DM841 (10 ECTS - autumn semester)

Heuristics and Constraint Programming for Discrete Optimization

[Heuristikker og Constraint Programmering for Diskret Optimering] (Gamle DM811 + DM826)

Marco Chiarandini
lektor, IMADA
www.imada.sdu.dk/~marco/DM841

Problems with Constraints

Social Golfer Problem

- 9 golfers: 1, 2, 3, 4, 5, 6, 7, 8, 9
- wish to play in groups of 3 players in 4 days
- such that no golfer plays in the same group with any other golfer more than just once.

Is it possible?

Problems with Constraints

Social Golfer Problem

- 9 golfers: 1, 2, 3, 4, 5, 6, 7, 8, 9
- wish to play in groups of 3 players in 4 days
- such that no golfer plays in the same group with any other golfer more than just once.

Is it possible?

	Group 1	Group 2	Group 3
Day 0	? ? ?	? ? ?	? ? ?
Day 1	? ? ?	? ? ?	? ? ?
Day 2	? ? ?	? ? ?	? ? ?
Day 3	? ? ?	? ? ?	? ? ?

Solution Paradigms

- Dedicated algorithms
- Integer Programming (DM545/DM554)
- Constraint Programming:
- Local Search & Metaheuristics
- Others (SAT, etc)

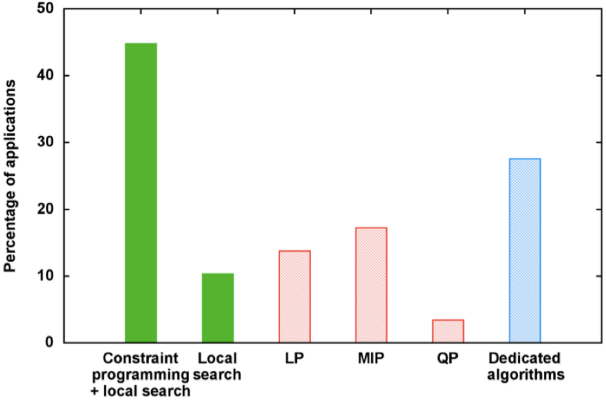
Solution Paradigms

- Dedicated algorithms
- Integer Programming (DM545/DM554)
- Constraint Programming:

  representation (language) + reasoning (search + propagation)
- Local Search & Metaheuristics
- Others (SAT, etc)

Applications

Distribution of technology used at Google for optimization applications developed by the operations research team



[Slide presented by Laurent Perron on OR-Tools at CP2013]

Constraint Programming

Modelling in MIP

Modelling in CP

Constraint Programming

Modeling

integer variables:
$X_{p,g}$ variable whose values are from the domain $\{1, 2, 3\}$

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}

Constraint Programming

Modeling

integer variables:
$X_{p,g}$ variable whose values are from the domain $\{1,2,3\}$

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}

- each group has exactly groupSize players
- each pair of players only meets once

Constraint Programming

Modeling

integer variables:
$X_{p,g}$ variable whose values are from the domain $\{1, 2, 3\}$

set variables:
$X_{g,d}$ variable whose values are subsets of $\{1, 2, ..., 9\}$

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}

Golfers

	Group 1	**Group 2**	**Group 3**
Day 0	0 1 2		
Day 1			
Day 2			
Day 3			

- each group has exactly groupSize players
- each pair of players only meets once

Constraint Programming

Modeling

integer variables:
$X_{p,g}$ variable whose values are from the domain $\{1, 2, 3\}$

set variables:
$X_{g,d}$ variable whose values are subsets of $\{1, 2, ..., 9\}$

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}

- each group has exactly groupSize players
- each pair of players only meets once

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2		
Day 1			
Day 2			
Day 3			

- In each day, groups must be disjoint and contain all players
- at most one player overlaps between groups

