Relational Neighborhood Inverse Consistency for Constraint Satisfaction:

A Structure-Based Approach for Adjusting Consistency & Managing Propagation

Robert J. Woodward

Constraint Systems Laboratory
Department of Computer Science & Engineering
University of Nebraska-Lincoln

Acknowledgements

- Software platform developed in collaboration with Shant Karakashian (to a large extent) & Chris Reeson
- Elizabeth Claassen & David B. Marx of the Department of Statistics @ UNL
- Experiments conducted at UNL's Holland Computing Center
- NSF Graduate Research Fellowship & NSF Grant No. RI-111795

Main Contributions

1. Relational Neighborhood Inverse Consistency (RNIC)
   - Characterization on binary & non-binary CSPs
   - An algorithm for enforcing RNIC
   - Comparison to other consistency properties
2. Variations of RNIC
   - Reformulation by redundancy removal, triangulation, both
   - A strategy for selecting the appropriate variation
3. Managing constraint propagation
   - Four queue-management strategies (QMSs)
4. Empirical evaluations on benchmark problems

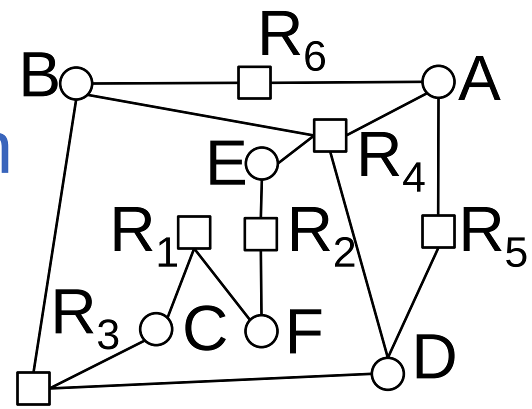
Outline

- Background
- Relational Neighborhood Inverse Consistency (RNIC)
  - Property, characterization
- Dual Graphs of Binary CSPs
  - Complete constraint network
  - Non-complete constraint network
  - RNIC on binary CSPs
- Enforcing RNIC
  - Algorithm for RNIC
  - Dual-graph reformulation
  - Selection strategy
- Evaluating RNIC
- Propagation-Queue Management
- Conclusions & Future Work

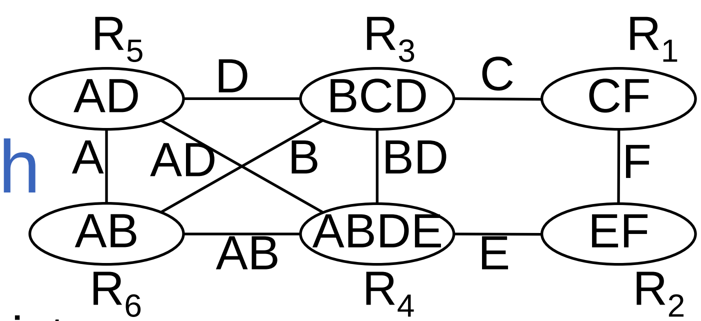
Constraint Satisfaction Problem

- CSP
  - Variables, Domains
  - Constraints: Relations & scopes
- Representation
  - Hypergraph
  - Dual graph
- Solved with
  - Search
  - Enforcing consistency
  - Lookahead = Search + enforcing consistency
- Warning
  - Consistency properties vs. algorithms

Hypergraph

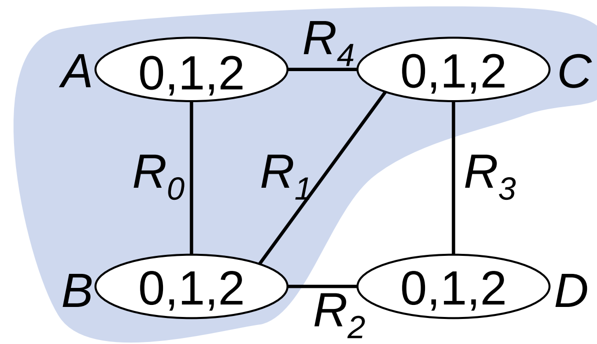


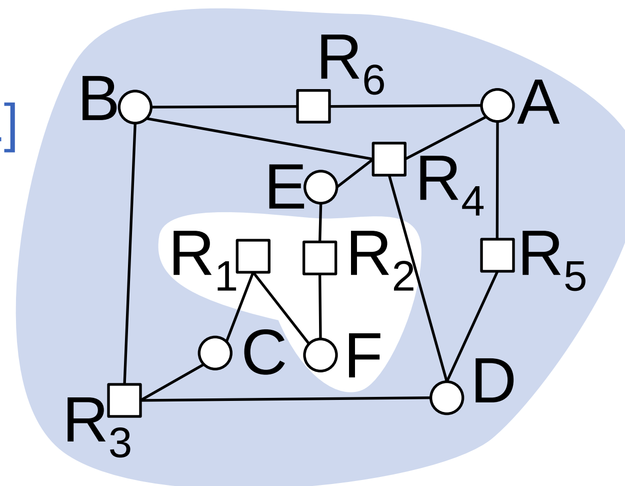
Dual graph



Neighborhood Inverse Consistency

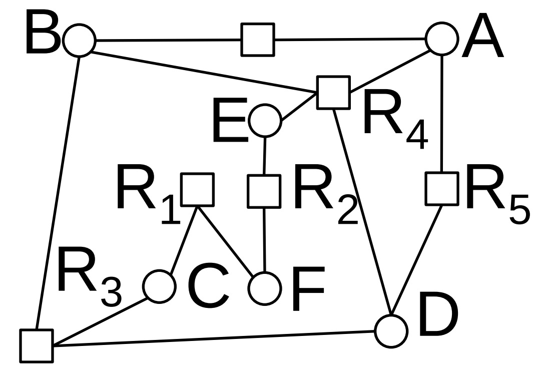
- Property [Freuder+ 96]
  - ↪ Every value can be extended to a solution in its variable's neighborhood
  - ↪ Domain-based property
- Algorithm
  - + No space overhead
  - + Adapts to graph connectivity
- Binary CSPs [Debruyene+ 01]
  - − Not effective on sparse problems
  - − Too costly on dense problems
- Non-binary CSPs?
  - − Neighborhoods likely too large



