Chapter 9: Choosing the Model (Sports Scheduling)

Helmut Simonis

Cork Constraint Computation Centre
Computer Science Department
University College Cork
Ireland

ECLiPSe ELearning Overview

Licence

Outline

What we want to introduce

- How to come up with a model for a problem
- Why choosing a good model is an art
- Channeling
- Projection
- Redundant constraints

Sports Scheduling

Tournament Planning

We plan a tournament with 8 teams, where every team plays every other team exactly once. The tournament is played on 7 days, each team playing on each day. The games are scheduled in 7 venues, and each team should play in each venue exactly once.

As part of the TV arrangements, some preassignments are done: We may either fix the game between two particular teams to a fixed day and venue, or only state that some team must play on a particular day at a given venue. The objective is to complete the schedule, so that all constraints are satisfied.

Example

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

Solution

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		6, 8		1, 2	5, 7		3, 4
Day 2	2, 3	1, 5			4, 8	6, 7	
Day 3	1, 7	2, 4	3, 8				5, 6
Day 4			4, 7		2, 6	3, 5	1, 8
Day 5	5, 8			3, 6		1, 4	2, 7
Day 6		3, 7	1, 6	4, 5		2, 8	
Day 7	4, 6		2, 5	7, 8	1, 3		

A More Abstract Formulation

Rooms Puzzle, (Thomas G. Room, 1955)

Place numbers 1 to 8 in cells so that each row and each column has each number exactly once, each cell contains either no numbers or two numbers (which must be different from each other), and each combination of two different numbers appears in exactly one cell.

Puzzle presented by R. Finkel

How to come up with a model

- What are the variables/what are their values?
- How can we express the constraints?
- Do we have these constraints in our system?
- Does this do good propagation?
- Backtrack to earlier step as required

Requirements

1. There are 8 teams, seven days and seven locations
2. Each team plays each other team exactly once
3. Each team plays 7 games (redundant)
4. Each team plays in each location exactly once
5. Each team plays on each day exactly once
6. A game consists of two (different) teams
7. There are four games on each day (redundant)
8. There are four games at each location (redundant)
9. In any location there is atmost one game at a time

Idea 1

- Matrix *Day* $\times$ *Game* (7×4)
- Each cell contains two variables, denoting teams
- Easy to say that team plays once on each day, `alldifferent`
- Columns don't have significance
- Model does not mention location, how to add this?
- How to express that each team plays each other once?

Idea 2, Change problem structure

- Matrix of *Day* $\times$ *Location* (7×7)
- Each cell contains two variables, each denoting a team
- How do we avoid symmetry inside cell?
- Need special value (0) to denote that there is no game
- In one cell, either both or none of the variables are 0
- Easy to say that each row and column contains each team exactly once
- Except for value 0, can not use `alldifferent`
- Link between two variables in cell to state that game needs two different teams
- How to express that each (ordered) pair occurs exactly once?

Idea 3, Add location variables

- Model as in Idea 1, matrix $Day \times Game$
- Each cell contains two variables for teams and one for location
- Easy to state that games on one day are in different locations
- How to express condition that each team plays in each location once?
- Also, how to express that each team plays each other exactly once?

Cork Constraint Computation Centre

Problem
Model
Program
Search
Redundant Modelling
Exploring Ideas
Expanding Idea 7
Comparing Ideas
Channeling
Selected Model

Idea 4, Use variables for pairs

- Matrix $Day \times Location$
- Each cell contains one variable ranging over (sorted) pairs of teams, and special value 0 (no game)
- Each pair value occurs once, except for 0
  - Special constraint `alldifferent0`
  - Or use `gcc`
- How to state that each team plays once per day?
- How to state that each team plays in each location?

Cork Constraint Computation Centre

Idea 5: If all else fails, use binary variables

- Binary variable stating that team *i* plays in location *j* at day *k*
- Three dimensional matrix
- Each team plays once on each day
- Each team plays once in each location
- Each game has two (different) teams, needs auxiliary variable
- Each pair of team meets once, needs auxiliary variables

Idea 6: An even bigger binary model

- Use four dimensions
- Team *i* meets team *j* in location *k* on day *l*
- 3136 = 8*8*7*7 variables
- Constraints all linear
- Why use finite domain constraints?

Idea 7: A different mapping

- Each team plays each other exactly once, one variable for each combination (8*7/2=28 variables)
- Decide when and where this game is played, values range over combinations of days and locations (7*7=49 values)
- All variables must be different (no two games at same time and location)
- Each team plays 7 games, by construction
- How to express that each team plays once per day?
- How to express that each team plays in each location once?

Expand Idea 7 into Full Model

Numbering Values

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Four games on each day

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

- Day 1 corresponds to values 1..7
- Four variables can take these values
- Day 2 corresponds to values 8..14, etc
- One constraint per day
- Exactly four of all variables take their value in the set ...
- Seven such constraints

Four games at each location

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

- City 1 corresponds to values
  - 1, 8, 15, 22, 29, 36, 43
- Four variables can take these values
- City 2 corresponds to values
  - 2, 9, 16, 23, 30, 37, 44
- One constraint per location
- Exactly four of all variables take their value in the set ...
- Seven such constraints over 28 variables each

Teams plays once on a day (at a location)

- Select those variables which correspond to Team *i*
- Exactly one of those variables takes its value in the set 1..7
- Same for all other days
- Same for all other teams
- 56 Constraints over 7 variables each
- Similar for teams and locations, another 56 constraints

Are we there yet?

- 28 variables with 49 possible values
- 1 alldifferent
- 7 exactly constraints over all variables (Days)
- 7 exactly constraints over all variables (Locations)
- 56 exactly constraints over 7 variables each (Days)
- 56 exactly constraints over 7 variables each (Locations)
- Forgotten anything?
- Check the requirements

Do we satisfy the requirements?

1. There are 8 teams, seven days and seven locations
2. Each team plays each other team exactly once
3. Each team plays 7 games (redundant)
4. Each team plays in each location exactly once
5. Each team plays on each day exactly once
6. A game consists of two (different) teams
7. There are four games on each day (redundant)
8. There are four games at each location (redundant)
9. In any location there is atmost one game at a time

What about the `exactly` constraint?

