Chapter B: Lessons Learned

Helmut Simonis

Cork Constraint Computation Centre
Computer Science Department
University College Cork
Ireland

ECLiPSe ELearning Overview

Licence

Outline

1. Overview of Course
2. What do you need to know about CP?
3. What is CP good for?
4. A Core Set of Global Constraints
5. Visualization

Main Message

- New ELearning course for ECLiPSe
- Modelling and programming with constraints
- Based on sample problems solved and explained in detail
- *A view* on core constraint programming skills
- Strong dependence on visualization to explain behavior

Interesting Things to Come

- Which global constraints a system should contain
- How to interpret search trees
- The open challenge of debugging global constraints

Outline

1. Overview of Course
2. What do you need to know about CP?
3. What is CP good for?
4. A Core Set of Global Constraints
5. Visualization

Background

- Gift grant from Cisco Systems/Silicon Valley Community Foundation
- Cisco owns open-sourced ECLiPSe system
- How to expand user-base?
- Self-taught course in constraint programming
- Intended for Cisco engineers/programmers
- Open source/available to community

Format

- Video lectures
- Slides
- Handout
- Exercises

Examples

- Website `http://4c.ucc.ie/~hsimonis/ELearning/index.htm`
- Slides `../introduction/slides.pdf`
- Handout `../introduction/handout.pdf`
- Video `../wavedemo/DEMO/web/web.html`

Outline

1. Overview of Course
2. What do you need to know about CP?
3. What is CP good for?
4. A Core Set of Global Constraints
5. Visualization

Central Topics

- Basic structure of constraint programs
- Global constraints
- User-defined search
- Optimization
- Symmetry breaking
- Choosing the right model
- Limits of propagation

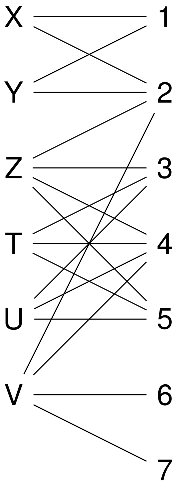
Explaining Concepts

- Explain as you go
- Constraints introduced when used by application
- Concepts/algorithms explained by example

Example: Domain consistent `alldifferent`

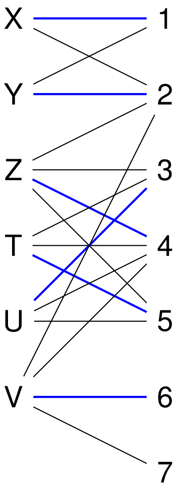
```
:-lib(ic).
:-lib(ic_global_gac).

top:-
    [X,Y] :: 1..2,
    Z :: 2..5,
    [T,U] :: 3..5,
    V :: [2,4,6,7],
    ic_global_gac:alldifferent([X,Y,Z,T,U,V]).
```

Making constraint domain consistent



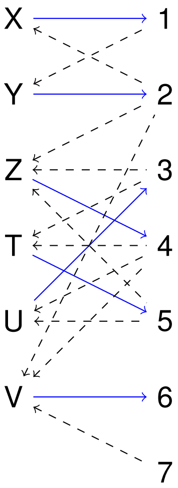
Problem shown as bipartite graph

Making constraint domain consistent



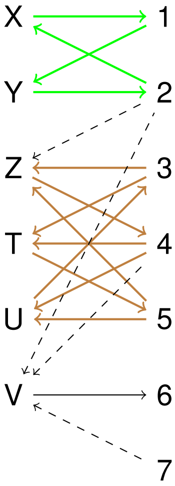
Find maximal matching (in blue)

Making constraint domain consistent



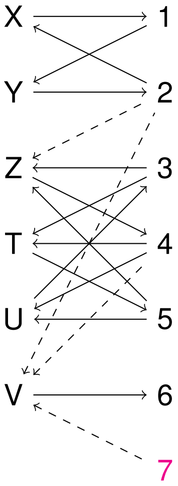
Orient graph (edges in matching from variables to values, all others from values to variables), mark edges in matching

Making constraint domain consistent



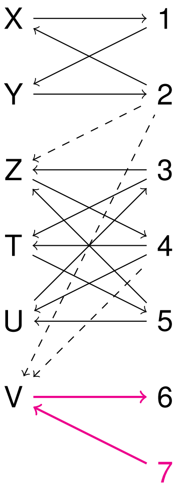
Find strongly connected components (green and brown), mark their edges

