPL code for the problem:

```
range RollIndex 1..20;
range PaperIndex 1..40;
int+ largeRollWidth = 3000;

int+ demand[PaperIndex];
initialize{
   forall(i in PaperIndex)
    demand[i] = rand(100)+1;
};

int+ paperWidth[PaperIndex];
initialize{
   forall(i in PaperIndex)
    paperWidth[i] = rand(30)+1;
};

var int+ y[RollIndex] in 0..1;
var int+ x[PaperIndex, RollIndex] in 0..largeRollWidth;

minimize sum(k in RollIndex)y[k]
subject to{
   forall(i in PaperIndex)
     sum(k in RollIndex) x[i,k] >= demand[i];

   forall(k in RollIndex)
     sum(i in PaperIndex) paperWidth[i]*x[i,k] <= largeRollWidth*y[k];
};
```

SLIDE 9

Scenario II: Change width of the roll from 3000 to 150.

Rolls	Items	constr	variables	CPU (s)
70	10	80	770	3.18 - never
140	20	160	2940	58.28 - never
210	30	240	6510	Out of memory

The performance has deteriorated. Why?

SLIDE 10

How good is the LP relaxation?

Observation: $Z_{LP} = \frac{\sum_i w_i n_i}{W}$. $\boxed{\text{N2}}$

This bound is trivial: The objective is to

$$\min \sum_{k \in \mathcal{K}} y^k$$

the optimal solution will satisfy:

- Choose y^k as small as possible. Therefore

$$\sum_i w_i x_i^k = W y^k$$

  for all k

SLIDE 11

- Choose x_i^k as small as possible. Therefore

$$\sum_k x_i^k = n_i$$

for all i.

$$\begin{aligned}\sum_{k\in\mathcal{K}} y^k &= \sum_{k\in\mathcal{K}} \frac{\sum_i w_i x_i^k}{W} = \sum_i \sum_{k\in\mathcal{K}} \frac{w_i}{W} x_i^k \\ &= \sum_i \frac{w_i}{W} \sum_{k\in\mathcal{K}} x_i^k = \sum_i \frac{w_i n_i}{W}\end{aligned}$$

Note 2 ***Proof***

We have the constraints

$$\sum_i w_i x_i^k \leq W y^k.$$

In the LP, since the objective is to minimize $\sum_k y^k$, the optimal LP solution will be such that

$$\frac{\sum_i w_i x_i^k}{W} = y^k.$$

y^k will be small if the x_i^k values are small. At the same time, because

$$\sum_{k\in\mathcal{K}} x_i^k \geq n_i,$$

to make x_i^k small, at the optimal solution, we must have

$$\sum_{k\in\mathcal{K}} x_i^k = n_i.$$

So the objective function, for the optimal LP solution, reduces to

$$\begin{aligned}\sum_{k\in\mathcal{K}} y^k &= \sum_{k\in\mathcal{K}} \frac{\sum_i w_i x_i^k}{W} \\ &= \sum_i \sum_{k\in\mathcal{K}} \frac{w_i}{W} x_i^k \\ &= \sum_i \frac{w_i}{W} \sum_{k\in\mathcal{K}} x_i^k \\ &= \sum_i \frac{w_i n_i}{W}\end{aligned}$$

SLIDE 12

Another Example: $W = 273$

Quantity Ordered	Order Width (inches)
233	18
310	91
122	21
157	136
120	51

- LP: solved in 0.27s. Solution 228.7106.
- IP: halted after 12 hours of computational time!