- ECLiPSe doesn't provide this constraint
  - Other system might do, could switch system
- Implement it
  - Extend `gcc` to allow multiple values
  - Should be last resort
- Emulate constraint with others

Idea 8: Mapping games to days and locations

- For each game to be played, we have two variables
  - One ranges over the days
  - The other over the locations
- Easy to state that there are four games per day an location
- Easy to state that each team plays once per day and location
- How do we express that no two games are played at the same location and the same time?
  - If we had an `alldifferent` over pairs of variables...
  - Not in ECLiPSe

We have four games on each day

- Each row value is taken four times amongst the variables
- `gcc([gcc(4,4,1),...,gcc(4,4,7)],Rows)`
- Similar for columns:
- `gcc([gcc(4,4,1),...,gcc(4,4,7)],Cols)`

Reminder: `gcc(Pattern, Variables)`

- `gcc` *global cardinality constraint*
- `Pattern` is list of terms `gcc(Low, High, Value)`
- The overall number of variables taking value `Value` is between `Low` and `High`
- Generalization of `alldifferent`
- Domain consistent version in ECLiPSe

Each team plays once per day

- For the seven variables which describe games of a team
- Each row value is taken exactly once amongst the variables
- Could use `gcc([gcc(1,1,1),...,gcc(1,1,7)],Vars)`
- But `alldifferent(Vars)` is more compact
- Similar for columns

Cork Constraint Computation Centre

How do the models differ?

Idea	Mapping
1	$D \times G \times \{f, s\} \rightarrow T$
2	$D \times L \times \{f, s\} \rightarrow T \cup \{0\}$
3	$D \times G \times \{f, s\} \rightarrow T$ $D \times G \rightarrow L$
4	$D \times L \rightarrow T \triangle T \cup \{0\}$
5	$T \times D \times L \rightarrow \{0, 1\}$
6	$T \times T \times D \times L \rightarrow \{0, 1\}$
7	$T \triangle T \rightarrow D \times L$
8	$T \triangle T \rightarrow D$ $T \triangle T \rightarrow L$

D	Days
T	Teams
L	Locations
G	Games

Cork Constraint Computation Centre

Requirements Capture

Idea	Requirement								
	1	2	3	4	5	6	7	8	9
1	N	?	Y	?	Y	Y	Y	?	?
2	C	?	Y	Y	Y	Y	Y	Y	Y
3	C	?	Y	?	Y	Y	Y	Y	Y
4	C	Y	Y	Y	Y	Y	Y	Y	Y
5	C	NL	L	L	L	NL	L	L	NL
6	C	L	L	L	L	L	L	L	L
7	C	C	C	E	E	C	E	E	A
8	C	C	C	A	A	C	G	G	?

Comments on models

Idea	Main point
1	missing locations, first second symmetry
2	spare value, first second symmetry
3	first second symmetry
4	spare value
5	0/1, non-linear constraints
6	0/1, large matrix
7	needs exactly constraint
8	needs alldifferent on tuples

Viewpoints and Channeling

- Instead of expressing all constraints over one set of variables
- Use multiple sets of variables (*viewpoints*)
- Decide which constraint to express over which variables
- Allows more freedom on how to express problem
- Link the different variables with *channeling* constraints

Cork Constraint Computation Centre

In Our Case

- Combine ideas 7 and 8
- One set of variables ranging over pairs
- Another using two variables per game for day and location
- How to combine variables?
- Minimize loss of information

Cork Constraint Computation Centre

Projection

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

- Link pair variables to row and column variables
- Pair variable uses cell numbers 1-49 as values
- Row and column variables indicate on which day (row) and in which location (column) the game is played
- Pair value 23 = row 4, column 2
- `element` constraint to link the variables
- Two projections from $D \times L$ space onto D and L

Cork Constraint Computation Centre

Mapping cells to rows and columns

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

```
element(Cell,[1,1,1,1,1,1,1,2,2,2,2,2,2,2,3,3,3,3,3,3,3,
              4,4,4,4,4,4,4,5,5,5,5,5,5,5,6,6,6,6,6,6,6,
              7,7,7,7,7,7,7],Row),
element(Cell,[1,2,3,4,5,6,7,1,2,3,4,5,6,7,1,2,3,4,5,6,7,
              1,2,3,4,5,6,7,1,2,3,4,5,6,7,1,2,3,4,5,6,7,
              1,2,3,4,5,6,7],Col),
```

Cork Constraint Computation Centre

Mapping cells to rows and columns

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

```
element(23  ,[1,1,1,1,1,1,1,2,2,2,2,2,2,2,3,3,3,3,3,3,3,
             4,4,4,4,4,4,4,5,5,5,5,5,5,5,6,6,6,6,6,6,6,
             7,7,7,7,7,7,7],4  ),
element(23  ,[1,2,3,4,5,6,7,1,2,3,4,5,6,7,1,2,3,4,5,6,7,
             1,2,3,4,5,6,7,1,2,3,4,5,6,7,1,2,3,4,5,6,7,
             1,2,3,4,5,6,7],2  ),
```


Channeling Constraints

- This is one common type, a *projection*
- Another common type is the *inverse*
  - Link a variable $A \to B$ to another $B \to A$
  - Typically used for bijective mappings
  - Built-in `inverse/2`
- Also used: *Boolean* channeling
  - Link variables $A \to B$ and $A \times B \to \{0, 1\}$
  - Built-in `bool_channeling/3`

Selected Model

- Two sets of variables (Req 1, 2, 3, 6, by construction)
- Pair variables ($T \triangle T \to D \times L$)
  - `alldifferent` (Req 9)
- Day and Location variables ($T \triangle T \to D$), ($T \triangle T \to L$)
  - `gcc` (Req 4, 5)
  - `alldifferent` (Req 7, 8)
- Channeling Constraints
  - `element` projection from pairs onto rows and columns
- Search only on pair variables

Handling of hints (I)

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

- This value (17) can not be used by pairs not involving team 8
- One of the pairs involving team 8 must use this value (17)

Handling of hints (II)

