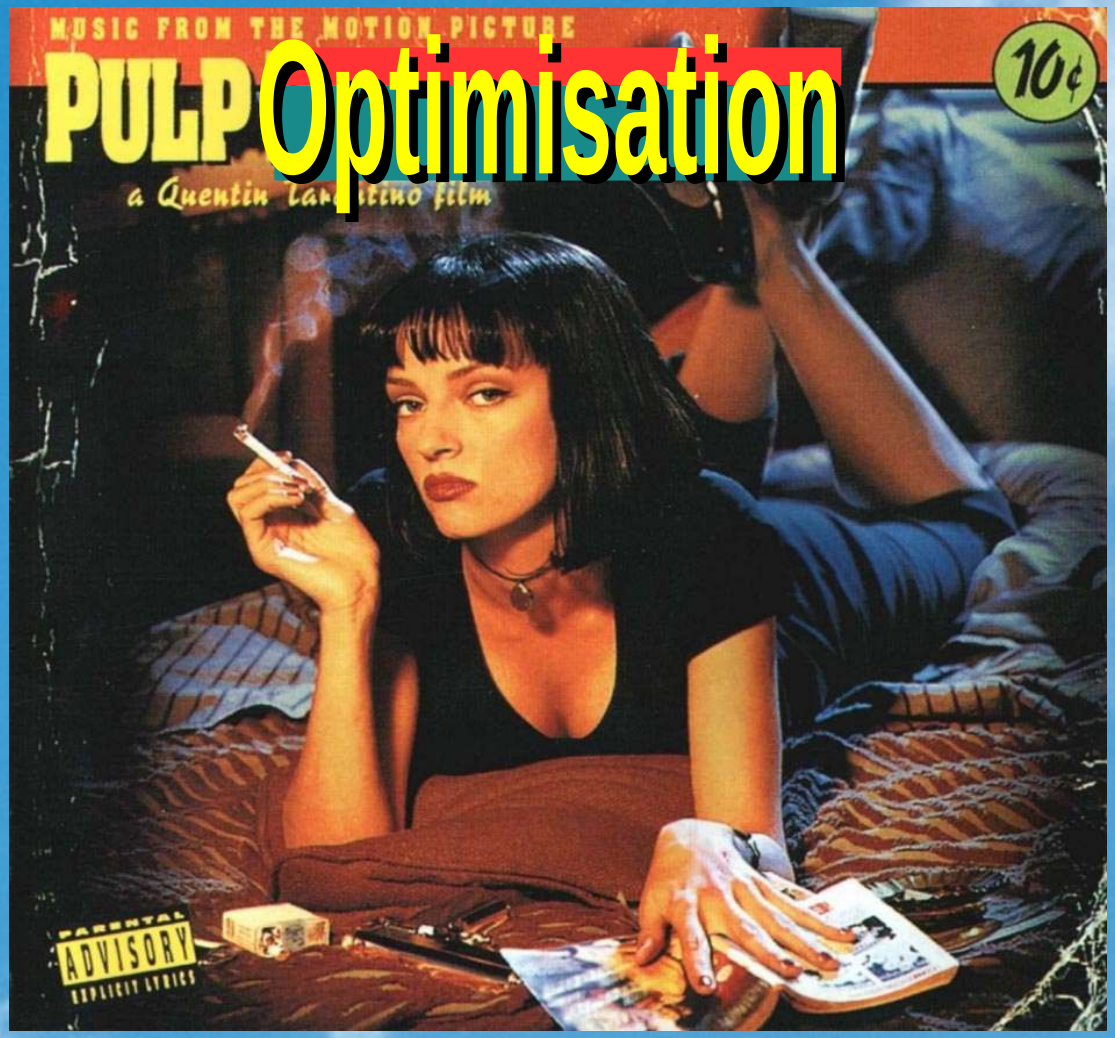


Optimisation

Dr. Stuart Mitchell
LMRC
University of Auckland
s.mitchell@auckland.ac.nz

Kiwi Pycon 2009

THE UNIVERSITY OF AUCKLAND
NEW ZEALAND

LIGHT METALS RESEARCH CENTRE · THE UNIVERSITY OF AUCKLAND

Contents of presentation

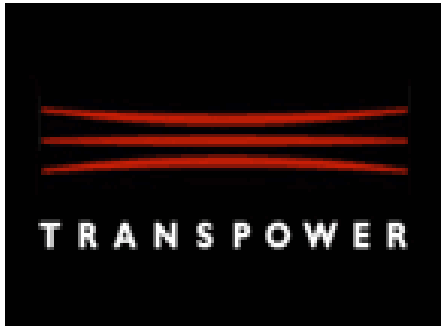
What is Mathematical Programing

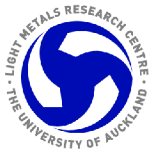
When is it useful

The Wedding Guest Problem

What is Mathematical programing

- A simple mathematically precise way of stating optimisation problems.
- Use mathematically rigorous way to find a solution






THE UNITED Countries OF Baseball
MARINERS
GIANTS
Athletics
UNINCORPORATED TERRITORIES
ROCKIES
Dodgers
ANGELS
Padres
Dbacks
Twins
Brewers
Tigers
Jays
RED SOX
Yankees
Phillies
Mets
Indians
PIRATES
Orioles
NATIONALS
Royals
REDS
Cardinals
Rangers
Braves
Astros
RAYS
MARLINS
Fan Loyalty Circa 2007
THESE BORDERS ARE REAL FOR SEVEN MONTHS A YEAR. FROM OPENING DAY TO THE WORLD SERIES. THE MAJORITY OF PEOPLE OF EACH COUNTRY PLEDGE THEIR ALLEGIANCE TO THE TEAM SHOWN. REGARDLESS OF PLACEMENT IN THE STANDINGS, QUESTIONABLE TRADES, DRAFT PICKS, PITCHING ROTATIONS, UNIFORM REDESIGNS, OR MASCOT BEHAVIOR. AND THESE LINES WILL STAY TRUE UNTIL THE CITIZENS OF EACH COUNTRY VOTE TO REDRAW THE BORDERS FOR NEXT SEASON.