4 Approach II

4.1 Gilmore and Gomory

4.1.1 Set Covering

SLIDE 13

P. C. GILMORE AND R. E. GOMORY, A linear programming approach to the cutting-stock problem, *Oper. Res.*, **8** (1961), pp. 849-859.

x_j = number of times pattern j is used

a_{ij} = number of times item i is cut in pattern j

For example, a large roll of width 100 can be cut into

- 4 rolls each of width $w_i = 25$ (pattern j, $a_{ij} = 4$)
- 2 rolls each of width $w_k = 35$ (pattern l, $a_{kl} = 2$)

SLIDE 14

$$\begin{array}{lrcll} \min & \sum_{j=1}^n x_j & & & \\ \text{s.t.} & \sum_{j=1}^n a_{ij}x_j & \geq & n_i, & i = 1, \dots, m, \\ & x_j & \in & Z^+, & j = 1, \dots, n \end{array}$$

SLIDE 15

Example: An instance: $W = 100$, $m = 3$.

	pattern						
w_i	**1**	**2**	**3**	**4**	**5**	**6**	n_i
25	4	2	2	1	0	0	**150**
35	0	1	0	2	1	0	**200**
45	0	0	1	0	1	2	**300**

SLIDE 16

Another way to formulate the cutting stock problem:

		minimize			$\sum_{j=1}^6 x_j$	
x_1	x_2	x_3	x_4	x_5	x_6	RHS
$4x_1$ +	$2x_2$ +	$2x_3$ +	$1x_4$ +	$0x_5$ +	$0x_6 \geq$	**150**
$0x_1$ +	$1x_2$ +	$0x_3$ +	$2x_4$ +	$1x_5$ +	$0x_6 \geq$	**200**
$0x_1$ +	$0x_2$ +	$1x_3$ +	$0x_4$ +	$1x_5$ +	$2x_6 \geq$	**300**

x_j = number of rolls to be cut using pattern j.

4.1.2 Computational Issues

SLIDE 17

Linear relaxation: (LP)

$$\begin{array}{lrcll} \min & \sum_{j=1}^n x_j & & & \\ \text{s.t.} & \sum_{j=1}^n a_{ij}x_j & \geq & n_i, & i = 1, \dots m, \\ & x_j & \geq & 0, & j = 1, \dots, n \end{array}$$

- LP solution provides a lower bound to IP.

SLIDE 18

How to solve (LP)?

- **Feasible patterns**: all nonnegative integer vectors $(z_1, \ldots, z_m)$ satisfying

$$\sum_{i=1}^{m} w_i z_i \leq W$$

- Not all feasible patterns are needed in the above formulation.

Issue: In the constraint matrix $\mathbf{A}$, the number of decision variables (i.e., number of feasible patterns) is **large**!

5 Cutting Stock Problem

5.1 Column Generation

5.1.1 Algorithm

SLIDE 19

1. Start with a basic feasible solution $\mathbf{B}$.

   For example, use the simple pattern to cut a roll into $\lfloor W/w_i \rfloor$ rolls of width w_i. (The basis matrix is a diagonal matrix.)

2. For any pattern j, reduced cost is $1 - \sum_{i=1}^{m} \pi_i a_{ij}$, where $(\pi_1, \ldots, \pi_m)$ $(=\mathbf{c_B B^{-1}})$ is the simplex multipliers vector associated with the current basis.

3. Identify a **pattern** with **negative** reduced cost, or prove that none exists. Update basis and repeat.

SLIDE 20

For instance, we may want to find the column with **most** negative reduced cost:

$$\begin{array}{rl} Z^*(\pi) = \min & 1 - \sum_{i=1}^{m} \pi_i x_i \\ \text{subject to} & \sum_i w_i x_i \leq W, x_i \text{ integral.} \end{array}$$

- If $Z^*(\pi) \geq 0$, all columns have negative reduced cost!
- Otherwise, the solution gives rise to a column with negative reduced cost!

5.1.2 Identifying Columns

SLIDE 21

$$\begin{array}{rl} \min & 1 - \sum_{i=1}^{m} \pi_i x_i \\ & \text{subject to } \sum_i w_i x_i \leq W, \quad x_i \text{ integral.} \end{array}$$

is equivalent to solving

$$\begin{array}{rcl} Z'(\pi) & = & \max \sum_{i=1}^{m} \pi_i x_i \\ \text{subject to} & & \sum_i w_i x_i \leq W, \; x_i \text{ integral.} \end{array}$$

5.1.3 Knapsack problem

Slide 22

$$\begin{aligned} Z'(\pi) = \max \quad & \sum_{i=1}^{m} \pi_i x_i \\ \text{subject to} \quad & \sum_i w_i x_i \le W, \\ & x_i \text{ integral.} \end{aligned}$$

- The column generation method depends critically on how fast we can solve the knapsack problem.
- How difficult is it to solve the knapssack problem?