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

- The pair involving teams 5 and 7 must take value 5, fixes variable

Problem Data

```
hint(1,8,[2-[8],5-[5,7],8-[2],9-[1,5],15-[7],
          17-[8],26-[2],27-[5],28-[1],29-[8],
          34-[1],39-[4,5],43-[4],47-[1,3]]).
```

Main Program

```
top(Problem,L):-
    hint(Problem,N,Hints),
    N1 is N-1,
    N2 is N//2,
    NrVars is N*N1//2,
    SizeDomain is N1*N1,
    length(L,NrVars),
    L :: 1..SizeDomain,
    create_pairs(N,Contains,Names),
    ic_global_gac:alldifferent(L),
    process_hints(L,Contains,Hints),
    ...
```


Main Program (continued)

```
    project_row_cols(L,N1,Rows,Cols),
    limit(Rows,N2,N1),
    limit(Cols,N2,N1),
    separate(Contains,Rows,N,SplitRows),
    separate(Contains,Cols,N,SplitCols),
    (foreach(K,SplitRows) do
        ic_global_gac:alldifferent(K)
    ),
    (foreach(K,SplitCols) do
        ic_global_gac:alldifferent(K)
    ),
    search(L,0,input_order,indomain,
           complete,[]).
```

Create Pairs and Names

```
create_pairs(N,Contains,Names):-
    (for(I,1,N-1),
     fromto(Names,A1,A,[]),
     fromto(Contains,B1,B,[]),
     param(N) do
        (for(J,I+1,N),
         fromto(A1,[Name|AA],AA,A),
         fromto(B1,[I-J|BB],BB,B),
         param(I) do
            concat_string([I,J],Name)
        )
    ).
```


Projecting Rows and Columns

```
project_row_cols(L,N,Rows,Cols):-
    generate_tables(N,RowTable,ColTable),
    (foreach(X,L),
     foreach(R,Rows),
     foreach(C,Cols),
     param(RowTable,ColTable) do
        element(X,RowTable,R),
        element(X,ColTable,C)
    ).
```

Generating Projection Tables

```
generate_tables(N,RowTable,ColTable):-
     (for(I,1,N),
      fromto(RowTable,A1,A,[]),
      fromto(ColTable,B1,B,[]),
      param(N) do
          (for(J,1,N),
           fromto(A1,[I|AA],AA,A),
           fromto(B1,[J|BB],BB,B),
           param(I) do
               true
          )
     ).
```


Extract row variables

```
separate(Contains,Rows,Values,SplitRows):-
     (for(Value,1,Values),
      foreach(SplitRow,SplitRows),
      param(Contains,Rows) do
          (foreach(A-B,Contains),foreach(V,Rows),
           fromto([],R,R1,SplitRow),param(Value) do
               (memberchk(Value,[A,B]) ->
                    R1 = [V|R]
               ;
                    R1 = R
               )
          )
     ).
```

Set up gcc constraint

```
limit(L,Bound,Values):-
    (for(I,1,Values),
     foreach(gcc(Bound,Bound,I),Pattern),
     param(Bound) do
        true
    ),
    gcc(Pattern,L).
```


Problem
Model
Program
Search
Redundant Modelling

Setting up hints

```
process_hints(L,Contains,Hints):-
    (foreach(Pos-Values,Hints),
     param(L,Contains) do
        process_hint(Pos,Values,L,Contains)
    ).

process_hint(Pos,[A,B],L,Contains):- % clause 1
    !,
    match_hint(A-B,Contains,L,X),
    X #= Pos.
```

Setting up hints

```
process_hint(Pos,[Value],L,Contains):- % clause 2
    (foreach(X,L),
     foreach(A-B,Contains),
     fromto([],R,R1,Required),
     param(Pos,Value) do
        (not_mentioned(A,B,Value) ->
            X #\= Pos,
            R1 = R
        ;
            R1 = [X|R]
        )
    ),
    occurrences(Pos,Required,1).
```

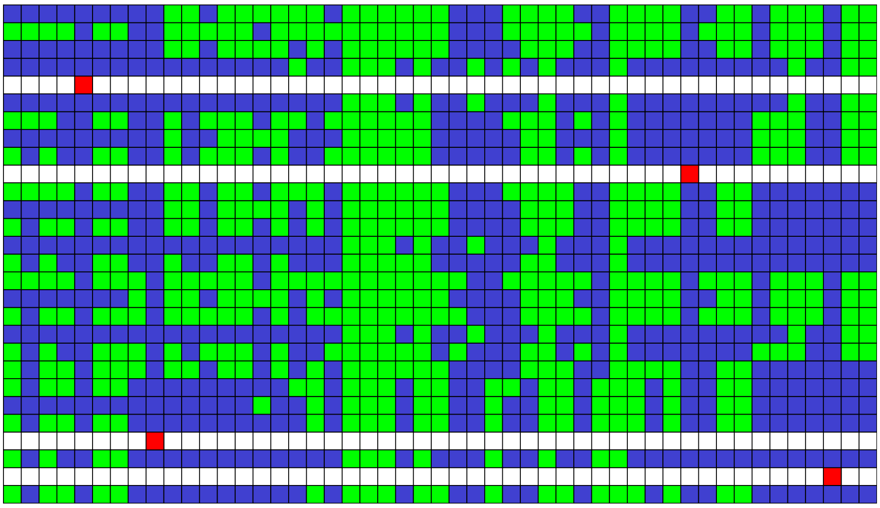

Setting up hints

```
not_mentioned(A,B,V):-
    A \= V,
    B \= V.

match_hint(H,[H|_],[X|_],X):-
    !.
match_hint(H,[_|T],[_|R],X):-
    match_hint(H,T,R,X).
```

