Fagdag

Combinatorial Optimization

29. november 2019

Tommy Odland and Alf Hammerseth

Plan for today

1. You get a problem. (30 min)
2. You compete to solve it. (3-4 hours)
3. We discuss our solutions and I talk about theory. (1-2 hours)

Problem description

A conference planner (e.g. JavaZone) has asked us to help set up a conference schedule:

Go	Unix	...	Testing
DevOps	ML	...	Java
Azure	.net	...	git

Tracks

Times

Time slot

Tracks $1, 2, \ldots, i, \ldots, I$

Times $1, 2, \ldots, j, \ldots, J$

Presentations $1, \ldots, p, \ldots, P$

People $1, 2, \ldots, k, \ldots, K$

Full description in repo.

Input data

- Schedule requirements (e.g. # tracks, # times)
- A list of presentations which the people attending are interested in.
- There are 2 problem instances:
  - small
    3 tracks, 15 times, 100 people
  - large
    8 tracks, 15 times, 4000 people

Minimal example

Tracks : 2

Times : 2

Presentations : 4

People : 2

$$\text{Presentations} = \{1, 2, 3, 4\}$$

Preferences :

Person	Preference
1	$\{1, 2\}$
2	$\{2, 3\}$

Solutions

1	3
2	4

Person	Score
1	1 + 0
2	1 + 1

3

1	2
3	4

Person	Score
1	1 + 1
2	1 + 1

4

Tracks

1	2
4	3

Times

Person	Score
1	1+1
2	0+1

3

Comments

- To calculate score:

  $\boxed{\text{for every time, you get 1 point for every person that has at least one interesting presentation to watch.}}$

- Any programming language and resource allowed.
- How many schedules are there?

$$\boxed{\frac{(\text{tracks} \cdot \text{times})!}{(\text{tracks}!)^{\text{times}} \cdot \text{times}!}}$$

Small $\dfrac{(3 \cdot 15)!}{(15!)^3 \cdot 3!} \cong 10^{32}$

large $\dfrac{(8 \cdot 15)!}{(15!)^8 \cdot 8!} \cong 10^{117}$

SONAT


Good luck!
I should look at repo now.

Solution approaches

Solution approaches

① Mixed Integer Programming (MIP) model

- optimal solution, different solvers, super-flexible approach, early termination

② Heuristics

- can use MIP model, observing humans solve, etc

③ Metaheuristics

- applicable to many similar problems

Mixed Integer Programming

Mixed Integer Programming

General

maximize $\sum_i c_i x_i$

subject to $\sum_j a_{ij} x_j \leq b_i \quad \forall i$

$x_j \geq 0 \quad \forall j$

$x_j \in \mathbb{R}$ or $x_j \in \mathbb{Z} \quad \forall j$

Problem data

Real numbers (polynomial)

Integers (NP-hard)

Example

maximize $4x_1 + 2x_2$

subject to

$x + 2y \leq 8$

$x + 0y \leq 4$

$x_1 \in \mathbb{R}$

$x_2 \in \mathbb{Z}$

$x_1, x_2 \geq 0$

MIP example

Days j

People i

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

Availability matrix a_{ij}

$$x_{ij} = \begin{cases} 1 & \text{if person } i \text{ assigned to day } j \\ 0 & \text{otherwise} \end{cases}$$

A personal trainer asks her clients when they are available.
Maximize appointments.

MIP model

maximize $\sum_i \sum_j a_{ij} x_{ij}$

subject to

$x_{ij} \le a_{ij} \quad \forall\, i,j$

At most D_i appointments/person i → $\sum_j x_{ij} \le D_i \quad \forall\, i$

At most 2 appointments/day → $\sum_i x_{ij} \le 2 \quad \forall\, j$

MIPs

- Extremely expressive
- Ridiculously underused
- Commercial solvers available: CPLEX, GUROBI
- Always create a MIP
  - common language
  - insight via optimal solutions to small problems
  - intelligent use: warm start, early termination, column generation, ...

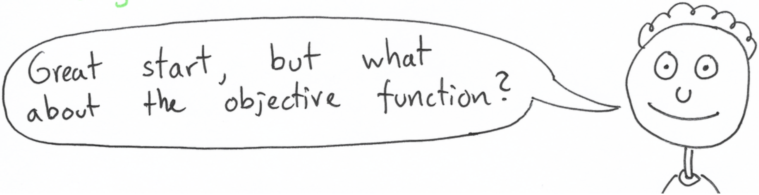
Vehicle routing
Inventory planning
Workforce planning
Shift scheduling
Supply chain
Facility location
Timetabling
Traveling salesman
Time-window routing

Conference planning as a MIP

$$x_{pj} = \begin{cases} 1 & \text{if presentation } p \text{ assigned to time } j \\ 0 & \text{otherwise} \end{cases}$$