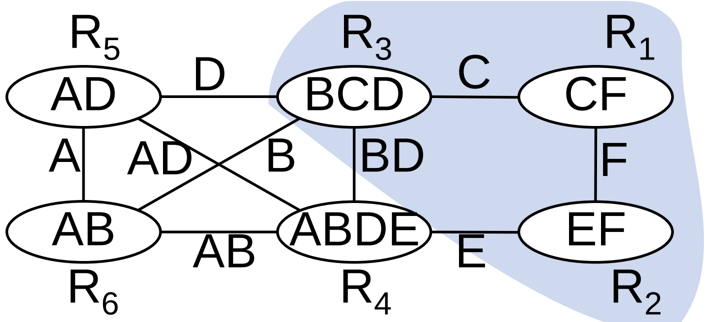


Relational NIC

- Property
  - ↪ Every tuple can be extended to a solution in its relation's neighborhood
  - ↪ Relation-based property
- Algorithm
  - Operates on dual graph
  - Filters relations
  - Does not alter topology of graphs
- Domain filtering
  - Property: RNIC+DF
  - Algorithm: Projection



Hypergraph



Dual graph

From NIC to RNIC

- Neighborhood Inverse Consistency (NIC) [Freuder+ 96]
  - Proposed for binary CSPs
  - Operates on constraint graph
  - Filters domain of variables
- Relational Neighborhood Inverse Consistency (RNIC)
  - Proposed for both binary & non-binary CSPs
  - Operates on dual graph
  - Filters relations; last step projects updated relations on domains
- Both
  - Adapt consistency level to local topology of constraint network
  - Add no new relations (constraint synthesis)
- NIC was shown to be ineffective or costly, we show that RNIC is worthwhile

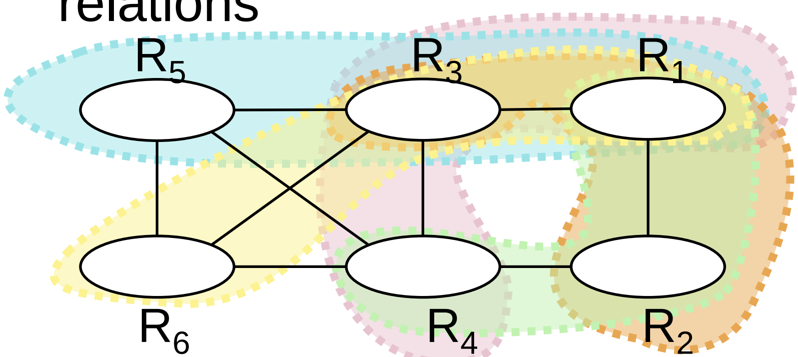
Characterizing RNIC: Binary CSPs

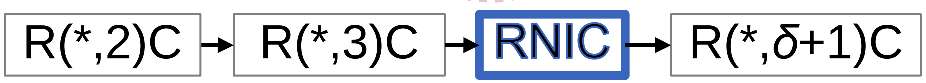
- On binary CSPs [Luchtel, 2011]
  - NIC (on the constraint graph) and RNIC (on the dual graph) are not comparable
  - Empirically, RNIC does more filtering than NIC

Characterizing RNIC (I): Nonbinary CSPs

R(*,*m*)C [Karakashian+ 10]

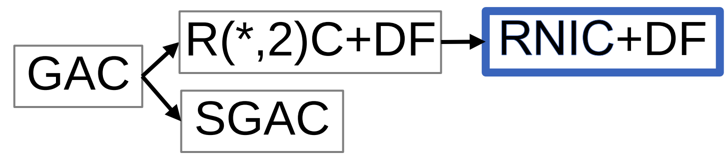
- Relation-based property
- Every tuple has a support in every subproblem induced by a combination of *m* connected relations





GAC, SGAC

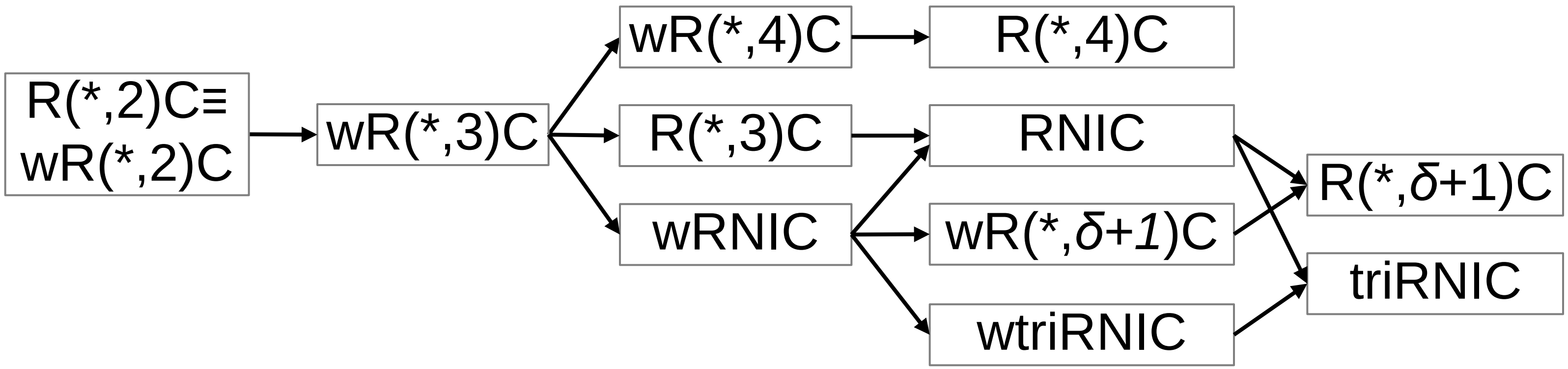
- Variable-based properties
- So far, most popular for non-binary CSPs



