



# Bicriteria Robustness versus Cost Optimisation in the Generation of Aircrew Pairings

David M Ryan and Matthias Ehrgott

Department of Engineering Science  
University of Auckland

{d.ryan,m.ehrgott}@auckland.ac.nz

Part 1: The Tour of Duty (ToD) problem of aircrew scheduling

Model, properties, computation, results

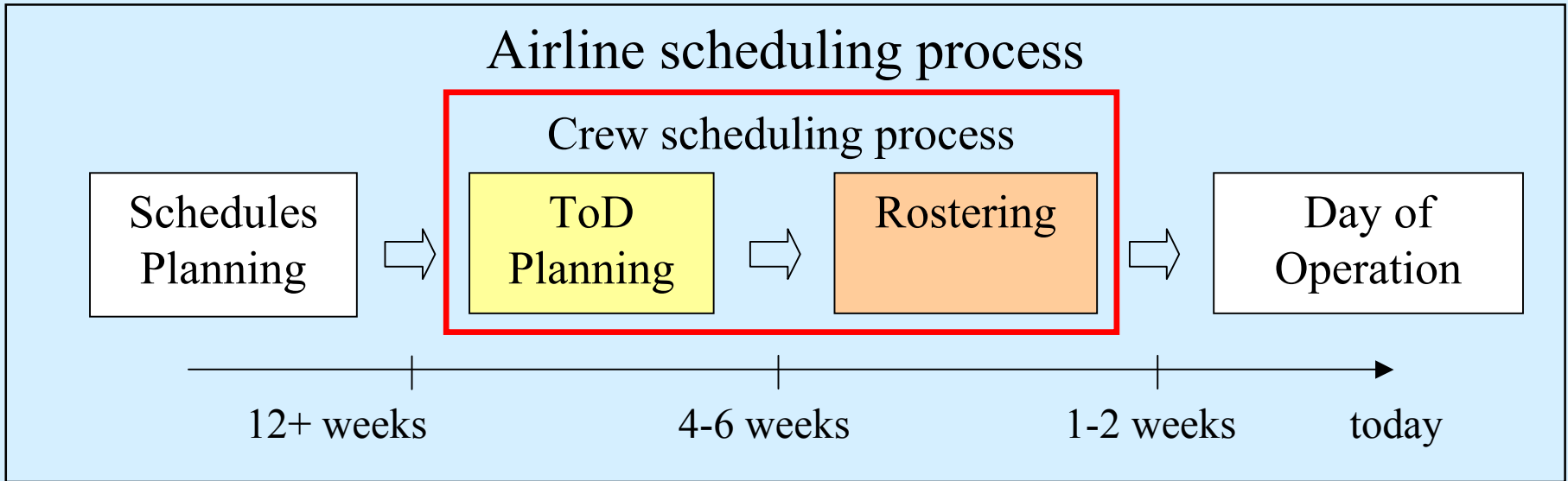
Minimum cost  $\equiv$  maximum potential disaster

Part 2: Non-robustness – the consequences of minimum cost

A non-robustness measure

The min cost / max robustness ToD problem

# The airline crew scheduling problems



The crew scheduling process has two phases:

- Tour of Duty (ToD) planning
  - ◆ Involves the construction of ToDs (or pairings)
- Rostering
  - ◆ Allocation of ToDs to crew members to create rosters

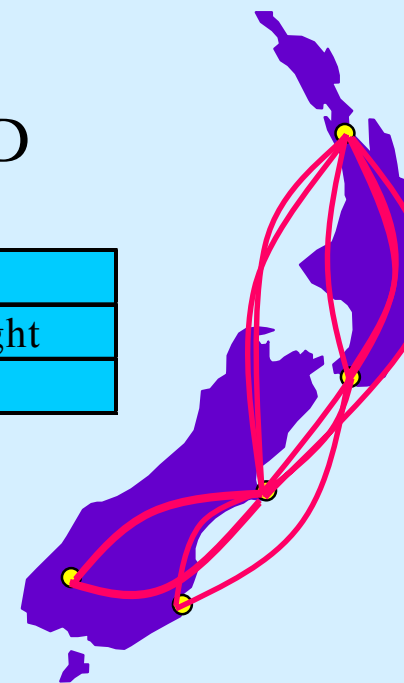
# Tour of Duty planning (Pairings)

## ■ Tour of Duty definition

- ◆ An alternating sequence of duty periods and rest periods
- ◆ Duty periods contain one or more flights
- ◆ Duty periods may include passengering (deadheading)
- ◆ Each ToD begins and ends at a crew base
- ◆ Generic - not associated with any crew member
- ◆ Many rules and regulations govern legality of a ToD

12:50 - 14:15	15:30 - 16:30	17:00 - 18:15	19:00 - 19:45	20:15 - 21:35	21:35 - 06:00	
AKL CHC	AKL WLG	WLG DUD	DUD CHC	CHC AKL	CHC	Overnight
NZ 527	NZ 445	NZ 445	NZ 554	NZ 554		

14:45 - 15:35	16:10 - 16:55	18:50 - 19:35	20:30 - 21:30
CHC ZQN	ZQN CHC	CHC WLG	WLG AKL
NZ 651	NZ 542	NZ 468	NZ 478

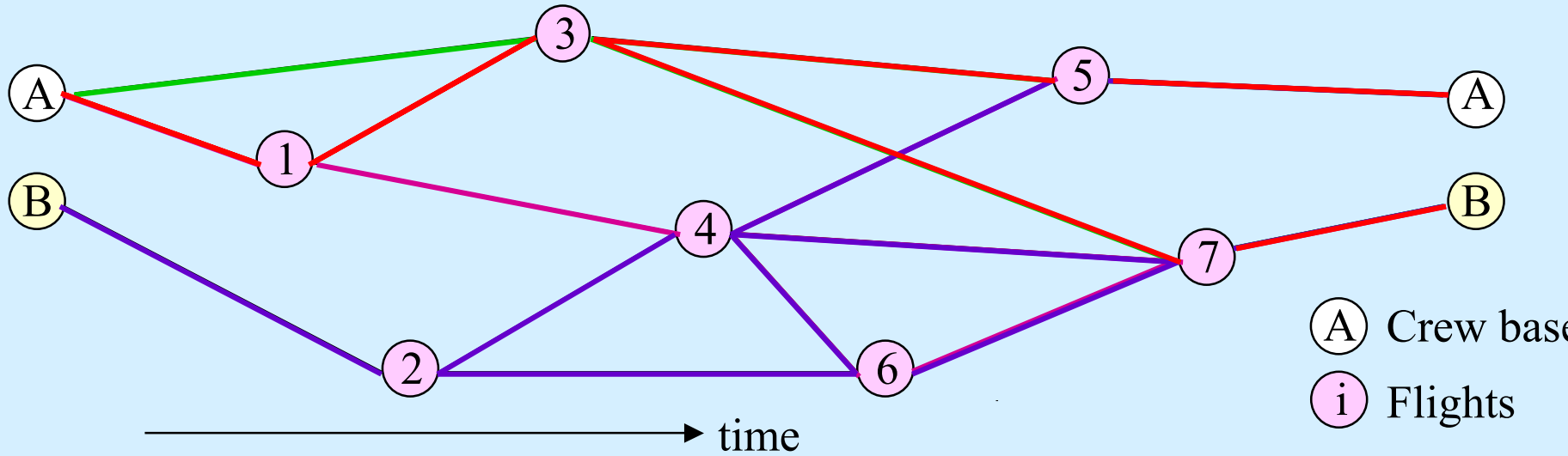


# The Set Partitioning optimisation model

SPP: Minimise  $z = \mathbf{c}^T \mathbf{x}$   
subject to  $\mathbf{A} \mathbf{x} = \mathbf{e} = (1,1,1,\dots,1)^T$   
and  $\mathbf{x} \in \{0,1\}^n$   
where  $c_j =$  the cost of variable  $j$   
 $a_{ij} = 1$  if column  $j$  covers row  $i$ , and  
 $0$  otherwise

- Sometimes in practice:
  - ◆ not all constraints are equalities - might have  $\leq$  (packing) or  $\geq$  (covering) constraints
  - ◆ some right-hand-sides might not be unit valued (but +ve integer)
- We call such models **Generalised Set Partitioning Models**
- Special forms of GSPP for ToD planning (and Rostering)

# The ToD model



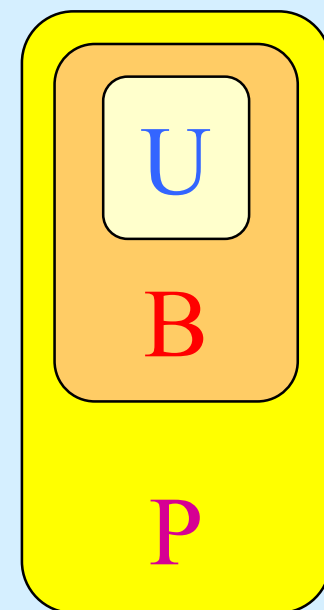
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	=	1
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	=	1
1	1	1	0	0	0	1	1	1	1	1	0	0	0	0	=	1
0	0	0	1	1	1	1	1	1	1	0	1	1	1	0	=	1
1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	=	1
0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	=	1
0	1	1	0	1	1	1	1	1	0	1	1	1	1	1	=	1
1	1	0	1	1	0	1	0	1	0	1	1	0	0	0	=	3
0	0	1	0	0	1	0	1	0	0	1	0	1	1	1	≤	6

# The Tour of Duty SPP model

- Rows (constraints) correspond to flights
  - ◆ Over one day or one week or some specified time period
  - ◆ Also have base constraints
    - limit number of ToDs or total ToD hours at a base
- Columns (variables) represent feasible ToDs
  - ◆ Each column must satisfy many rules (constraints) which are implicitly represented
- Up to 2000 constraints ( $m$ ), many millions (or billions!!) of variables ( $n$ )
- The objective is to minimise the total dollar cost
  - ◆ Costs include allowances, hotel and meal costs, ground transport costs, passengering costs, paid duty hours
  - ◆ The cost of each legal ToD can be calculated

# Natural integer properties of the SPP LP

- Consider the SPP LP relaxation polytope (ie  $\mathbf{x} \geq 0$ )
- Three classes of zero-one matrices are known to ensure that all extreme points (ie basic feasible solutions) are naturally integer
  - ◆ A is **totally unimodular**
    - Hoffman and Kruskal (1956) – applies more generally
    - Smallest class, difficult to check or use in practice
  - ◆ A is **balanced**
    - Berge (1972); relevant in ToD Planning
  - ◆ A is **perfect**
    - Padberg (1974); relevant in Rostering
- To solve such SPPs, just solve the LP relaxation
  - ◆ Unfortunately, the real world is neither balanced nor perfect!



# Balanced zero-one matrices

- A zero-one matrix is **balanced** iff it **does not contain any odd order 2-cycle submatrices** (ie odd order submatrix with row and column sums equal to 2).

