

620-362
Applied Operations Research

“Good” Model vs “Bad” Model

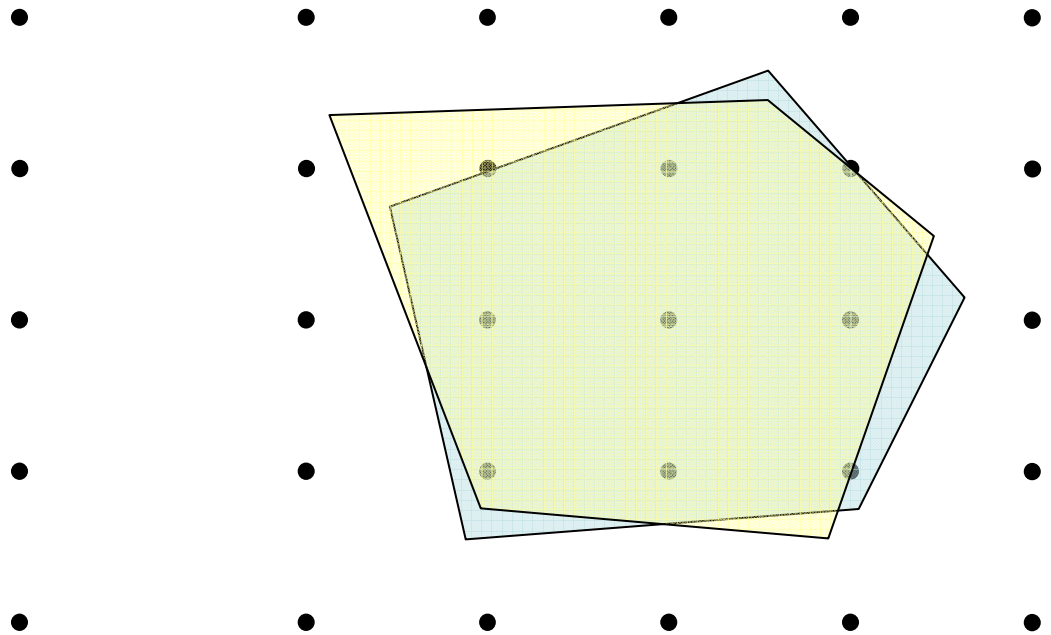
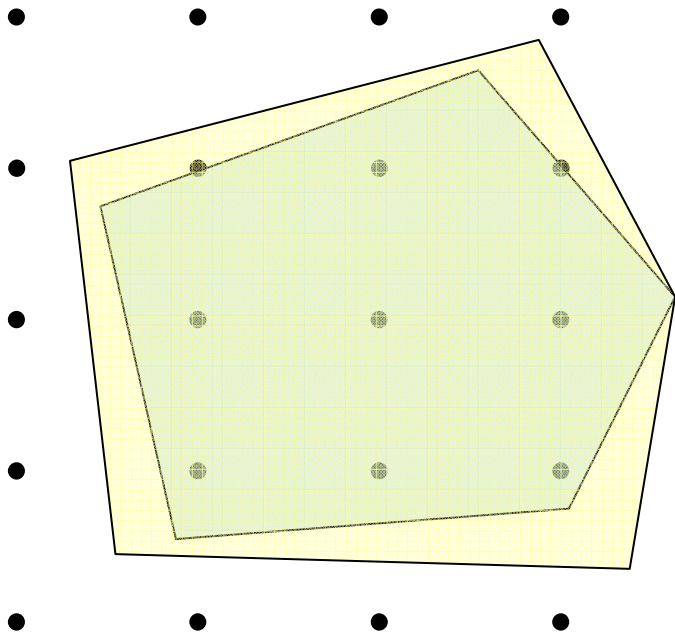
Department of Mathematics and Statistics
The University of Melbourne

This presentation has been made in accordance with the provisions of Part VB of the copyright act for the teaching purposes of the University.

*For use of students of the University of Melbourne enrolled in the subject 620-362.
Copyright©2008 by Heng-Soon Gan*

Some contents of this presentation are adapted from year 2005 course notes for 620-362 Applied Operations Research, Department of Mathematics and Statistics, The University of Melbourne (compiled by Prof Natasha Boland and Dr Renata Sotirov)

Which model is better?