Constraint Programming

Model with Integer Variables

```
players = 9;
groupSize = 3;
days = 4;

groups = players/groupSize;

# === Variables ===============
assign = m.intvars(players * days, 0, groups-1)
schedule = Matrix(players, days, assign)

# === Constraints =============
# C1: Each group has exactly groupSize players
for d in range(days):
   m.count(schedule.col(d), [groupSize, groupSize, groupSize]);

# C2: Each pair of players only meets once
p_pairs = [(a,b) for a in range(players) for b in range(players) if p1<p2]
d_pairs = [(a,b) for a in range(days) for b in range(days) if d1<d2]
for (p1,p2) in p_pairs:
   for (d1,d2) in d_pairs:
      b1 = m.boolvar()
      b2 = m.boolvar()
      m.rel(assign(p1,d1), IRT_EQ, assign(p2,d1), b1)
      m.rel(assign(p1,d2), IRT_EQ, assign(p2,d2), b2)
      m.linear([b1,b2], IRT_LQ, 1)

m.branch(assign, INT_VAL_MIN_MIN, INT_VAL_SPLIT_MIN)
```

Constraint Programming

Model with Set Variables

```
p = 9 # number of players
g = 3 # number of groups
w = 4 # number of days

s = p/g # size of groups

# === Variables ===============
groups = m.setvars(g*w, intset(), 0, p-1, s, s)
schedule = Matrix(g, w, groups)
allPlayers = m.setvar(0, p-1, 0, p)

# === Constraints =============
# In each day, groups must be disjoint and contain all players
for i in range(g):
z1 = m.setvars(g, intset(), 0, p-1, 0, p)
m.rel(SOT_DUNION, schedule[i].row(i), z1[i])
m.rel(z1[i], SRT_EQ, allPlayers)

# at most one player overlaps between groups
for i,j in itertools.combinations(range(g*w), 2):
    z2 = m.setvar(intset(), 0, p-1, 0, p))
    m.rel(groups[i], SOT_INTER, groups[j], SRT_EQ, z2)
    m.cardinality(z2, 0, 1)

m.branch(groups, SET_VAR_MIN_MIN, SET_VAL_MIN_INC);
```

Constraint Programming

Solution: Assign and Propagate

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2		
Day 1			
Day 2			
Day 3			

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	{2,3}	{1,2,3}	{1,2,3}	{1,2,3}

Constraint Programming

Solution: Assign and Propagate

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2	3 4 5	
Day 1			
Day 2			
Day 3			

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	{3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	{3}	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	{3}	{1,2,3}	{1,2,3}	{1,2,3}

Constraint Programming

Solution: Assign and Propagate

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2	3 4 5	6 7 8
Day 1			
Day 2			
Day 3			

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 1	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 3	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	3	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	3	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	3	{1,2,3}	{1,2,3}	{1,2,3}

Constraint Programming

Solution: Assign and Propagate

Groups

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2	3 4 5	6 7 8
Day 1	0		
Day 2			
Day 3			

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	1	{1,2,3}	{1,2,3}
Golfer 1	1	{2,3}	{1,2,3}	{1,2,3}
Golfer 2	1	{2,3}	{1,2,3}	{1,2,3}
Golfer 3	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	3	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	3	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	3	{1,2,3}	{1,2,3}	{1,2,3}

Constraint Programming

Solution: Assign and Propagate

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2	3 4 5	6 7 8
Day 1	0	1	
Day 2			
Day 3			

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	1	{1,2,3}	{1,2,3}
Golfer 1	1	2	{1,2,3}	{1,2,3}
Golfer 2	1	{3}	{1,2,3}	{1,2,3}
Golfer 3	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 4	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 5	2	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 6	3	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 7	3	{1,2,3}	{1,2,3}	{1,2,3}
Golfer 8	3	{1,2,3}	{1,2,3}	{1,2,3}