Making constraint domain consistent



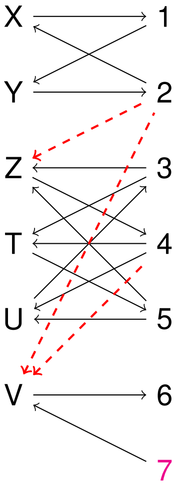
Find unmatched value nodes (here node 7, magenta)

Making constraint domain consistent



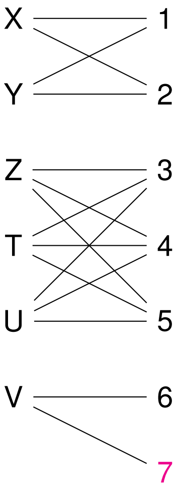
Find alternating paths from such nodes (in magenta), mark their edges

Making constraint domain consistent



All unmarked edges can be removed

Making constraint domain consistent

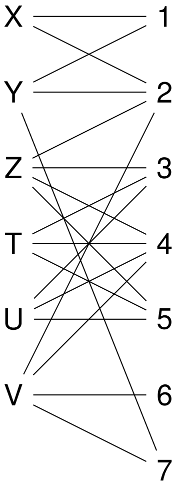


Resulting graph, constraint is domain consistent

Extended Example

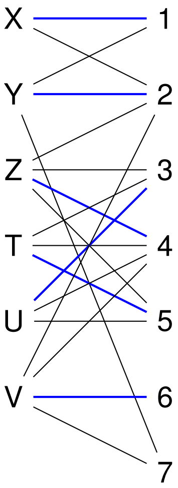
```
:-lib(ic).
:-lib(ic_global_gac).

top:-
    X :: 1..2,
    Y :: [1,2,7],
    Z :: 2..5,
    [T,U] :: 3..5,
    V :: [2,4,6,7],
    ic_global_gac:alldifferent([X,Y,Z,T,U,V]).
```

No propagation in expanded example



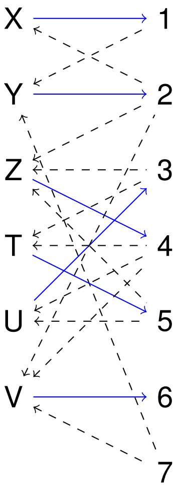
Problem shown as bipartite graph

No propagation in expanded example



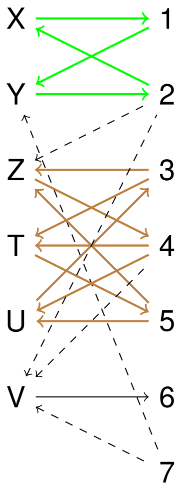
Find maximal matching (in blue)

No propagation in expanded example



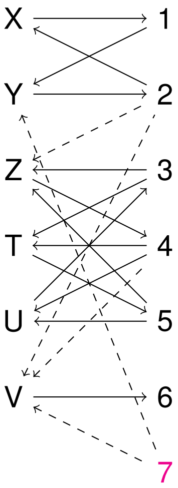
Orient graph (edges in matching from variables to values, all others from values to variables), mark edges in matching

No propagation in expanded example



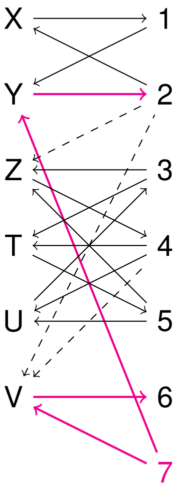
Find strongly connected components (green and brown), mark their edges

No propagation in expanded example



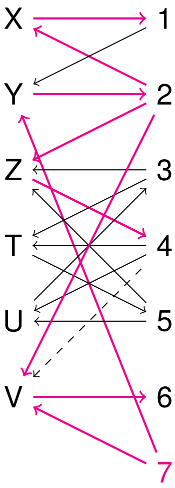
Find unmatched value nodes (here node 7, magenta)