A Formal Definition

Consider two models for the same IP

$$M_1: \quad A^1x \leq b^1, \quad x \in Z_+^n$$

$$M_2: \quad A^2x \leq b^2, \quad x \in Z_+^n$$

where its integer feasible region

$$S = \{x \in Z_+^n : A^1x \leq b^1\} = \{x \in Z_+^n : A^2x \leq b^2\}$$

Let R_1 and R_2 be the LP-relaxations of M_1 and M_2 respectively, i.e. let

$$R_1 = \{x \in R_+^n : A^1x \leq b^1\}$$

$$R_2 = \{x \in R_+^n : A^2x \leq b^2\}$$

A Formal Definition

- Model M_1 is at least as good as model M_2 if

$$R_1 \subseteq R_2$$

- Model M_1 is better than model M_2 if

$$R_1 \subset R_2$$

- Model M_1 is better than model M_2 with respect to objective $\mathbf{c}x$ if

$$\max_{x \in R_1} \mathbf{c}x < \max_{x \in R_2} \mathbf{c}x$$

LP Relaxation

Compare

$$y_1 + y_2 \leq 2x$$

VS

$$y_1 \leq x$$

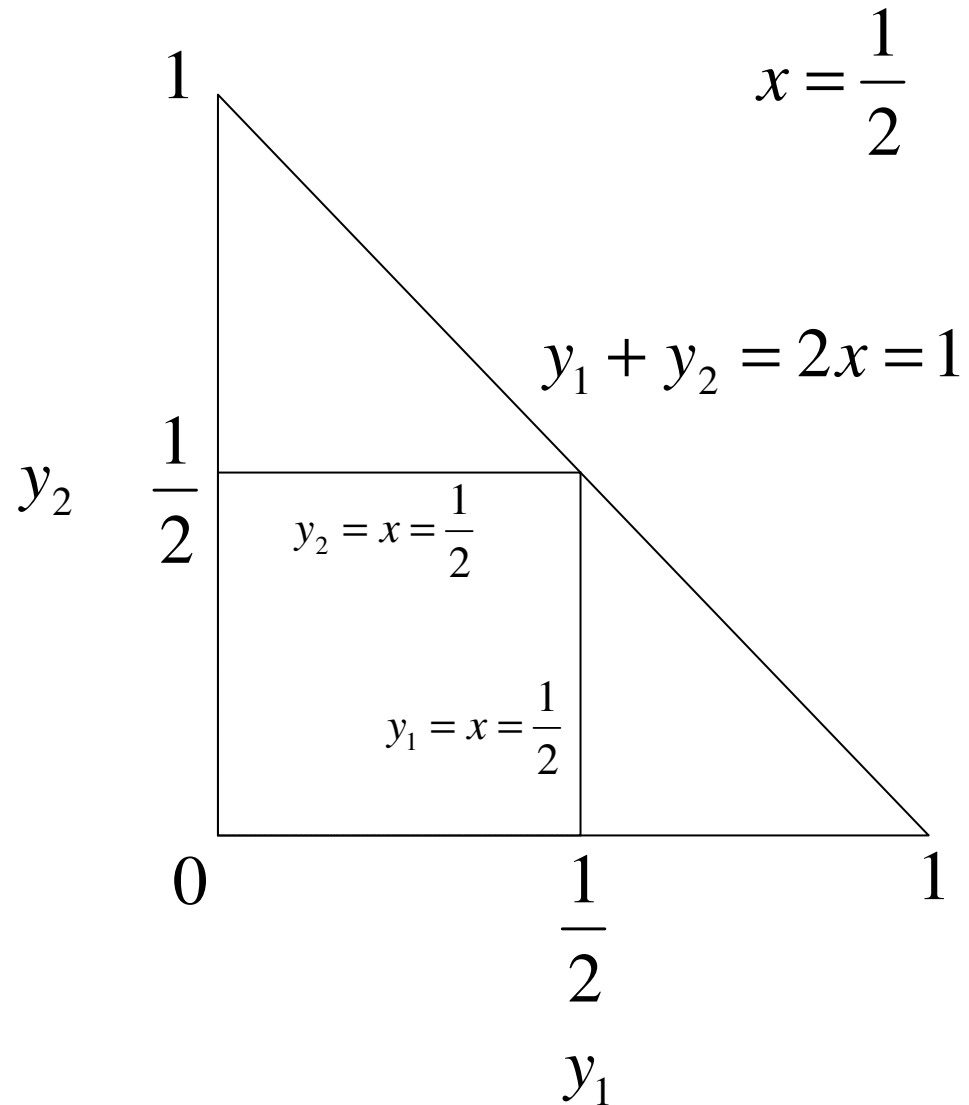
$$y_2 \leq x$$

e.g.

$$x = 1/2$$

$$y_1 = 1/3$$

$$y_2 = 2/3$$



Uncapacitated Facility Location (UFL)

A company is considering potential sites at which to construct new facilities.

Given:

I = set of n candidate sites

J = set of m clients

c_{ij} = cost of serving client j from site i

f_i = cost of operating a facility at site i

Variables:

y_{ij} = fraction service to client j from site i

x_i = 1, if a facility is constructed at site i ; 0, otherwise.

Uncapacitated Facility Location (UFL)

$$\begin{aligned} \min \quad & \sum_{i \in I, j \in J} c_{ij} y_{ij} + \sum_{i \in I} f_i x_i \\ \text{s.t.} \quad & \sum_{i \in I} y_{ij} = 1, \quad \forall j \in J \\ & \sum_{j \in J} y_{ij} \leq m x_i, \quad \forall i \in I \\ & y_{ij} \geq 0, \quad \forall i \in I, j \in J \\ & x_i \in \{0,1\}, \quad \forall i \in I \end{aligned}$$