$p \rightarrow p'$: p is strictly weaker than p'

Characterizing RNIC (II): Nonbinary CSPs

- The fuller picture, details are in the thesis



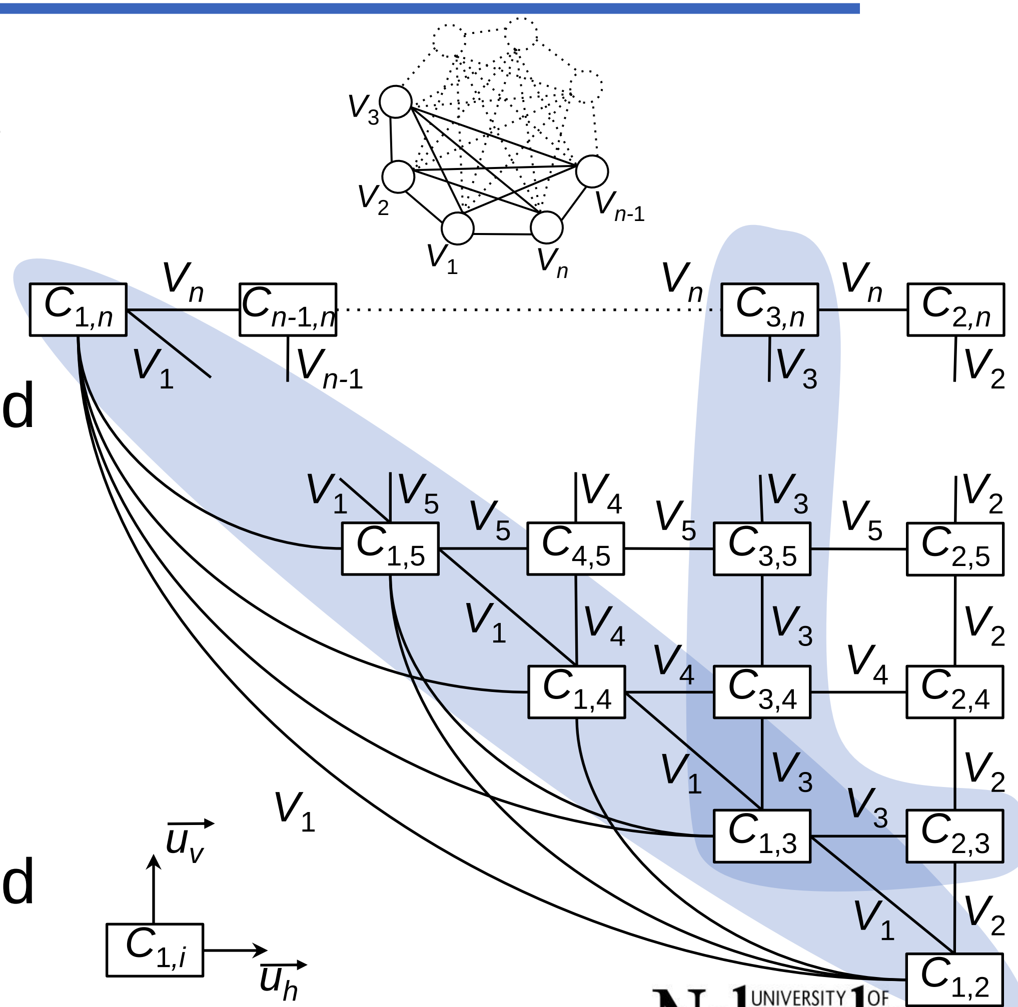
  - w: Property weakened by redundancy removal
  - tri: Property strengthened by triangulation
  - *δ*: Degree of dual network

Outline

- Background
- Relational Neighborhood Inverse Consistency
  - Property, characterization
- **Dual Graphs of Binary CSPs**
  - **Complete constraint network**
  - **Non-complete constraint network**
  - **RNIC on binary CSPs**
- Enforcing RNIC
  - Algorithm for RNIC
  - Dual-graph reformulation
  - Selection strategy
- Evaluating RNIC
- Propagation-Queue Management
- Conclusions & Future Work

