

## 15-853: Algorithms in the Real World

### Linear and Integer Programming III

- Integer Programming
  - Applications
  - Algorithms

15-853

Page1

## Integer (linear) Programming

$$\begin{array}{ll} \text{minimize:} & c^T x \\ \text{subject to:} & Ax \leq b \\ & x \geq 0 \\ & x \in \mathbb{Z}^n \end{array}$$

### Related Problems

- Mixed Integer Programming (MIP)
- Zero-one programming
- Integer quadratic programming
- Integer nonlinear programming

15-853

Page2

## History

- Introduced in 1951 (Dantzig)
- TSP as special case in 1954 (Dantzig)
- First convergent algorithm in 1958 (Gomory)
- General branch-and-bound technique 1960 (Land and Doig)
- Frequently used to prove bounds on approximation algorithms (late 90s)

15-853

Page3

## Current Status

- Has become "dominant" over linear programming in past decade
- Saves industry Billions of Dollars/year
- Can solve 10,000+ city TSP problems
- 1 million variable LP approximations
- Branch-and-bound, Cutting Plane, and Separation all used in practice
- General purpose packages do not tend to work as well as with linear programming --- knowledge of the domain is critical.

15-853

Page4

## Subproblems/Applications

- **Facility location**  
Locating warehouses or franchises (e.g. a Burger King)
- **Set covering and partitioning**  
Scheduling airline crews
- **Multicommodity distribution**  
Distributing auto parts
- **Traveling salesman and extensions**  
Routing deliveries
- **Capital budgeting**
- **Other Applications**  
VLSI layout, clustering

15-853

Page5

## Knapsack Problem

### Integer (zero-one) Program:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to:} & ax \leq b \\ & x \text{ binary} \end{array}$$

### where:

b = maximum weight

$c_i$  = utility of item i

$a_i$  = weight of item i

$x_i = 1$  if item i is selected, or 0 otherwise

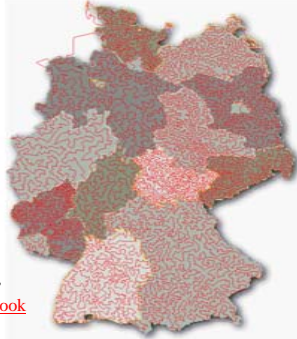
The problem is NP-hard.

15-853

Page6

## Traveling Salesman Problem

Find shortest tours that visit all of n cities.



courtesy: [Applegate](#),  
[Bixby](#), [Chvátal](#), and [Cook](#)

15-853

Page7

## Traveling Salesman Problem

$$\text{minimize: } \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij}$$

$$\text{subject to: } \sum_{j=0}^n x_{ij} = 2 \quad 1 \leq i \leq n \quad (\text{path enters and leaves})$$

$$x_{ji} = x_{ij}, \text{ binary}$$

$c_{ij} = c_{ji}$  = distance from city i to city j  
(assuming **symmetric version**)

$x_{ij}$  if tour goes from i to j or j to i, and 0 otherwise

### Anything missing?

15-853

Page8

## Traveling Salesman Problem

**minimize:** 
$$\sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij}$$

**subject to:** 
$$\sum_{j=0}^n x_{ij} = 1 \quad 1 \leq i \leq n \quad (\text{out degrees} = 1)$$

$$\sum_{i=0}^n x_{ij} = 1 \quad 1 \leq j \leq n \quad (\text{in degrees} = 1)$$

$$t_i - t_j + nx_{ij} \leq n-1 \quad 2 \leq i, j \leq n \quad (??)$$

$c_{ij}$  = distance from city  $i$  to city  $j$   
 $x_{ij}$  = 1 if tour visits  $i$  then  $j$ , and 0 otherwise (binary)  
 $t_i$  = arbitrary real numbers we need to solve for

15-853

Page9

## Traveling Salesman Problem

The last set of constraints:  $t_i - t_j + nx_{ij} \leq n-1 \quad 2 \leq i, j \leq n$  prevents "subtours":



Consider a cycle that goes from some node 4 to 5,  
 $t_4 - t_5 + nx_{4,5} \leq n-1 \Rightarrow t_5 \geq t_4 + 1$   
 Similarly  $t$  has to increase by 1 along each edge of the cycle that does not include vertex 1.

Therefore, for a tour of length  $m$  that does not go through vertex 1,  $t_4, t_4 + m$ , a contradiction.

**Every cycle must go through vertex 1.**

Together with other constraints, it forces one cycle.

15-853

Page10

## Traveling Salesman Problem

Many "Real World" **applications** based on the TSP.

- They typically involve more involved constraints
- Not just routing type problems.