Number of constraints =

$$m + n$$

Implications from second constraint: for all $j \in J$

$$x_i = 0 \quad \Rightarrow \quad y_{ij} = 0$$

Disaggregated model for UFL

$$\begin{aligned} \min \quad & \sum_{i \in I, j \in J} c_{ij} y_{ij} + \sum_{i \in I} f_i x_i \\ \text{s.t.} \quad & \sum_{i \in I} y_{ij} = 1, \quad \forall j \in J \\ & y_{ij} \leq x_i, \quad \forall i \in I, j \in J \\ & y_{ij} \geq 0, \quad \forall i \in I, j \in J \\ & x_i \in \{0,1\}, \quad \forall i \in I \end{aligned}$$

Number of constraints:

$$m + mn$$

Too many constraints \Rightarrow only add violated ones

UFL – Mosel Code

```
model "UFL"  
  uses "mmxprs"; !gain access to the Xpress-Optimizer solver  
  
  parameters  
    INFILE= "ufl.txt"  
  end-parameters  
  
  !sample declarations section  
  declarations  
    SITES: set of integer  
    CLIENTS: set of integer  
  
    ServiceCost: array(SITES,CLIENTS) of real  
    OperatingCost: array(SITES) of real  
  end-declarations  
  
  initialisations from INFILE  
    ServiceCost  
    OperatingCost  
  end-initialisations  
  
  finalize(SITES)  
  finalize(SITES)  
  
  declarations  
    y: array(SITES,CLIENTS) of mpvar !fraction of service to client j from site i  
    x: array(SITES) of mpvar !1 if facility is constructed at site i; 0 otherwise  
  end-declarations  
  
  forall(j in CLIENTS) do  
    sum(i in SITES) y(i,j) = 1  
  end-do  
  
  forall(i in SITES) do  
    sum(j in CLIENTS) y(i,j) <= getsize(CLIENTS) * x(i)  
  
    x(i) is_binary  
  end-do  
  
  Obj:= sum(i in SITES, j in CLIENTS) ServiceCost(i,j) * y(i,j) + sum(i in SITES) OperatingCost(i) * x(i)  
  
  minimise(Obj)  
  
end-model
```

