2011

A Reformulation Strategy for Multi-Dimensional CSPs: The Case Study of the SET Game

Amanda Swearngin
University of Nebraska-Lincoln, aswearng@cse.unl.edu

Berthe Y. Choueiry
University of Nebraska - Lincoln, choueiry@cse.unl.edu

Robert J. Woodward
University of Nebraska - Lincoln, rwoodwar@cse.unl.edu

Eugene C. Freuder
University College Cork, e.freuder@cs.ucc.ie

A Reformulation Strategy for Multi-Dimensional CSPs: The Case Study of the SET Game

Presented by: Robert J. Woodward,

Amanda Swearngin[1] Berthe Y. Choueiry[2] Eugene C. Freuder[3]

[1]ESQuaReD Laboratory, University of Nebraska-Lincoln
[2]Constraint Systems Laboratory, University of Nebraska-Lincoln
[3]Cork Constraint Computation Centre, University College Cork

Acknowledgements

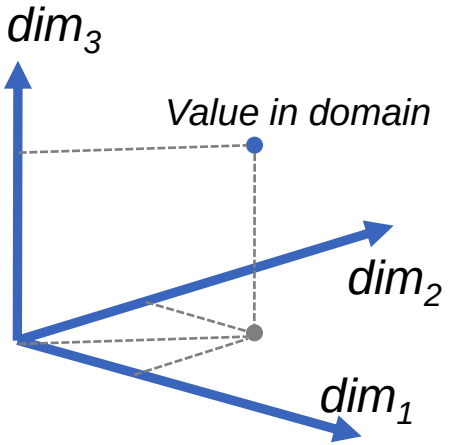
- National Science Foundation under grants CCF-0747009, CNS-0855139, and RI-1117956
- Science Foundation Ireland under grant 05/IN/1886

Outline

- General reformulation strategy for CSPs
  - Multidimensional CSPs (MD-CSPs)
  - Problem reformulation by value interchangeability
  - A general reformulation strategy for MD-CSPs
- Game of Set: A new toy problem
  - Game, CSP model
  - Problem reformulation
  - Algorithms & Results
- Conclusions

Multi-Dimensional CSPs [Yoshikawa+ 1992]

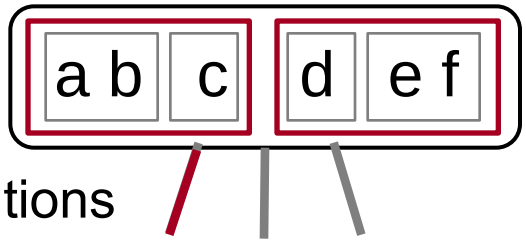
- All variables have the same domain
- Domain is multi-dimensional
  - A set of dimensions
  - Each domain value is described by a combination of dimensions values



- In MD-CSPs, a constraint can be
  - One-dimensional: defined over a single dimension
  - Multi-dimensional, otherwise
- Typical applications
  - Scheduling, resource allocation, configuration, etc.

Reformulation by value interchangeability

- Value interchangeability [Freuder 91]
  - Domain abstraction: equivalent values
  - 'Perfect' equivalence rare, small domain partitions
  - Ignoring some constraints yields larger domain partitions, smaller CSPs, smaller search space [Haselboeck 93, Choueiry+ 94]



- Abstraction in MD-CSPs [Freuder+ 95,97]
  - Abstract domains based on a dimension, P_r
  - Solve reformulated CSP
  - Use solution of P_r to guide solving original CSP, P_o

