Constraint Tightness versus Global Consistency

Peter van Beek
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada T6G 2H1
vanbeek@cs.ualberta.ca

Rina Dechter
Department of Computer and Information Science
University of California, Irvine
Irvine, California, USA 92717
dechter@ics.uci.edu

Abstract

Constraint networks are a simple representation and reasoning framework with diverse applications. In this paper, we present a new property called *constraint tightness* that can be used for characterizing the difficulty of problems formulated as constraint networks. Specifically, we show that when the constraints are tight they may require less preprocessing in order to guarantee a backtrack-free solution. This suggests, for example, that many instances of crossword puzzles are relatively easy while scheduling problems involving resource constraints are quite hard. Formally, we present a relationship between the tightness or restrictiveness of the constraints, and the level of local consistency sufficient to ensure global consistency, thus ensuring backtrack-freeness. Two definitions of local consistency are employed. The traditional variable-based notion leads to a condition involving the tightness of the constraints, the level of local consistency, and the arity of the constraints, while a new definition of *relational* consistency leads to a condition expressed in terms of tightness and local-consistency level, alone. New algorithms for enforcing relational consistency are introduced and analyzed.

1 Introduction

Constraint networks are a simple representation and reasoning framework. A problem is represented as a set of variables, a domain of values for each variable, and a set of constraints between the variables, and the reasoning task is to find an instantiation of the variables that satisfies the constraints. In spite of the simplicity of the framework, many interesting problems can be formulated as constraint networks, including graph coloring [Montanari, 1974], scene labeling [Waltz, 1975], natural language parsing [Maruyama, 1990], and temporal reasoning [Allen, 1983; Dechter *et al.*, 1991; Meiri, 1991; van Beek, 1992].

Constraint networks are often solved using a backtracking algorithm. However, backtracking algorithms are susceptible to "thrashing:" discovering over and over again the same reason for reaching a dead end in the search for a solution. To ameliorate this thrashing behavior, algorithms for preprocessing a constraint network by removing local inconsistencies have been proposed and studied (e.g., [Dechter and Meiri, 1989; Mackworth, 1977; Montanari, 1974]). Sometimes a certain level of local consistency is enough to guarantee that the network is globally consistent. A network is globally consistent if any solution for a subnetwork can always be extended to a solution for the entire network. Hence, if a network is globally consistent, a solution can be found in a backtrack-free manner.

In this paper, we present a relationship between the tightness or restrictiveness of the constraints, the arity of the constraints, and the level of local consistency sufficient to ensure global consistency. Specifically, in any constraint network where the constraints have arity r or less and the constraints have tightness of m or less, if the network is strongly $((m+1)(r-1)+1)$-consistent, then the network is globally consistent. Informally, a network is strongly k-consistent if any consistent instantiation of any $k-1$ or fewer variables can be extended consistently to any additional variable. Also informally, given an r-ary constraint and an instantiation of $r-1$ of the variables that participate in the constraint, the parameter m is an upper bound on the number of instantiations of the rth variable that satisfy the constraint.

We also present a new definition of local consistency called *relational m-consistency.* The virtue of this definition is that, firstly, it allows expressing the relationship between tightness and local consistency in a way that avoids an explicit reference to the arity of the constraints. Secondly, it is operational, thus generalizing the concept of the composition operation defined for binary constraints, and can be incorporated naturally in algorithms for enforcing desired levels of relational consistency. Thirdly, it unifies known operators such as resolution in theorem proving, joins in relational databases, and variable elimination for solving equations and inequalities. Finally, it allows identifying those formalisms for which consistency can be decided by enforcing pairwise consistency, like propositional databases and linear equalities and inequalities, from general databases requiring higher levels of local consistency.

The results we present are particularly useful in applications where a knowledge base will be queried over and over and we desire that queries be answered quickly. In such applications the preprocessing time to enforce local consistency is of less importance. What is of importance is knowing what level of local consistency will guarantee that queries can be answered quickly.

2 Background

We begin with some needed definitions and describe related work.