Disaggregated UFL – Mosel Code

```
model "UFL-Disagg"
  uses "mumxprs"; !gain access to the Xpress-Optimizer solver

  parameters
    INFILE= "ufl.txt"
  end-parameters

  !sample declarations section
  declarations
    SITES: set of integer
    CLIENTS: set of integer

    ServiceCost: array(SITES,CLIENTS) of real
    OperatingCost: array(SITES) of real
  end-declarations

  initialisations from INFILE
    ServiceCost
    OperatingCost
  end-initialisations

  finalize(SITES)
  finalize(SITES)

  declarations
    y: array(SITES,CLIENTS) of mpvar      !fraction of service to client j from site i
    x: array(SITES) of mpvar              !1 if facility is constructed at site i; 0 otherwise
  end-declarations

  forall(j in CLIENTS) do
    sum(i in SITES) y(i,j) = 1
  end-do

  forall(i in SITES, j in CLIENTS) do
    y(i,j) <= x(i)
  end-do

  forall(i in SITES) do
    x(i) is_binary
  end-do

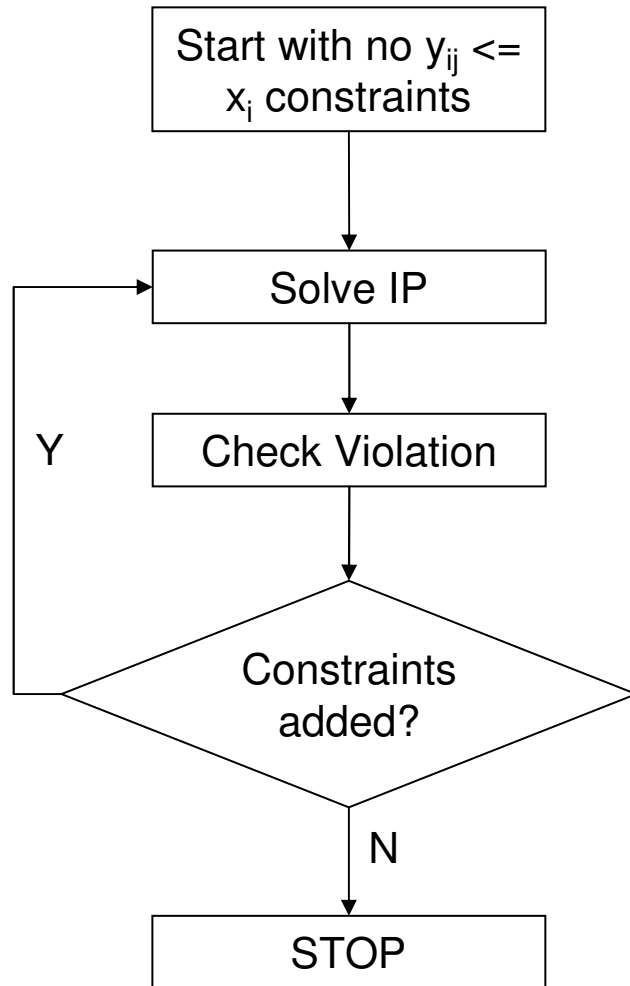
  Obj:= sum(i in SITES, j in CLIENTS) ServiceCost(i,j) * y(i,j) + sum(i in SITES) OperatingCost(i) * x(i)

  minimise(Obj)

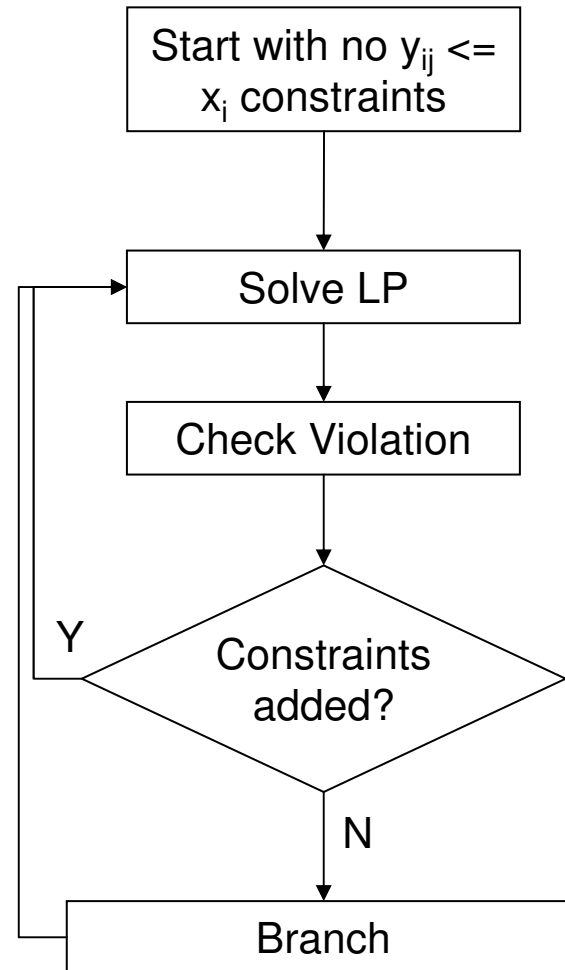
end-model
```