30 teams – 162 games

THE UNITED Countries OF Baseball

THE UNITED COUNTRIES OF BASEBALL

Fan Loyalty Circa 2007

THESE BORDERS ARE REAL FOR SEVEN MONTHS A YEAR. FROM OPENING DAY TO THE WORLD SERIES, THE MAJORITY OF PEOPLE OF EACH COUNTRY PLEDGE THEIR ALLEGIANCE TO THE TEAM SHOWN. REGARDLESS OF PLACEMENT IN THE STANDINGS, QUESTIONABLE TRADES, DRAFT PICKS, PITCHING ROTATIONS, UNIFORM REDESIGNS, OR MASCOT BEHAVIOR. AND THESE LINES WILL STAY TRUE UNTIL THE CITIZENS OF EACH COUNTRY VOTE TO REDRAW THE BORDERS FOR NEXT SEASON.

reds.Com reds.Com reds.Com

CINCINNATI REDS

APRIL 2010

SUN	MON	TUE	WED	THU	FRI	SAT
				1	2	3
4	5 STL	6	7 STL	8 STL	9 CHI	10 CHI
11 CHI	12 FLA	13 FLA	14 FLA	15 FLA	16 PIT	17 PIT
18 PIT	19	20 LA	21 LA	22 LA	23 SD	24 SD
25 SD	26	27 HOU	28 HOU	29 HOU	30 STL	

JULY 2010

SUN	MON	TUE	WED	THU	FRI	SAT
				1 CHI	2 CHI	3 CHI
4 CHI	5 NYM	6 NYM	7 NYM	8 PHI	9 PHI	10 PHI
11 PHI	12	13 ★	14	15	16 COL	17 COL
18 COL	19 WSH	20 WSH	21 WSH	22 WSH	23 HOU	24 HOU
25 HOU	26 MIL	27 MIL	28 MIL	29	30 ATL	31 ATL