Definition 1 (binary constraint network; Montanari [1974]) *A* binary constraint network *consists of a set X of n variables $\{x_1, x_2, \ldots, x_n\}$, a domain D_i of possible values for each variable, and a set of binary constraints between variables. A* binary constraint *or relation, R_{ij}, between variables x_i and x_j, is any subset of the product of their domains (i.e., $R_{ij} \subseteq D_i \times D_j$). An* instantiation *of the variables in X is an n-tuple $(X_1, X_2, \ldots, X_n)$, representing an assignment of $X_i \in D_i$ to x_i. A* consistent instantiation *of a network is an instantiation of the variables such that the constraints between variables are satisfied. A consistent instantiation is also called a* solution.

Mackworth [1977] defines three properties of networks that characterize local consistency of networks: *node*, *arc*, and *path consistency*. Freuder [1978] generalizes this to k-consistency.

Definition 2 (k-consistency; Freuder [1978]) *A network is k-*consistent *if and only if given any instantiation of any $k-1$ variables satisfying all the direct relations among those variables, there exists an instantiation of any kth variable such that the k values taken together satisfy all the relations among the k variables. A network is* strongly k-consistent *if and only if it is j-consistent for all $j \leq k$.*

Node, arc, and path consistency correspond to strongly one-, two-, and three-consistent, respectively. A strongly n-consistent network is called *globally consistent.* Globally consistent networks have the property that any consistent instantiation of a subset of the variables can be extended to a consistent instantiation of all the variables without backtracking [Dechter, 1992b].

Following Montanari [1974], a binary relation R_{ij} between variables x_i and x_j is represented as a (0,1)-matrix with $|D_i|$ rows and $|D_j|$ columns by imposing an ordering on the domains of the variables. A zero entry at row a, column b means that the pair consisting of the ath element of D_i and the bth element of D_j is not permitted; a one entry means the pair is permitted. A concept central to this paper is the tightness of constraints.

Definition 3 (m-tight)
*A binary constraint is m-*tight *if every row and every column of the (0,1)-matrix that defines the constraint has at most m ones, where $0 \leq m \leq |D|-1$. Rows and columns with exactly $|D|$ ones are ignored in determining m. A binary constraint network is m-tight if all its binary constraints are m-tight.*

Example 1. We illustrate some of the definitions using a variant of n-queens proposed by Nadel [1989] called confused n-queens. The problem is to find all ways to place n-queens on an $n \times n$ chess board, one queen per column, so that each pair of queens *does* attack each other. One possible constraint network formulation of the problem is as follows: there is a variable for each column of the chess board, $x_1, \ldots, x_n$; the domains of the variables are the possible row positions, $D_i = \{1, \ldots, n\}$; and the binary constraints are that two queens should attack each other. The (0,1)-matrix representation of the constraints between two variables x_i and x_j is given by,

$$R_{ij,ab} = \begin{cases} 1 & \text{if } a = b \vee |a-b| = |i-j| \\ 0 & \text{otherwise,} \end{cases}$$

for $a, b = 1, \ldots, n$. For example, consider the constraint R_{12} between x_1 and x_2: $R_{12,34} = 1$, which states that putting a queen in column 1, row 3 and a queen in column 2, row 4 is allowed by the constraint since the queens attack each other.

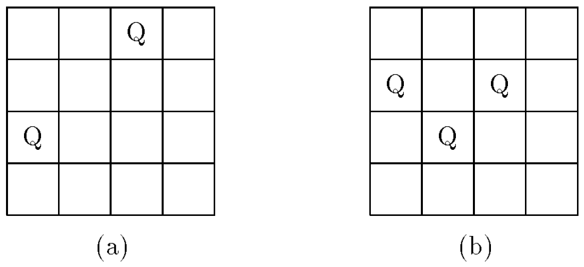


Figure 1: (a) not 3-consistent; (b) not 4-consistent

It can be seen that the networks for the confused n-queens problem are 2-consistent since, given that we have placed a single queen on the board, we can always place a second queen such that the queens attack each other. However, the networks are not 3-consistent. For example, for the confused 4-queens problem shown in Fig. 1a, there is no way to place a queen in the last column that is consistent with the previously placed queens. Similarly the networks are not 4-consistent (see Fig. 1b). Finally, every row and every column of the (0,1)-matrices that define the constraints has at most 3 ones. Hence, the networks are 3-tight.