1	0	1
1	1	0
0	1	1

2 from 3

$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
---------------	---------------	---------------

1	0	0	0	1
1	1	0	0	0
0	1	1	0	0
0	0	1	1	0
0	0	0	1	1

2 from 5

$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
---------------	---------------	---------------	---------------	---------------

etc

1	0	1	1
1	1	0	1
1	1	1	0
0	1	1	1

3 from 4

$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
---------------	---------------	---------------	---------------

# Subsequence

- The **subsequence count** for any row  $s$  of a zero-one  $m \times n$  matrix  $A$  is given by

$$[a_{sj} = 1, \quad a_{ij} = 0 \text{ for } s < i < t, \quad a_{tj} = 1]$$

# Subsequence

- The **subsequence count** for any row  $s$  of a zero-one  $m \times n$  matrix  $A$  is given by

$$SC(s) = |\{t \mid [a_{sj} = 1, a_{ij} = 0 \text{ for } s < i < t, a_{tj} = 1], j = 1, \dots, n\}|$$

$s \rightarrow$

1	1	1	1	1	1	1
0	0	0	0	0	0	0
0	0	0	0	1	1	0
1	0	1	0	1	0	1
0	0	0	0	0	0	1
0	1	1	0	0	1	0

$$SC(s) = 3$$

- Note:  $0 \leq SC(s) \leq m-s$
- A matrix  $A$  has **unique subsequence** iff  $SC(s) \leq 1$  for all  $s = 1, \dots, m$

# Subsequence

- The **subsequence count** for any row  $s$  of a zero-one  $m \times n$  matrix  $A$  is given by

$$SC(s) = |\{t \mid [a_{sj} = 1, a_{ij} = 0 \text{ for } s < i < t, a_{tj} = 1], j = 1, \dots, n\}|$$

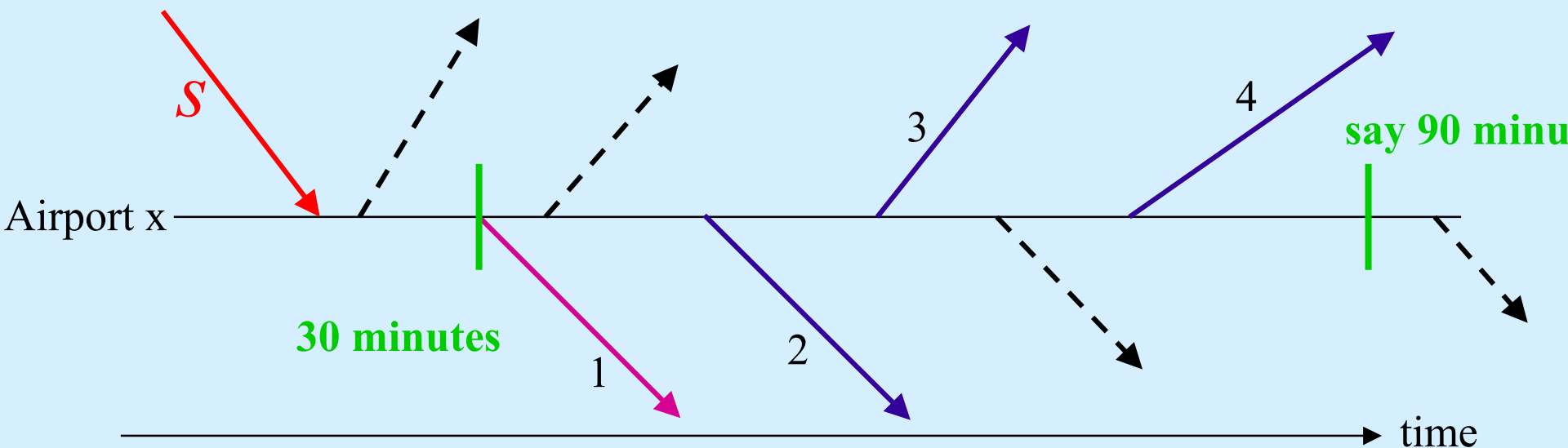
$s \rightarrow$

1	1	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	1	1	0
0	0	0	0	0	0	1	0	0	0
0	1	1	0	0	1	0	1	0	1

- Note:  $0 \leq SC(s) \leq m-s$
- A matrix  $A$  has **unique subsequence** iff  $SC(s) \leq 1$  for all  $s = 1, \dots, m$
- A matrix with unique subsequence is balanced (unimodular)

# The ToD problem and limited subsequence

- Order constraints (flights) by increasing departure time
  - ◆ a sequence of flights is represented by row order within the column
- The “next available” flights following flight  $s$  correspond to the “subsequences” for flight  $s$ 
  - ◆ limit the number of possible subsequent flights
  - ◆ some flights might not be included as subsequences



# The ToD problem and limited subsequence

- The first subsequence is usually to follow the aircraft
  - ◆ ie stay on the same aircraft and operate the following flight
    - least expensive and most robust
  - ◆ a significant proportion of subsequences in optimal ToD solutions follow the aircraft
- Shallow subsequences correspond to changing aircraft but with short idle (ground) time between flights within a ToD
  - ◆ attractive in terms of cost but unattractive in terms of robustness
- Deeper subsequences correspond to longer (more expensive) idle periods between flights within a ToD
  - ◆ unattractive in terms of cost but attractive in terms of robustness
- Limited subsequence SPPs have fewer variables, have “nice” fractions and are easier to solve

# Some computational techniques for SPP

- Solve the LP relaxation and then use Branch and Price
- Limited subsequence column generation
  - ◆ A priori matrix generation is not possible (too many variables)
  - ◆ We use dynamic column generation during the LP convergence
  - ◆ Solve resource constrained shortest path problems using dynamic programming on a “limited subsequence” activity network
  - ◆ ToD Planning
    - activity network based on flights
    - ToD generation by crew base (and possibly by day) – partial pricing
- Branch and Price strategies
  - ◆ Branch using the Ryan and Foster constraint branch
    - conventional variable branching does not work
    - branch on subsequent flight pairs

# Fractional Solutions

- Fractions can only occur because of the existence of odd-order 2-cycles (Berge (1972))

1	0	1
1	1	0
0	1	1

2 from 3

$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
---------------	---------------	---------------

1	0	0	0	1
1	1	0	0	0
0	1	1	0	0
0	0	1	1	0
0	0	0	1	1

2 from 5

$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
---------------	---------------	---------------	---------------	---------------

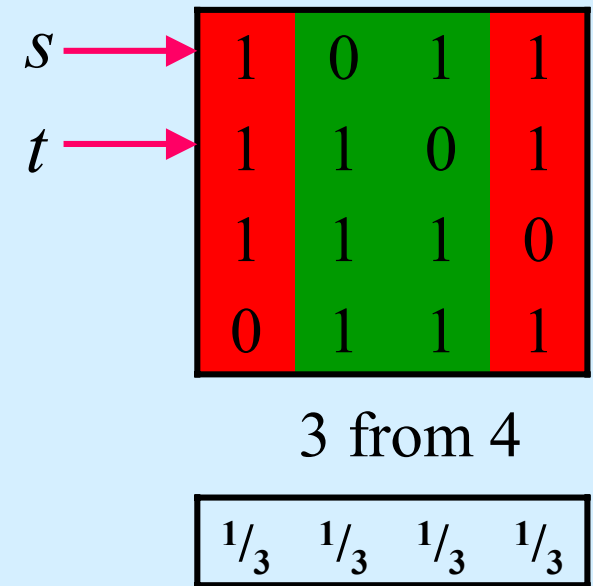
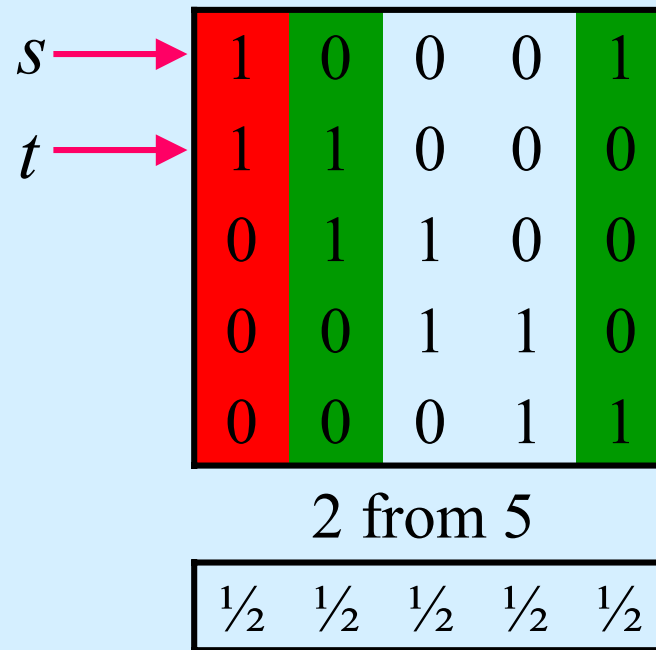
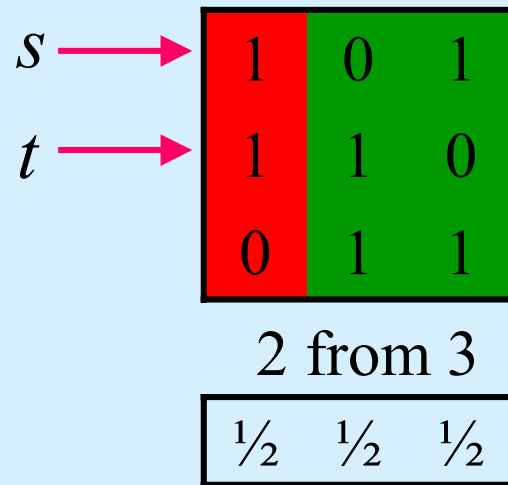
1	0	1	1
1	1	0	1
1	1	1	0
0	1	1	1

3 from 4

$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
---------------	---------------	---------------	---------------

# Constraint branch on flights $s$ and $t$

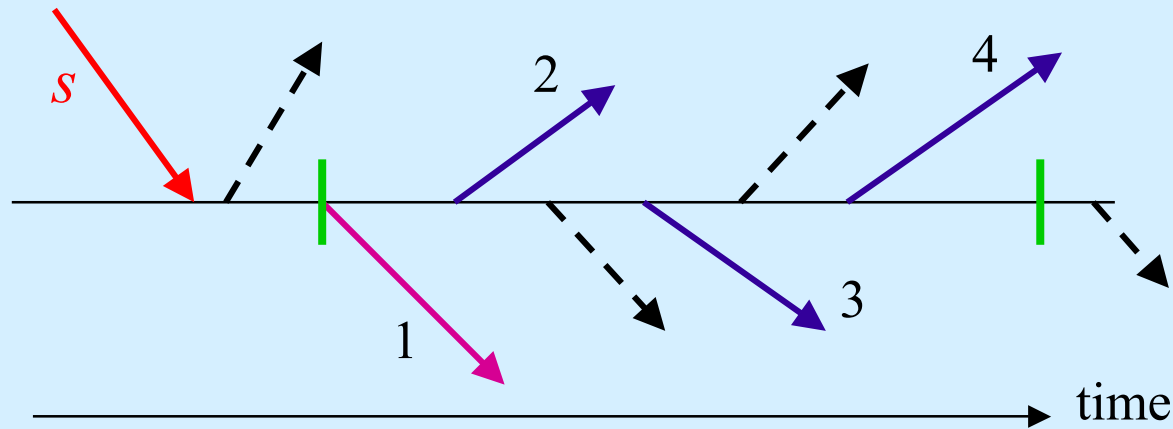
- Fractions can only occur because of the existence of odd-order 2-cycles (Berge (1972))



- Branch on pairs of constraints to eliminate odd-order 2-cycle (eg flight  $s$  is either **followed by  $t$  in same ToD** or **not**)

# Constraint branching strategies for ToD

- Constraint branch on successive flights within a ToD
  - ◆ for example: suppose  $(s,1)$  is 0.05,  $(s,2)$  is 0.9 and  $(s,4)$  is 0.05
  - ◆ choose  $s$  followed by 2 - this imposes a unique subsequence on  $s$