No propagation in expanded example



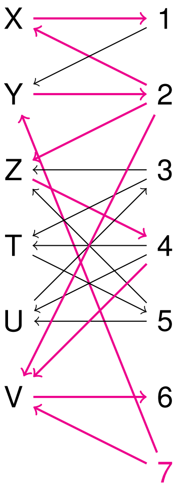
Find alternating paths from such nodes (in magenta), mark their edges

No propagation in expanded example



Continue with alternating paths

No propagation in expanded example



Continue with alternating paths, all edges marked, no propagation, constraint is domain consistent

No Theoretical Background

- Target audience
  - Engineers
  - Application programmers
- Not a course in theoretical computer science
- Different from current textbooks on CP

Outline

1. Overview of Course
2. What do you need to know about CP?
3. What is CP good for?
4. A Core Set of Global Constraints
5. Visualization

This was easy ten years ago:

- Based on industrial experience
    - Assignment problems
    - Scheduling
    - Transportation
    - Personal planning

In the meantime

- Dramatic improvements for competitors
  - MILP
  - SAT
- Improvements in hardware performance

New Areas

- Configuration
  - In search of a nice example
- Placement
  - Not in ECLiPSe
- Networks
  - In form of MIP/FD hybrids

Problems Handled in Course

- Must have puzzles!
- Send+More=Money
- Sudoku
- N-queens
- Shikaku (see Wednesday's talk)

Practical Example Problems

- Test plan generation (BIBD)
- Progressive party problem
- Routing and wavelength assignment
- Optical network design
- Car sequencing
- Costas arrays
- Sports scheduling
- Still to come
  - Production scheduling
  - Nurse rostering
  - Airport stand allocation

Intention

- Realistic, life like problems
- Must address scalability issues
- Often, problem not completely specified
- Issue: Hard to verify by hand
- Complexity still limited, not real problems
- No attempt at integration

What is CP good for: A Problematic View

- CP works if it out-performs everything else
- See which technique produces best (optimal) solutions
- Which technique runs fastest
- A publication game (I like to play too!)
    - Somebody defines the problem
    - Minute detail, no choices left
    - Counting runtime, choices, backtracks
    - Tiny improvements count

From RWA Chapter

Network	Dem.	Complete MIP		Decomposition MIP-MIP		MIP-FD		MIP-SAT	
		Opt	Avg	Opt	Avg	Opt	Avg	Opt	Avg
brezil	100	100	277.14	100	0.91	100	0.01	100	0.03
brezil	200	-	-	100	4.45	100	0.03	100	0.07
brezil	300	-	-	100	8.08	99	0.07	100	0.15
brezil	400	-	-	100	10.93	100	0.13	100	0.27
brezil	500	-	-	100	13.09	100	0.23	100	0.44
brezil	600	-	-	100	16.77	100	0.31	100	0.69
eon	100	100	33.62	100	1.51	100	0.01	100	0.04
eon	200	100	65.51	100	5.27	100	0.04	100	0.10
eon	300	100	121.27	100	5.60	100	0.09	100	0.24
eon	400	100	116.64	100	7.38	100	0.16	100	0.45
eon	500	100	162.55	100	9.58	100	0.29	100	0.76
eon	600	100	232.91	99	14.04	100	0.40	100	1.20
mci	100	100	20.27	100	2.08	100	0.01	100	0.05
mci	200	100	38.79	100	5.36	100	0.05	100	0.12
mci	300	100	55.78	100	5.83	100	0.10	100	0.29
mci	400	100	109.85	100	8.71	100	0.19	100	0.56
mci	500	100	129.90	100	13.89	100	0.29	100	0.97
mci	600	100	257.70	100	22.56	100	0.45	100	1.55
nsf	100	100	8.17	100	2.38	100	0.02	100	0.05
nsf	200	100	12.75	100	1.81	100	0.05	100	0.15
nsf	300	100	17.01	100	1.98	100	0.10	100	0.35
nsf	400	100	27.36	100	3.54	100	0.17	100	0.71
nsf	500	100	54.60	100	5.77	100	0.31	100	1.26
nsf	600	100	88.72	100	9.09	100	0.43	100	2.07

Question

- Is MIP-FD decomposition really better than MIP-SAT?
- Both are good enough
- Both are clearly better than complete MIP
- This is only one possible problem variant
  - For other variant difference is much more significant

A More Meaningful Evaluation

- When is a solution good enough?
- How long does it take to get there?
  - Total development time, not runtime
  - For whom: beginner, expert, genius?
- What happens if you change the problem?
- Can you explain what is happening?
- How easy it is to integrate into workflow?