Disaggregated UFL (on-the-fly)

Scheme 1



Scheme 2



Disaggregated UFL (Scheme 1)

Mosel Code

UFL – Computational Example

20 sites
40 clients

840 variables

	UFL	UFL - Disaggregated Model	UFL - on-the-fly Scheme 1 (final iteration)
# constraints	60	820	375
LP value (root)	103.198	300.579	300.302
Optimal MIP value	313.64	313.64	313.64
# nodes searched	55	11	17

- Solution time?
- Scheme 1: only need 414 variables (presolve)
 - column generation?

Capacitated Facility Location

A company is considering potential sites at which to construct new facilities. Each site has a maximum capacity which it can support. At which sites should it locate facilities and how much should each facility supply to each client?

Given:

I = set of candidate sites

J = set of clients

c_{ij} = cost per unit of supplying client j from site i

f_i = cost of operating a facility at site i

b_i = capacity of site i

d_j = demand of client j

Variables:

y_{ij} = units supplied to client j from site i

$x_i = 1$, if a facility is constructed at site i ; 0, otherwise

Capacitated Facility Location

$$\text{Minimise } \sum_{i \in I} f_i x_i + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$$

s.t.

$$\sum_{i \in I} y_{ij} = d_j, \quad \forall j \in J$$

$$\sum_{j \in J} y_{ij} \leq b_i x_i, \quad \forall i \in I$$

$$x_i \in \{0,1\}, \quad \forall i \in I$$

$$y_{ij} \geq 0, \quad \forall i \in I, j \in J$$

From constraint 2: $x_i = 0 \Rightarrow y_{ij} = 0$

Linear model: $y_{ij} \leq Mx_i$ for M large

For good model, choose M as small as possible, e.g.
 $M = \min(b_i, d_j)$

Single Facility Supply Variant

Demand of each client must be met **completely** by **one** facility.

$$\text{Minimise } \sum_{i \in I} f_i x_i + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$$

s.t.

$$\sum_{i \in I} y_{ij} = 1, \quad \forall j \in J$$

$$\sum_{j \in J} d_j y_{ij} \leq b_i x_i, \quad \forall i \in I$$

$$x_i \in \{0,1\}, \quad \forall i \in I$$

$$y_{ij} \in \{0,1\}, \quad \forall i \in I, j \in J$$

Lot-Sizing Problem

To develop a production plan such that the total cost is minimised. Total cost consists of inventory holding, production setup and production costs.

Given:

$1, \dots, T$ time periods

d_t = demand in period t

p_t = production cost in period t

h_t = holding cost in period t

c_t = set up cost in period t

Lot-Sizing Problem – Formulation 1

Variables:

y_t = production in period t

s_t = inventory at the end of period t

$x_t = 1$, if set up in period t ; 0, otherwise.

$$\min \sum_{t=1}^T (p_t y_t + h_t s_t + c_t x_t)$$

s.t.

$$y_1 = d_1 + s_1$$

$$s_{t-1} + y_t = d_t + s_t, \quad t = 2, \dots, T-1$$

$$s_{T-1} + y_T = d_T$$

$$y_t \leq M x_t, \quad t = 1, \dots, T$$

$$y_t, s_t \geq 0, x_t \in \{0,1\}, \quad t = 1, \dots, T$$

$$M = \sum_{k=1}^T d_k$$

A tighter bound :

$$M_t = \sum_{k=t}^T d_k$$

Lot-Sizing Problem – Formulation 2

Variables:

q_{it} = production in period i for demand in period t , ($i \leq t$)

x_i = 1, if set up in period i ; 0, otherwise

$$\min \sum_{t=1}^T \left[\left[\sum_{i=1}^t \left(p_i + \sum_{k=i}^{t-1} h_k \right) q_{it} \right] + c_t x_t \right]$$

s.t.