Slide 23

Computational Result on random instances using the MIP solver from CPLEX:

n	CPU (s)
1,000	0.22
10,000	1.04
100,000	75.52

More specialized algorithm can be used to solve the Knapsack problem efficiently in practice.

5.1.4 How Good is the bound?

Slide 24

Number of items = 5.
Tested on several instances:

Optimal	Col Gen (LP)
15	14.0533
11	10.4733
34	33.0989
19	18.2867

Round Up Conjecture:

$$Z_{IP} \le \lceil Z_{LP} \rceil ?$$

Slide 25

Unfortunately, this is not true:

- $W = 273$
-

$w_1 = 18$	$n_1 = 233$
$w_2 = 91$	$n_2 = 310$
$w_3 = 21$	$n_3 = 122$
$w_4 = 136$	$n_4 = 157$
$w_5 = 51$	$n_5 = 120$

$Z_{LP}(CG) = 228.9982$. $Z_{IP} = 230$. $\boxed{\text{N3}}$

Note 3 ***Round Up Conjecture***

In 1985 the theoretical result of Marcotte (The cutting stock problem and integer rounding, Math. Programming, 33 (1985), pp. 82-92) shed light on the relationship between solutions of the LP relaxation and the cutting stock problem itself. She proved that for some practical instances of the problem a so-called round-up property is valid. It means that to find the optimal value of the cutting stock problem, it is sufficient just to solve the LP relaxation and to round up the value of the objective function.
Unfortunately, this conjecture is not true for all instances of the cutting stock problem. Fieldhouse (The duality gap in trim problems, SICUP-Bulletin No. 5, 1990) presents an example of the cutting stock problem with a gap of 1.0333. This gives rise to the "modified" round up conjecture.

5.1.5 Modified Round Up Conjecture

SLIDE 26

$$Z_{IP} \leq \lceil Z_{LP} \rceil + 1?$$

- This conjecture has not been answered.
- Can you disprove it?

5.1.6 Getting Intergal Solution

SLIDE 27

The solution obtained from solving the column generation problem may fractional.
How to obtain integral solution?

- round up the fractional solution. (e.g., change 18.3 to 19).

- round down the fractional solution, and resolve the problem with smaller set of demand.

- branch and bound to obtain the optimal integral solution

5.1.7 Rounding Up

SLIDE 28

- Let x_j be the (fractional) LP solution obtained from the column generation method.
- Let $x'_j = \lceil x_j \rceil$. x'_j integral.
- $\sum_j a_{i,j} x_j \geq n_i$ implies $\sum_j a_{i,j} x'_j \geq n_i$. So x'_j defined in this way is a feasible integral solution.
- How good is this heuristic?

SLIDE 29

$W = 273$	
$w_1 = 18$	$n_1 = 233$
$w_2 = 91$	$n_2 = 310$
$w_3 = 21$	$n_3 = 122$
$w_4 = 136$	$n_4 = 157$
$w_5 = 51$	$n_5 = 120$

- $Z_{LP}(CG) = 228.9982$.
- Round-Up produces a solution of 231

SLIDE 30

	Round-Up	Fractional	**18**	**91**	**21**	**136**	**51**
cut	0	0.0000	15	0	0	0	0
cut	104	103.3333	0	3	0	0	0
cut	9	8.2363	0	0	13	0	0
cut	79	78.5000	0	0	0	2	0
cut	0	0.0000	0	0	0	0	5
cut	24	24.0000	1	0	0	0	5
cut	15	14.9286	14	0	1	0	0

Disadvantages of the round-up heuristic?