Each time j must have "tracks" presentations assigned to it. $\rightarrow \sum_p x_{pj} = \text{tracks} \quad \forall j$

Each presentation can be assigned to one and only one time. $\rightarrow \sum_j x_{pj} = 1 \quad \forall p$



Conference planning as a MIP ( )

$$C_{kp} = \begin{cases} 1 & \text{if person } k \text{ is interested in presentation } p \\ 0 & \text{otherwise} \end{cases}$$

Problem data

$$\sum_p C_{kp} X_{pj} = \text{number of presentations that person } k \text{ is interested in in time } j$$

Introduce new variables

y_{kj} = score assigned to person k in time j

$y_{kj} \in \{0, 1\} \quad \forall\ k, j$

To model

$$\sum_p C_{kp} X_{pj} = 0 \Rightarrow y_{kj} = 0$$

add constraints

$$y_{kj} \leq \sum_p C_{kp} X_{pj} \quad \forall\ k, j$$

Conference planning as a MIP

maximize $\sum_k \sum_j y_{kj}$ ← Score assigned to person k in time j

subject to

$$\sum_p x_{pj} = \text{tracks} \quad \forall j$$ ← Times

$$\sum_j x_{pj} = 1 \quad \forall p$$ ← Presentations

The objective is to set these to 1, which is only possible if $\sum_p c_{kp} x_{pj} \geq 1$.

$$y_{kj} \leq \sum_p c_{kp} x_{pj} \quad \forall k, j$$

$$y_{kj}, x_{pj} \in \{0, 1\}$$

The full specification is 5 lines of math.
The solver does the rest!

SONAT

Possibilities when modeling MIPs

- linearize logical expressions { IF, IF-ELSE, AND, OR, XOR, k-out-of-n
- approximate convex/non-convex objectives

- optimize over ℓ_1 and ℓ_∞ norms
- optimize fractional objectives

Any LP? Any NP-hard problem?
Are MIPs ever <u>not</u> the solution?

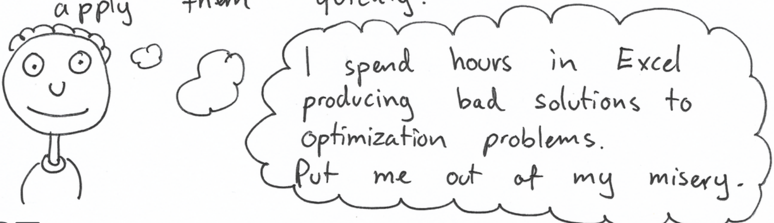
For practical problems, MIPs are bad:
- when the problem is "too easy" (sorting)
- when the problem is "too hard" (too large)
- when more specialized solvers are more appropriate (SAT, TSP)

Heuristics and greedy algorithms

Heuristics

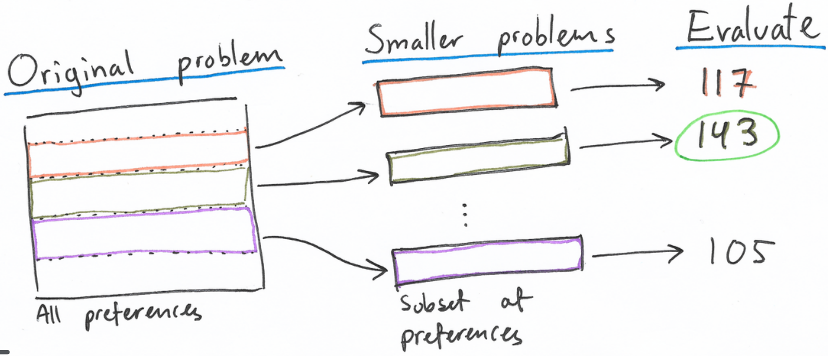
metric TSP

- no guarantee of optimality
  (but some algorithms can be bounded by factor)
- on small instances, comparing heuristics with optimal solution gives insights
- watching human solve can be insightful
- even bad heuristics can outperform humans since a computer can apply them quickly.



Heuristic: sample preferences

- sample a subset of preferences such that the instance is small enough to be solved by MIP
- repeat many times



Heuristic : maximize minimum frequency

F_p = frequency (count) of presentation p among the preferences

Assume 4 presentations with $F = \{1, 1, 100, 100\}$

Times j — minimal sum is 2

100	1
100	1

Times j — minimal sum is 101

100	100
1	1

← Much better solution in the worst case.

min $100 + 1 = 101$ $\qquad$ $100 + 100 = 200$

max $200 + 2 = 202$ $\qquad$ $101 + 101 = 202$

Idea: maximize minimal sum of frequencies in each time j
(we get an approximation ratio too)

MIP model: maximize minimum frequency

$$X_{pj} = \begin{cases} 1 & \text{if presentation } p \text{ is assigned to time } j \\ 0 & \text{otherwise} \end{cases}$$

$$\sum_p F_p X_{pj} = \text{sum of frequencies in time } j$$

Introduce a variable y and add constraints

$$y \leq \sum_p F_p X_{pj} \quad \forall j$$

and then maximize y.