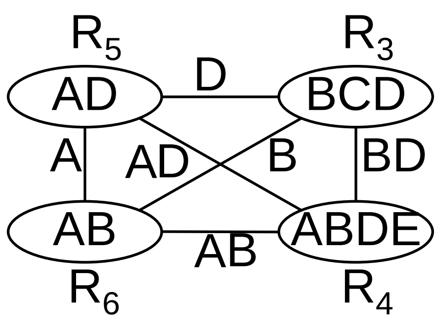
Complete Binary CSPs

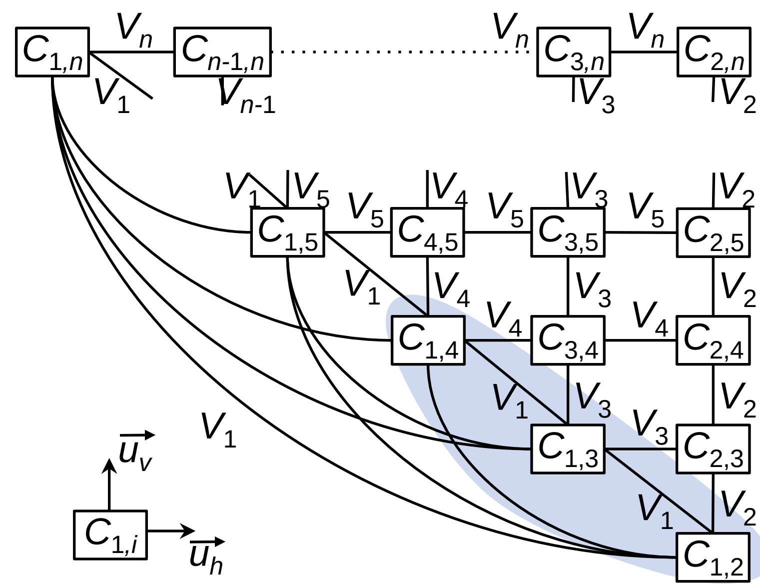
- Triangle-shaped grid
- n-1 vertices for V_1
  - $C_{1,i}$ $i \in [2,n]$
  - Completely connected
- n-1 vertices for $V_{i \geq 2}$
  - Centered on $C_{1,i}$
  - i-2 along horizontal
  - n-i along vertical
  - Completely connected
    - Not shown for clarity



Complete Binary CSPs: RR (I)

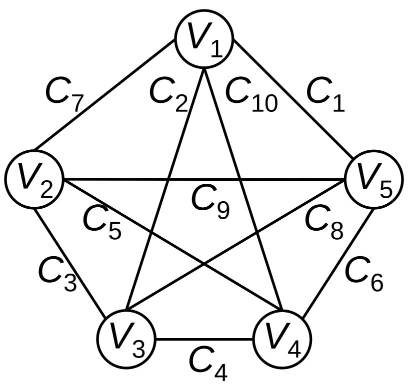
- An edge is redundant if
  - There exists an alternate path between two vertices
  - Shared variables appear in every vertex in the path
- A triangle-shaped grid
  - Every CSP variable annotates a chain of length *n*-2
  - Remove edges that link two non-consecutive vertices

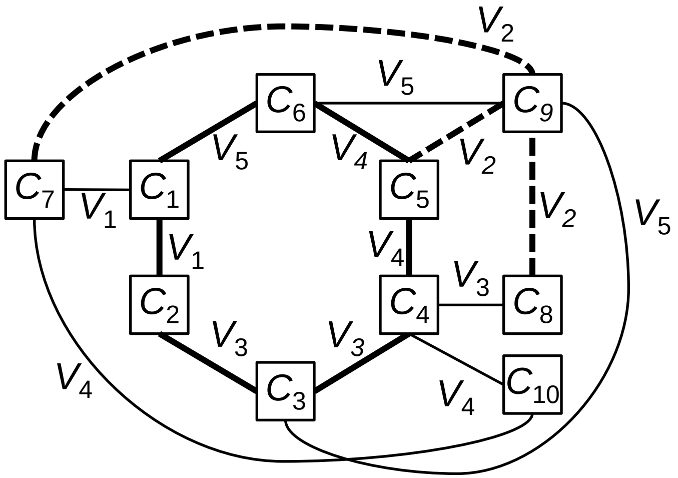




Complete Binary CSPs: RR (II)

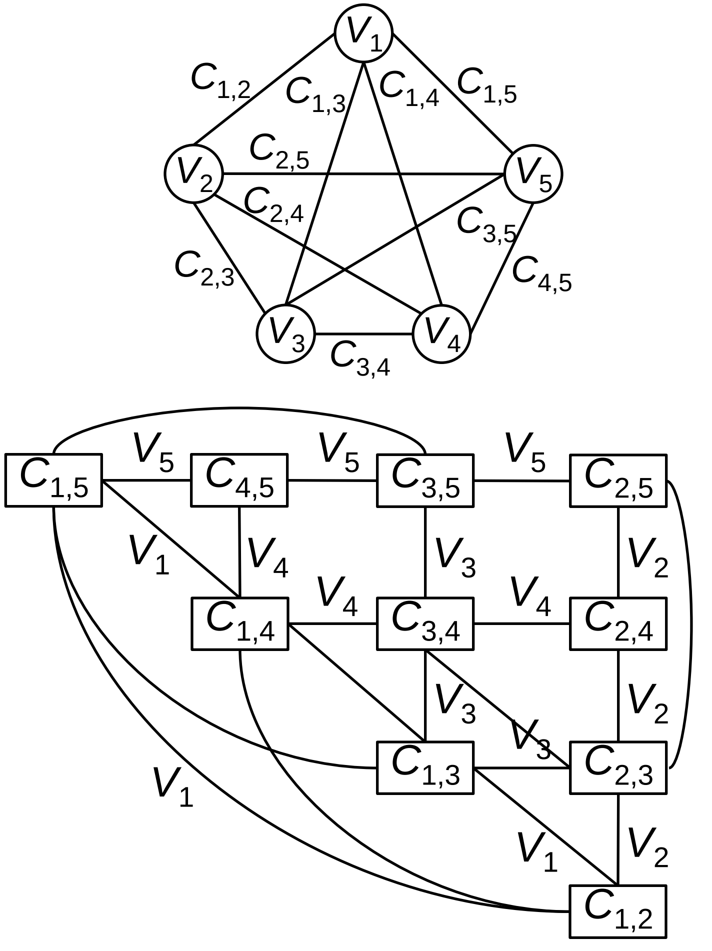
- A redundancy-free dual graph is not unique
- No chain for V_2, but a star