Cork Constraint Computation Centre

Problem: How do we measure/report this?

- Usability labs?
- Instrument IDE?
- Cost of parallel development
- Commercial sensitivities
- Can this be published?

Outline

1. Overview of Course
2. What do you need to know about CP?
3. What is CP good for?
4. A Core Set of Global Constraints
5. Visualization

Cork Constraint Computation Centre

Global Constraint Catalog (A-C)

- all differ from at least k pos
- all equal
- all min dist
- alldifferent
- alldifferent between sets
- alldifferent consecutive values
- alldifferent cst
- alldifferent except 0
- alldifferent interval
- alldifferent modulo
- alldifferent on intersection
- alldifferent partition
- alldifferent same value
- allperm
- among
- among diff 0
- among interval
- among low up
- among modulo
- among seq
- among var
- and
- arith
- arith or
- arith sliding
- assign and counts
- assign and nvalues
- atleast
- atleast nvalue
- atleast nvector
- atmost
- atmost1
- atmost nvalue
- atmost nvector
- balance
- balance interval
- balance modulo
- balance partition
- between min max
- bin packing
- bin packing capa
- binary tree
- bipartite
- calendar
- cardinality atleast
- cardinality atmost
- cardinality atmost partition
- change
- change continuity
- change pair
- circuit
- circuit cluster
- circular change
- clause and
- clause or
- clique
- colored matrix
- coloured cumulative
- coloured cumulatives
- common
- common interval
- common modulo
- common partition
- cond lex cost
- cond lex greater
- cond lex greatereq
- cond lex less
- cond lex lesseq

Core Global Constraints

- `element`
- `alldifferent`
- `gcc`
- `lex-ordering`
- `bin packing`
- `sequence`
- `cumulative`
- `regular`
- `diffn/disjoint/geost`

Not sure about:

- `min weight alldifferent`
- `gcc with cost`
- `cycle`
- `nvalue`

Sources

- Good survey papers and PhD thesis
  - W. van Hoeve: `alldifferent`
  - C. Quimper: `alldifferent`, `gcc`
  - Z. Kiziltan: `lex ordering`
- Clear descriptions of specific constraints
  - P. Shaw: `bin packing`
  - G. Pesant: `regular`

Families of Constraints

- Matching/flow based global constraints
    - Build graph
    - Find matching/flow
    - Reorient links
    - Find SCC
    - (BFS reachable)
    - Remove unmarked edges

Examples

- `alldifferent`
- `gcc`
- `alldifferent matrix`
- `gcc matrix`
- `same`
- `sequence`

Some Constraints Still Opaque

- cumulative
- cycle
- geost

Which Variant to Implement

- `gcc`
  - Low/high limits or variables for counters
  - Open/closed version
- `cumulative`
  - Capacity limit or variable
  - End limit
  - Task surface

Use in Course

Constraint	Sudoku	N-queens	RWA	BIBD	Party	Sports	Car	Skikaku	Network Design	Costas	Scheduling	Airport	Nurse
element						x	x					x	x
alldifferent	x	x	x		x	x				x		x	
gcc			x			x	x	x					x
lex				x		x			x				x
sequence							x						x
bin packing					x								
cumulative					x						x		
somedifferent			x		x							x	

Open Challenge

- How do we implement/test global constraints?
- No publication, ever
- No methodology

Techniques Used

- Profiling
- Line coverage
- Generic GAC

Profiling

//C/Documents and Settings/hsimonis/My Documents/courseware/bin_packing.ecl - Mozilla Firefox