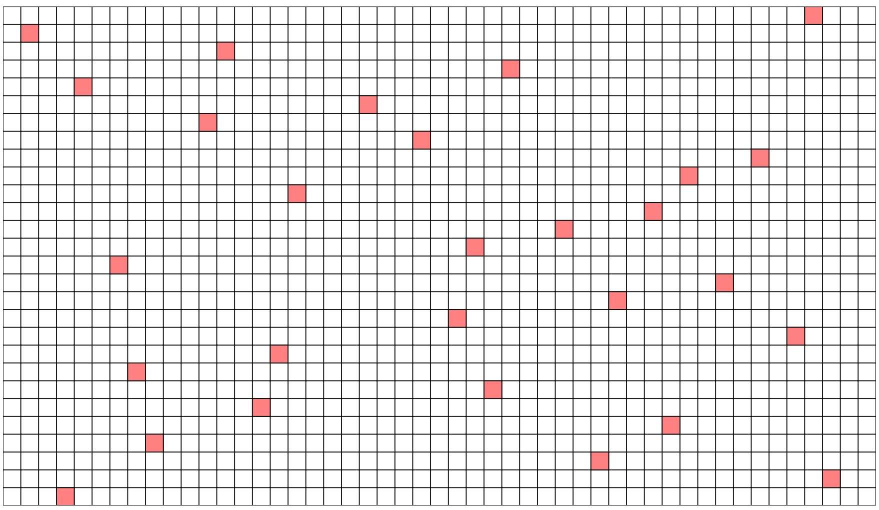
Before Search

Values

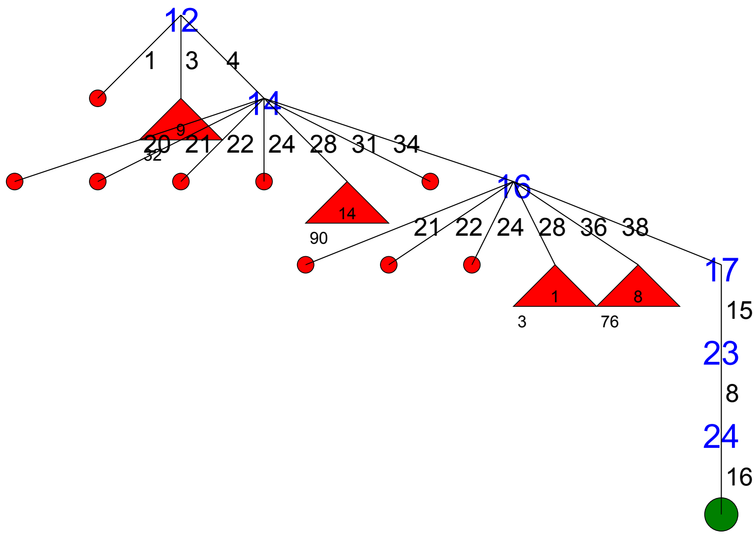


Vars

Solution



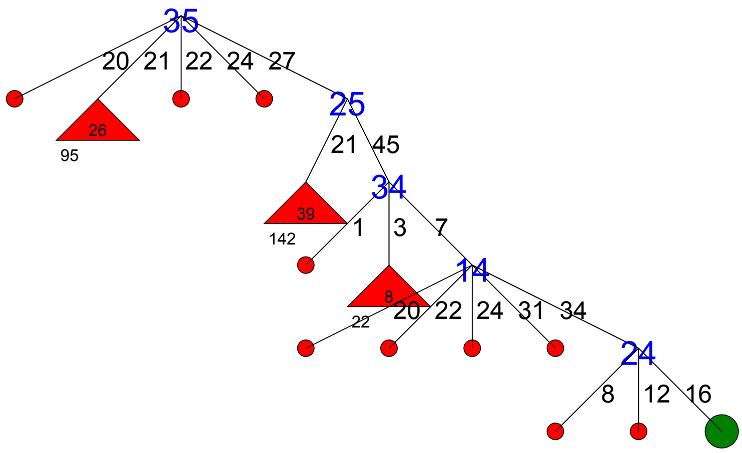
Search Tree with input order



How to improve?

- Try different search strategy
- Use `first_fail` dynamic variable selection

Search Tree with first fail

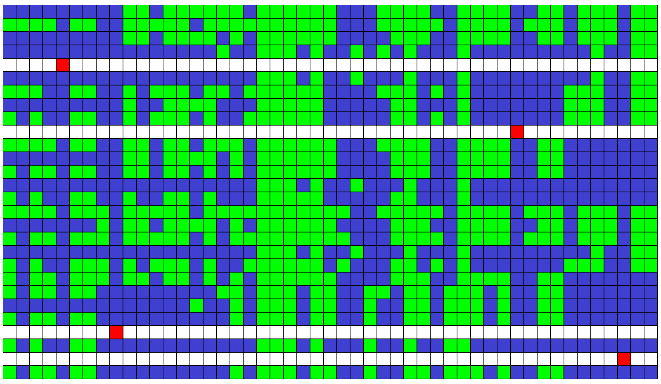


Observation

- It does not work
- Search tree is slightly larger than before!

Missing Propagation

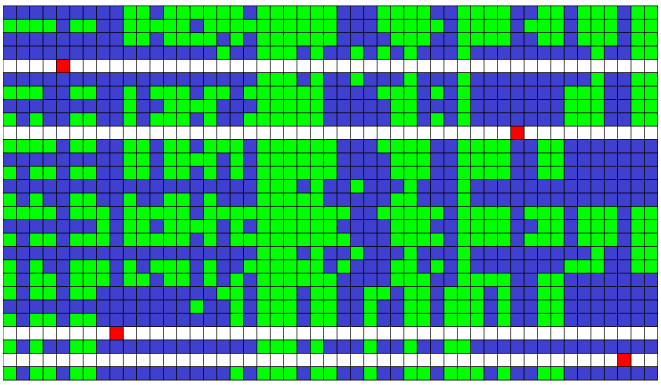
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Missing Propagation

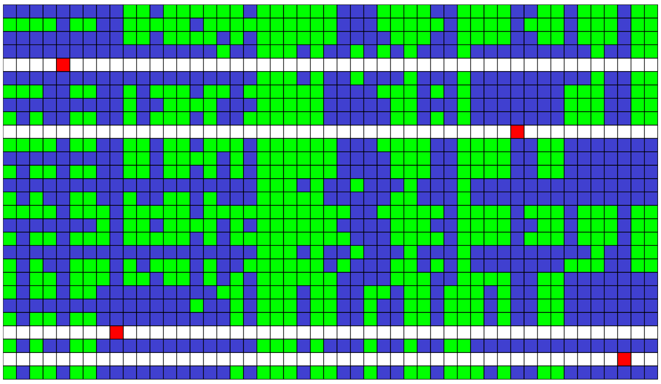
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Missing Propagation