6 Column Generation

6.1 Dual Perspective

6.1.1 Lagrangian Relaxation

SLIDE 31

How would you use LR to solve the cutting stock problem?

$$
\begin{array}{rc}
\min & \sum_{k=1}^{K} y^k \\
s.t. & \sum_{k=1}^{K} x_i^k \geq n_i \quad \forall \quad i = 1, 2, \ldots, m, \\
& \sum_{i=1}^{m} x_i^k w_i \leq W y^k \quad \forall \quad k = 1, \ldots, K, \\
& y^k \in \{0, 1\} \ \forall \ k, \\
& x_i^k \geq 0, x_i^k \text{ integral}
\end{array}
$$

Which constraints would you relax?

SLIDE 32

Suppose you relax the first class of constraints:

$$
\begin{array}{rc}
L(\mathbf{u}) = \min & \sum_{k=1}^{K} y^k + \sum_{i=1}^{m} u_i \left(n_i - \sum_{k=1}^{K} x_i^k \right) \\
s.t. & \sum_{i=1}^{m} x_i^k w_i \leq W y^k \quad \forall \quad k = 1, \ldots, K, \\
& y^k \in \{0, 1\} \ \forall \ k, \\
& x_i^k \geq 0, x_i^k \text{ integral}
\end{array}
$$

where $\mathbf{u}_i \geq 0$ for all i. SLIDE 33

$$L(\mathbf{u}) = \sum_{k=1}^{K} \mathcal{L}_k(\mathbf{u}) + \sum_{i=1}^{m} u_i n_i$$

where

$$\begin{array}{rl} \mathcal{L}_k(\mathbf{u}) = \min & y^k - \sum_{i=1}^m u_i x_i^k \\ s.t. & \sum_{i=1}^m x_i^k w_i \leq W y^k \\ & y^k \in \{0,1\} \\ & x_i^k \geq 0, x_i^k \text{ integral} \end{array}$$

$\mathcal{L}_k(\mathbf{u})$ is the minimum of the two values: zero (when $y^k = 0$), or 1-Z^* (when $y^k = 1$) , where Z^* is obtained by solving the knapsack problem SLIDE 34

$$\begin{array}{rl} Z^* = \max & \sum_{i=1}^m u_i x_i^k \\ s.t. & \sum_{i=1}^m x_i^k w_i \leq W \\ & x_i^k \geq 0, x_i^k \text{ integral} \end{array}$$

The subproblem in Lagragian Relaxation reduces again to a Knapsack problem!

SLIDE 35

Proposition: $\boxed{\text{N4}}$

- $\max_{\mathbf{u} \geq 0} L(\mathbf{u}) = Z_{LP}(CG)$.
- The optimal Lanagrangian multiplers is the LP dual multiplers to the constraints $\sum_{j=1}^N a_{i,j} K \lambda_j \geq n_i$ in the column formulation.
- Lagrangian relaxation solves the dual of the column formulation!

Note 4 ***Derivation of proposition***

$$L(\mathbf{u}) = \sum_{k=1}^{K} \mathcal{L}_k(\mathbf{u}) + \sum_{i=1}^{m} \mathbf{u}_i n_i$$

The value $\mathcal{L}_k(\mathbf{u})$ does not depend on the index k.

$$L(\mathbf{u}) = K\mathcal{L}(\mathbf{u}) + \sum_{i=1}^{m} \mathbf{u}_i n_i$$

where

$$\begin{array}{rl} \mathcal{L}(\mathbf{u}) = \min & y - \sum_{i=1}^m u_i x_i \\ s.t. & \sum_{i=1}^m x_i w_i \leq W y \\ & y \in \{0,1\} \\ & x_i \geq 0, x_i \text{integral} \end{array}$$