Consider a drug company with  $k$  drugs they can make at a lab. They can only make the drugs one at a time. The cost of converting the equipment from making drug  $i$  to drug  $j$  is  $c_{ij}$

Current best solutions are based on IP

- Applegate, Bixby, et. al., have solutions for more than 15K cities in Germany  
 > 150,000 CPU hours (**more info**)
- Involves "branch-and-bound" and "cutting planes"

15-853

Page11

## Set Covering Problem

Find cheapest sets that cover all elements



Courtesy: Darmstadt University of Technology

15-853

Page12



## Constraints Expressible with IP

### Logical constraints ( $x_1, x_2$ binary):

Either  $x_1$  or  $x_2 \Rightarrow x_1 + x_2 = 1$

If  $x_1$  then  $x_2 \Rightarrow x_1 - x_2 \leq 0$

### Combining constraints:

Either  $a_1x \leq b_1$  or  $a_2x \leq b_2 \Rightarrow$

$$\begin{array}{rcl} a_1x - My & \leq & b_1 \\ a_2x - M(1-y) & \leq & b_2 \end{array}$$

$y$  is a binary variable,  $M$  needs to be "large",  
 $a_1, a_2$ , and  $x$  can be vectors

15-853

Page17

## Algorithms

1. Use a linear program
  - round to integer solution (what if not feasible?)
2. Search
  - Branch and bound (integer ranges)
  - Implicit (0-1 variables)
3. Cutting planes
  - Many variants

15-853

Page18

## Important Properties

- LP solution is an upper bound on IP solution (assuming maximization)
- If LP is infeasible then IP is infeasible
- If LP solution is integral (all variables have integer values), then it is the IP solution.

15-853

Page19

## Linear Programming Solution

1. Some LP problems will always have integer solutions
  - transportation problem
  - assignment problem
  - min-cost network flowThese are problems with a unimodular matrix  $A$ . (unimodular matrices have  $\det(A) = 1$ ).
2. Solve as linear program and round. Can violate constraints, and be non-optimal. Works OK if
  - integer variables take on large values
  - accuracy of constraints is questionable

15-853

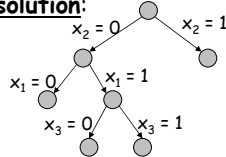
Page20

## Branch and Bound

Lets first consider **0-1** programs.

**Exponential solution:** try all  $\{0,1\}^n$

**Branch-and-bound solution:**



Traverse tree keeping current best solution.

If it can be shown that a subtree never improves on the current solution, **or** is infeasible, **prune** it.

15-853

Page21

## Zero-One Branch and Bound

**minimize:**  $z = c^T x$ , subject to:  $Ax \leq b, x \geq 0, x \in \{0,1\}^n$

Assume all elements of  $c$  are non-negative

**function**  $ZO_r(A, b, c, x_f, z^*)$

//  $x_f$ : a fixed setting for a subset of the variables

//  $z^*$  is the cost of current best solution

$x = x_f + 0$  // set unconstrained variables to zero

**if**  $(cx \geq z^*)$  **or** (no feasible completion of  $x_f$ ) **return**  $z^*$

**if**  $(Ax \leq b)$  **then return**  $cx$

pick an unconstrained variable  $x_i$  from  $x$

$z_0^* = ZO_r(A, b, x_f \cup \{x_i = 0\}, c, z^*)$

$z_1^* = ZO_r(A, b, x_f \cup \{x_i = 1\}, c, z_0^*)$

**return**  $z_1^*$

**function**  $ZO(A, b, c) = ZO_r(A, b, c, \emptyset, 1)$

Page22

## Zero-One Branch and Bound

**Checking for feasible completions:** check each constraint and find if minimum of left is greater than right.

**Example:**

$$x_f = \{x_1 = 1, x_3 = 0\}$$

and one of the constraints is

$$3x_1 + 2x_2 - x_3 + x_4 \leq 2$$

then

$$3 + 2x_2 - 0 + x_4 \leq 2$$

$$2x_2 + x_4 \leq -1$$

which is impossible.

15-853

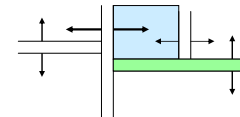
Page23

## Integer Branch and Bound

The zero-one version is sometimes called "implicit enumeration" since it might enumerate all possibilities.

An integer version **cannot** branch on all possible integer values for a variable. Even if the integer range is bounded, it is not practical.

Will "bound" by adding inequalities to split the two branches.