- Use the practical application to support the choice of  $s$  and  $t$ 
  - ◆ follow-the-aircraft branching preferred – most “robust”
  - ◆ provided fractional sum of variables using  $(s,t)$  is large (ie near 1)
  - ◆ avoid  $(s,t)$  pairs involving aircraft change and short ground time

# Summary of minimum cost TOD features

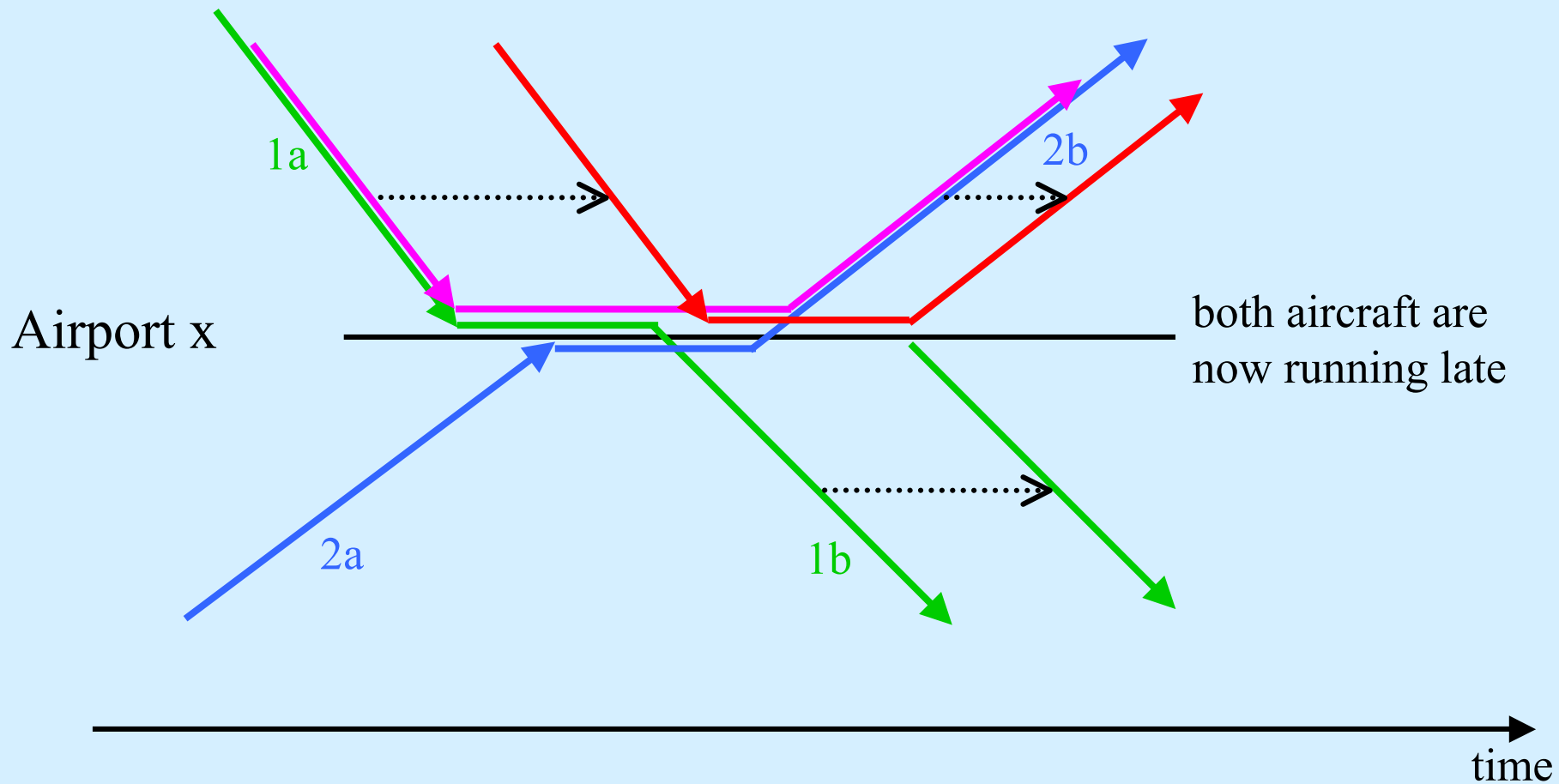
- Minimum cost ToDs have the following features:
  - ◆ Follow the aircraft most often – most cost efficient use of crew
    - Between 60% and 75% of optimal subsequences follow the aircraft
  - ◆ Aircraft changes are usually required within a ToD
    - Limit on duty time
    - Meal breaks
    - Crew-base imbalances – over-nighting in multi-day ToDs
    - ToDs must start and finish at the crew base
  - ◆ Aircraft changes usually involve short ground time
    - short ground time  $\equiv$  shallow subsequence  $\equiv$  lack of robustness
  - ◆ Small number of aircraft changes may involve longer ground time
    - long ground time  $\equiv$  deeper subsequence  $\equiv$  more robust

# The effect of late aircraft arrival

Aircraft 1 

Aircraft 2 

ToD 



# Tour of Duty planning (Pairings)

## ■ Tour of Duty definition

- ◆ A sequence of duty periods and rest periods
- ◆ Duty periods contain one or more flights
- ◆ Duty periods may include passengering (deadheading)
- ◆ Each ToD begins and ends at a crew base
- ◆ Generic - not associated with any crew member
- ◆ Many rules and regulations govern legality of a ToD

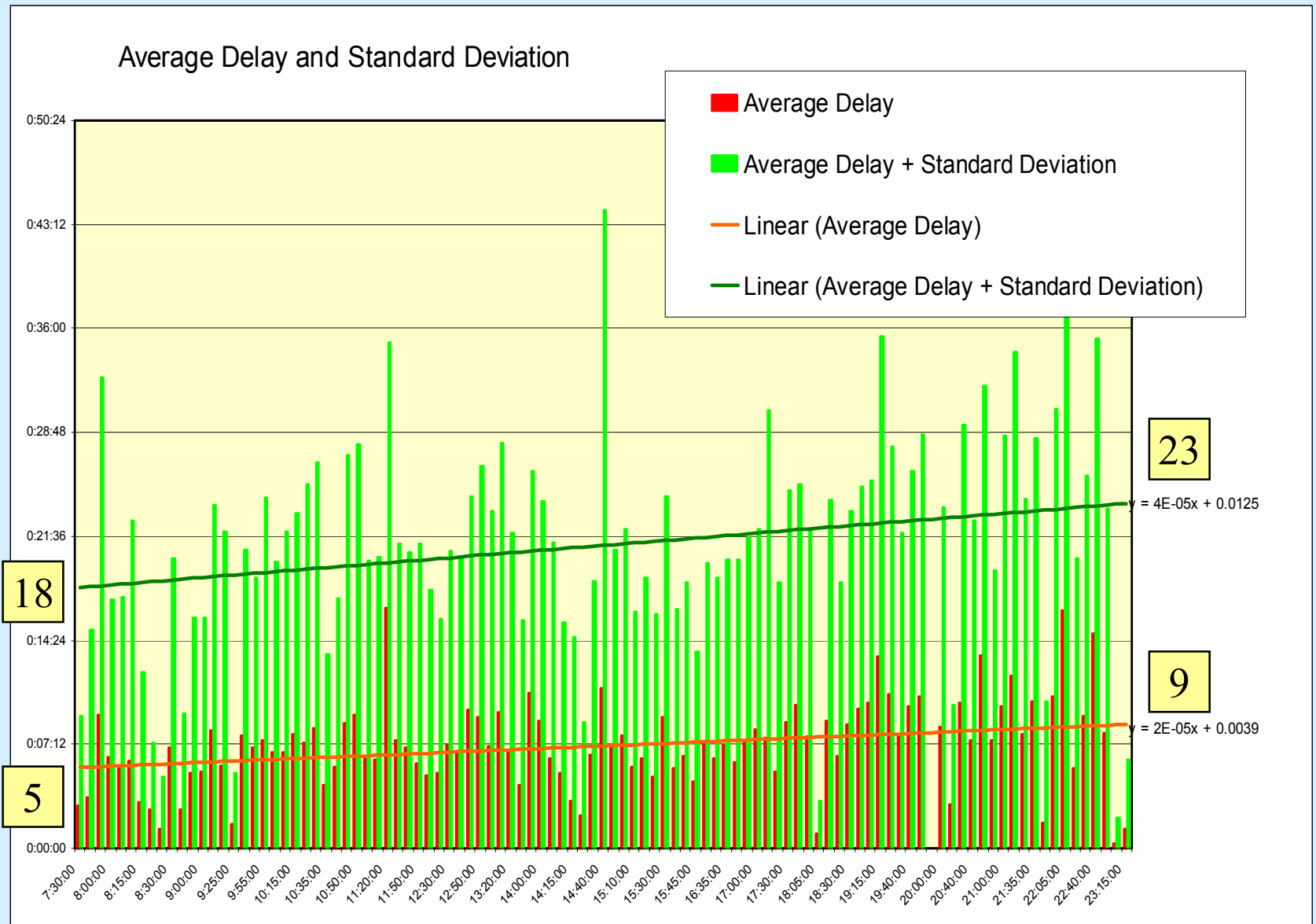
(35, 90, 16+hrs, 11

12:50 - 14:10	14:45 - 15:35	16:10 - 16:55	18:25 - 19:45	20:15 - 21:35	
AKL CHC	CHC ZQN	ZQN CHC	CHC AKL	AKL CHC	Overnight
NZ 527	NZ 651	NZ 542	NZ 548	NZ 559	

14:45 - 15:35	16:10 - 16:55	18:50 - 19:35	20:30 - 21:30
CHC ZQN	ZQN CHC	CHC WLG	WLG AKL
NZ 651	NZ 542	NZ 468	NZ 478

Aircraft change: delays propagate through schedule, not robust

# Delay data (over 46,000 flights)



# How can we introduce robustness into ToD build?

- Yen and Birge - *A Stochastic Programming Approach to the Airline Crew Scheduling Problem*
  - ◆ Formulate the ToD problem as a two-stage stochastic binary optimisation problem with recourse:
    - the first stage is based on the GSPP formulation
  - ◆ The recourse problem measures cost of delays
    - evaluation requires solution of one LP for each scenario
    - determines a so-called “switching cost” associated with aircraft change
    - “switching cost” is used to remove expensive aircraft changes in next GSPP
  - ◆ The results show:
    - small decrease in number of aircraft changes
    - longer ground time when changing aircraft within a ToD
  - ◆ A rather computationally expensive means of treating the problem
    - requires repeated solution of the deterministic GSPP

# How can we introduce robustness into ToD build?

- Schaefer, Johnson, Kleywegt and Nemhauser
  - *Airline Crew Scheduling under Uncertainty*
  - ◆ Use an “operational cost” instead of a “planned cost” for each ToD
  - ◆ Monte Carlo simulation to estimate “operational cost”
  - ◆ The simulation (using SimAir) focuses on one ToD at a time and ignores the interactive effects between ToDs
    - Schaefer et al suggest that this approach can be shown to account for approximately 90% of the total disruption costs
  - ◆ This approach is not compatible with column generation since the “operational cost” is only available after simulation
  - ◆ Penalizes ToDs with expensive “operational cost”
  - ◆ The observed effect of this approach is to produce ToDs with lower operational cost than those produced with planned cost