- How to "use solution of P_r to solve P_o"? Hard to automate

Reformulation Strategy for MD-CSPs

- Process

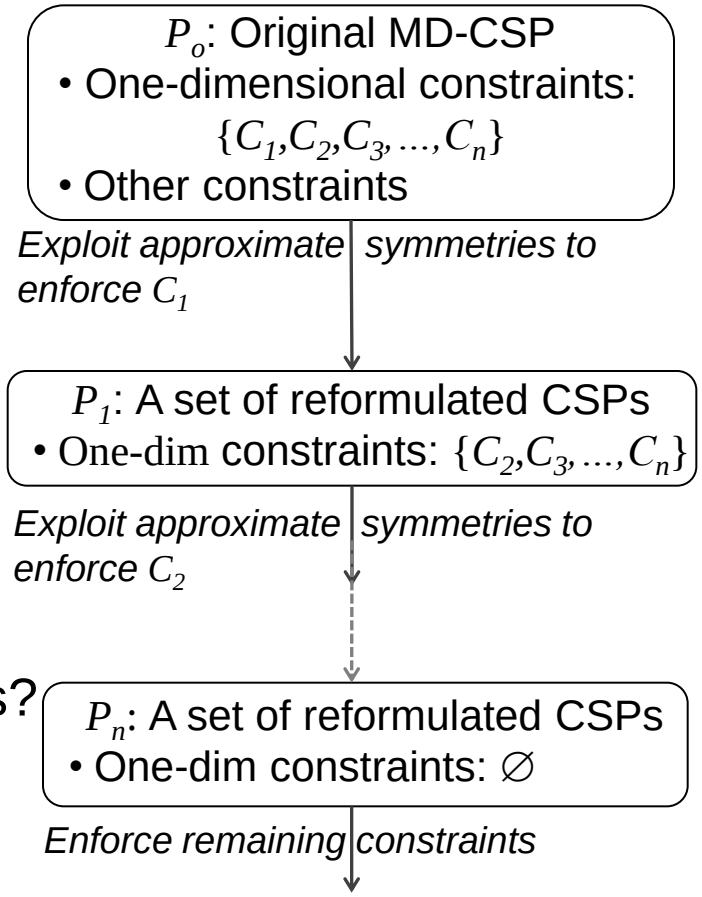
For each one-dimensional constraint

Abstract domains using interchangeability

Enforce one-dimensional constraint

Solve remaining CSPs with some solver

- Questions
  - Which 1-dim constraint to use first?
  - How to process reformulated problems?
- Case study of the Set game



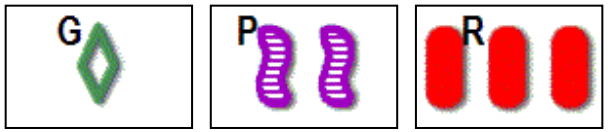
Outline

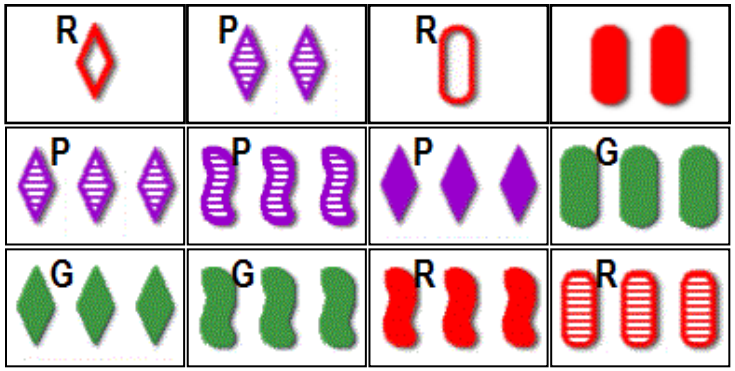
- General reformulation strategy for CSPs
  - Multidimensional CSPs (MD-CSPs)
  - Problem reformulation by value interchangeability
  - A general reformulation strategy for MD-CSPs
- **Game of Set: A new toy problem**
  - Game, CSP model
  - Problem reformulation
  - Algorithms & Results
- Conclusions

Game of Set [Falco 74]

- Deck of 81(=3^4) cards, each card with a unique combination of 4 attributes values
  1. *Number* ∈ {1,2,3}
  2. *Color* ∈ {*green*,*purple*,*red*}
  3. *Filling* ∈ {*empty*,*stripes*, *full*}
  4. *Shape* ∈ {*diamond*,*squiggle*,*oval*}
- Solution set: 3 cards