MAY 2010

SUN	MON	TUE	WED	THU	FRI	SAT
						1 STL
2 STL	3 NYM	4 NYM	5 NYM	6	7 CHI	8 CHI
9 CHI	10 PIT	11 PIT	12 PIT	13	14 STL	15 STL
16 STL	17 MIL	18 MIL	19 ATL	20 ATL	21 CLE	22 CLE
CLE 23	PIT 24	25 PIT	26 PIT	27 PIT	28 HOU	29 HOU
HOU 30	STL 31					

AUGUST 2010

SUN	MON	TUE	WED	THU	FRI	SAT
1 ATL	2 PIT	3 PIT	4 PIT	5	6 CHI	7 CHI
8 CHI	9 STL	10 STL	11 STL	12	13 FLA	14 FLA
15 FLA	16	17 ARI	18 ARI	19 ARI	20 LA	21 LA
22 LA	23 SF	24 SF	25 SF	26	27 CHI	28 CHI
29 CHI	30 MIL	31 MIL				

JUNE 2010

SUN	MON	TUE	WED	THU	FRI	SAT
		1 STL	2 STL	3	4 WSH	5 WSH
6 WSH	7 SF	8 SF	9 SF	10 SF	11 KC	12 KC
13 KC	14	15 LA	16 LA	17 LA	18 SEA	19 SEA
20 SEA	21 OAK	22 OAK	23 OAK	24	25 CLE	26 CLE
27 CLE	28 PHI	29 PHI	30 PHI			

SEPT. - OCT. 2010

SUN	MON	TUE	WED	THU	FRI	SAT
			1 MIL	2	3 STL	4 STL
5 STL	6 COL	7 COL	8 COL	9 COL	10 PIT	11 PIT
12 PIT	13 ARI	14 ARI	15 ARI	16 ARI	17 HOU	18 HOU
19 HOU	20 MIL	21 MIL	22 MIL	23	24 SD	25 SD
SD 26	27	28 HOU	29 HOU	30 HOU	1 MIL	2 MIL
MIL 3						

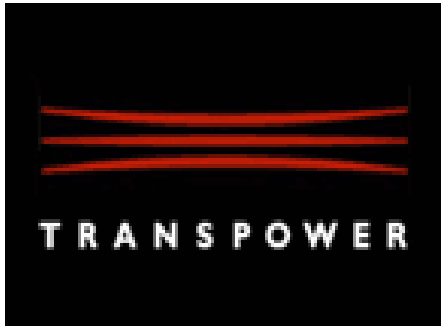
HOME AWAY

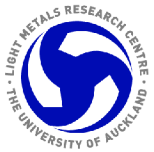
SEPT. 15, 2009

SCHEDULE SUBJECT TO CHANGE. GAME TIMES ARE TBD.
All games can be heard on the Reds on Radio Network.

What is Mathematical programing

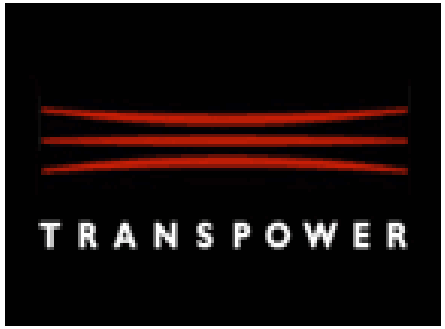
- A simple mathematically precise way of stating optimisation problems.
- Use mathematically rigorous way to find a solution

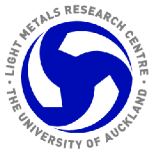




What is Mathematical programing

- A simple mathematically precise way of stating optimisation problems.
- Use mathematically rigorous way to find a solution