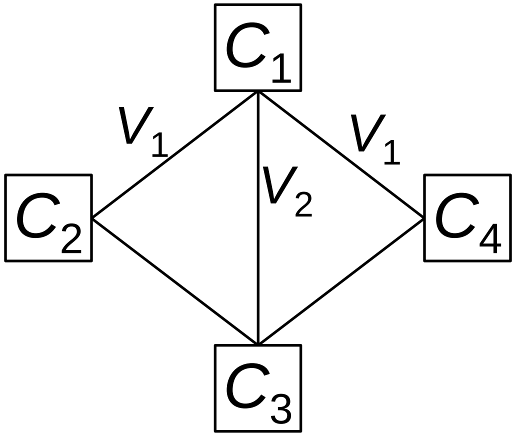
Non-complete Binary CSPs

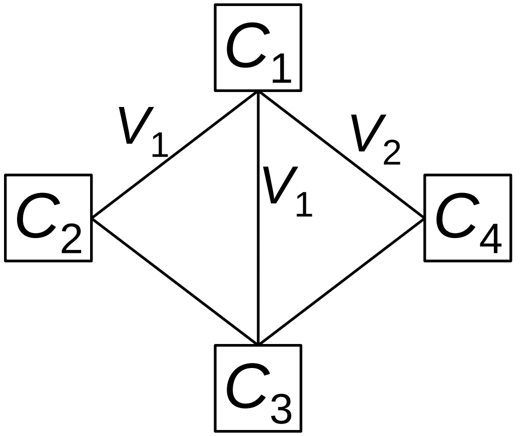
- Non-complete binary CSP
  - Is a complete binary constraint graph with missing edges
- In the dual graph
  - There are missing dual vertices
  - The dual vertices with variable V_i in their scope are completely connected
- Redundancy-free dual graph
  - Can still form a chain using alternate edges



RNIC on Binary CSPs

- After RR, RNIC is never stronger than R(*,3)C
- Configurations for R(*,4)C
  - C_1 has three adjacent constraints C_2, C_3, C_4
  - C_1 is not an articulation point
- Two configurations, neither is possible



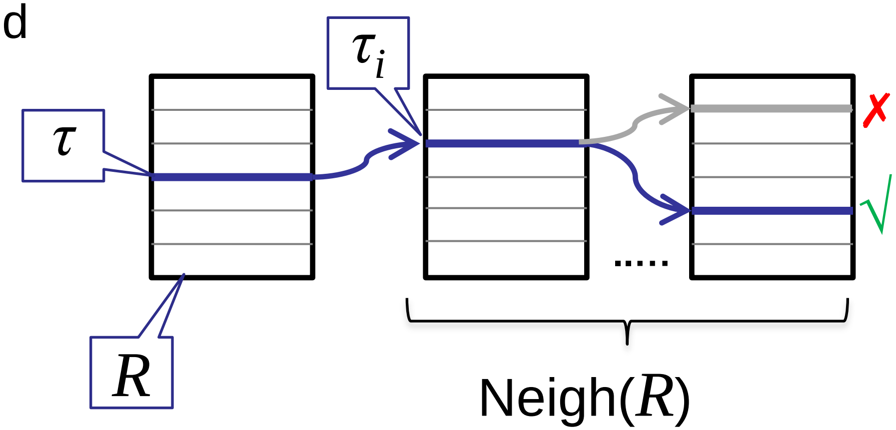


Outline

- Background
- Relational Neighborhood Inverse Consistency
  - Property, characterization
- Dual Graphs of Binary CSPs
  - Complete constraint network
  - Non-complete constraint network
  - RNIC on binary CSPs
- **Enforcing RNIC**
  - **Algorithm for RNIC**
  - **Dual-graph reformulation**
  - **Selection strategy**
- Evaluating RNIC
- Propagation-Queue Management
- Conclusions & Future Work

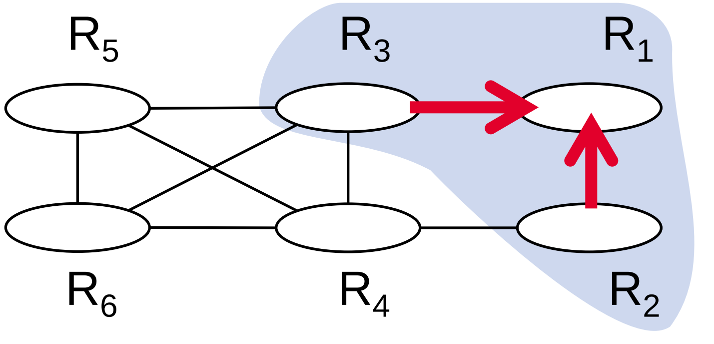
Algorithm for Enforcing RNIC

- Two queues
  1. Q: relations to be updated
  2. $Q_t(R)$: The tuples of relation R whose supports must be verified
- SEARCHSUPPORT(τ,R)
  - Backtrack search on Neigh(R)
- Loop until all $Q_t(\cdot)$ are empty
- Complexity
  - Space: $O(ket\delta)$
  - Time: $O(t^{\delta+1}e\delta)$
  - Efficient for a fixed δ



Improving Algorithm's Performance

1. Use IndexTree [Karakashian+ AAAI10]
   – To quickly check consistency of 2 tuples
2. Dynamically detect dangles
   – Tree structures may show in subproblem @ each instantiation
   – Apply directional arc consistency