2.1 Related work

Much work has been done on identifying relationships between properties of constraint networks and the level of local consistency sufficient to ensure global consistency. This work falls into two classes: identifying topological properties of the underlying graph of the network and identifying properties of the constraints. Here we review only the literature for constraint networks with finite domains.

For work that falls into the class of identifying topological properties, Freuder [1982; 1985] identifies a relationship between the *width* of a constraint graph and the level of local consistency needed to ensure a solution can be found without backtracking. As a special case, if the constraint graph is a tree, arc consistency is sufficient to ensure a solution can be found without backtracking. Dechter and Pearl [1988] provide an adaptive scheme where the level of local consistency is adjusted on a node-by-node basis. Dechter and Pearl [1989] generalize the results on trees to hyper-trees which are called acyclic databases in the database community [Beeri *et al.*, 1983].

For work that falls into the class of identifying properties of the constraints (the class into which the present work falls), Montanari [1974] shows that path consistency is sufficient to guarantee that a binary network is globally consistent if the relations are monotone. Van Beek and Dechter [1994] show that path consistency is sufficient if the relations are row convex. Dechter [1992b] identifies a relationship between the size of the domains of the variables, the arity of the constraints, and the level of local consistency sufficient to ensure the network is globally consistent. She proves the following result.

Theorem 1 (Dechter [1992b]) *Any $|D|$-valued r-ary constraint network that is strongly $(|D|(r-1)+1)$-consistent is globally consistent. In particular, any $|D|$-valued binary constraint network that is strongly $(|D|+1)$-consistent is globally consistent.*

For some networks, Dechter's theorem is tight in that the level of local consistency specified by the theorem is really required (graph coloring problems formulated as constraint networks are an example). For other networks, Dechter's theorem overestimates. Our results should be viewed as an improvement on Dechter's theorem. In particular, our main theorem, by taking into account the tightness of the constraints, always specifies a level of strong consistency that is less than or equal to the level of strong consistency required by Dechter's theorem.

3 Binary constraint networks

In this section we restrict our attention to binary constraint networks and present a relationship between the tightness of the constraints and the level of local consistency sufficient to ensure a network is globally consistent. The results are generalized to constraint networks with constraints of arbitrary arity in the next section.

The following lemma is needed in the proof of the main result for constraint networks with binary constraints and in a later proof of the result generalized to constraint networks with constraints of arbitrary arity. The lemma is really about the "tightness" of constraints and the sufficiency of a certain level of consistency. We state the lemma in more colloquial terms to make the proof more understandable.

Lemma 1 *Suppose there are fan clubs that like to meet and talk about famous people, and the following conditions.*

1. *There are n fan clubs and d famous people.*
2. *Each fan club meets and talks about at most m, $m < d$, famous people.*
3. *For every set of $m+1$ or fewer fan clubs, there exists at least one famous person that every club in the set talks about.*

Then, there must exist at least one famous person that every fan club talks about.

Proof. The proof is by contradiction and uses a proof technique discovered by Dechter for Theorem 1. Assume to the contrary that no such famous person exists. Then, for each famous person, f_i, there must exist at least one fan club that does not talk about f_i. Let c_i denote one of the fan clubs that does not talk about f_i. By construction, the set $c = \{c_1, c_2, \ldots, c_d\}$ is a set of fan clubs for which there does not exist a famous person that every club in the set talks about (every candidate f_i is ruled out since c_i does not talk about f_i). For every possible value of m, this leads to a contradiction.

Case 1 ($m = d - 1$): The contradiction is immediate as $c = \{c_1, c_2, \ldots, c_d\}$ is a set of fan clubs of size $m+1$ for which there does not exist a famous person that every club in the set talks about. This contradicts condition (3).