MIP model: maximize minimum frequency

maximize y

subject to

$$\sum_p x_{pj} = \text{tracks} \quad \forall j$$

$$\sum_j x_{pj} = 1 \quad \forall p$$

Same as earlier.

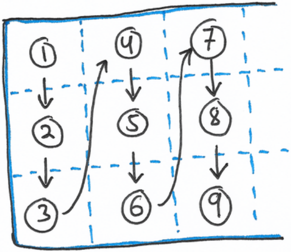
"minimax" → $$y \le \sum_p F_p x_{pj} \quad \forall j$$

$$x_{pj} \in \{0,1\}$$

$$y \in \mathbb{R}$$

Variable types
x_{pj} = binary
y = real

Greedy algorithm : choose best presentation



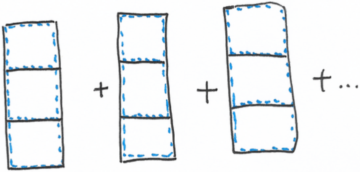
Iterate over time slots, choose the presentation that increases the objective function value (score) the most at each time slot.

Number of choices : $P + (P-1) + (P-2) + \ldots + 1 = \frac{P(P+1)}{2}$

Extensions

- Fixate first presentation in each time j
- Add probabilities : instead of choosing best presentation, choose a random presentation weighted by how much it increases score (then run the algorithm many times)

Greedy algorithm: optimize time-by-time



Iterate over times, look at every possible combination of presentations. Choose the one that increases objective (score) the most.

Number of choices: $\binom{P}{3} + \binom{P-3}{3} + \binom{P-6}{3} + \dots$

Extensions

- Fixate first presentation in each time j
- Add probabilities, or only consider a subset of the $\binom{P}{3}$ combinations at each step

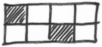
Local search and metaheuristics

Local search

1. Produce an initial solution
2. Define a set of local moves that might improve the solution
3. How to use the moves (local heuristics) can be defined using a metaheuristic

random initial solution or greedy solution, etc

Local moves



swap 2 random

swap k (permute)
1→2 2→3 3→1

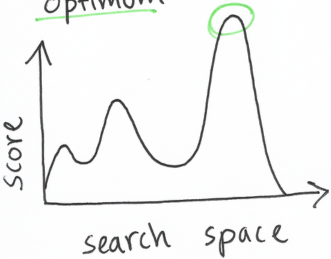
swap from times with lowest score

Score: 1 5 2 7

many possibilities for local moves. be creative

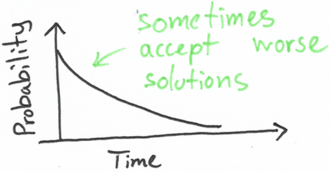
Metaheuristics

- The problem:
  sequence of good local moves do not necessarily lead to global optimum



Approaches

- randomized restarts/moves
- simulated annealing



- tabu search
  avoid cycling back to previous solutions
- multi-armed bandit
  when a move does well, increase likelihood of using it again in the future.

Summary

Practical summary

- check if the problem can be reduced to a known problem
- try formulating it as a MIP or SAT if possible
- if the problem is too large, try an approximation algorithm (bound on performance)
- if no approximation algorithm is available, try heuristics

Proposition: a MIP solver will do amazingly well on a wide range of real world business problems.

Algorithm	Small	Large
Mean of random solutions	984	50043
Best of 100 random solutions	1066	51449
Exact MIP model (1 min time limit)	1185	N/A
Heuristic MIP model (1 min time limit)	930	50064
Greedy time slot by time slot (track-by-track)	1167	54604
Greedy time slot by time slot (time-by-time)	1136	52861
Greedy time by time	1157	N/A
Simulated annealing	1198	54638

References

- The algorithms perspective
  - ***Algorithms*** by Dasgupta et al.
  - *Introduction to algorithms* by Cormen et al
  - *The Algorithm Design Manual* by Skiena
- The operations research perspective (discrete/real)
  - ***Optimization modeling*** by Sarker et al.
  - *Applied integer modeling* by Chen et al
  - *Linear and integer optimization* by Sierksma
  - Stackexchange, forums and the blog by Kalvelagen are also good.
- The numerical perspective (real)
  - *Convex Optimization* by Boyd et al
  - *Numerical optimization* by Wright et al
- The AI perspective
  - *Artificial Intelligence* by Norvig et al.