  ∀ attribute, the 3 cards have either the same value or all different values

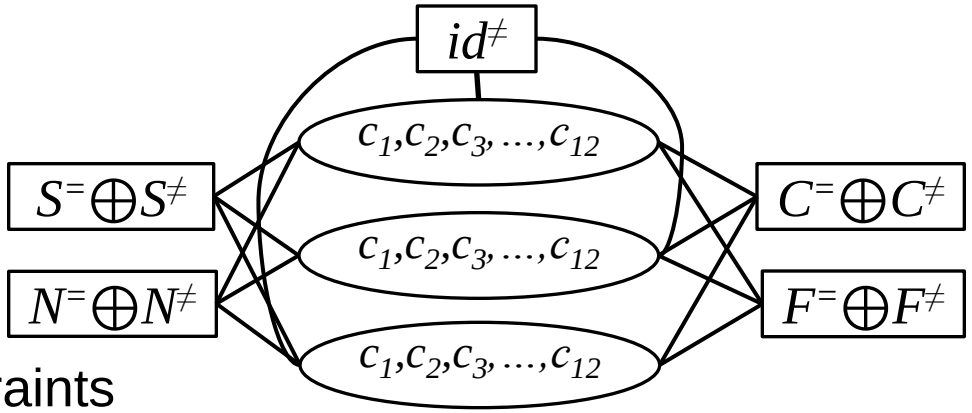




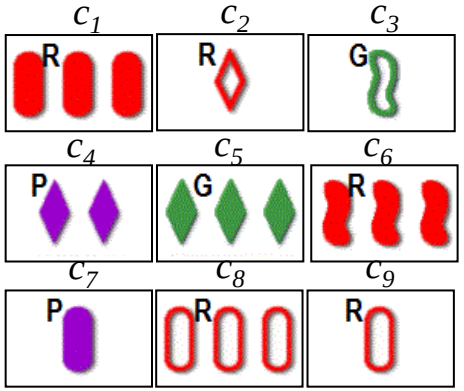
- 12 cards are dealt, on table [3,21]
- Recreational game, favorite of children & CS/math students
- New toy problem for AI: a typical multi-dimensional CSP

Set as an MD-CSP

- Model
  - Three variables
  - Same domain (12 cards)
  - One ‘physical’ constraints
  - Four 1-dimensional constraints