# How can we introduce robustness into ToD build?

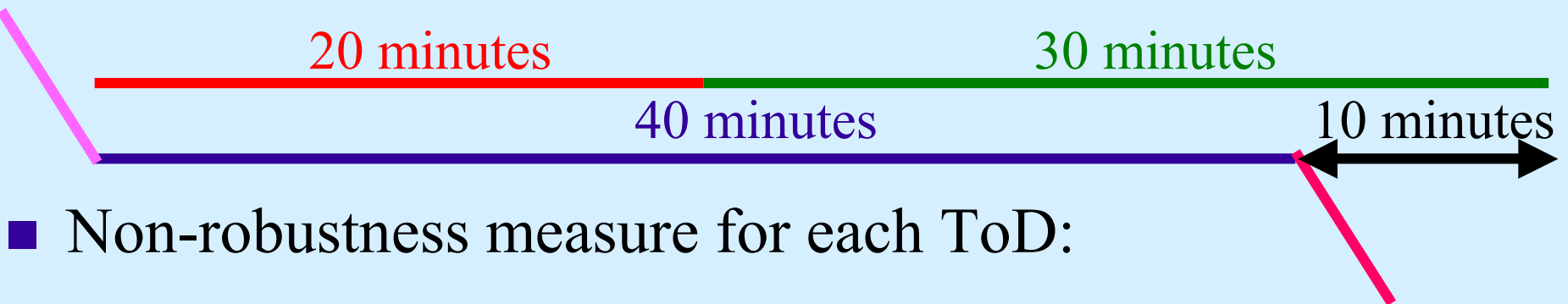
- Our approach is to develop a measure of “**non-robustness**” for each ToD based on the effect of potential delays within the ToD
  - ◆ If the ToD stays with the aircraft there will be no penalty
  - ◆ If the ToD changes aircraft, the penalty will reflect the potential disruption effect of the possible delay at the aircraft change
  - ◆ Ignore the interactive effects between ToDs
- Then treat the non-robustness measure as a second objective
  - ◆ The “non-robustness” objective measure should be additive across the ToD
  - ◆ So it is compatible with column generation solution procedures

# A non-robustness measure

Penalty  $p = \text{Max} [- \{$

Ground time (between consecutive flights on different aircraft)

- Measure of delay of incoming flight (eg mean + 2 \* deviations)
- Ground duty time (e.g. 30 or 60 for meal break)  $\}, 0 ]$



■ Non-robustness measure for each ToD:

$$r_j = \sum \alpha_i * p_i$$

- ◆ weighted sum of penalties over consecutive flight pairs in a ToD
- ◆ zero penalty if consecutive flights are on same aircraft
- ◆ weights  $\alpha_i$  reflect time of day and possibly airport
  - early delays – larger weight; late delays – smaller weight

# Bicriteria optimisation in ToD planning

- Minimise cost and minimise penalty for *non-robust* solutions
- The model

$$\min \quad \mathbf{r}^T \mathbf{x} \quad (\text{non-robustness objective})$$

$$\min \quad \mathbf{c}^T \mathbf{x} \quad (\text{cost objective})$$

$$\text{s.t.} \quad \mathbf{A} \mathbf{x} = \mathbf{e} \quad (\text{flight and base constraints})$$

$$\mathbf{x} \in \{0,1\}^n$$

- A solution  $\mathbf{x}^*$  is said to be Pareto optimal iff there does not exist any other  $\mathbf{x}$  which is
  - ◆ at least as good as  $\mathbf{x}^*$  with respect to both objectives
  - ◆ strictly better than  $\mathbf{x}^*$  with respect to at least one objective

# Bicriteria optimisation in ToD planning

## ■ The model

$$\min \quad \mathbf{r}^T \mathbf{x} \quad (\text{non-robustness objective})$$

$$\min \quad \mathbf{c}^T \mathbf{x} \quad (\text{cost objective})$$

$$\text{s.t.} \quad \mathbf{A} \mathbf{x} = \mathbf{e} \quad (\text{flight and base constraints})$$

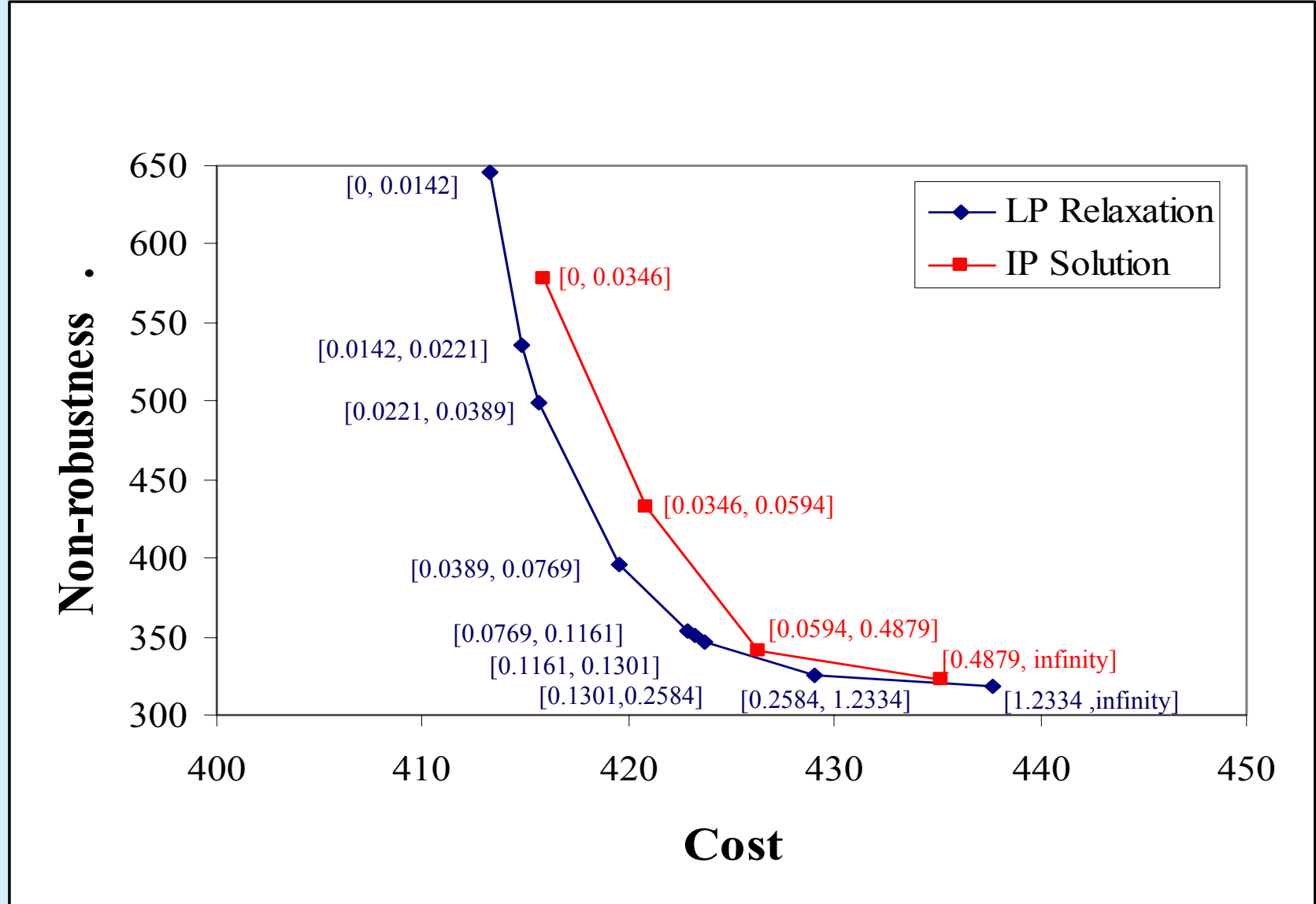
$$\mathbf{x} \in \{0,1\}^n$$

## ■ We could form a weighted sum of objectives

$$\min \quad \mathbf{c}^T \mathbf{x} + t * \mathbf{r}^T \mathbf{x}$$

- ◆ Choice of weight value  $t$  is not clear
- ◆ Solutions (both LP and IP) are sensitive choice of weight value
- ◆ Cannot generate all Pareto optimal integer solutions

# Results of the weighted sum method



# Preferred solution approach

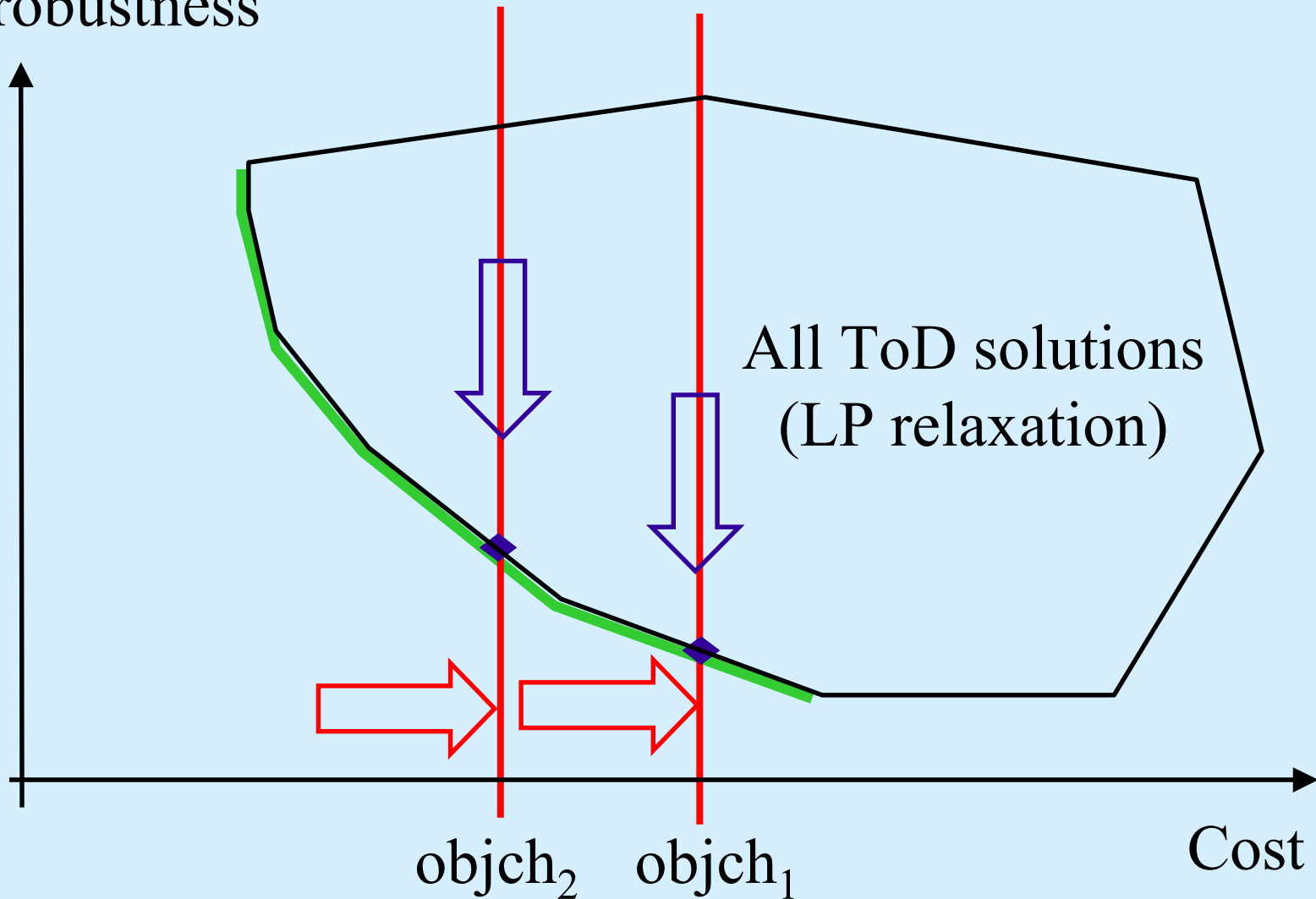
- Use the so-called  $\varepsilon$ -constraint method for multi-criteria optimisation
- Treat the cost objective as a constraint

$$\begin{aligned} \min \quad & \mathbf{r}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{c}^T \mathbf{x} \leq \varepsilon \\ & \mathbf{A} \mathbf{x} = \mathbf{e} \\ & \mathbf{x} \in \{0,1\}^n \end{aligned}$$