File Edit View History Bookmarks Tools Help

file:///C:/Documents and Settings/hsimonis/My Documents/courseware/coverage/bin_packing.html

```
bin_packing(L, _S, B, _Total, _NrVars, NrBins, Items) :-
        82 hash create(Hash),
        82 check bin packing(B, NrBins, Items, Hash),
        82 append(L, B, AllVars),
        82 ( ground(AllVars) ->
            9 true 9
        ;
            73 suspend(update bin packing(AllVars, B, NrBins, Items, Hash, Susp), 5, [(AllVars -> inst), (AllVars -> ic : min), (AllVars -> ic : max), (L ->
        ) 82 .

:- demon update_bin_packing / 6.
update_bin_packing(AllVars, B, NrBins, Items, Hash, Susp) :-
        42513 check bin packing(B, NrBins, Items, Hash),
        34400 ( ground(AllVars) ->
            73 kill suspension(Susp) 73
        ;
            34327 true 34327
        ) 34400 .

check_bin_packing(Bins, NrBins, Items, Hash) :-
        42595 l2 limit(Bins, NrBins, Items),
        42595 no sum reasoning(Bins, 1, Items, Hash),
        34482 true 34482 .

no_sum_reasoning([], _J, _Items, _32243) 34482 .
no_sum_reasoning([Bin|Bins], J, Items, Hash) :-
        488020 get bounds(Bin, BMin, BMax),
        488020 candidates(Items, J, Variables, Candidates, 0, _Possible, 0, Fixed),
        488020 ( Candidates = [] ->
            152137 true 152137
        ;
            335883 Alpha is BMin - Fixed,
            335883 Beta is BMax - Fixed,
            335883 sum cand(Candidates, 0, Sum, 0, N),
            335883 no sum(Hash, Candidates, Sum, N, Alpha, Beta, Result, _Alpha1, _Beta1),
            335883 ( Result = infeasible ->
                1717 fail 0
            ;
                334166 true 334166
            ),
```

Done

Generic GAC: `alldifferent_sum`

```
alldifferent_sum(L,N):-
    (ic:alldifferent(L),
     sum(L) #= N,
     labeling(L)) infers ac.
```

Propia library (Le Provost, Wallace)

Worst Case Scenario: Progressive Party Problem

- Progressive Party Problem
- Adding stronger global constraint, bin packing (P. Shaw)
- Found different solution
  - Variable selection changed by improved propagation?
  - Consistent value removed by bug in constraint?

Outline

1. Overview of Course
2. What do you need to know about CP?
3. What is CP good for?
4. A Core Set of Global Constraints
5. Visualization

How do we understand behavior?

- Mental model
- Formal analysis
- Debugging
- Tracing
- Life visualization
- Post-mortem analysis

Design Choices

- No deep integration with solver
- Post-mortem visualization
- Intermediate file format
- No view of detailed propagation
  - Tool not intended for debugging constraint engine

Conceptual Model

- Stable state at defined program points
- Granularity
  - Assign value
  - Post constraint
- Show stable state after propagation
- Do not show individual propagation steps