Same domain for all 3 variables

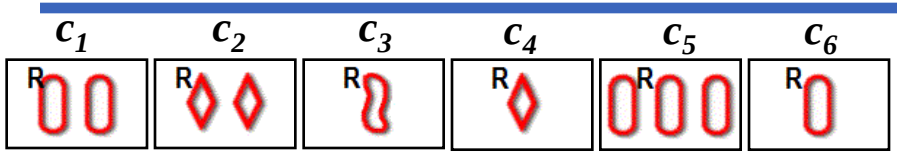


Domain Table

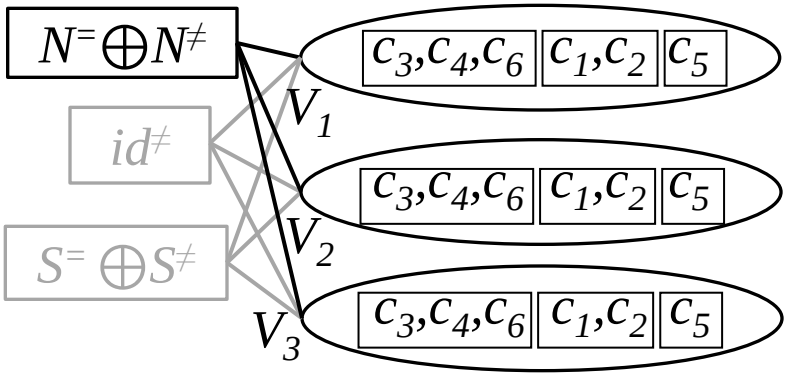
Domain dimensions

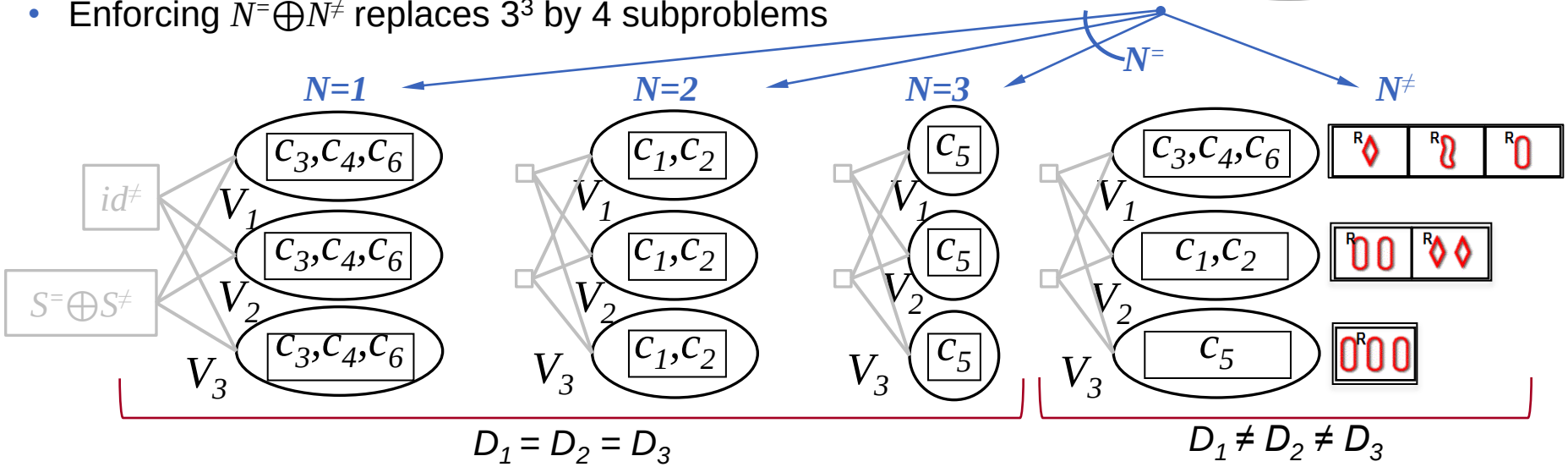
		c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
Number	*1*	0	1	1	0	0	0	1	0	1
	2	0	0	0	1	0	0	0	0	0
	3	1	0	0	0	1	1	0	1	0
Color	*r*	1	1	0	0	0	1	0	1	1
	g	0	0	1	0	1	0	0	0	0
	p	0	0	0	1	0	0	1	0	0
Filling	*f*	1	0	0	1	1	1	1	0	0
	e	0	1	1	0	0	0	0	1	1
	s	0	0	0	0	0	0	0	0	0
Shape	*s*	0	0	1	0	0	1	0	0	0
	o	1	0	0	0	0	0	1	1	1
	d	0	1	0	1	1	0	0	0	0