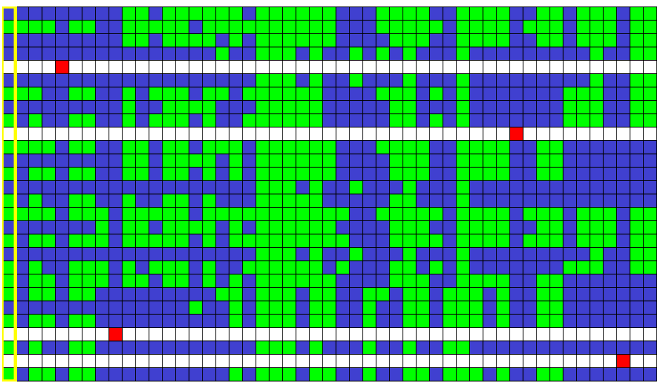
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Missing Propagation

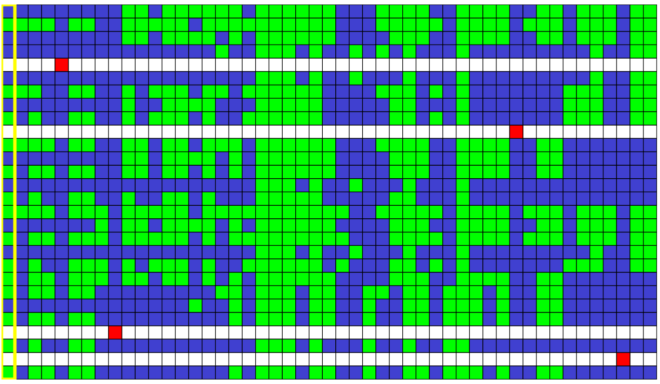
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

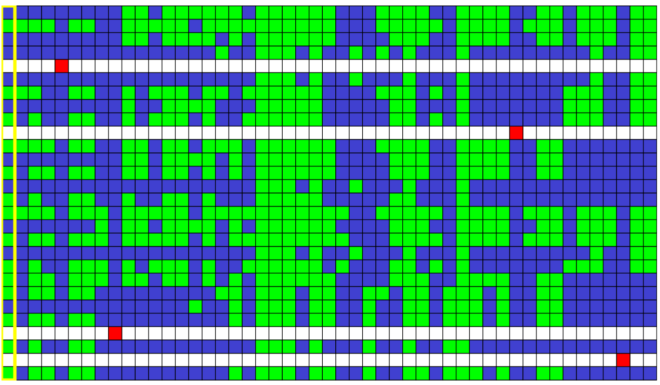


	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Cork Constraint Computation Centre

Missing Propagation

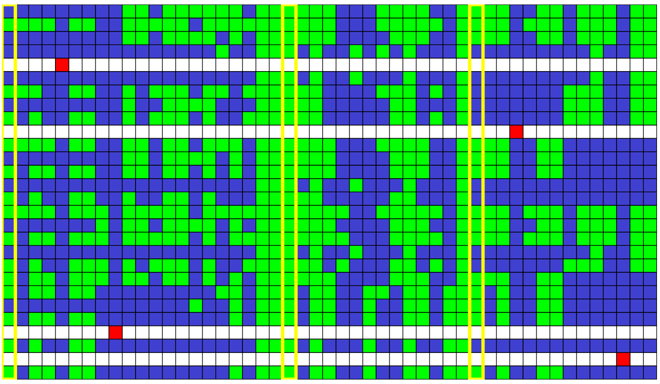
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Cork Constraint Computation Centre

Missing Propagation



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Why is this?

- Constraints involved:
  - `gcc` constraint on row: four variables can use values from this row
  - four `occurrence` constraints for hints: One of the variables must take this value
- No interaction between constraints, only between constraints and variables
- We do not detect that value 1 can not be used
- Eventual solution respects condition, model is correct
- We are concerned about propagation, not just correctness

Adding Redundant Constraints

- Add constraints which do more propagation, but do not affect solutions
- Lead to smaller search tree, hopefully faster solution
- Introduction requires understanding of (lack of) propagation
- Visualization is key to detect missing propagation

First Attempt: Adding 0/1 Viewpoint

- *Day* × *Location* matrix of 0/1 variables
- Indicates if there is a game on this day at this location
- Row/column sums: 4 games in each row/column
- Hint given for cell: Game variable is 1

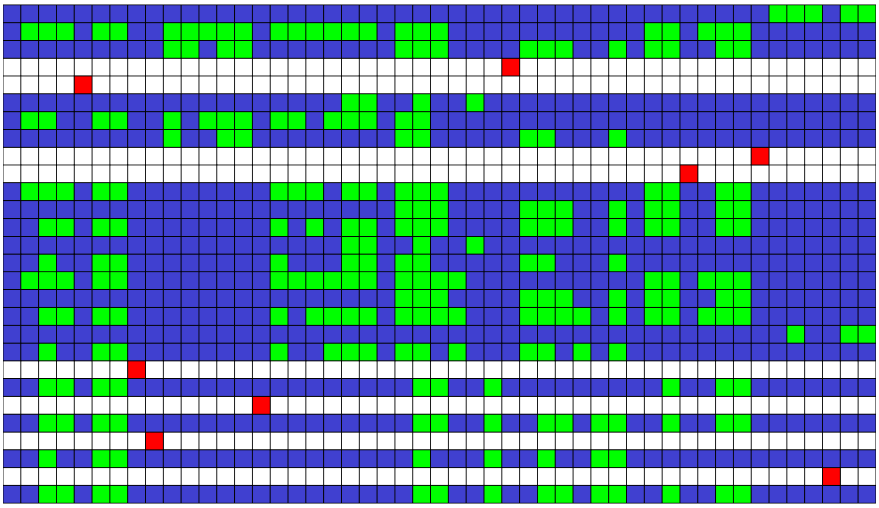
Channeling Constraint

- Link pair variables P_i to 0/1 variables Y_j as *value-index*
- $(\exists i \text{ s.t. } P_i = v) \Leftrightarrow Y_v = 1$
- Propagation:
  - $P_i = v \Rightarrow Y_v = 1$
  - $Y_v = 0 \Rightarrow \forall i: \; P_i \neq v$
  - $(\forall i: \; v \notin d(P_i)) \Rightarrow Y_v = 0$
  - $Y_v = 1 \Rightarrow \text{occurrence}(V, P_1 ... P_n, N), N \geq 1$