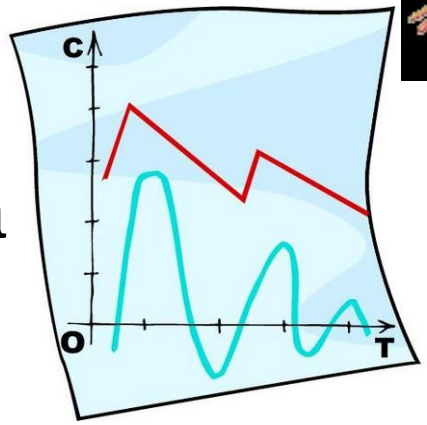




When is it useful

- There is a decision problem

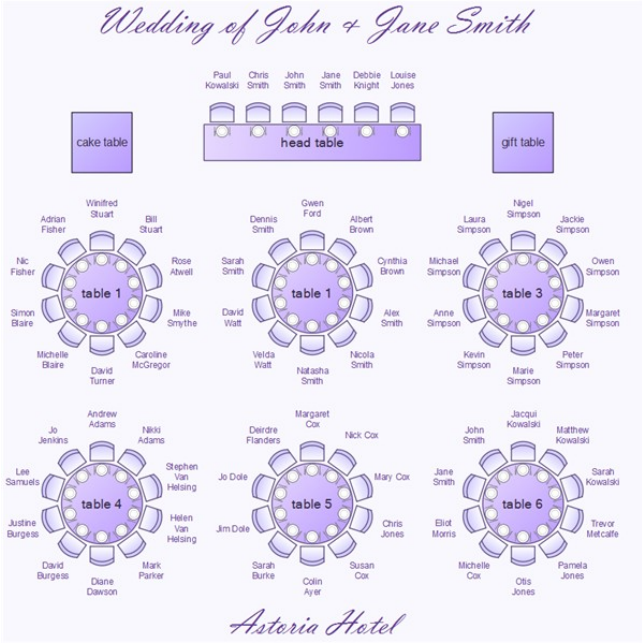
- There is lots of data



- There is money to be made

The wedding guest problem

- Similar to work at LMRC
- A wedding planner whats to work out seating allocations

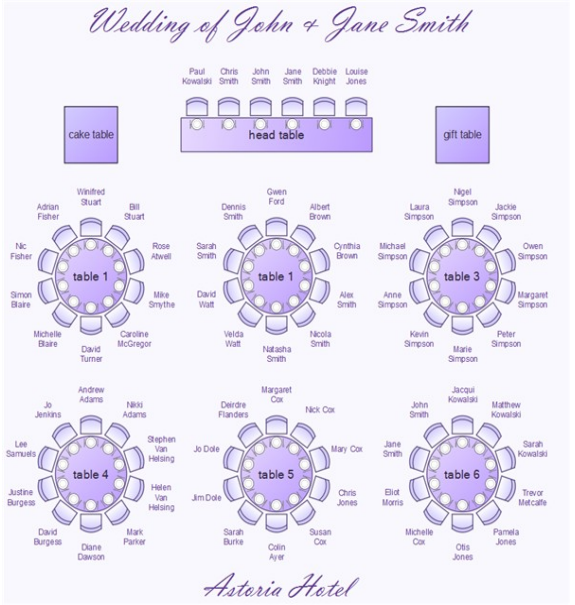


http://weddingelegante.com

* Tables near the head table (or sweetheart table) should be reserved for your closest family and friends
* If you're using round tables, you should generally assign males and females to alternating seats
* Don't forget to use your guests' ages and interests when assigning
* Grouping your guests into identifiable cliques (grooms college friends, work colleagues, friends of your parents, etc.) will make grouping guests together at specific tables easier
* Seat children under 8 years of age at the same table as their parents, but if you're expecting a great deal of 8+ children to attend, you may consider creating a "kids-only" table
* Leave 2-3 tables empty for vendors during their breaks or for unexpected guests
* Have a large diagram of the completed seating chart on hand at the reception venue on your wedding day to solve any confusion

The wedding guest problem

- We need to define
  - A solution – what is the result we want



The wedding guest problem

- We need to define
  - A solution – what is the result we want
  - Variables – parts of the solution

