Adaptive Neighborhood Inverse Consistency as Lookahead for Non-Binary CSPs

Robert J. Woodward[1] **Shant Karakashian**[1] **Berthe Y. Choueiry**[1] **Christian Bessiere**[2]

[1]Constraint Systems Laboratory, University of Nebraska-Lincoln, USA
{*rwoodwar*|*shantk*|*choueiry*}*@cse.unl.edu*

[2]LIRMM-CNRS, University of Montpellier, France
bessiere@lirmm.fr

Abstract

Freuder and Elfe (1996) introduced Neighborhood Inverse Consistency (NIC) for binary CSPs. In this paper, we introduce RNIC, the extension of NIC to non-binary CSPs, and describe a practical algorithm for enforcing it. We propose an adaptive strategy to weaken or strengthen this property based on the connectivity of the network. We demonstrate the effectiveness of RNIC as a full lookahead strategy during search for solving difficult benchmark problems.

1 Introduction

Solving difficult Constraint Satisfaction Problems (CSPs) remains a challenge today despite the dramatic advances in hardware technology. To counter the exponential growth of the size of the search space of CSPs, consistency properties and algorithms for implementing them have been proposed since the inception of Constraint Programming (CP). While lower levels of consistency, such as Arc Consistency (AC) for binary constraints and Generalized Arc Consistency (GAC) for non-binary constraints, are commonly and advantageously used, solving difficult problems often requires enforcing higher orders of consistency, which typically increases time and/or space requirements. Freuder and Elfe (1996) introduced Neighborhood Inverse Consistency (NIC) for binary CSPs as a particularly promising consistency property because (1) it has no space overhead (it is enforced by filtering the variables domains), and (2) the enforced consistency level depends directly on the connectivity of each variable with its neighborhood. Despite its promise and filtering effectiveness, NIC remained relatively unexploited because the algorithm for enforcing it is too costly, which prevented its use on dense networks or in a lookahead scheme during backtrack search.

In this paper, we introduce Relational Neighborhood Inverse Consistency (RNIC) as a generalization of this property to non-binary CSPs. We also introduce weakened and strengthened variations of RNIC to cope with the difficulties raised by the topology of the dual graph, and propose a strategy for automatically choosing the property to enforce.

2 Background

A CSP is defined by $\mathcal{P} = (\mathcal{V}, \mathcal{D}, \mathcal{C})$ where $\mathcal{V}$ is a set of variables, $\mathcal{D}$ is a set of domains, and $\mathcal{C}$ is a set of constraints or relations. Each variable $V_i \in \mathcal{V}$ has a finite domain $D_i \in \mathcal{D}$, and is constrained by a subset of the relations in $\mathcal{C}$. In a relation R, tuple $\tau \in R$ is a combination of allowed values for the variables in the scope of R. Solving a CSP corresponds to assigning a value to each variable such that all the constraints are satisfied. We denote by $\mathcal{P}^{\mathcal{D}}$ the dual encoding of a CSP $\mathcal{P}$. $\mathcal{P}^{\mathcal{D}}$ is a binary CSP whose variables are the relations of $\mathcal{P}$, their domains are the tuples of those relations, and the constraints enforce *equalities* over the shared variables. Finally, the *dual graph* of a CSP is a graph whose vertices represent the relations of the CSP, and whose edges connect two vertices corresponding to relations whose scopes overlap. $\mathrm{Neigh}(R_i)$ denotes the set of relations adjacent to a relation R_i in the dual graph.

The consistency property R($*$,m)C introduced in (Karakashian et al. 2010) ensures that each tuple in each relation can be extended in a consistent assignment to every combination of $m-1$ relations in the problem.

3 Relational NIC

Definition 1 *A relation R_i is said to be* RNIC *iff each tuple in R_i can be extended to the variables in $\bigcup_{R_j \in \mathrm{Neigh}(R_i)} scope(R_j) \setminus scope(R_i)$ in an assignment that simultaneously satisfies all the relations in* $\mathrm{Neigh}(R_i)$. *A network is* RNIC *iff every relation is* RNIC.

Informally, each tuple τ_i in each relation R_i can be extended to a tuple τ_j in each $R_j \in \mathrm{Neigh}(R_i)$ such that together all those tuples are consistent with all the relations in $\mathrm{Neigh}(R_i)$. Such a set of tuples $\{\tau_j\}$ is called a *support* of τ_i. RNIC is enforced by filtering the existing relations and without introducing any new relations to the CSP. A straightforward algorithm for enforcing RNIC applies Expression (1) to each R_i until a fixed point is reached:

$$R_i \leftarrow \pi_{scope(R_i)}(\bowtie_{R_j \in \{R_i\} \cup \mathrm{Neigh}(R_i)} R_j) \qquad (1)$$

where π and $\bowtie$ are the relational operators project and join. The space requirement of this algorithm is prohibitive.

Algorithm for enforcing RNIC For each relation R_i of a CSP $\mathcal{P}$, our algorithm for enforcing RNIC conducts a backtrack search the subproblem induced by $\{R_i\} \cup \mathrm{Neigh}(R_i)$

on the dual graph, $\mathcal{P}^{\mathcal{D}}$, of $\mathcal{P}$ in order to ensure that each tuple $\tau_i \in R_i$ has a valid support. The algorithm maintains a queue, $\mathcal{Q}_R$, of the relations to be revised and, for each relation, a queue of tuples whose support must be verified. Whenever a relation is empty, the algorithm halts and returns false, indicating that $\mathcal{P}$ is not consistent. When $\mathcal{Q}_R$ is empty, it terminates successfully indicating that $\mathcal{P}$ is RNIC. Note that $\mathrm{Neigh}(R_i)$ is determined by the topology of the dual graph, which we will alter in Section 4.