Reformulation by Value Interchangeability

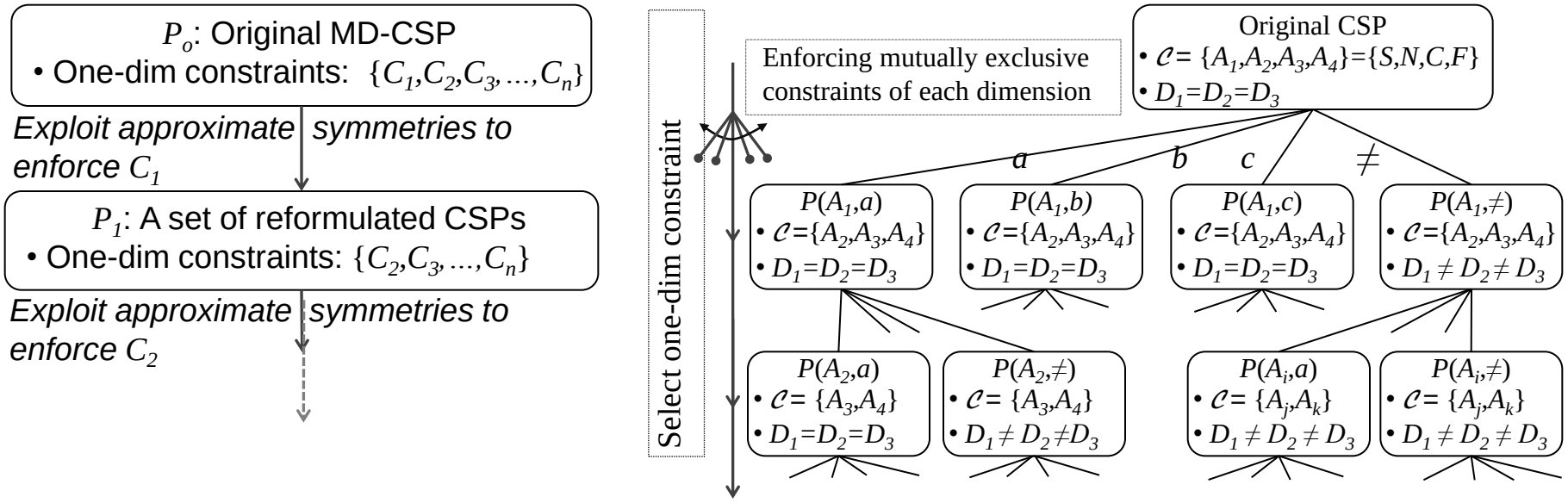


- *Filling≡empty, Color≡red*
- *Number* yields 3 domain partitions by neighborhood interchangeability (meta), replacing 6^3 solutions by 3^3 subproblems
- Enforcing $N^{=} \oplus N^{\neq}$ replaces 3^3 by 4 subproblems





Reformulation Strategy for Set

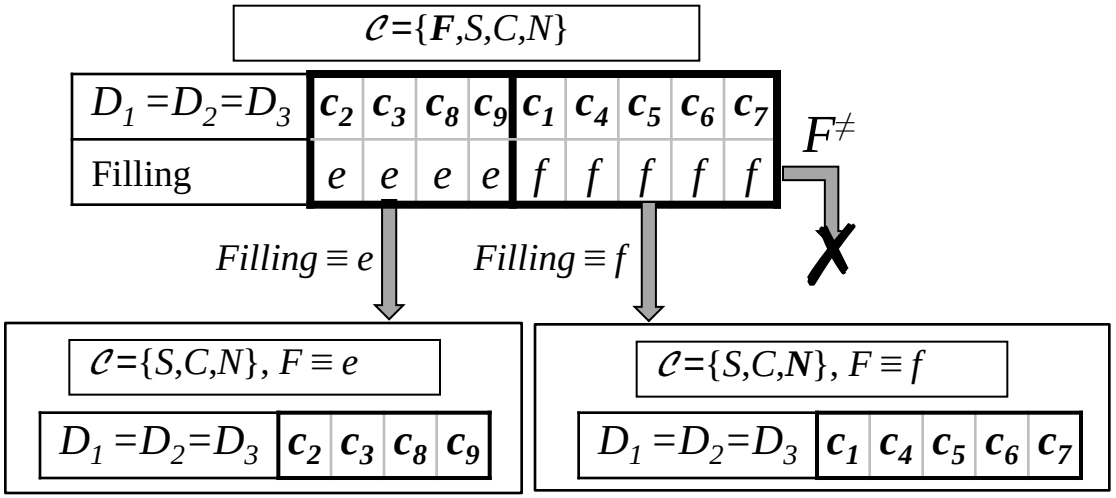


- Which dimension to choose first?
  - ↪ For Set, heuristics based on data in 'Domain Table:'
    Fewest subproblems first (infamously, Fail First Principle)

Selecting Domain Dimension

- Goal: Reduce branching factor
  - In example below, no card in domain has a shaded filling, thus, subproblems for $Filling{=}s$ and $F^{\neq}$ do not exist

Domain		c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	Σ
Number	*1*	0	1	1	0	0	0	0	0	1	3
	2	0	0	0	1	0	0	0	0	0	1
	3	1	0	0	0	1	1	1	1	0	5
Color	*r*	1	1	0	0	0	1	0	1	1	5
	g	0	0	1	0	1	0	0	0	0	2
	p	0	0	0	1	0	0	1	0	0	2
Filling	*f*	1	0	0	1	1	1	1	0	0	5
	e	0	1	1	0	0	0	0	1	1	4
	s	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Shape	*s*	0	0	1	0	0	1	0	0	0	2
	o	1	0	0	0	0	0	1	1	1	3
	d	0	1	0	1	1	0	0	0	0	4