Added Program

```
value_set_channeling(L,Hints):-
    dim(Matrix,[7,7]),
    Matrix[1..7,1..7] :: 0..1,
    flatten_array(Matrix,ValueSet),
    value_set_channel(L,ValueSet,1),
    (for(I,1,7),param(Matrix) do
        sumlist(Matrix[I,1..7],4),
        sumlist(Matrix[1..7,I],4)
    ),
    (foreach(K-_,Hints),param(Matrix) do
        coor(K,I,J),
        subscript(Matrix,[I,J],1)
    ).
```

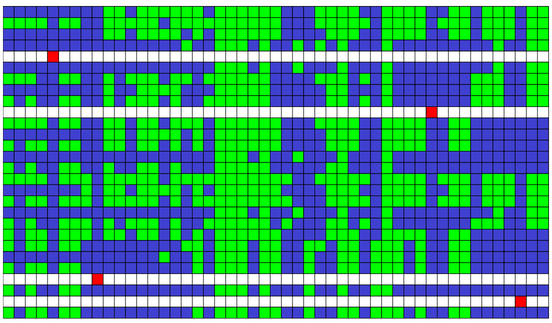
Before Search



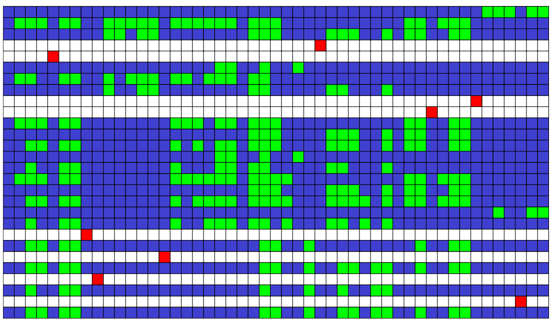
Cork Constraint Computation Centre

Impact of Redundant Constraints

Without

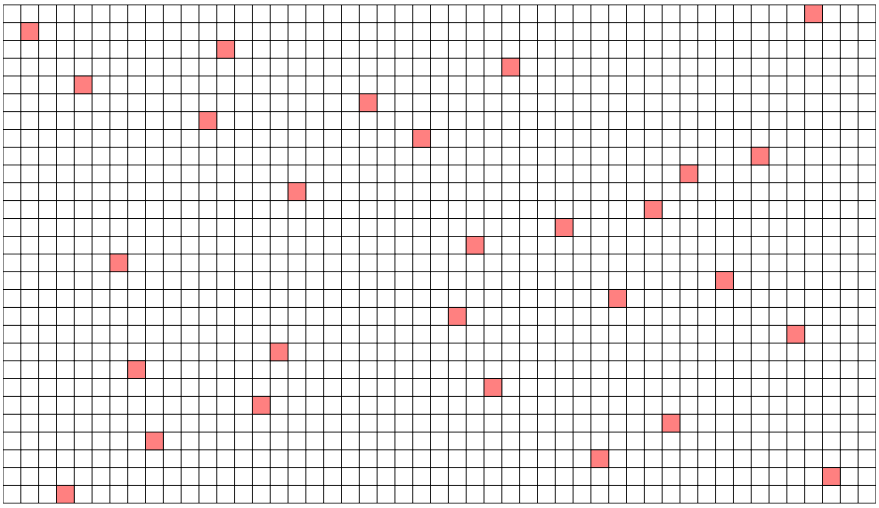


With value index channeling

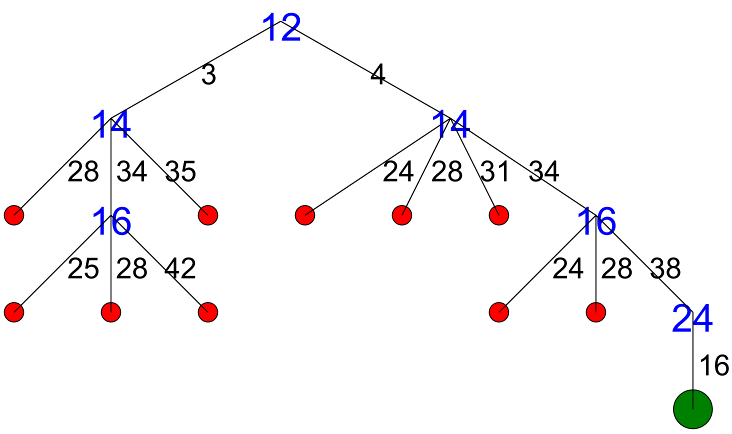


Cork Constraint Computation Centre

Solution

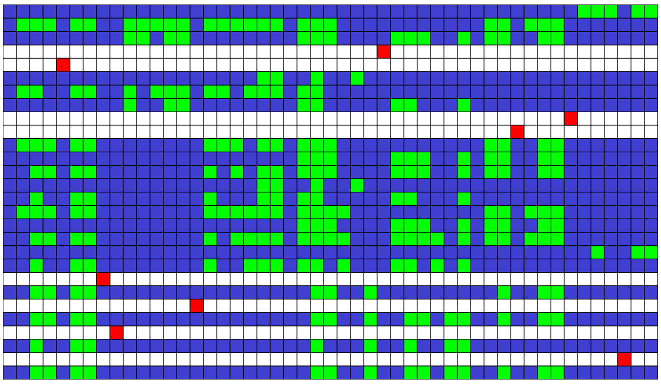


Search Tree



Still Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



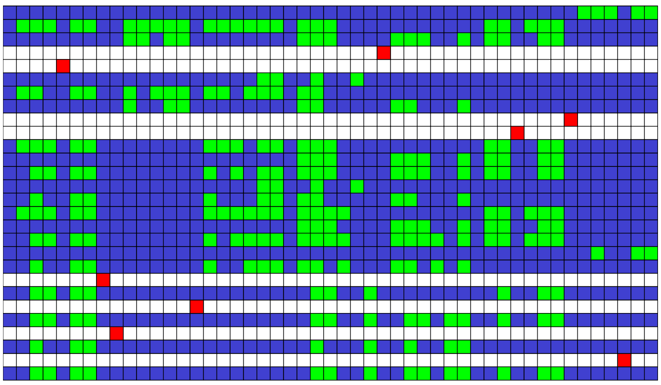
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Game 12 can not be played on day 1 at locations 5 or 6