- Let $\mathbf{z}_j = (a_{1,j}, \ldots, a_{m,j})$, $j = 1, \ldots, N$ be the extreme points of the polytope
$$\sum_{i=1}^{m} x_i w_i \leq W,\ x_i \geq 0,\ x_i \text{ integral.}$$
- $\mathcal{L}(\mathbf{u}) = \min(0, 1 - \max_{j=1,\ldots,N} \sum_{i=1}^{m} a_{i,j}\mathbf{u}_i) = \min_{j=1,\ldots,N} \min(0, 1 - \sum_{i=1}^{m} a_{i,j}\mathbf{u}_i)$

$$L(\mathbf{u}) = K\mathcal{L}(\mathbf{u}) + \sum_{i=1}^{m} \mathbf{u}_i n_i$$

The Lagrangian Dual

$$\begin{aligned}\max_{\mathbf{u}\geq 0} L(\mathbf{u}) &= \max_{\mathbf{u}\geq 0} \min_{j=1,\ldots,N} \Bigg(K \min(0, 1 - \sum_{i=1}^{m} a_{i,j}\mathbf{u}_i) \\ &\qquad + \sum_{i=1}^{m} \mathbf{u}_i n_i \Bigg)\end{aligned}$$

reduces to

$$\begin{array}{lcl} & \max\ \ \mathbf{y} & \\ s.t. & \mathbf{y} \leq \sum_{i=1}^{m} \mathbf{u}_i n_i & \text{for the zero extreme point} \\ & \mathbf{y} \leq K(1 - \sum_{i=1}^{m} \mathbf{u}_i a_{i,j}) + \sum_{i=1}^{m} \mathbf{u}_i n_i & \text{for the } j\text{th extreme point} \\ & \mathbf{u}_i \geq 0\ \forall\ \ i = 1, \ldots, m. & \end{array}$$

Equivalently,

$$\begin{array}{cc} & \max\ \ \ \mathbf{y} \\ (\lambda_0) & \mathbf{y} - \sum_{i=1}^{m} \mathbf{u}_i n_i \leq 0 \\ (\lambda_j) & \mathbf{y} + K(\sum_{i=1}^{m} \mathbf{u}_i a_{i,j}) - \sum_{i=1}^{m} \mathbf{u}_i n_i \leq K \\ & \mathbf{u}_i \geq 0\ \forall\ \ i = 1, \ldots, m. \end{array}$$

λ_0, λ_j are the associated dual variables.
The dual of this problem:

$$\begin{array}{cc} \min & \sum_{j=1}^{N} K\lambda_j \\ & \lambda_0 + \sum_{j=1}^{N} \lambda_j = 1 \\ & -n_i(\lambda_0 + \sum_{j=1}^{N} \lambda_j) + \sum_{j=1}^{N} a_{i,j} K\lambda_j \geq 0. \\ & \lambda_j \geq 0\ \forall\ \ j = 1, \ldots, N. \end{array}$$

This is just

$$\min \qquad \sum_{j=1}^{N} \Big(K\lambda_j \Big)$$

$$\lambda_0 + \sum_{j=1}^N \lambda_j = 1$$
$$\sum_{j=1}^N a_{i,j} K\lambda_j \geq n_i$$
$$\lambda_j \geq 0 \;\forall\;\; j = 1, \ldots, N.$$

For K large, the constraint $\lambda_0 + \sum_{j=1}^N \lambda_j = 1$ is redundant – as long as there is a feasible solution λ_j with $\sum_{j=1}^N \lambda_j \leq 1$
Let $x_j = K\lambda_j$. We obtain the column formulation of the cutting stock problem !

6.2 Lagrangian Relaxation

6.2.1 Comparison

SLIDE 36

- The bounds obtained by both methods are identical.
- Which method is better?