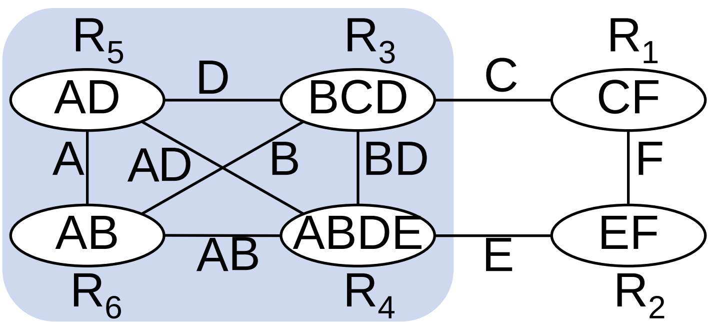


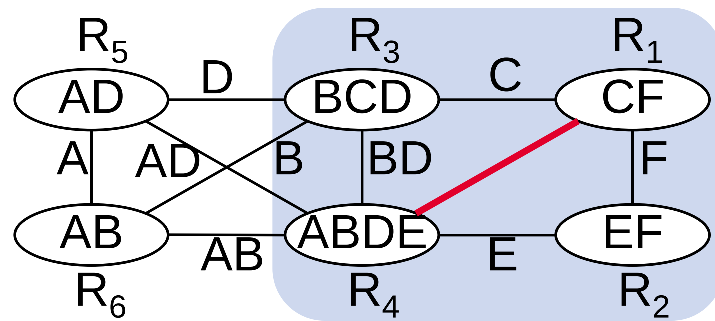
Note that exploiting dangles is

– Not useful in R(*,*m*)C: small value of *m*, subproblem size
– Not applicable to GAC: does not operate on dual graph

Reformulating the Dual Graph

- High degree
  - Large neighborhoods
  - High computational cost
- Redundancy Removal (wRNIC)
  - Use minimal dual graph
- Cycles of length ≥ 4
  - Hampers propagation
  - RNIC≡R(*,3)C
- Triangulation (triRNIC)
  - Triangulate dual graph



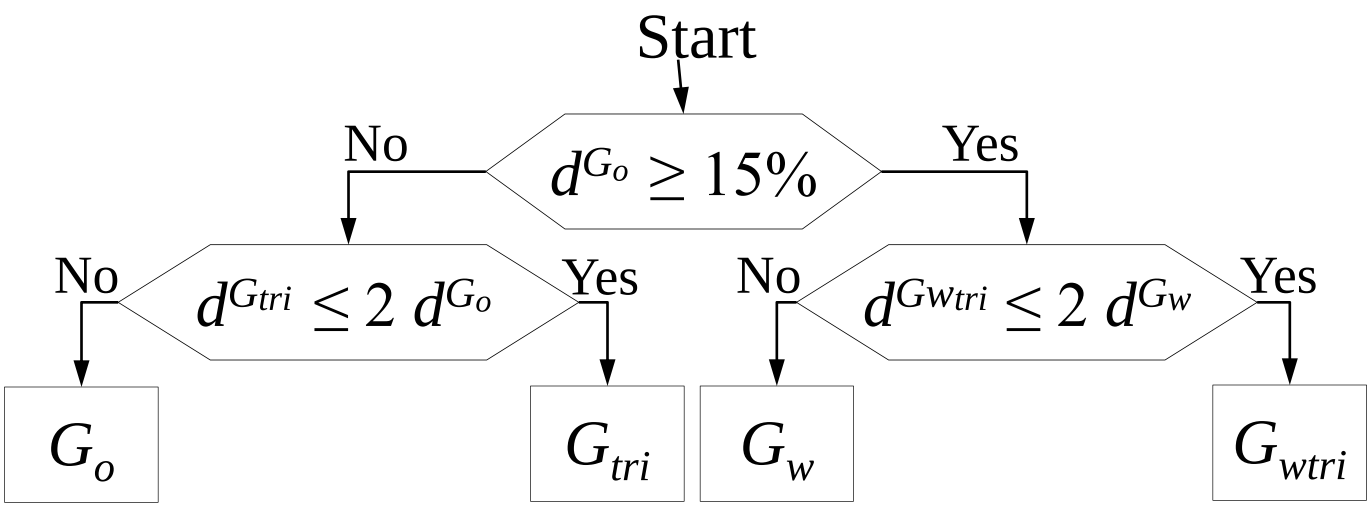


RR+Triangulation (wtriRNIC)

- Local, complementary, do not 'clash'

Selection Strategy: Which? When?

- Density of dual graph ≥ 15% is too dense
  - Remove redundant edges
- Triangulation increases density no more than two fold
  - Reformulate by triangulation
- Each reformulation executed at most once



Outline

- Background
- Relational Neighborhood Inverse Consistency
  - Property, characterization
- Dual Graphs of Binary CSPs
  - Complete constraint network
  - Non-complete constraint network
  - RNIC on binary CSPs
- Enforcing RNIC
  - Algorithm for RNIC
  - Dual-graph reformulation
  - Selection strategy
- **Evaluating RNIC**
- Propagation-Queue Management
- Conclusions & Future Work

Experimental Setup

- Backtrack search with full lookahead
- We compare
  - wR(*,m)C for $m = 2,3,4$
  - GAC
  - RNIC and its variations
- General conclusion
  - GAC best on random problems
  - RNIC-based best on structured/quasi-structued problems
- We focus on non-binary results (960 instances)
  - triRNIC theoretically has the least number of nodes visited
  - selRNIC solves most instances backtrack free (652 instances)

Category	#Binary	#Non-binary
Academic	31	0
Assignment	7	50
Boolean	0	160
Crossword	0	258
Latin square	50	0
Quasi-random	390	25
Random	980	290
TSP	0	30
Unsolvable		
Memory	10	60
All timed out	447	87

Experimental Results

- Statistical analysis on CP benchmarks
- **Time**: Censored data calculated mean
- **Rank**: Censored data rank based on probability of survival data analysis
- **#F**: Number of instances fastest
- **[·]$_{CPU}$**: Equivalence classes based on CPU
- **[·]$_{Completion}$**: Equivalence classes based on completion
- **#C**: Number of instances completed
- **#BT-free**: # instances solved backtrack free