Case 2 ($m = d - 2$): The nominal size of the set $c = \{c_1, c_2, \ldots, c_d\}$ is $m+2$. We claim, however, that there is a repetition in c and that the true size of the set is $m+1$. Assume to the contrary that $c_i \neq c_j$ for $i \neq j$. Recall c_i is a club that does not talk about f_i, $i = 1, \ldots, d$ and consider $\{c_1, c_2, \ldots, c_{d-1}\}$. This is a set of $m+1$ fan clubs so by condition (3) there must exist an f_i that every club in the set talks about. The only possibility is f_d. Now consider $\{c_1, \ldots, c_{d-2}, c_d\}$. Again, this is a set of $m+1$ fan clubs so there must exist an f_i that every club in the set talks about. This time the only possibility is f_{d-1}. Continuing in this manner, we can show that fan club c_1 must talk about exactly $m+1$ famous people. This contradicts condition (2). Therefore, it must be the case that $c_i = c_j$ for some $i \neq j$. Thus, the set c is of size $m+1$ and this contradicts condition (3).

Case 3 ($m = d-3$), ..., **Case d-1** ($m = 1$): The remaining cases are similar. In each case we argue that (i) there are repetitions in the set $c = \{c_1, c_2, \ldots, c_d\}$, (ii) the true size of the set c is $m+1$, and (iii) a contradiction is derived by appealing to condition (3).

Thus, there exists at least one famous person that every fan club talks about. $\Box$

We now state the theorem for binary constraint networks.

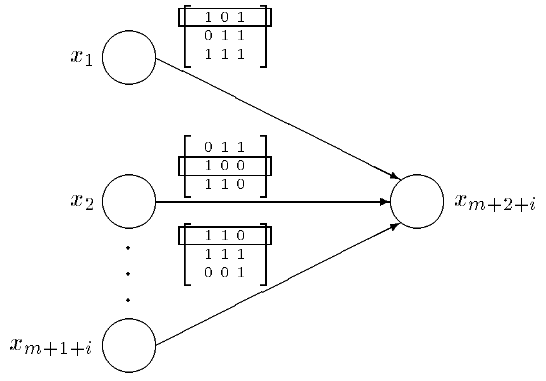
Theorem 2 *If a binary constraint network, R, is m-tight, and if the network is strongly $(m+2)$-consistent, then the network is globally consistent.*

Proof. We show that any network with $\leq m$ ones in every row that is strongly $(m+2)$-consistent is $(m+2+i)$-consistent for any $i \geq 1$. Suppose that variables $x_1, \ldots, x_{m+1+i}$ can be consistently instantiated with values $X_1, \ldots, X_{m+1+i}$. To show that the network is $(m+2+i)$-consistent, we must show that there exists at least one instantiation, X_{m+2+i}, of variable x_{m+2+i} such that

$$(X_j, X_{m+2+i}) \in R_{j,m+2+i} \qquad j = 1, \ldots, m+1+i$$

is satisfied. Let v_j be the (0,1)-vector given by row X_j of the (0,1)-matrix $R_{j,m+2+i}$, $j = 1, \ldots, m+1+i$ (see Figure 2 for an illustration; the v_j are shown boxed). The one entries in the v_j are the allowed instantiations of x_{m+2+i}, given the instantiations $X_1, \ldots, X_{m+1+i}$. That there exists a consistent instantiation of x_{m+2+i} follows from Lemma 1 where (i) $X_1, \ldots, X_{m+1+i}$ are the fan clubs, (ii) $1, \ldots, d$, the domain elements of x_{m+2+i}, are the famous people, (iii) the one entries in the v_j's are the famous people that fan club X_j talks about, and (iv) condition (3) of Lemma 1 follows from the assumption of strong $(m+2)$ consistency. Therefore, from Lemma 1 it follows that there exists at least one instantiation of x_{m+2+i} that satisfies all the constraints simultaneously. Hence, the network is $(m+2+i)$-consistent. $\Box$

Theorem 2 always specifies a level of strong consistency



x_1
$\begin{bmatrix} \boxed{1\ 0\ 1} \\ 0\ 1\ 1 \\ 1\ 1\ 1 \end{bmatrix}$
x_2
$\begin{bmatrix} 0\ 1\ 1 \\ \boxed{1\ 0\ 0} \\ 1\ 1\ 0 \end{bmatrix}$
$\begin{bmatrix} \boxed{1\ 1\ 0} \\ 1\ 1\ 1 \\ 0\ 0\ 1 \end{bmatrix}$
x_{m+1+i}
x_{m+2+i}

Figure 2: Instantiating x_{m+2+i}

that is less than or equal to the level of strong consistency required by Dechter's theorem (Theorem 1). The level of required consistency is equal only when $m = |D| - 1$ and is less when $m < |D| - 1$. As well, the theorem can sometimes be usefully applied if $|D| \geq n-1$, whereas Dechter's theorem cannot.