The wedding guest problem

- We need to define
  - A solution – what is the result we want
  - Variables – parts of the solution
  - Objective function – how to compare solutions

The wedding guest problem

- We need to define
  - A solution – what is the result we want
  - Variables – parts of the solution
  - Objective function – how to compare solutions
  - Constraints – define a solution
    - In terms of the variables

Mathematical expression of the problem

- This is a set partitioning problem.
  - We wish to partition the set of guests into subsets of tables.
  - We wish to maximise happiness of the tables.
  - We want everyone to be seated at exactly one table.
- This is implemented in the following python code

Wedding Planning Formulation

Let guests be the set of all guests.

```
"""
A set partitioning model of a wedding seating problem

Authors: Stuart Mitchell 2009
"""

import pulp

max_tables = 5
max_table_size = 4
guests = 'A B C D E F G I J K L M N O P Q R'.split()
```

Wedding Planning Formulation

Use pulp.allcombinations to generate a list of all possible table seatings.

```
20  #create list of all possible tables
21  possible_tables = [tuple(c) for c in pulp.allcombinations(guests,
22                                                  max_table_size)]
```

Wedding Planning Formulation

Then we create a binary variable that will be 1 if the table will be in the solution, or zero

```
24  #create a binary variable to state that a table setting is used
25  x = pulp.LpVariable.dicts('table', possible_tables,
26                            lowBound = 0,
27                            upBound = 1,
28                            cat = pulp.LpInteger)
```

Wedding Planning Formulation

We create the LpProblem and then make the objective function.

```
30  seating_model = pulp.LpProblem("Wedding Seating Model", pulp.LpMinimize)

32  seating_model += sum([happiness(table) * x[table] for table in possible_tables])
```

Wedding Planning Formulation

We create the LpProblem and then make the objective function.

```
30  seating_model = pulp.LpProblem("Wedding Seating Model", pulp.LpMinimize)

32  seating_model += sum([happiness(table) * x[table] for table in possible_tables])
```

```
13  def happiness(table):
14      """
15      Find the happiness of the table
16      - by calculating the maximum distance between the letters
17      """
18      return abs(ord(table[0]) - ord(table[-1]))
```

Wedding Planning Formulation

We specify the total number of tables allowed in the solution.

```
34  #specify the maximum number of tables
35  seating_model += sum([x[table] for table in possible_tables]) <= max_tables, \
36                                  "Maximum_number_of_tables"
```

Wedding Planning Formulation

Multiple similar constraints (one of each guest) guarantee that a guest is allocated to exactly one table.

```
38  #A guest must seated at one and only one table
39  for guest in guests:
40      seating_model += sum([x[table] for table in possible_tables
41                                    if guest in table]) == 1, "Must_seat_%s"%guest
```

Wedding Planning Formulation

Solve the problem and print out an answer.

```
43  seating_model.solve()

45  print "The choosen tables are:"
46  for table in possible_tables:
47      if x[table].value() == 1.0:
48          print table
```

The choosen tables are out of a total of 3213:
('M', 'N')
('E', 'F', 'G')
('A', 'B', 'C', 'D')
('I', 'J', 'K', 'L')
('O', 'P', 'Q', 'R')

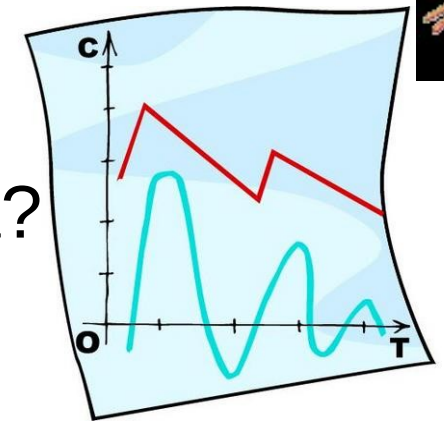
Why Pulp

- MIT Licence
- Installation (google-code or pypi)
- Syntax
- Interoperabilty
- Documentation

Next time you are talking to a client

- Is there a decision problem?

- Is there lots of data?



- Is there money to be made?