2011

Solving Difficult CSPs with Relational Neighborhood Inverse Consistency

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

Shant Karakashian
*University of Nebraska-Lincoln*, shantk@cse.unl.edu

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

Christian Bessiere
*University of Montpellier, France*, bessiere@lirmm.fr

Follow this and additional works at: [https://digitalcommons.unl.edu/cseconfwork](https://digitalcommons.unl.edu/cseconfwork)

Part of the *Computer Sciences Commons*


[https://digitalcommons.unl.edu/cseconfwork/183](https://digitalcommons.unl.edu/cseconfwork/183)

This Article is brought to you for free and open access by the Computer Science and Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in CSE Conference and Workshop Papers by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.
Solving Difficult CSPs with Relational Neighborhood Inverse Consistency

R.J. Woodward¹, S. Karakashian¹, B.Y. Choueiry¹ & C. Bessiere²

¹Constraint Systems Laboratory, University of Nebraska-Lincoln
²LIRMM-CNRS, University of Montpellier

Acknowledgements
• Elizabeth Claassen and David B. Marx of the Department of Statistics @ UNL
• Experiments conducted at UNL’s Holland Computing Center
• Robert Woodward supported by a B.M. Goldwater Scholarship and NSF Graduate Research Fellowship
• NSF Grant No. RI-111795
Outline

- Introduction
- Relational Neighborhood Inverse Consistency
  - Property, Algorithm, Improvements
- Reformulating the Dual Graph by
  1. Redundancy removal $\rightarrow$ property wRNIC
  2. Triangulation $\rightarrow$ property triRNIC
- Selection strategy: four alternative dual graphs
- Experimental Results
- Conclusion
Constraint Satisfaction Problem

- **CSP**
  - Variables, Domains
  - Constraints: Relations & scopes

- **Representation**
  - Hypergraph
  - Dual graph

- **Solved with**
  - Search
  - Enforcing consistency

- **Warning**
  - Consistency properties vs. algorithms
Neighborhood Inverse Consistency

• Property
  ↷ 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
Characterizing RNIC

**R(\(*,m\)C**

- 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

\[ R(\*,2)C + DF \rightarrow \text{RNIC+DF} \]

\[ \text{GAC} \]

\[ \text{SGAC} \]

\[ R(\*,\delta+1)C \]

\[ R(\*,3)C \rightarrow \text{RNIC} \rightarrow R(\*,2)C \]

\[ p \rightarrow p' : \text{ } p \text{ is strictly weaker than } p' \]
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 $(\tau,R)$
  - Backtrack search on $\text{Neigh}(R)$
- Loop $Q$ until is empty
- Complexity
  - Space: $O(\text{ket}\delta)$
  - Time: $O(t^{\delta+1}e\delta)$
  - Efficient for a fixed $\delta$
Improving Algorithm’s Performance

1. Use IndexTree
   - 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 $\geq 4$
  - Hampers propagation
  - $\text{RNIC} \equiv R^{(*,3)}C$

- Triangulation (triRNIC)
  - Triangulate dual graph

RR+Triangulation (wtriRNIC)
- Local, complementary, do not ‘clash’
Selection Strategy: Which? When?

- Density of dual graph $\geq 15\%$ is too dense
  - Remove redundant edges
- Triangulation increases density no more than two fold
  - Reformulate by triangulation
- Each reformulation executed at most once
# 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
- **EquivCPU**: Equivalence classes based on CPU
- **EquivCmp**: Equivalence classes based on completion
- **#C**: Number of instances completed
- **#BT-free**: # instances solved backtrack free

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Time</th>
<th>Rank</th>
<th>#F</th>
<th>EquivCPU</th>
<th>#C</th>
<th>EquivCmp</th>
<th>#BT-free</th>
</tr>
</thead>
<tbody>
<tr>
<td>wR(*,2)C</td>
<td>944924</td>
<td>3</td>
<td>52</td>
<td>A</td>
<td>138</td>
<td>B</td>
<td>79</td>
</tr>
<tr>
<td>wR(*,3)C</td>
<td>925004</td>
<td>4</td>
<td>8</td>
<td>B</td>
<td>134</td>
<td>B</td>
<td>92</td>
</tr>
<tr>
<td>wR(*,4)C</td>
<td>1161261</td>
<td>5</td>
<td>2</td>
<td>B</td>
<td>132</td>
<td>B</td>
<td>108</td>
</tr>
<tr>
<td>GAC</td>
<td>1711511</td>
<td>7</td>
<td>83</td>
<td>C</td>
<td>119</td>
<td>C</td>
<td>33</td>
</tr>
<tr>
<td>RNIC</td>
<td>6161391</td>
<td>8</td>
<td>19</td>
<td>C</td>
<td>100</td>
<td>C</td>
<td>66</td>
</tr>
<tr>
<td>triRNIC</td>
<td>3017169</td>
<td>9</td>
<td>9</td>
<td>C</td>
<td>84</td>
<td>C</td>
<td>80</td>
</tr>
<tr>
<td>wRNIC</td>
<td>1184844</td>
<td>6</td>
<td>8</td>
<td>B</td>
<td>131</td>
<td>B</td>
<td>84</td>
</tr>
<tr>
<td>wtriRNIC</td>
<td>937904</td>
<td>2</td>
<td>3</td>
<td>B</td>
<td>144</td>
<td>B</td>
<td>129</td>
</tr>
<tr>
<td>selRNIC</td>
<td>751586</td>
<td>1</td>
<td>17</td>
<td>A</td>
<td>159</td>
<td>A</td>
<td>142</td>
</tr>
</tbody>
</table>

169 instances: aim-100, aim-200, lexVg, modifiedRenault, ssa
Conclusions

• Contributions
  – RNIC
  – Algorithm
    • 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
  – Empirical evidence, supported by statistics

• Future work
  – Extend to singleton-type consistencies
Check Poster in Student Poster Session Wednesday, Aug 10 @ 6:30PM Grand Ballroom