- Our reformulation algorithm for Set
  - Uses 'Domain Table' & 'Summary of Domain Table'
  - Has 4 tests & 5 heuristics

Algorithms: Finding all Solutions

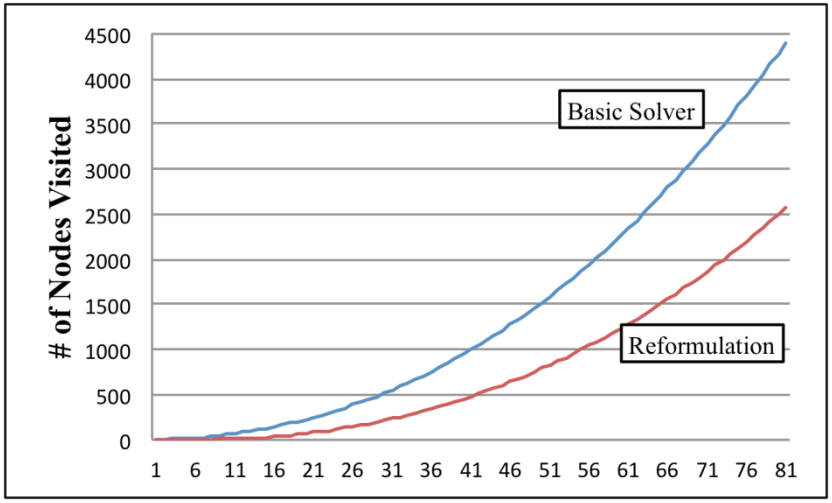
1. Brute-force search (BF)
   - 3-nested *for*-loops generate all combinations, then test for solutions
   - Contradicts 40+ years of CP research & experience ☹
   - Does not scale ($12^3 \sim d^n$)
2. Backtrack search (Basic Solver)
   - Symmetry breaking (lexicographic ordering)
   - Both forward-checking (equality) & back-checking (All-diff constraints)
3. Reformulation-based algorithm
   - Uses 2 data structures: 'Domain Table' & 'Summary of Domain Table'
   - Includes 5 selection heuristics
   - Open subproblems maintained in an agenda: room for heuristics (1Sol)

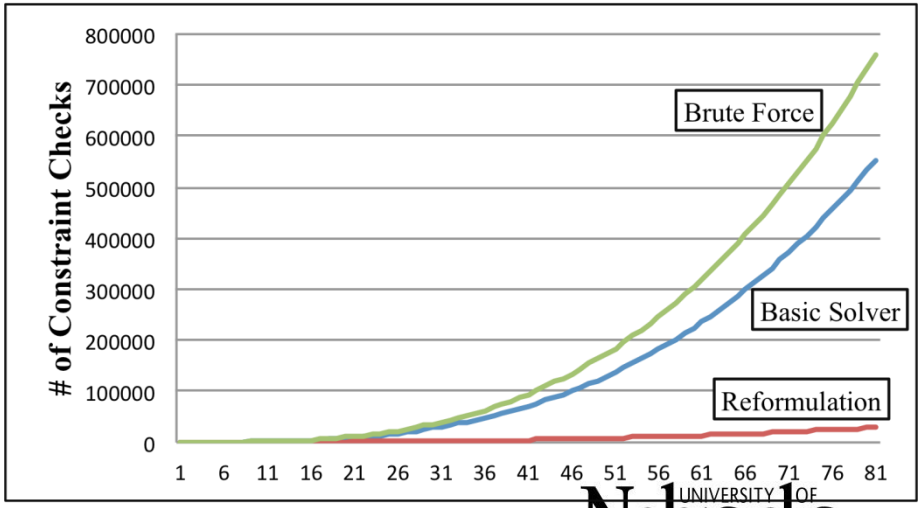
- Empirical tests: randomly selected 'hands' of 3 to 81 cards, results averaged over of 1,000 runs