Column Generation	**Langrangean Relaxation**
Primal, dual optimal solution	Dual but not primal solution
(Primal) Bounds monotone	(Dual) Bounds zig-zag
Dual solution zig-zag	Dual solution suitably selected
LP solver needed	Easy to implement

6.3 Speeding Up

6.3.1 Column Selection

SLIDE 37

- **Prevent generation of redundant columns**.
  Instead of $\min_j \left\{ c_j - \pi^{\mathbf{T}}\mathbf{N_j} \right\}$, solve

$$\min_{j:\pi^T N_j > 0} \frac{c_j}{\pi^T N_j},$$

or

$$\min_j \left\{ \frac{c_j - \pi^{\mathbf{T}}\mathbf{N_j}}{1^T N_j} \right\}.$$

SLIDE 38

- $\min_j \left\{ c_j - \pi^{\mathbf{T}}\mathbf{N_j} \right\}$ versus $\min_j \left\{ \frac{c_j - \pi^{\mathbf{T}}\mathbf{N_j}}{1^T N_j} \right\}$

	Round-Up	Fractional	**18**	**91**	**21**	**136**	**51**
cut	0	0.0000	15	0	0	0	0
cut	104	103.3333	0	3	0	0	0
cut	9	8.2363	0	0	13	0	0
cut	79	78.5000	0	0	0	2	0
cut	0	0.0000	0	0	0	0	5
cut	24	24.0000	1	0	0	0	5
cut	15	14.9286	14	0	1	0	0

- **Maintain a column pool**.

  Check for columns with negative reduced cost in the column pool before solving the pricing subproblem. Replenish the column pool everytime you solve a pricing subproblem.

6.3.2 Dual Selection

SLIDE 39

Restrict domain of dual prices

Use properties of optimal dual prices to restrict the domain.
In the cutting stock problem, suppose the orders are ranked such that $w_1 < w_2 < \ldots < w_m$, then it is easy to see that the dual prices satisfy:

$$\pi_1 \leq \pi_2 \leq \ldots \leq \pi_m.$$

Translating into the primal, this is equivalent to adding the following columns with zero cost to the primal problem:

$$\begin{pmatrix} 1 \\ -1 \\ 0 \\ \cdots \\ \cdots \\ \cdots \\ 0 \end{pmatrix}, \quad \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \\ \cdots \\ \cdots \\ 0 \end{pmatrix}, \quad \ldots \begin{pmatrix} 0 \\ \cdots \\ \cdots \\ \cdots \\ 0 \\ 1 \\ -1 \end{pmatrix}$$

7 Column Generation

7.1 Applications

7.1.1 Examples

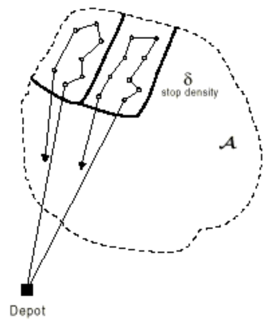
SLIDE 40

Other application of column generation:

Vehicle Routing with time window or other types of constraint:

- A set of m customers.
- Each customer must be served within certain time window.
- Find a set of routes to serve all the customers, so that each customer will be visited by a vehicle within the stipulated time window.

Here each column may represent a feasible trip. One likely objective is to find minimal number of vehicles to cover all the customers. SLIDE 41



SLIDE 42

Column generation phase now reduces to the following: Given a profit π_i for each demand point, find a route that satisfies:

- feasibility constraints (meet the time window constraints);
- total profits accrued by serving the demand points on the route is maximum - a variant of the TSP problem.

N5

Note 5 ***Papers***

- J. Desrosiers, Y. Dumas,F. Soumis & M. Solomon. Time Constrained Routing and Scheduling, **Handbooks in OR & MS**, 8 (1995)
- G. Desaulniers et al. A Unified Framework for Deterministic Vehicle Routing and Crew Scheduling Problems T. Crainic & G. Laporte (eds) Fleet Management & Logistics (1998).

8 Conclusions

SLIDE 43

- Column Generation has been successfully used to solve many large scale integer programming problem arising in the industry.
- Able to handle large scale model that standard commercial MIP solver cannot handle.
- Ability to solve the pricing subproblem efficiently is key to the approach
- Connection between Column generation and Lagragian Relaxation
- Non-linearities occuring in practical problems taken care of in the subproblem (next lecture)