Since solutions are integral, each split can remove a strip of width 1

15-853

Page24

## Integer Branch and Bound

**maximize:**  $z = c^T x$ , subject to:  $Ax \leq b, x \geq 0, x \in \mathbb{Z}^n$

**function**  $IP_r(A_e, b_e, c, z^*)$

//  $A_e, b_e$  are  $A$  and  $b$  with additional constraints

//  $z^*$  is the cost of current best solution

$z, x, f = LP(A, b, c)$  //  $f$  indicates whether feasible

**if** not( $f$ ) or ( $z < z^*$ ) **return**  $z^*$

**if** (integer( $x$ )) **return**  $z$

pick a non-integer variable  $x_i'$  from  $x$

$z_1^* = IP(\text{extend } A_e, b_e \text{ with } x_i \leq \lfloor x_i' \rfloor, c, z^*)$

$z_2^* = IP(\text{extend } A_e, b_e \text{ with } -x_i \leq -\lceil x_i' \rceil, c, z_1^*)$

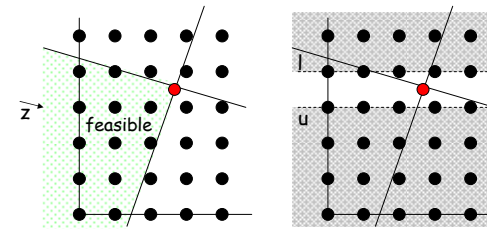
**return**  $z_g^*$

**function**  $IP(A, b, c) = IP_r(A, b, c, -1)$

15-853

Page25

## Example



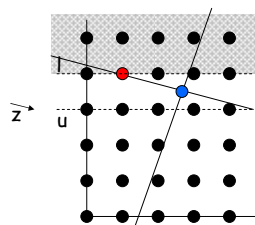
Find optimal solution.

Cut along  $y$  axis, and make two recursive calls

15-853

Page26

## Example



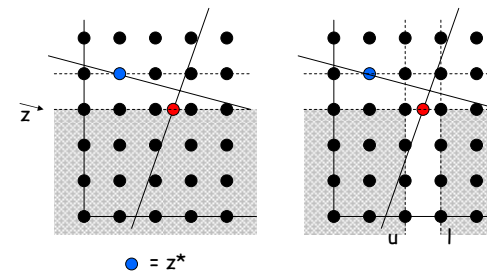
Find optimal solution.

Solution is integral, so return it as current best  $z^*$

15-853

Page27

## Example



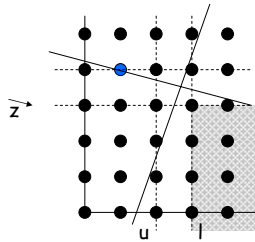
Find optimal solution. It is better than  $z^*$ .

Cut along  $x$  axis, and make two recursive calls

15-853

Page28

### Example

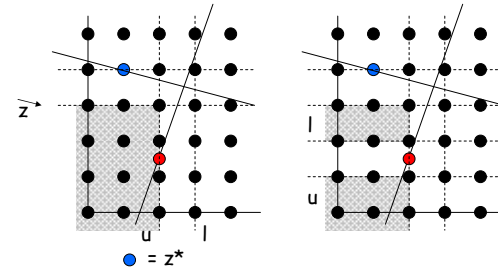


Infeasible, Return.

15-853

Page29

### Example

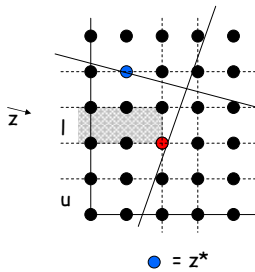


Find optimal solution. It is better than  $z^*$ .  
Cut along  $y$  axis, and make two recursive calls

15-853

Page30

### Example

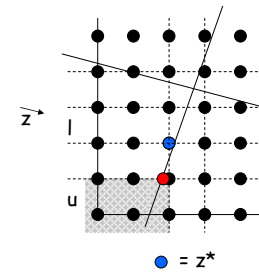


Find optimal solution. Solution is integral and better than  $z^*$ . Return as new  $z^*$ .

15-853

Page31

### Example



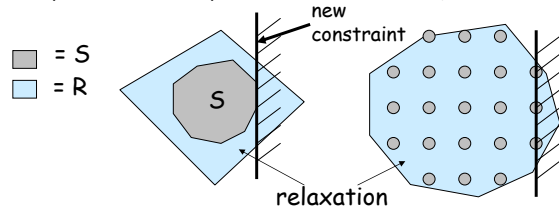
Find optimal solution. Not as good as  $z^*$ , return.

15-853

Page32

## Cutting Plane

The idea is to start with a "relaxation"  $R$  of the problem and then add constraints on the fly to find an actual feasible solution in  $S$ .