$$\sum_{i=1}^t q_{it} = d_t, \quad t = 1, \dots, T$$

$$q_{it} \leq d_t x_i, \quad i = 1, \dots, T, t = 1, \dots, T$$

$$q_{it} \geq 0, \quad i = 1, \dots, T, t = 1, \dots, T$$

$$x_i \in \{0,1\}, \quad i = 1, \dots, T$$

Lot-Sizing Problem – Computational Example

40 time periods

	Formulation 1	Formulation 2
# constraints	80	859
# variables	120	860
LP value (root)	117,268	124,258
Optimal MIP value	124,258	124,258

Lot-Sizing Problem – Formulation 2

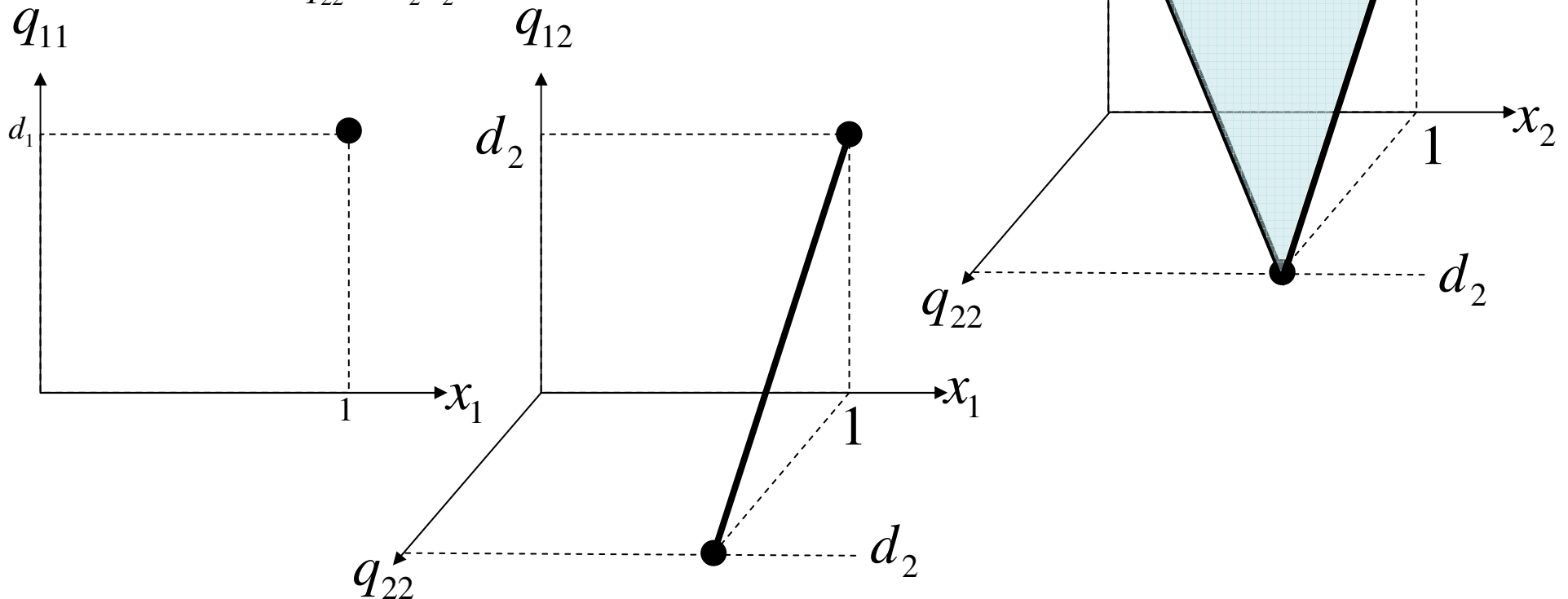
For $t = 1, 2$: $q_{11} = d_1$

$$q_{12} + q_{22} = d_2$$

$$q_{11} \leq d_1 x_1 \quad \Rightarrow \quad x_1 = 1 \Rightarrow q_{12} \leq d_2$$

$$q_{12} \leq d_2 x_1$$

$$q_{22} \leq d_2 x_2$$



Extreme points of this formulation are integer points!! (for integer variables)

Set Covering & Partitioning Models

- airline crew scheduling
- bus crew scheduling
- train driver rostering
- vehicle routing
- binary cutting stock

For all of these problems there is more than one formulation that seems reasonable.