Cork Constraint Computation Centre

Still Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



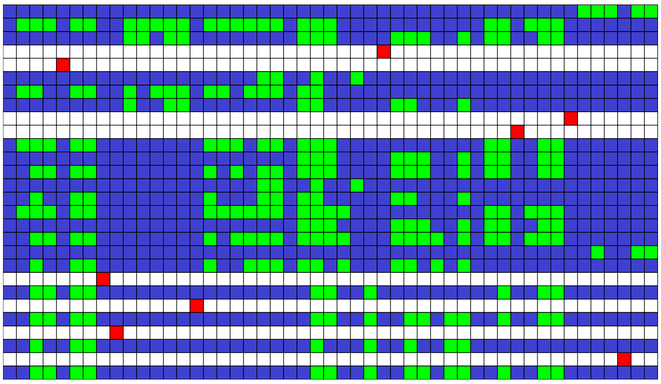
	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Game 12 can not be played on day 1 at locations 5 or 6

Cork Constraint Computation Centre

Still Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

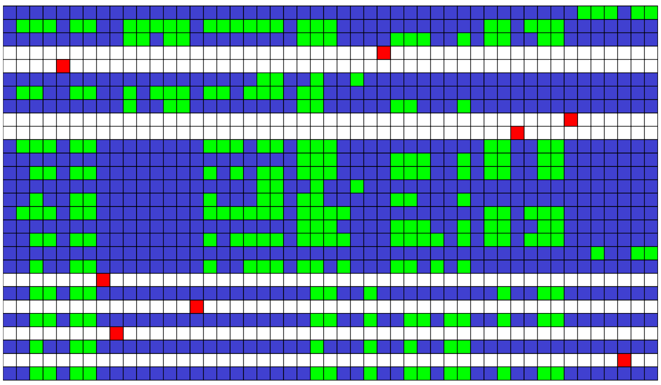


	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Game 12 can not be played on day 1 at locations 5 or 6

Still Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

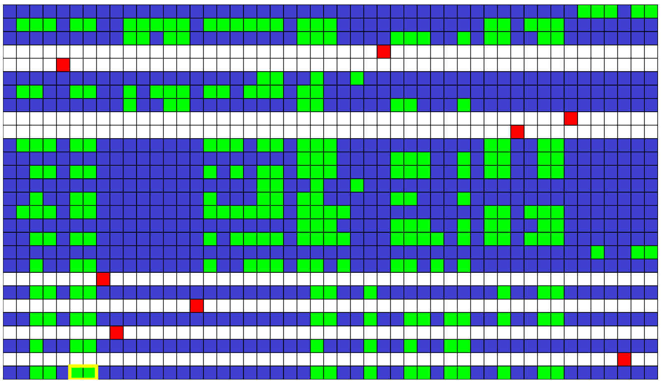


	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Game 12 can not be played on day 1 at locations 5 or 6

Still Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

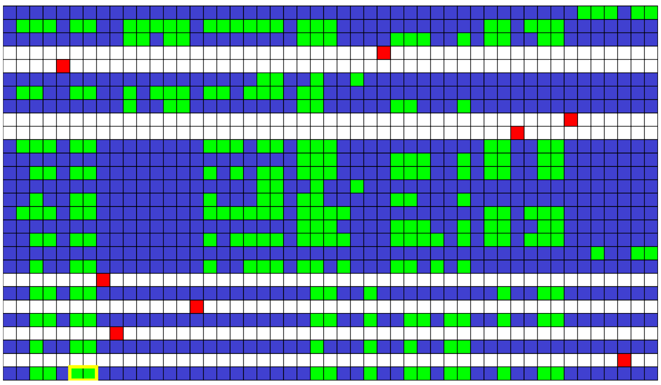


	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Game 12 can not be played on day 1 at locations 5 or 6

Still Missing Propagation

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		



	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1	1	2	3	4	5	6	7
Day 2	8	9	10	11	12	13	14
Day 3	15	16	17	18	19	20	21
Day 4	22	23	24	25	26	27	28
Day 5	29	30	31	32	33	34	35
Day 6	36	37	38	39	40	41	42
Day 7	43	44	45	46	47	48	49

Game 12 can not be played on day 1 at locations 5 or 6

Our model does not deal well with hints

- Preset game is ok, leads to variable assignment
- Preset team is weak, adds new constraint
- As there is no interaction of this constraint with the other constraints, there is no initial domain restriction
- Model is correct, but lazy

Second Attempt: Improving the handling of hints

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

- This value can not be used by pairs not involving team 8
- One of the pairs involving team 8 must use this value
- These values can not be used by any pair involving team 8

Redundant Constraints

- Red value can not be used by pairs not involving team 8
  - disequalities
- One of the pairs involving team 8 must use red value
  - occurrences(gcc) constraint
- Yellow values can not be used by any pair involving team 8
  - disequalities

	City 1	City 2	City 3	City 4	City 5	City 6	City 7
Day 1		8			7, 5		
Day 2	2	1, 5					
Day 3	7		8				
Day 4					2	5	1
Day 5	8					1	
Day 6				5, 4			
Day 7	4				1, 3		