4 Variations on RNIC

The following two conditions of the topology of the dual graph seriously hinder the performance of enforcing RNIC: **(1)** high density and **(2)** the existence of cycles of size 4 or more. The former increases the neighborhood of a vertex in the dual graph, thus increasing the cost of RNIC. The latter prevents the neighbors of a given relation R_i in the cycle from 'communicating' and reduces RNIC to R(∗,2)C. To address those issues, we propose three variations on RNIC, and a selection procedure to automatically decide, at the preprocessing stage, which of the four properties to enforce.

Use a minimal dual graph: To address (1), we enforce RNIC on G_w a minimal dual graph obtained from the original dual graph G_o by removing redundant edges (Janssen et al. 1989). The resulting property, wRNIC, is strictly weaker than RNIC.

Triangulate the dual graph: To address (2), we enforce RNIC on G_{tri}, a triangulation of G_o, thus boosting propagation but also raising the consistency level. The resulting property, triRNIC, is strictly stronger than RNIC.

Triangulate a minimal dual graph: We denote wtriRNIC the consistency enforced on G_{wtri}, a triangulation of G_w.

Figure 1 shows RNIC and R(∗,m)C-based properties in a partial order, where δ is the degree of the dual graph. Ideally,

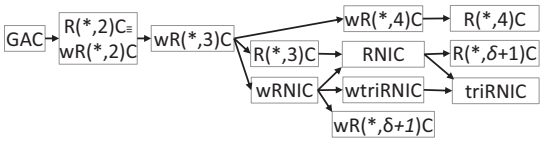

Figure 1: A partial order on RNIC, R(∗,m)C, and their variations.

one should adjust the strength of propagation to the topology of the problem. We propose to measure the density of the original dual graph, G_o, and determine whether to remove redundant edges and/or triangulate the graph. The selection policy of Figure 2 automatically chooses the appropriate dual graph by comparing the density d^G of two dual graphs. The study of 1689 dual CSPs showed a sharp threshold at

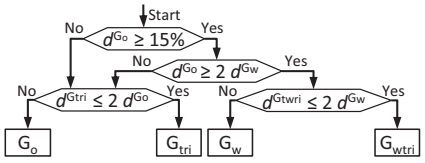

Figure 2: selRNIC enforces RNIC on the chosen dual graph.

15% density. We empirically determined the factor 2 above. Both values are extremely stable. The resulting mechanism, selRNIC, applies triangulation and redundancy removal at most once, and nicely ties together our techniques in a consistent and adaptive framework.

5 Experimental Results

In (Woodward et al. 2011), we compared the performance of the four variations of RNIC, GAC, and wR(∗,m)C for $m = 2, 3, 4$ of (Karakashian et al. 2010) as a full lookahead strategy during backtrack search for solving CSPs. We ran the experiments on the benchmarks of the CSP Solver Competition[1] with a time limit of one and a half hours per instance. On some of the benchmarks (e.g., aim-100), RNIC completed the largest number of instances, and yielded the largest number of backtrack-free searches. Interestingly, on lexVg, and despite the high density (72.6%) of the triangulated dual graph, selRNIC (= triRNIC) solved in a backtrack-free manner all but one of the instances of this benchmark, thus hinting to its tractability. The sheer number of relations in the dual graphs of the modifiedRenault benchmark prevents us from executing RNIC and triRNIC. This situation demonstrates the benefits of using wRNIC and wtriRNIC, which were indeed automatically chosen by selRNIC. Also noteworthy, wtriRNIC solved, backtrack free, all instances in the modifiedRenault benchmark. Finally, RNIC/selRNIC solved two large and difficult instances of the aim-200 benchmark and one instance of the ssa benchmark that *no other algorithm can solve*, the ssa instance being solved backtrack free.

6 Future Work & Conclusions

Our contribution adds to the state of the art of consistency properties and propagation algorithms. Our long-term goal is to enable a constraint solver to identify tractable problem classes and automatically select and apply the appropriate tools for solving them. In that sense, the ability of our approach to adapt to a problem's structure and solve many difficult instances backtrack free is perhaps most noteworthy and indicates that we may be one step closer to achieving this goal.

Acknowledgments Experiments were conducted at the Holland Computing Center facility of the Univ. of Nebraska. Robert Woodward was partially supported by a B.M. Goldwater Scholarship.

References

Freuder, E. C., and Elfe, C. D. 1996. Neighborhood Inverse Consistency Preprocessing. In *Proc. of AAAI-96*, 202–208.

Janssen, P.; Jégou, P.; Nougier, B.; and Vilarem, M. 1989. A Filtering Process for General Constraint-Satisfaction Problems: Achieving Pairwise-Consistency Using an Associated Binary Representation. In *IEEE Workshop on Tools for AI*, 420–427.

Karakashian, S.; Woodward, R.; Reeson, C.; Choueiry, B. Y.; and Bessiere, C. 2010. A First Practical Algorithm for High Levels of Relational Consistency. In *AAAI 10*, 101–107.

Woodward, R.; Karakashian, S.; Choueiry, B. Y.; and Bessiere, C. 2011. Solving Difficult CSPs with Relational Neighborhood Inverse Consistency. In *AAAI 11*, 1–8.

[1] http://www.cril.univ-artois.fr/CPAI09/