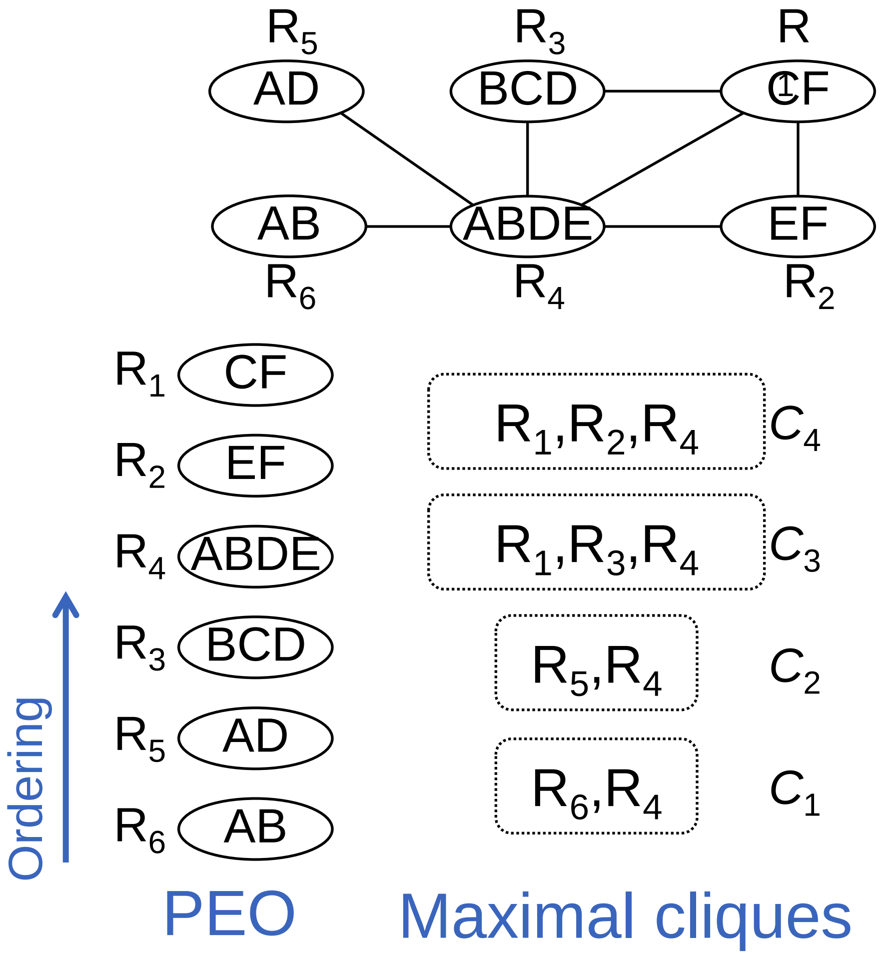
Algorithm	Time	Rank	#F	[·]$_{CPU}$	#C	[·]$_{Completion}$	#BT-free
169 instances: aim-100,aim-200,lexVg,modifiedRenault,ssa							
wR(*,2)C	944924	3	52	**A**	138	B	79
wR(*,3)C	925004	4	8	B	134	B	92
wR(*,4)C	1161261	5	2	B	132	B	108
GAC	1711511	7	**83**	C	119	C	33
RNIC	6161391	8	19	C	100	C	66
triRNIC	3017169	9	9	C	84	C	80
wRNIC	1184844	6	8	B	131	B	84
wtriRNIC	937904	2	3	B	144	B	129
selRNIC	**751586**	**1**	17	**A**	**159**	**A**	**142**

Outline

- Background
- Relational Neighborhood Inverse Consistency
  - Property, characterization
- Dual Graphs of Binary CSPs
  - Complete constraint network
  - Non-complete constraint network
  - RNIC on binary CSPs
- Enforcing RNIC
  - Algorithm for RNIC
  - Dual-graph reformulation
  - Selection strategy
- Evaluating RNIC
- **Propagation-Queue Management**
- Conclusions & Future Work

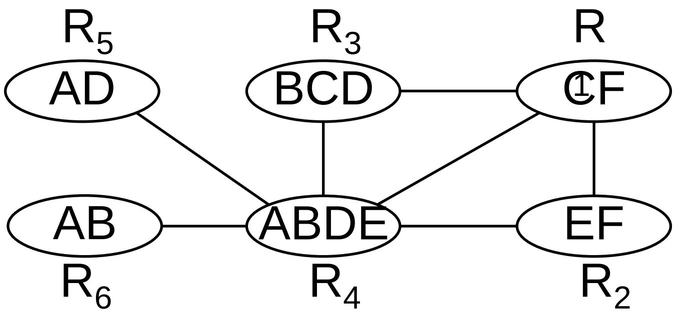
Propagation-Queue Management

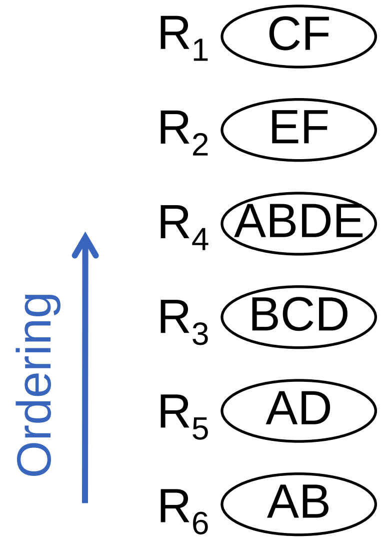
- Three directions for ordering the relations:
  1. Arbitrary ordering (previous)
  2. Perfect elimination ordering (PEO) of some triangulation
  3. Ordering of the maximal cliques, corresponds to a tree-decomposition ordering (TD)