Results

- **#CC,#NV**: Reformulation dramatically reduces # of combinations tested
- **CPU time** reflects the cost of setting up the data structures for the CSP & search

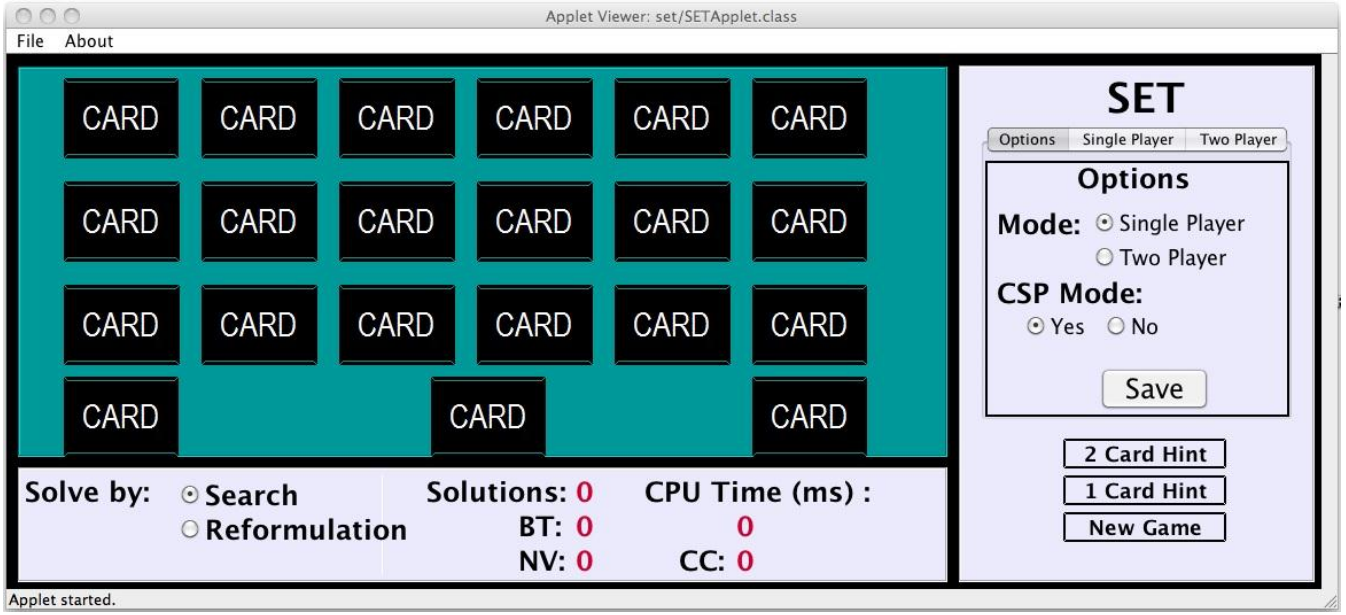
Algorithm	#Cards	#Sol	#CC	#NV	Time [msec]
Brute Force	12	2.77	1956.8	220	**0**
BT Search			1726.6	80.77	62.46
Reformulation			**85.1**	**12.65**	5.85
Brute Force	81	1080	758808	85320	**0**
BT Search			553365	4401	101.04
Reformulation			**31158**	**2565**	39.44





Online Game gameofset.unl.edu

- Game running online
- Interface explaining the reformulation still in development
- Advertzmt: minesweeper.unl.edu & sudoku.unl.edu (CP-based)



Conclusions

- Contributions
  - A systematic approach to reformulation and ‘conditional’ symmetries
  - Applicability to real-world problems highly promising
  - A new toy problem for AI research & education ☺
- Technical issues
  - Generalize heuristics for dimension selection and problem decomposition
  - Explore other types of interchangeability/symmetries
  - Extend definition of MD-CSP to allow unequal/all-diff domains
- Modeling lesson
  - CSP variables and values are often ‘objects’ with attributes
  - So far, we have integrated those attributes in the constraint definitions
  - Let’s rethink CSP modeling: Maybe multi-dimensional CSPs are more common than we thought they are..