In all cases, a set covering or partitioning model performs much better than the alternatives.

Set covering/partitioning models can be especially useful for eliminating or reducing *symmetry*, which causes problems for the branch-and-bound algorithm.

Human Resource Planning (HRP)

Tasks to be performed: 1, 2, ..., n

h_i = time required for task i (hours)

H = max number of hours worker can work in a day

What is the minimum workforce required?

An “assignment” model...

$z_{ij} = 1$, task i is done by worker j; 0, otherwise

$w_j = 1$, worker j used; 0, otherwise

forall all $j = 1, 2, \dots, m$ where

m = upper bound on number of workers required

(Note: obviously $m \leq n$)

HRP – Assignment Model

$$\min \sum_{j=1}^m w_j$$

s.t.

$$\sum_{j=1}^m z_{ij} = 1, \quad \forall i = 1, \dots, n$$

$$\sum_{i=1}^n h_i z_{ij} \leq H w_j, \quad \forall j = 1, \dots, m$$

$$z_{ij} \in \{0,1\}, \quad \forall i = 1, \dots, n; j = 1, \dots, m$$

$$w_j \in \{0,1\}, \quad \forall j = 1, \dots, m$$

...might have other restrictions, e.g. some jobs cannot be done by the same worker.

Any problems?

SYMMETRY – every worker is identical

(May also have symmetry if several jobs require same length of time.)

HRP – Assignment Model

How can we break symmetry?

One way is to include the following set of constraints:

$$\sum_{i=1}^n z_{i(j-1)} \geq \sum_{i=1}^n z_{ij}, \quad \forall j = 2, \dots, m$$

Considers only solutions where the number of jobs assigned to a worker is non-decreasing with increasing worker “index”.

- This is reasonable since any feasible solution obtained for this problem can be arranged in this order.

HRP – Set Partitioning Model

S is a valid task set if S is a set of tasks,

$S \subseteq \{1, \dots, n\}$ such that

$$\sum_{i \in S} h_i \leq H$$

J = set of all possible valid task sets

Variable:

$x_S = 1$, some worker does task set S ; 0, otherwise

HRP – Set Partitioning Model

$$\min \sum_{s \in J} x_s$$

s.t.

$$\sum_{s \in J: i \in S} x_s = 1, \quad \forall i = 1, \dots, n$$

$$x_s \in \{0,1\}, \quad \forall s \in J$$

How do we generate the set of task sets?

- heuristic: only consider a subset of J
 - create task sets from different “building heuristics”
 - partial enumeration
- exact: column generation (next lecture)

HRP – Set Partitioning Model

Consider the following methods to generate a subset of J :

- allocate the first unassigned task in list to first “available” worker
- sort task list in non-decreasing order, then allocate the first unassigned task in list to first “available” worker
- sort task list in non-increasing order, then allocate the first unassigned task in list to first “available” worker
- randomise order of tasks in list, then allocate the first unassigned task in list to first “available” worker
 - you can run this for n times

HRP – Computational example

Scenario		HRP (Assignment)	HRP (Assignment - SymBreak)	HRP (Partition)
40 tasks	# constraints	64	87	34
	# variables	984	984	399
	LP value	22.25	22.25	23
	MIP value	23	23	23
	# nodes (# simplex iterations)	471 (204)	1,079 (443)	1 (109)
50 tasks	# constraints	84	117	40
	# variables	1,734	1,734	382
	LP value	30.425	30.425	33
	MIP value	33 (~7.8%, 60s)	33	33
	# nodes (# simplex iterations)	63,023 (242)	141 (530)	1 (71)