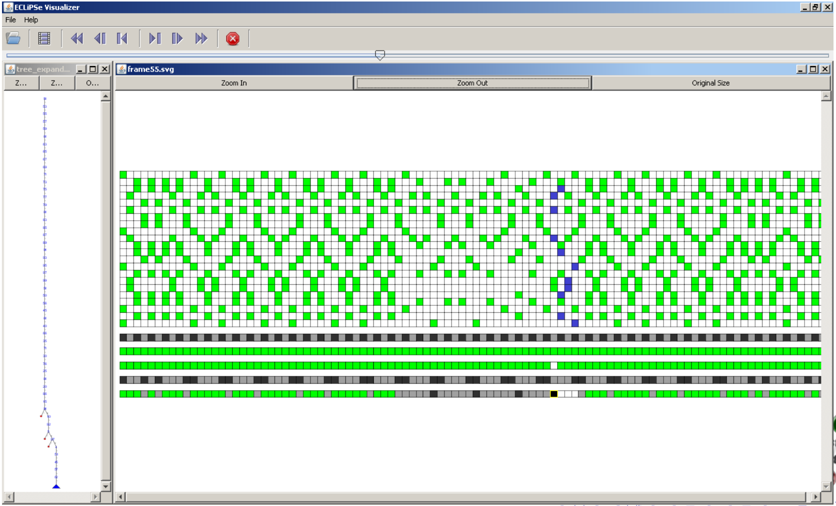
Visualizers

- Search tree
- Variables
- Constraints

Visualization Tool

- Developed in Java
- Show two panes: tree and state
- Navigate along timeline

Visualization Tool: Car Sequencing



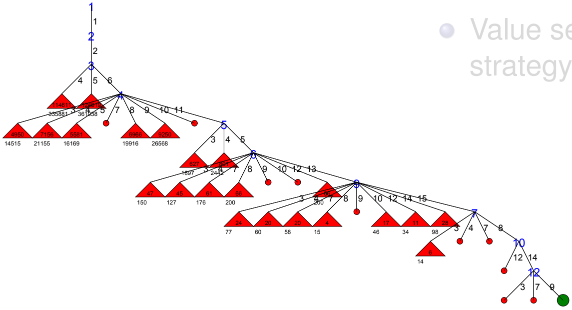
How many visualizers do we need?

- Develop few primitives
  - Cell based view
  - Domain vector
- Allow aggregation
  - Vector/matrix
  - General layout
- Which global constraints require more?
  - Task based view for `cumulative`
  - Matching/flow based representation does not scale

How to Interpret Visualization

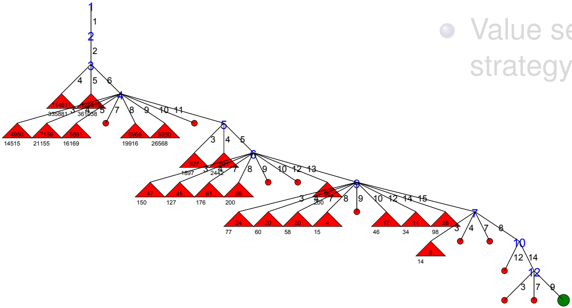
- Search tree
  - Good/bad choices
  - Place of backtracking
- State
  - Missing propagation

Costas Array Search tree (Size 16)



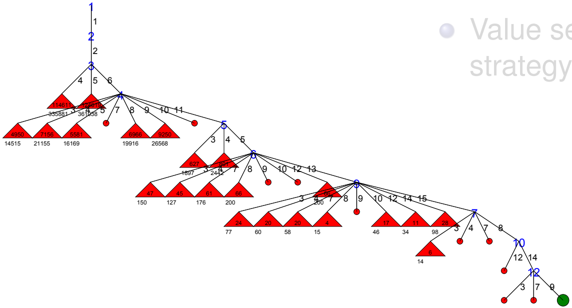
- Deep backtracking
- Third choice wrong
- Last choice wrong
- Value selection strategy useless

Costas Array Search tree (Size 16)



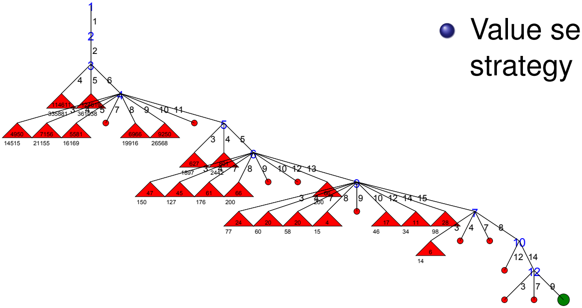
- Deep backtracking
- Third choice wrong
- Last choice wrong
- Value selection strategy useless

Costas Array Search tree (Size 16)



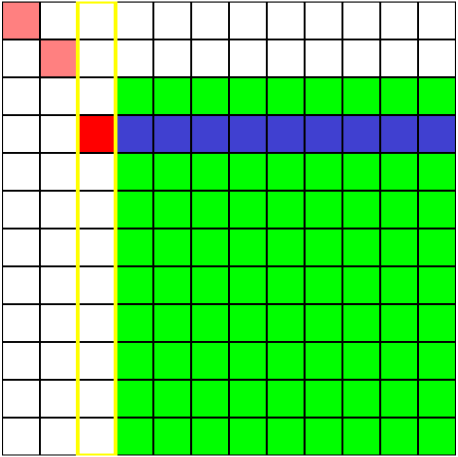
- Deep backtracking
- Third choice wrong
- Last choice wrong
- Value selection strategy useless

Costas Array Search tree (Size 16)