- Allow a (small) increase in cost to improve robustness
  - ◆ for example, choose  $\varepsilon = (1 + \text{objch} / 100) \text{CIP}$  (or even CLP)
  - ◆ a percentage increase of the minimal IP (LP) solution cost

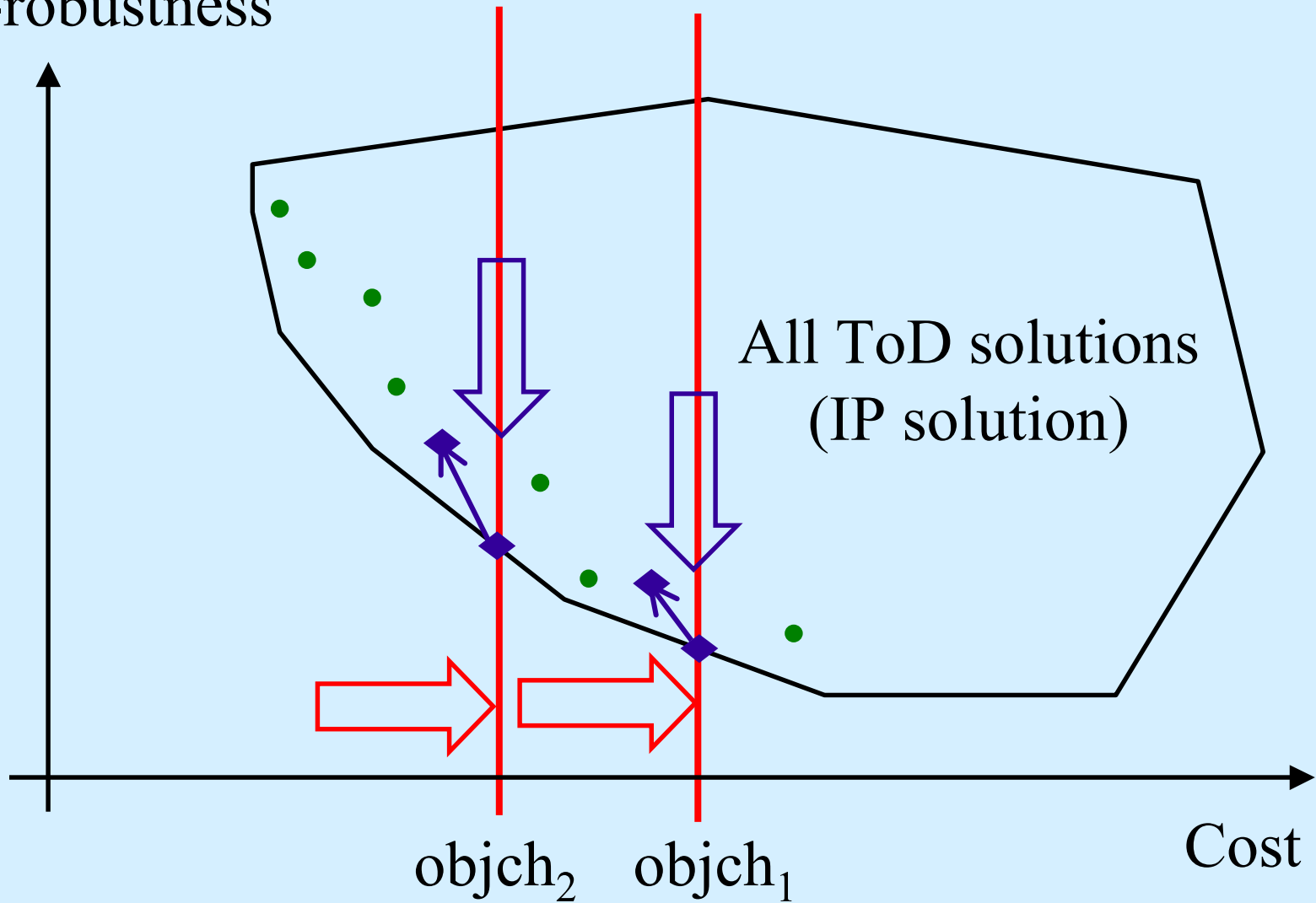
# Illustration (solution of LP relaxation)