Constraint Programming

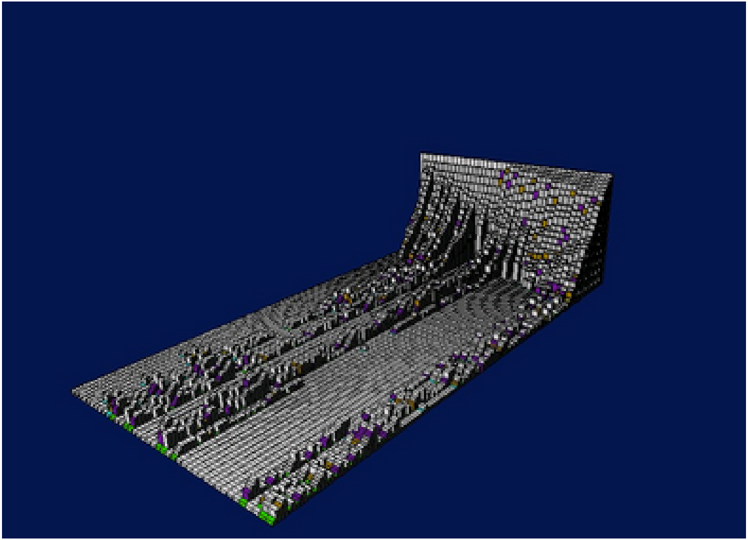
Solution: Assign and Propagate

Golfers

	Group 1	Group 2	Group 3
Day 0	0 1 2	3 4 5	6 7 8
Day 1	0 3 6	1 4 7	2 5 8
Day 2	0 4 8	1 5 6	2 3 7
Day 3	0 5 7	1 3 8	2 4 6

Groups

	Day 0	Day 1	Day 2	Day 3
Golfer 0	1	1		
Golfer 1	1	2		
Golfer 2	1	{3}		
Golfer 3	2			
Golfer 4	2			
Golfer 5	2			
Golfer 6	3			
Golfer 7	3			
Golfer 8	3			



Effect of constraint propagation on the domains
of variables during search in a placement problem.

Local Search

Solution: Trial and Error

	Group 1	**Group 2**	**Group 3**
Day 0	0 1 2	3 4 5	6 7 8
Day 1	**0 4** 6	1 **3 7**	2 5 8
Day 2	**0 4** 8	1 5 6	2 **3 7**
Day 3	0 5 7	1 3 8	2 4 6

Heuristic algorithms: compute, efficiently, good solutions to a problem (without caring for theoretical guarantees on running time and approximation quality).

Contents: Constraint Programming

- Modelling and Applications
  Integer variables, set variables, float variables, constraints
- Principles Consistency levels
- Filtering Algorithms
  Alldifferent, cardinality, regular expressions, etc.
- Search:
  Backtracking, Strategies
- Symmetry Breaking
- Restart Techniques
- Programming
  Gecode (C++)

Contents: Heuristics

- Construction Heuristics
- Local Search
- Metaheuristics
  - Simulated Annealing
  - Iterated Local Search
  - Tabu Search
  - Variable Neighborhood Search
  - Evolutionary Algorithms
  - Ant Colony Optimization
- Programming
  EasyLocal (C++)

Aims & Contents

- modeling problems with constraint programming
- design heuristic algorithms
- implement the algorithms
- assess the programs
- describe with appropriate language
- look at different problems

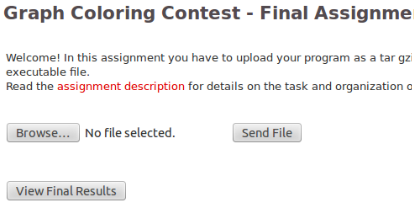
Course Formalities

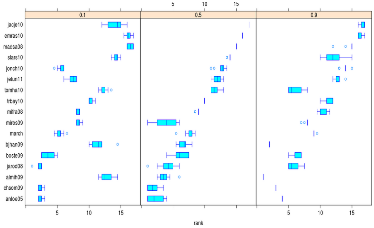
Prerequisites:	✓ Algorithms and data structures (DM507) ✓ Programming (DM502, DM503, DM550)
Credits:	10 ECTS
Language:	English and Danish
Classes:	intro phase $2h \times 24$; training phase $2h \times 10$
Material:	slides + articles + lecture notes + starting code

Assessment (10 ECTS)

5 obligatory assignments:

- individual
- deliverables: program + short written report
- graded with external censor,
  final grade given by weighted average





DM841 (10 ECTS - autumn semester)

Heuristics and Constraint Programming for Discrete Optimization

[Heuristikker og Constraint Programmering for Diskret Optimering] (Gamle DM811 + DM826)

Marco Chiarandini
lektor, IMADA
www.imada.sdu.dk/~marco/DM841