Added Program

```
improved_hint(Pos,[Value],L,Contains):-
     (foreach(X,L),foreach(A-B,Contains),
      fromto([],R,R1,Required),
      param(Pos,Value) do
          (not_mentioned(A,B,Value) ->
               X #\= Pos,R1 = R
          ;
               R1 = [X|R]
          )
     ),
     occurrences(Pos,Required,1),
      excluded_locations(Pos,Excluded),
      exclude_values(Required,Excluded).
```

Added Program

```
excluded_locations(Pos,Excluded):-
    coor(Pos,X,Y),
    (for(I,1,7),
     fromto([],A,A1,E1),
     param(Y,Pos) do
        coor(K,I,Y),
        (Pos = K ->
            A1 = A
        ;
            A1 = [K|A]
        )
    ),
    ...
```

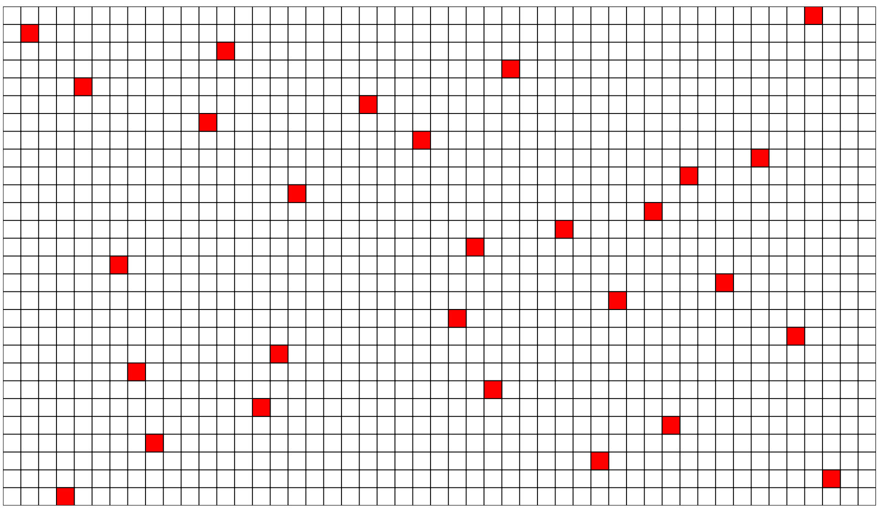

Added Program

```
    ...
    (for(J,1,7),
     fromto(E1,A,A1,Excluded),
     param(X,Pos) do
        coor(K,X,J),
        (Pos = K ->
            A1 = A
        ;
            A1 = [K|A]
        )
    ).
```

Added Program

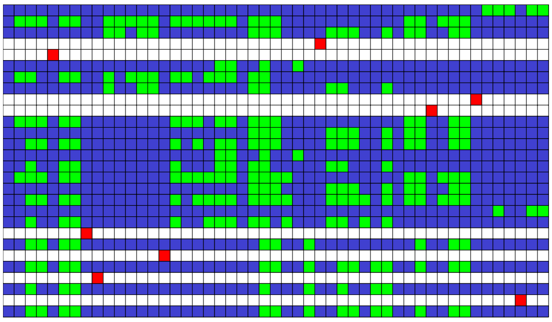
```
exclude_values(Vars,Values):-
     (foreach(X,Vars),
      param(Values) do
          (foreach(Value,Values),
           param(X) do
               X #\= Value
          )
     ).
```


Before Search

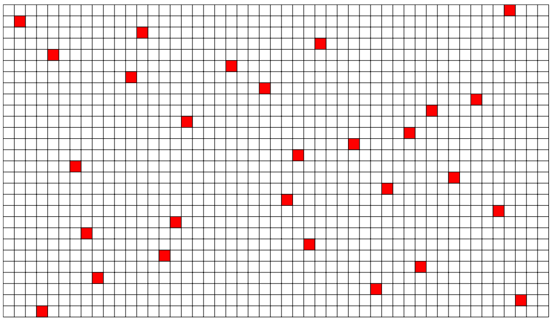


Impact of improved hint handling

With index set channeling



Improved Hints



Observation

- We don't need the value index channeling
- It is subsumed by the improved hint treatment
- Always worthwhile to check if constraints are still required after modifying model

Conclusions

- Many ways of modelling even simple problems
- Selection of “best” model difficult
  - Depends on constraints available
  - Often needs experimentation
- How do we measure if one model is “better” than another?
  - Execution time?
  - Size of search tree?
  - Scalability?
- Definition of variables is key
- Explore choices by considering mapping operators

Conclusions

- Channeling - Combining viewpoints
  - Express some constraints in one, others in second viewpoint
  - Channeling constraints to link the viewpoints
  - Decide which model to use for search
- Redundant Constraints - Improving constraint propagation
  - Constraints are logically implied by other constraints
  - Provide more propagation to reduce search space

More Information

- B. M. W. Cheng, Jimmy Ho-Man Lee, and J. C. K. Wu.
  Speeding up constraint propagation by redundant modeling.
  In Eugene C. Freuder, editor, *CP*, volume 1118 of *Lecture Notes in Computer Science*, pages 91–103. Springer, 1996.
- Barbara Smith.
  Modelling.
  In F. Rossi, P. van Beek, and T. Walsh, editors, *Handbook of Constraint Programming*. Elsevier, 2006.
- Helmut Simonis.
  Models for global constraint applications.
  Constraints, 12(1):63–92, 2007.

More Information

- Barbara Smith.
  Modelling for Constraint Programming.
  ACP Summerschool 2008, St Andrews, Scotland, 2008.
  http://www-circa.mcs.st-and.ac.uk/cpss2008/slides/SmithSummerSchool1.pdf
  http://www-circa.mcs.st-and.ac.uk/cpss2008/slides/SmithSummerSchool2.pdf
  http://www-circa.mcs.st-and.ac.uk/cpss2008/slides/SmithSummerSchool3.pdf
  http://www-circa.mcs.st-and.ac.uk/cpss2008/slides/SmithSummerSchool4.pdf

Exercises

1. Is there potential for symmetry breaking in this problem?
2. Develop one of the alternative models considered into a full program and compare its performance.
3. Consider the use of channeling for the N-queens problem, for the Sudoku puzzle. What could you use as a second viewpoint, and how to link the views together?

Cork Constraint Computation Centre