Non-robustness

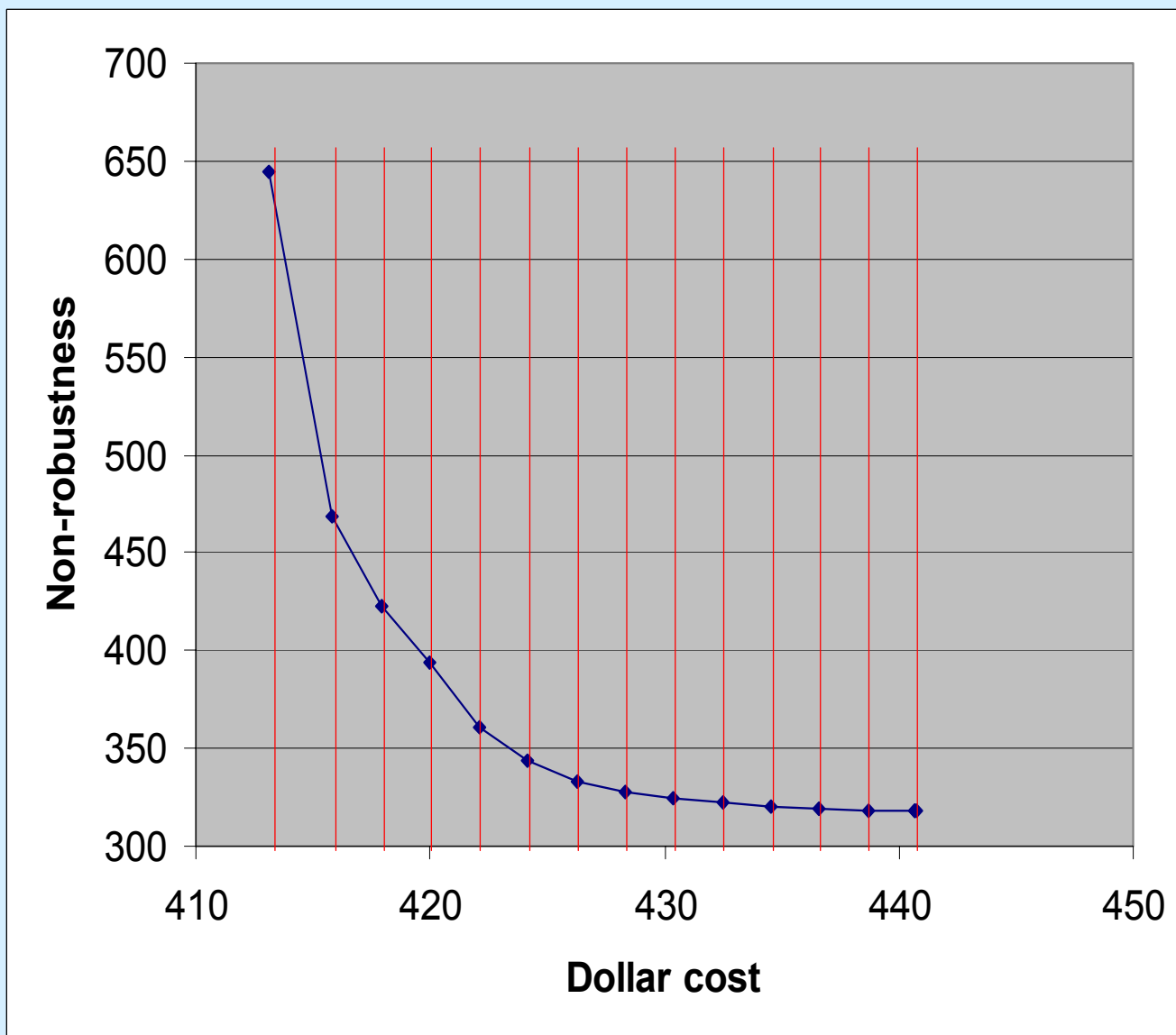


# Illustration (solution of integer problem)

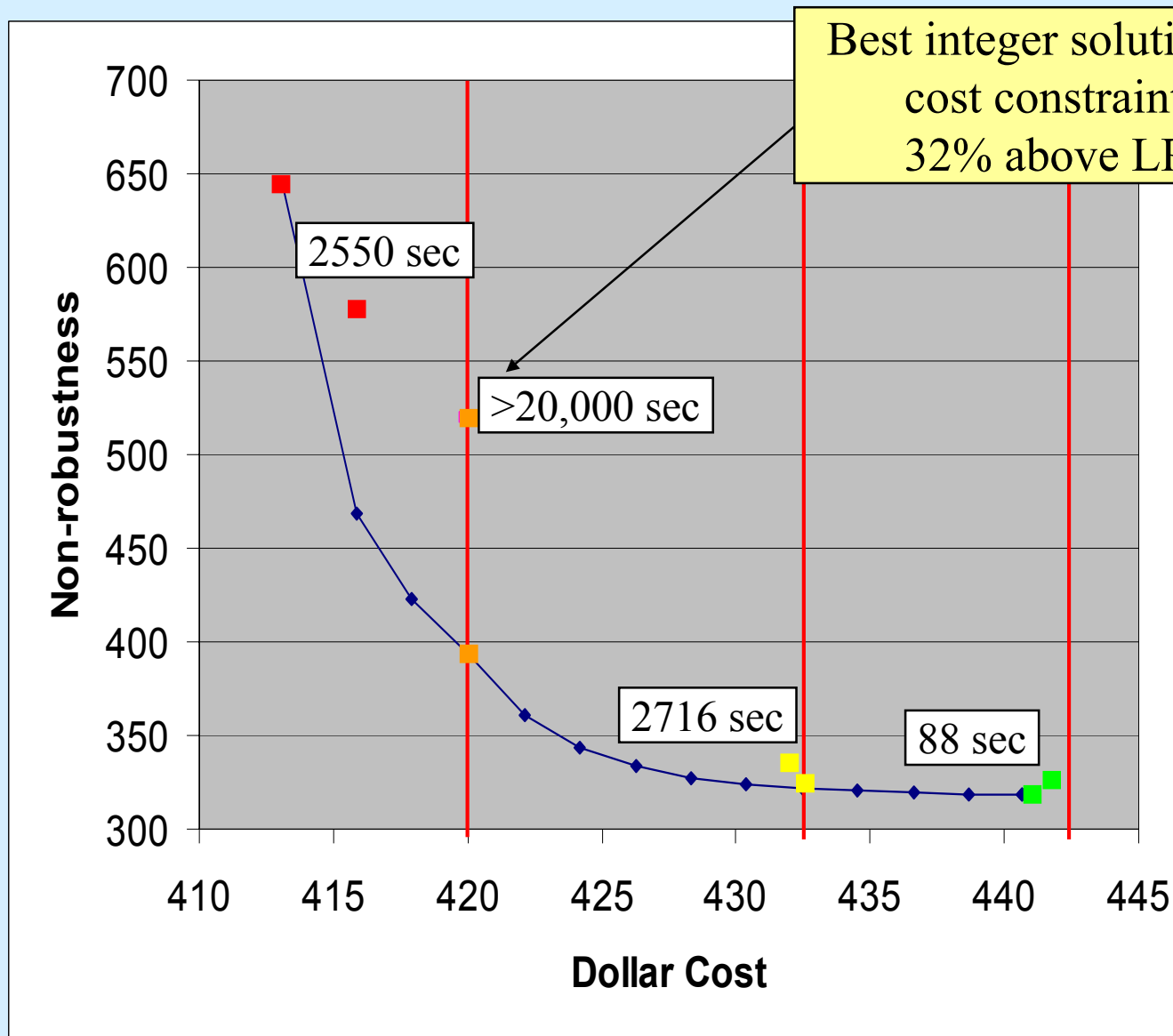
Non-robustness



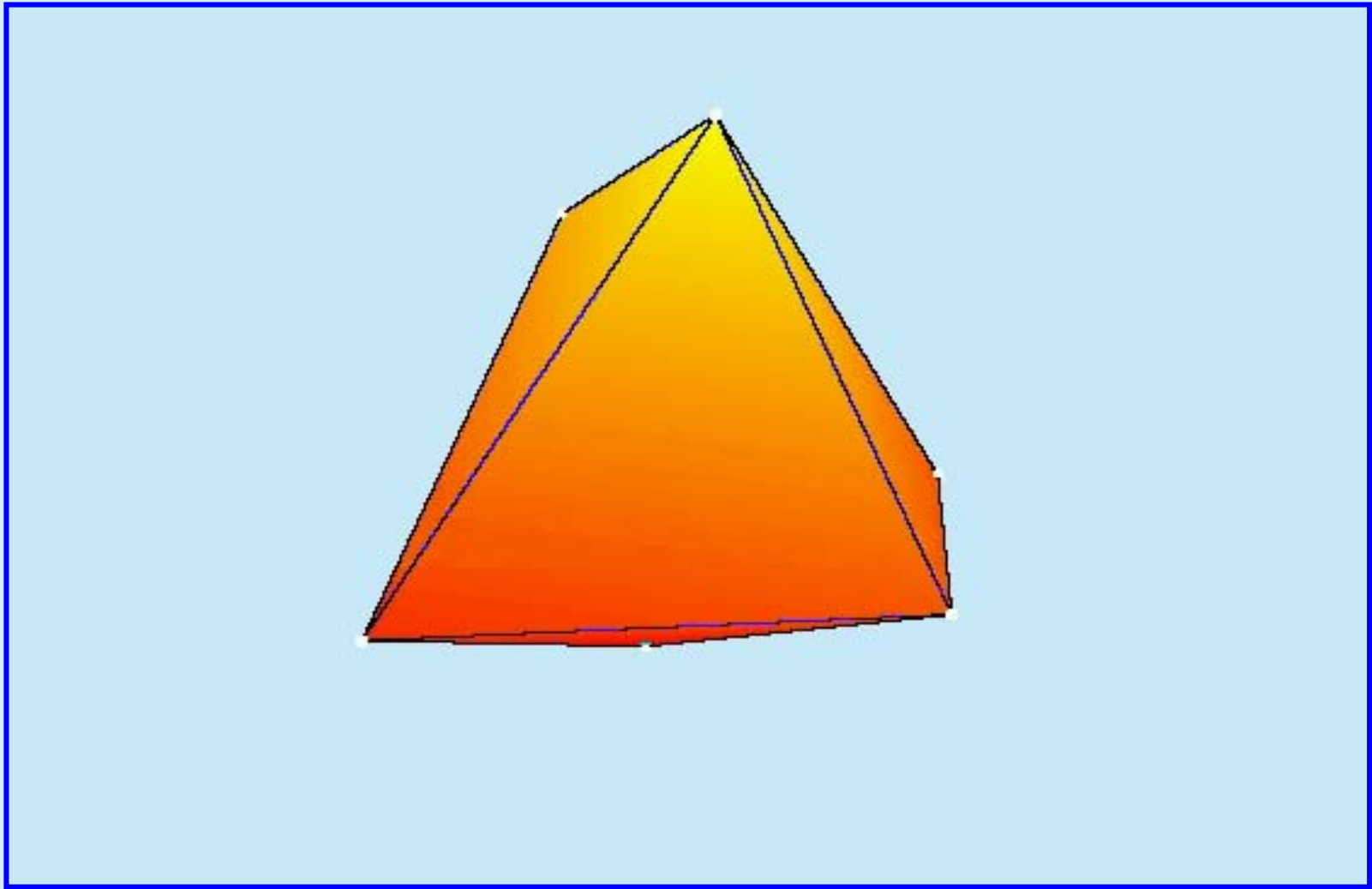
# Solution of LP Relaxation - trade off curve



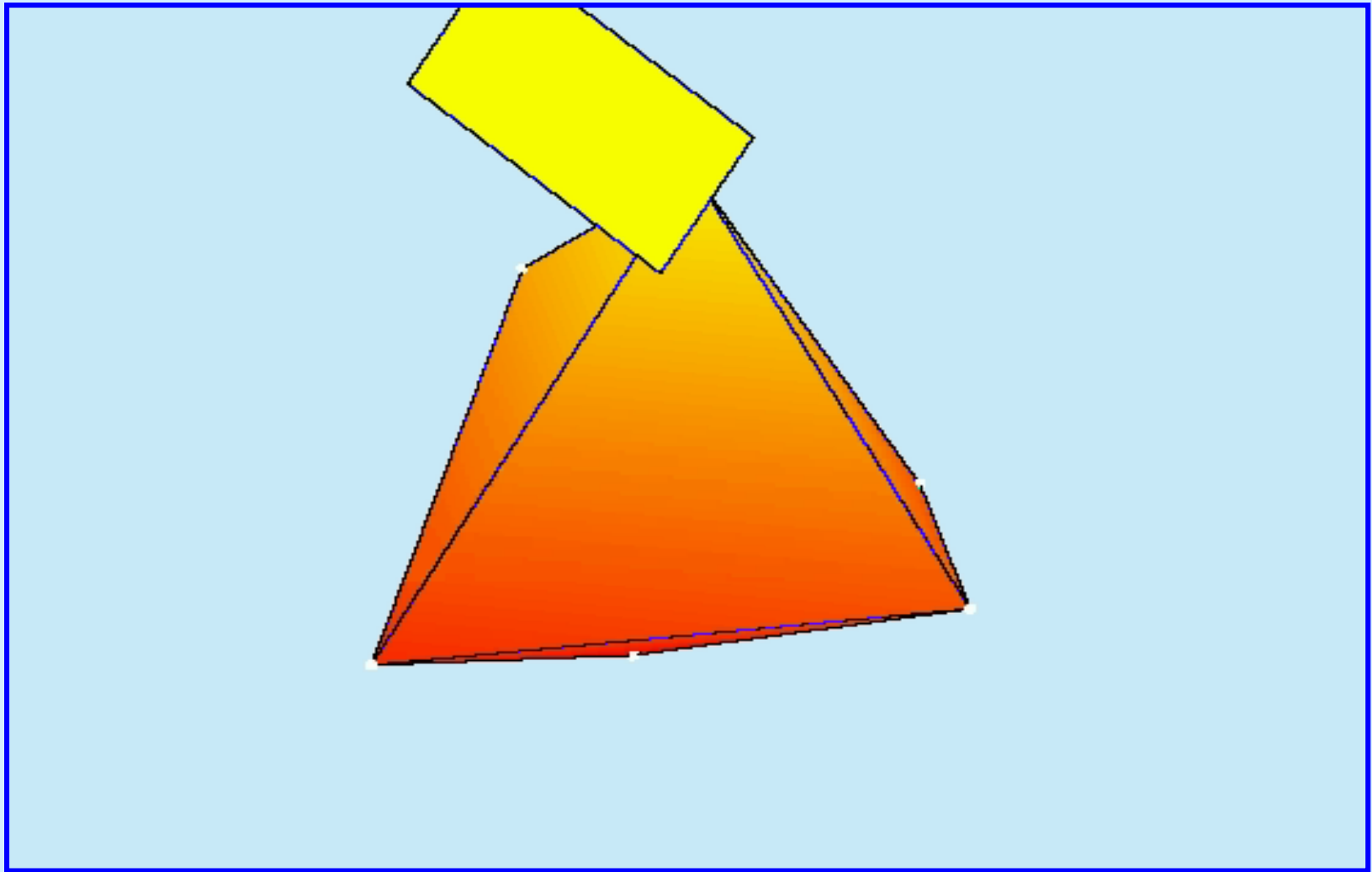
# Corresponding IP solutions



# The SPPLP polytope



# The effect of the cost objective cut

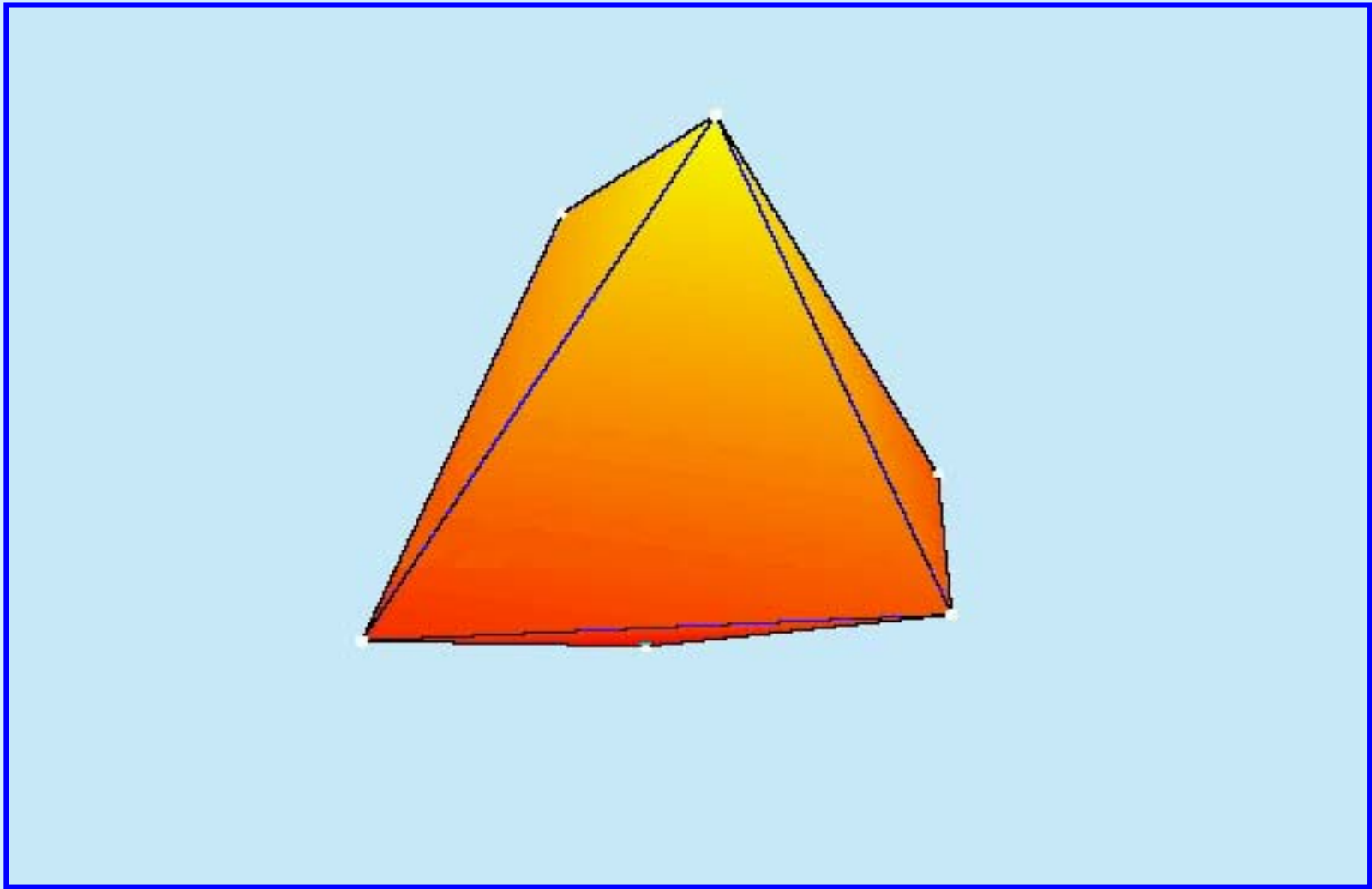


# The elastic cost constraint

- Treat the cost objective constraint as an elastic constraint
  - ◆ I first learnt about the benefits of the elastic constraint approach from Gerry Brown and Rick Rosenthal at the Naval Postgraduate School
- Permit a (small) violation of the cost constraint
  - ◆ but penalise the violation