Queue-Management Strategies

Exact	QMS_a	Arbitrary ordering of relations
	QMS_{PEO}	Perfect elimination ordering
	QMS_{TD}	Sequence of maximal cliques The cliques revised in sequence • Each clique is revised until quiescence • Revised back and forth until quiescence
Lazy	QMS_{LTD}	Same as QMS_{TD}, except • Cliques are traversed only once
	QMS_{L^2TD}	Same as QMS_{TD}, except traverses • each clique only once • each relation only once

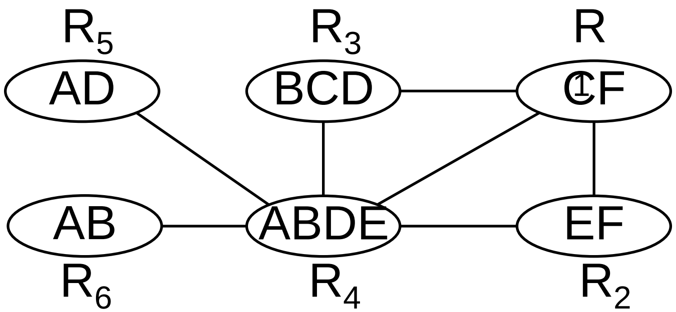


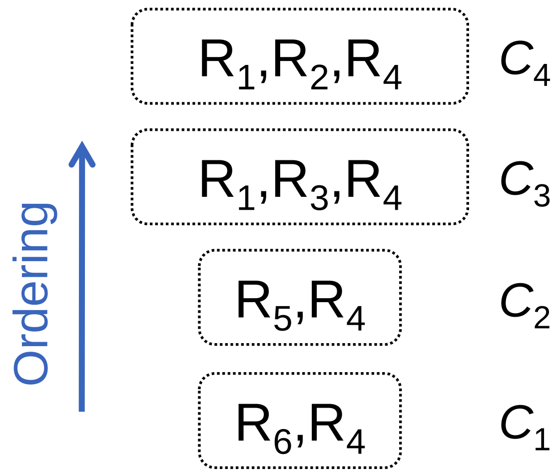


PEO

Queue-Management Strategies

Exact	QMS_a	Arbitrary ordering of relations
Exact	QMS_{PEO}	Perfect elimination ordering
Exact	QMS_{TD}	Sequence of maximal cliques The cliques revised in sequence • Each clique is revised until quiescence • Revised back and forth until quiescence
Lazy	QMS_{LTD}	Same as QMS_{TD}, except • Cliques are traversed only once
Lazy	QMS_{L^2TD}	Same as QMS_{TD}, except traverses • each clique only once • each relation only once





maximal cliques

Propagation-Queue Management

- Statistical analysis on CP benchmarks
- **Time**: Censored data rank based on probability of survival data analysis
- **$[\cdot]_{CPU}$**: Equivalence classes based on CPU
- **%**: Percent increased gain by the algorithm

triRNIC Pre-processing

				SAT	UNSAT
Strategy	**Time**	**$[\cdot]_{CPU}$**	**%**	$[\cdot]_{CPU}$	$[\cdot]_{CPU}$
QMS_a	1,410,292	C	-	**A**	C
QMS_{PEO}	1,186,691	B	16%	**A**	B
QMS_{TD}	**765,976**	**A**	**46%**	**A**	**A**

wtriRNIC Pre-processing

				SAT	UNSAT
Strategy	**Time**	**$[\cdot]_{CPU}$**	**%**	$[\cdot]_{CPU}$	$[\cdot]_{CPU}$
QMS_a	479,725	**A**	-	**A**	B
QMS_{PEO}	**467,747**	**A**	**2%**	**A**	**A**
QMS_{TD}	476,604	**A**	1%	**A**	B

triRNIC Search

				SAT	UNSAT
Strategy	**Time**	**$[\cdot]_{CPU}$**	**%**	$[\cdot]_{CPU}$	$[\cdot]_{CPU}$
QMS_a	1,243,917	C	-	**A**	C
QMS_{PEO}	900,069	B	28%	**A**	B
QMS_{TD}	416,464	**A**	67%	**A**	**A**
QMS_{LTD}	**403,766**	**A**	**68%**	**A**	**A**
QMS_{L2TD}	434,479	**A**	65%	**A**	**A**

wtriRNIC Search

				SAT	UNSAT
Strategy	**Time**	**$[\cdot]_{CPU}$**	**%**	$[\cdot]_{CPU}$	$[\cdot]_{CPU}$
QMS_a	628,523	C	-	**A**	C
QMS_{PEO}	582,629	B	7%	**A**	B
QMS_{TD}	**519,578**	**A**	**17%**	**A**	**A**
QMS_{LTD}	602,437	C	4%	B	**A**
QMS_{L2TD}	575,277	C	8%	B	**A**

Conclusions

- RNIC
- Structure of binary dual graph
- Algorithm for enforcing RNIC
  - Polynomial for fixed-degree dual graphs
  - BT-free search: hints to problem tractability
- Various reformulations of the dual graph
- Adaptive, unifying, self-regulatory, automatic strategy
- New propagation-queue management strategies
- Empirical evidence, supported by statistics

Future Work

- Extension to singleton-type consistencies
- Extension to constraints defined in intension
  - Possible by only domain filtering (weakening)
- Study influence of redundancy removal algorithms
  - Redundancy removal algorithm of [Janssen+ 89] seems to favor grids
- Evaluate new queue-management strategies on other consistency algorithms

Thank You!

Questions?