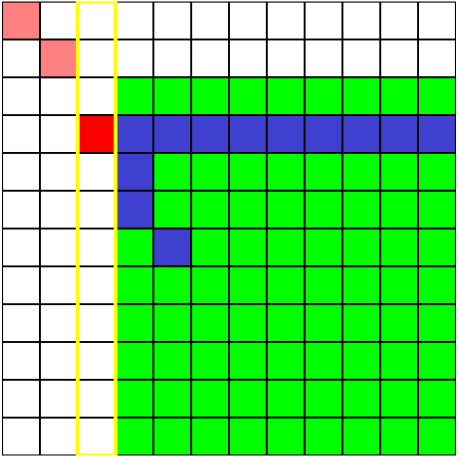
- Deep backtracking
- Third choice wrong
- Last choice wrong
- Value selection strategy useless

Missing Propagation



The model is doing this

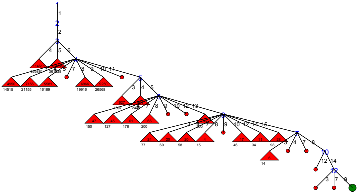
Missing Propagation



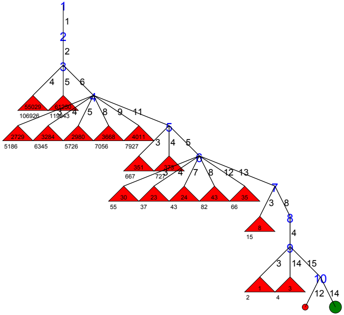
It could be doing that!

Comparison (Search Tree, size 16)

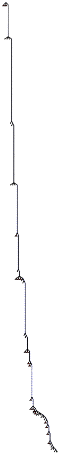
Initial Model



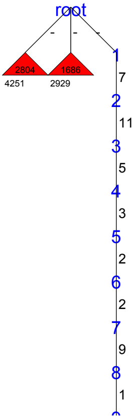
Improved Model



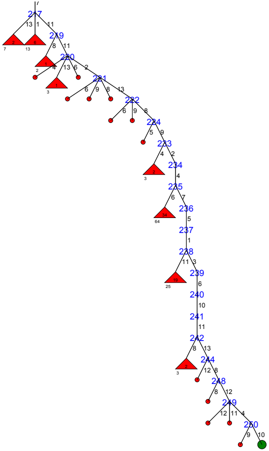
Progressive Party Problem, 9 Time Periods



2 Restarts Before Solution Found



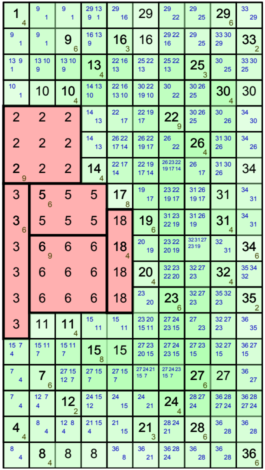
Value Choice Strategy Not Focused



Progressive Party

- Clearly impossible to explore search space
- Either many solutions or good value selection
- Value selection at end rather poor
- Probably many solutions

From Shikaku Presentation (Wednesday)



What is still missing?

- Explanation of failures
    - Procedural: Show propagation which leads to failure
    - Declarative: Find conflict set
- Comparison of trees

Outline

6 Some Technical Details

7 Conclusions

How was this generated?

- Slides produced with LaTeX `beamer` class
- Templates for chapters
- Visualization imported as pdf files
- Using `inkscape` as SVG to PDF converter
- Allows to produce multiple versions from one source
  - slides
  - handout
  - article

Videos

- Recorded and produced with Camtasia Studio
- Long takes, minimal editing
- Screen capture at full resolution
- Produce video in different target formats
    - Web based (640x480)
    - iPhone (480x320)
    - HD video possible

Metrics *without* Program Development

- Slides: 2-4 days per chapter
- Video: 1:10 ratio for finished product
  - 2-3 trials
  - No fine grain editing
  - Only for bulk production

Outline

6 Some Technical Details

7 Conclusions

Conclusions

- New ELearning course for ECLiPSe
- Open source material, Creative Commons BY-NC-SA license
    - Application driven
    - Modelling with global constraints
    - Customizing search
- Effort only justifiable through Cisco grant

Conclusions

- If you are not using visualization, why not?
- Generic tools, not just for ECLiPSe