$$\begin{aligned} \text{ESPPIP: } \quad & \min \quad \mathbf{r}^T \mathbf{x} + p \, su \\ & \text{s.t.} \quad \mathbf{c}^T \mathbf{x} + sl - su = (1 + \text{objch}/100) \text{CIP (or CLP)} \\ & \quad \quad \mathbf{A} \mathbf{x} = \mathbf{e} \\ & \quad \quad \mathbf{x} \in \{0,1\}^n, \quad sl, su \geq 0 \end{aligned}$$

# The ESPPLP polytope



# Theoretical result

## ■ Theorem:

- ◆ If  $p > 0$ , the optimal solution of ESPPIP is Pareto optimal.
- ◆ If  $\mathbf{x}^*$  is Pareto optimal, there exist **objch** and  $p^*$  such that  $\mathbf{x}^*$  is an optimal solution of ESPPIP for all  $p$  with  $p \geq p^*$ .

$$\begin{aligned} \text{ESPPIP: } \min \quad & \mathbf{r}^T \mathbf{x} + p \, su \\ \text{s.t.} \quad & \mathbf{c}^T \mathbf{x} + sl - su = (1 + \text{objch}/100) \text{ CIP} \\ & \mathbf{A} \mathbf{x} = \mathbf{e} \\ & \mathbf{x} \in \{0,1\}^n, \, sl, su \geq 0 \end{aligned}$$

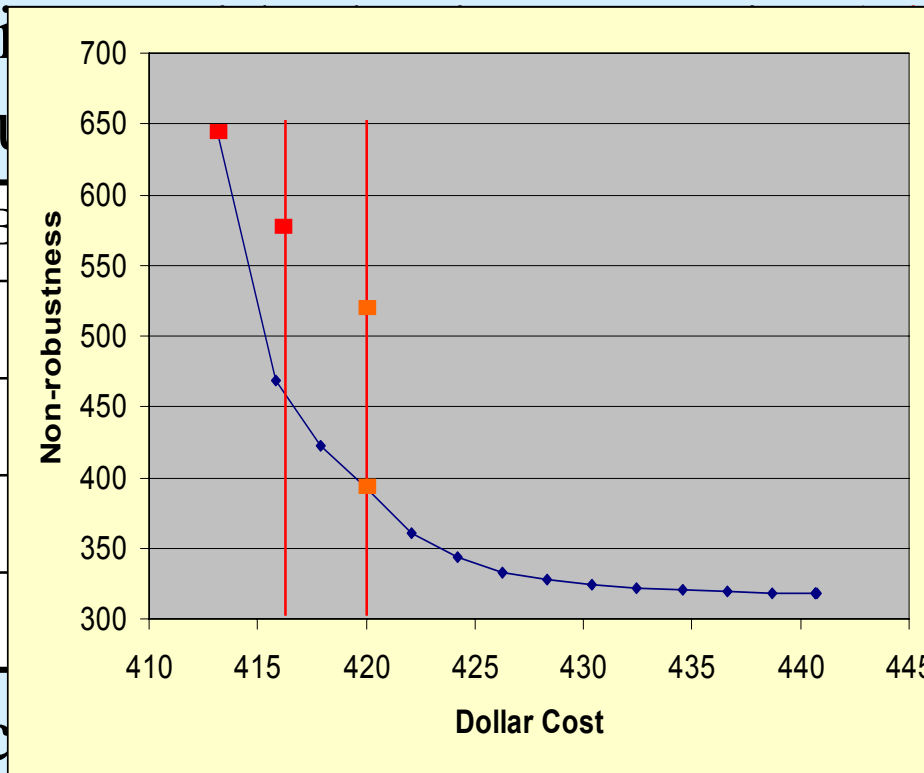
# Numerical results

## ■ Computing times with $\epsilon$ -constraint

- ◆ **objch** = 0: integer solution exists but not found after 1,000 nodes
- ◆ **objch** = 1: best integer after 1,000 nodes and >20,000 sec
  - non-robustness still 32% above LP lower bound