**Example 1**

**Example 2**

A "linear" relaxation

15-853

Page33

## Cutting Plane: general algorithm

**minimize:**  $z = c^T x$ , subject to  $x \in S$

**function**  $CP(R, c)$

//  $R$  a relaxed set of constraints  $Ax \leq b$   
s.t.  $S \frac{1}{2}$  polytope( $Ax \leq b$ )

**repeat:**

$x = LP(R, c)$

**if**  $x \notin S$  **return**  $x$

find an inequality  $r$  satisfied by  $S$ ,  
but violated by  $x$  ( $r$  **separates**  $x$  from  $S$ )

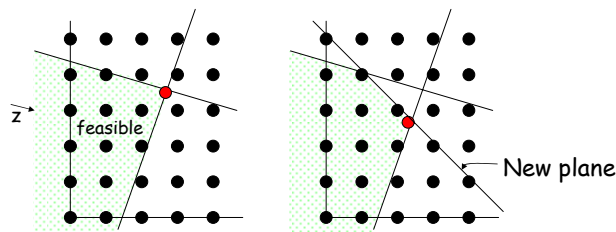
$R = R \cup \{r\}$

Can add multiple inequalities on each iteration

15-853

Page34

## Cutting Plane



Note that we are removing a corner, and no integer solutions are being excluded.

15-853

Page35

## Picking the Plane

**Method 1:** Gomory cuts (1958)

- Cuts are generated from the LP Tableau
- Each row defines a potential cut
- Guaranteed to converge on solution
- General purpose, but inefficient in practice

**Method 2:** problem specific cuts (templates)

- Consider the problem at hand and generate cuts based on its structure
- A **template** is a problem specific **set** of cuts (probably of exponential size) which  $S$  satisfies. Each round picks a cut from this set.

15-853

Page36

## Templates for the TSP problem

We consider some example templates used in solutions of the Traveling Salesman Problem.

Recall that  $x_{ij}$  indicates the edge from  $v_i$  to  $v_j$

Assume the symmetric TSP:  $x_{ij} = x_{ji}$

Consider subsets of vertices  $U \subseteq \frac{1}{2} V$ .

**define:**  $\delta_x(U) = \sum x_{ij}, v_i \in U, v_j \in V-U$   
(i.e. the number of times path crosses into/out of  $U$ )

**Degree Constraints:**  $\delta_x(\{v_i\}) = 2, 1 \leq i \leq n$

**Subtour Constraints:**  $\delta_x(U) \geq 2, U \subseteq \frac{1}{2} V$  **A template**

There are an exponential number of these

15-853

Page37

## Templates for the TSP problem

A set of constraints (a template) is **facet-defining** for  $S$  if each constraint is on a facet of the convex hull of  $S$ .

We would like templates which are facet defining since, intuitively, they will more quickly constrain us to the boundary of  $S$ .

The subtour template is facet defining.

In practice the subtour inequalities are not enough to constrain the solution to integral solutions.

Are there other sets of facet defining constraints?

15-853

Page38

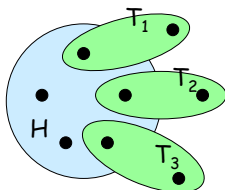
## Templates for the TSP problem

**Blossom inequalities** (Edmonds 1965):

Defined by  $H$  (handle) and  $T_1, \dots, T_k$  (teeth) satisfying:

$k \geq 3$  and odd,  $|T_i| = 2$

$T_i \cap T_j = \emptyset, |H \cap T_i| = 1, |T_i \setminus H| = 1$



$$\delta_x(H) + \sum_{i=1}^k \delta_x(T_i) \geq 3k + 1$$

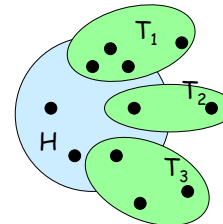
15-853

Page39

## Templates for the TSP problem

**Comb inequalities** (Grotschel 1977)

Just generalizes  $T_i$  to be any size. At least one element of each  $T$  has to be in and out of  $H$ .



$$\delta_x(H) + \sum_{i=1}^k \delta_x(T_i) \geq 3k + 1$$

15-853

Page40

## The art of Templates

Picking the right set of templates, and applying them in the right way is the art of solving NP-hard problems with integer programming.

Different problems have different templates.

One needs to find good algorithms for selecting a member of a template that separates  $x$  from  $S$  (can be quite complicated on its own).

Cutting planes often used in conjunction with branch and bound.

Can interleave template cuts with Gomory cuts (e.g. use Gomory cuts when the set of template cuts "dries out").

15-853

Page41

## Practical Developments

- **Good formulations**, heuristics and theory  
Goal: to get LP solution as close as possible to IP solution  
Disaggregation, adding constraints (cuts)
- **Preprocessing**  
Automatic methods for reformulation  
Some interesting graph theory is involved
- **Cut generation** (branch-and-cut)  
Add cuts during the branch-and-bound
- **Column generation**  
Improve formulation by introducing an exponential number of variables.

15-853

Page42