## ■ Computing times when minimizing cost

$p$	Branch and bound
0.1	
10	
22.15	
1,000.0	

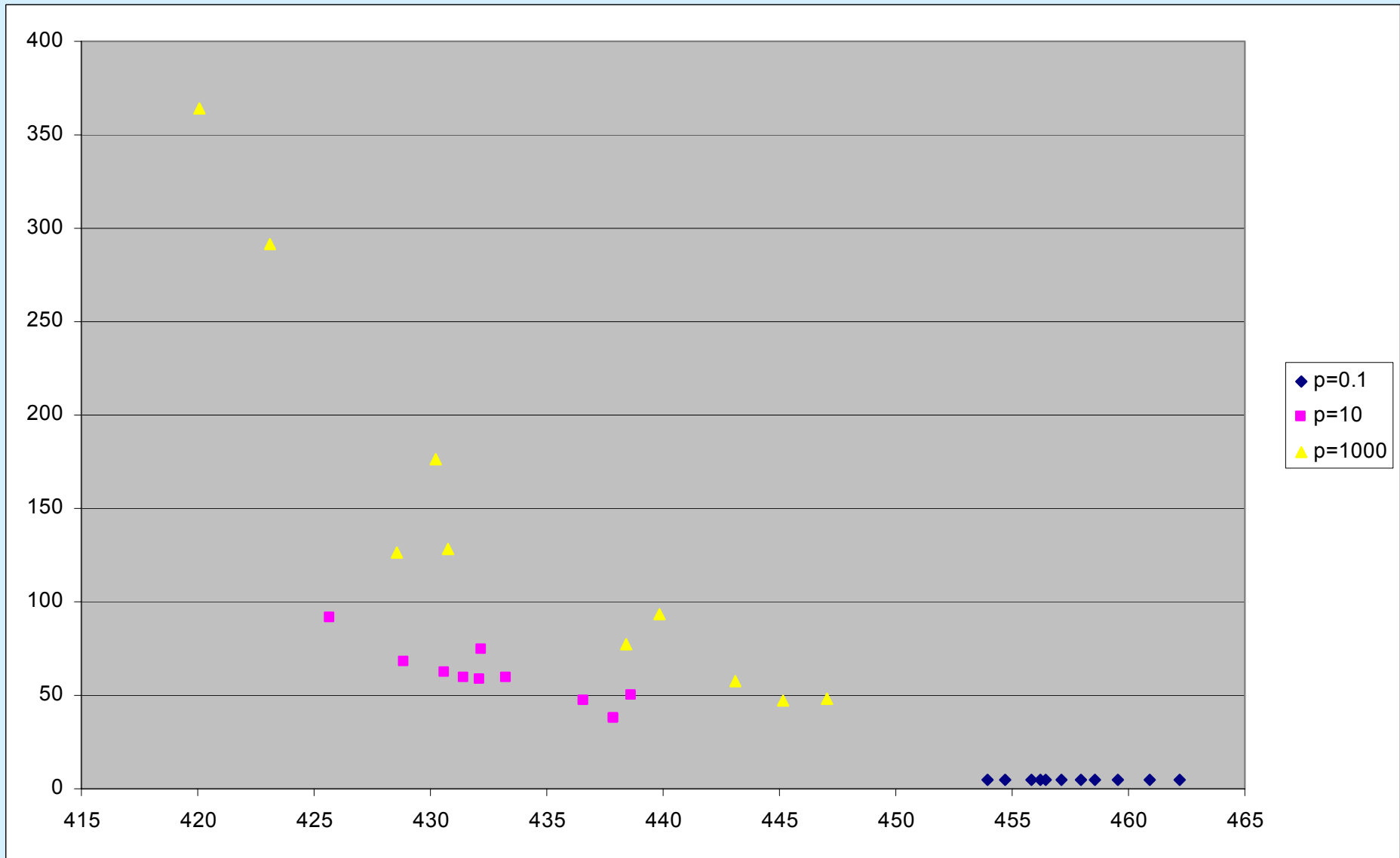


Cost % increase
10.27
4.19
1.74
1.00

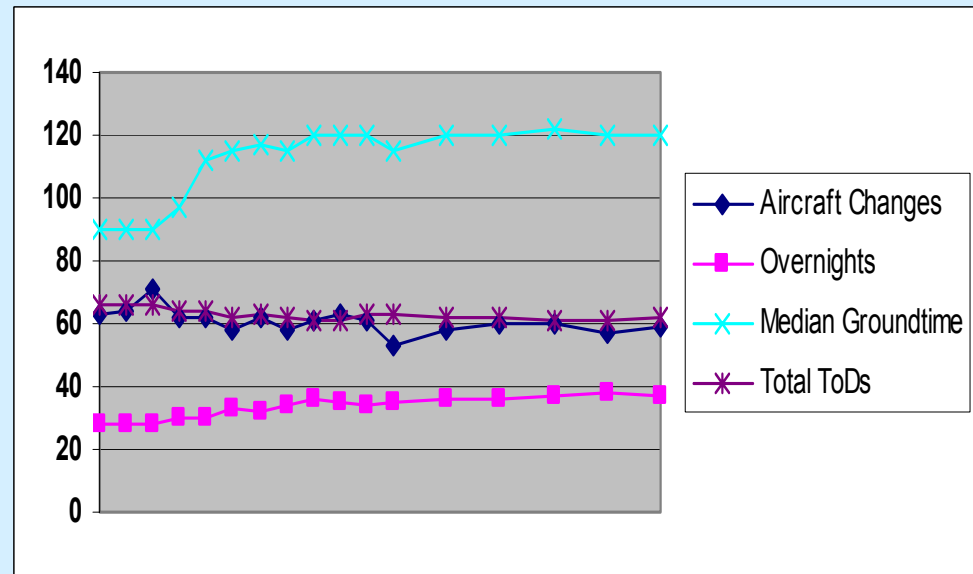
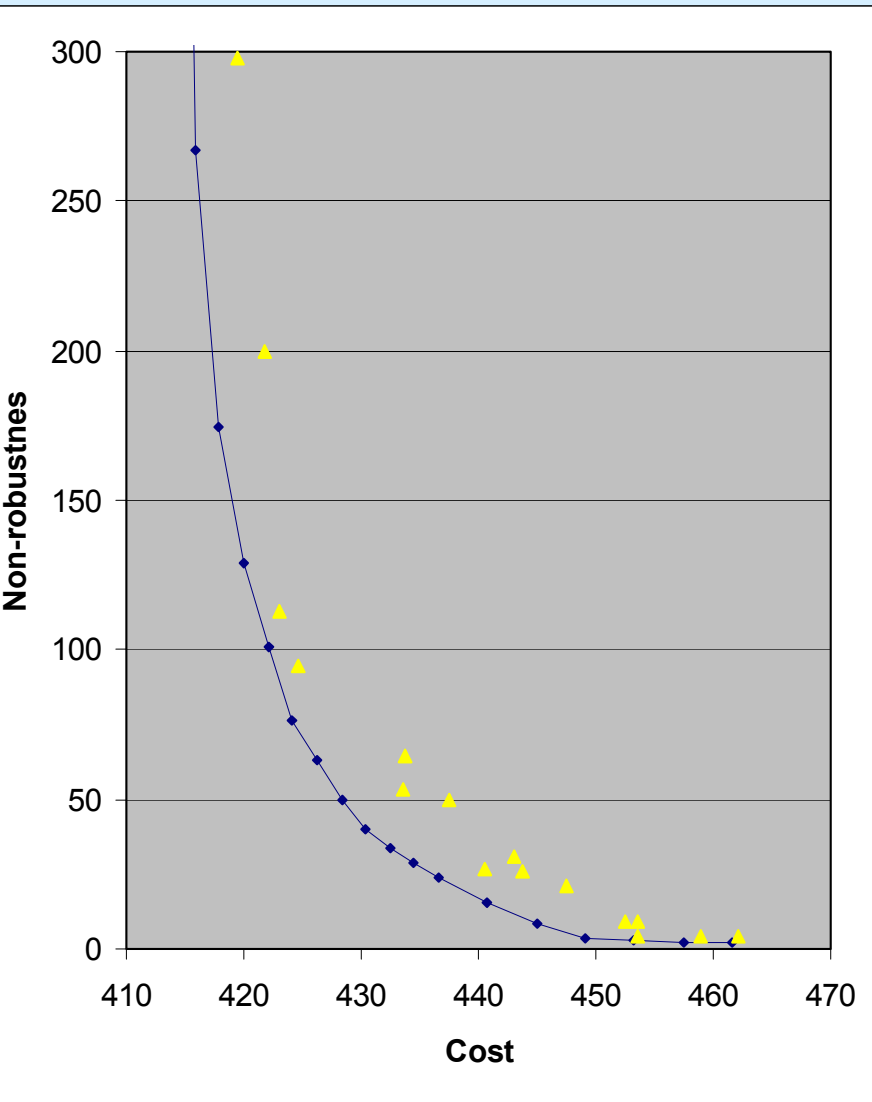
## ■ Elastic cost constraint

branch and bound

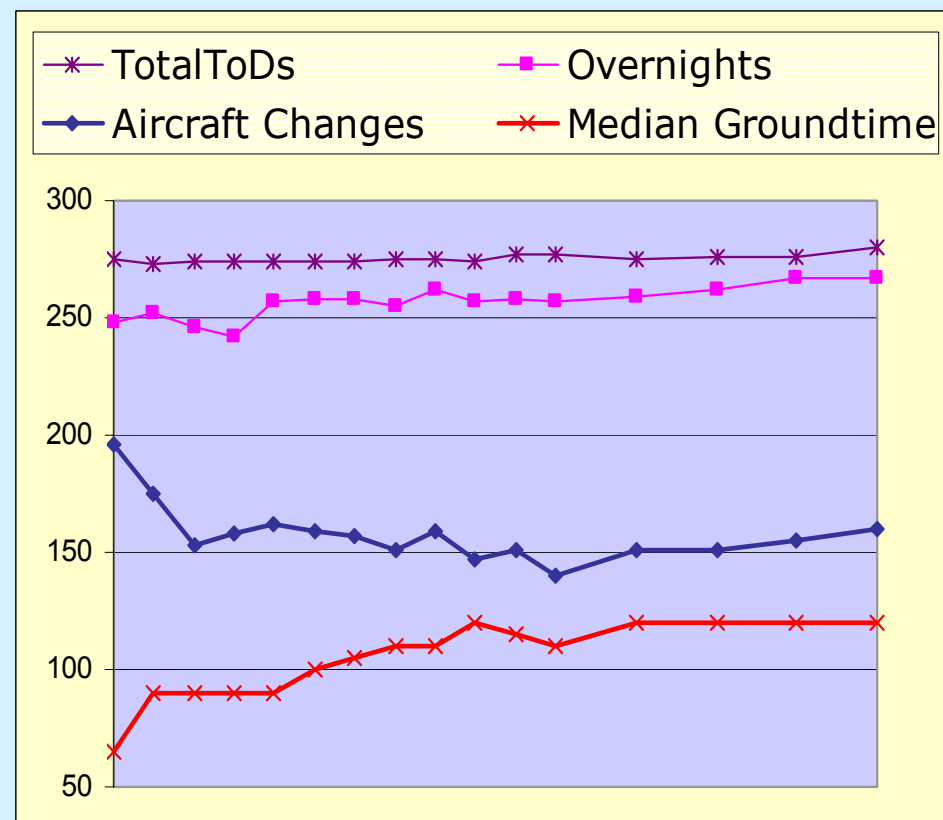
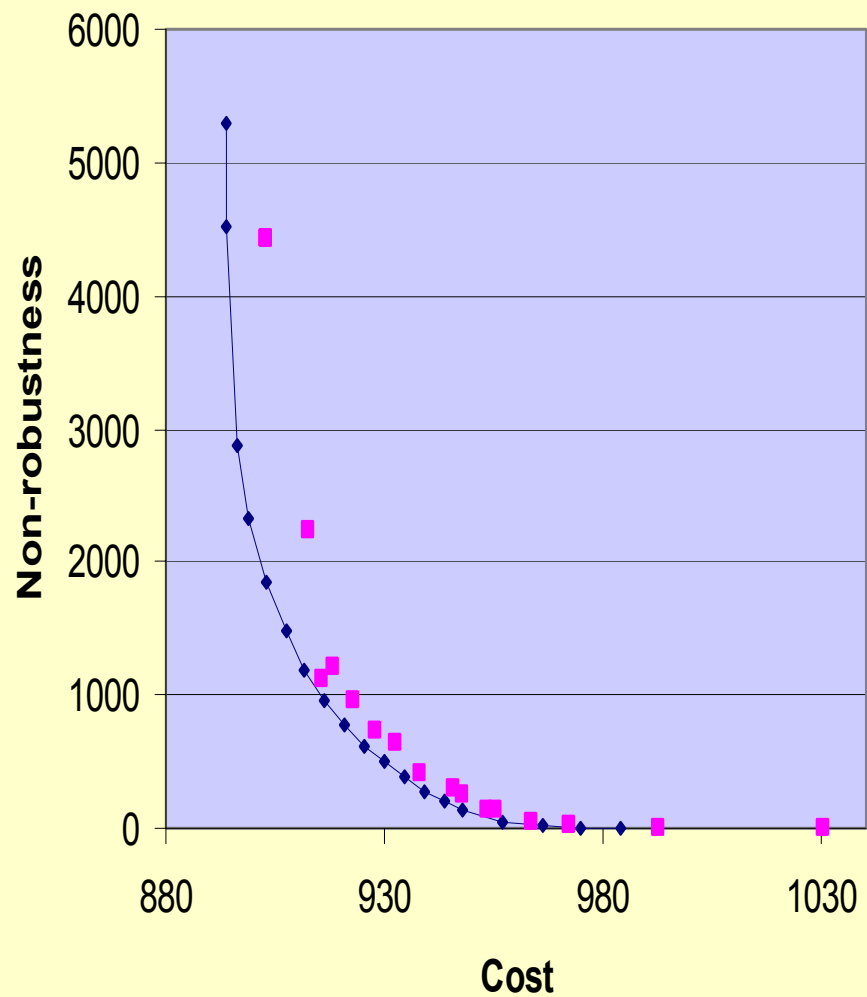
# Effect of $p = (0.1, 10, 1000)$



# From minimal cost to robust ToDs - technical crew



# Minimal cost versus robust ToDs – flight attendants



# From minimal cost to robust ToDs

Crew Type	Technical		
Solution	Min Cost	Objch=1%	Max Robustness
Cost	<b>418.9</b>	<b>423.1</b>	<b>453.6</b>
Non robustness	<b>679.9</b>	<b>112.8</b>	<b>4.3</b>

# From minimal cost to robust ToDs

Crew Type	Technical		
Solution	Min Cost	Objch=1%	Max Robustness
Cost	<b>418.9</b>	<b>423.1</b>	<b>453.6</b>
Non robustness	<b>679.9</b>	<b>112.8</b>	<b>4.3</b>
Aircraft changes	<b>62</b>	<b>62</b>	<b>60</b>
Median groundtime	<b>90</b>	<b>97.5</b>	<b>122.5</b>
Passengering	<b>20</b>	<b>18</b>	<b>20</b>
Overnights	<b>28</b>	<b>30</b>	<b>37</b>

# From minimal cost to robust ToDs

Crew Type	Technical			Flight Attendants		
Solution	Min Cost	Objch=1%	Max Robustness	Min Cost	Objch=1%	Max Robustness
Cost	<b>418.9</b>	<b>423.1</b>	<b>453.6</b>	<b>902.8</b>	<b>915.64</b>	<b>992.7</b>
Non robustness	<b>679.9</b>	<b>112.8</b>	<b>4.3</b>	<b>4418.0</b>	<b>1115.23</b>	<b>4.8</b>
Aircraft changes	<b>62</b>	<b>62</b>	<b>60</b>	<b>196</b>	<b>158</b>	<b>155</b>
Median groundtime	<b>90</b>	<b>97.5</b>	<b>122.5</b>	<b>65</b>	<b>90</b>	<b>120</b>
Passengering	<b>20</b>	<b>18</b>	<b>20</b>	<b>21</b>	<b>22</b>	<b>33</b>
Overnights	<b>28</b>	<b>30</b>	<b>37</b>	<b>248</b>	<b>242</b>	<b>267</b>

# Conclusions

- Elasticizing constraints helps preserve nice integer structure
- Considerable robustness gains from small increase in cost
  - ◆ Robustness gains are reflected in longer ground times between successive flights on different aircraft within a ToD and fewer aircraft changes
  - ◆ Cost increases are due to slightly longer duty times, a few more ToDs, a very small increases in overnighing and passengering
- Robust solutions have deeper subsequences
  - ◆ no first subsequences that involve aircraft change
- Our approach treats cost and robustness separately
- Our approach is fully compatible with current deterministic SPP models and column generation methods

# Future work

- System should be easy to use:
  - ◆ Permitted cost increase (**objch**) is specified by management
  - ◆ Dynamically modify the elastic penalty during the optimisation to control cost violation
    - keep elastic penalty cost as small as possible
- Need to determine weights in  $r_j = \sum \alpha_i * p_i$ 
  - ◆ Equal unit weights used so far
- Compare results of all three approaches
  - ◆ Use the SimAir package currently being developed by Georgia Tech and the National University of Singapore