As the following example illustrates, both r, the arity of the constraints, and m can change if the level of consistency required by the theorem is not present and must be enforced. The parameter r can only increase; m can decrease, as shown below, but also increase. The parameter m will increase if all of the following hold: (i) there previously was no constraint between a set of variables, (ii) enforcing a certain level of consistency results in a new constraint being recorded between those variables and, (iii) the new constraint has a larger m value than the previous constraints.

Example 2. Consider again the confused n-queens problem introduced in Example 1. The problem is worth considering, as Nadel [1989] uses confused n-queens in an empirical comparison of backtracking algorithms for solving constraint networks. Thus it is important to analyze the difficulty of the problems to set the empirical results in context. As well, the problem is interesting in that it provides an example where Theorem 2 can be applied but Dechter's theorem can not (since $|D| \geq n-1$). Independently of n, each row of the constraints has ≤ 3 ones. Hence, the networks are 3-tight and the theorem guarantees that if the network for the confused n-queens problem is strongly 5-consistent, the network is globally consistent.

First, suppose that n is even and we attempt to either verify or achieve this level of strong consistency by applying successively stronger local consistency algorithms. Kondrak [1993] has shown that the following analysis holds for all n, n even.

1. Applying an arc consistency algorithm results in no changes as the network is already arc consistent.

2. Applying a path consistency algorithm does tighten the constraints between the variables. Once the network is made path consistent, each row has ≤ 2 ones. Now the theorem guarantees that if the constraint network is strongly 4-consistent, the network is globally consistent.
3. Applying a 4-consistency algorithm results in no changes as the network is already 4-consistent. Thus, the network is strongly 4-consistent and therefore also globally consistent.

Second, suppose that n is odd. This time, after applying path consistency, the networks are still 3-tight and it can be verified that the networks are not 4-consistent. Enforcing 4-consistency would require non-binary constraints, hence Theorem 2 no longer applies. We take this example up again in the next section where the results are generalized to non-binary constraints. There we show that recording 3-ary constraints is sufficient.

Recall that Nadel [1989] uses confused n-queens problems to empirically compare backtracking algorithms for finding all solutions to constraint networks. Nadel states that these problems provide a "non-trivial test-bed" [1989, p.190]. We believe the above analysis indicates that these problems are quite easy and that any empirical results on these problems should be interpreted in this light. Easy problems potentially make even naive algorithms for solving constraint networks look promising. To avoid this potential pitfall, backtracking algorithms should be tested on problems that range from easy to hard. In general, hard problems are those that require a high level of local consistency to ensure global consistency. Note also that these problems are trivially satisfiable.

Example 3. The graph k-colorability problem can be viewed as a problem on constraint networks: there is a variable for each node in the graph; the domains of the variables are the possible colors, $D = \{1, \ldots, k\}$; and the binary constraints are that two adjacent nodes must be assigned different colors. Graph k-colorability provides examples of networks where both Theorems 1 and 2 give the same bound on the sufficient level of local consistency (since $|D| = k$ and $m = |D| - 1$). Further, as Dechter [1992b] shows, the bound is tight. For example, consider coloring a complete graph on five nodes with four colors. The network is 3-tight and strongly 4-consistent, but not strongly 5-consistent and not globally consistent. Hence, when $m = |D| - 1$, the level of local consistency specified by Theorem 2 is as strong as possible and cannot be lowered.

We can also construct examples to show that Theorem 2 is as strong as possible for all $m < |D| - 1$. This can be done by "embedding" graph coloring constraints into the constraints for the new network. For example, consider the network where the domains are $D = \{1, \ldots, 5\}$ and the constraints between all variables is given by,

$$R_{ij} = \begin{bmatrix} 1\,0\,0\,0\,1 \\ 0\,0\,1\,1\,0 \\ 0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0 \\ 1\,0\,0\,0\,1 \end{bmatrix}.$$

The inner 3×3 matrix is the 3-coloring constraint. The network is 2-tight and strongly 3-consistent, but not strongly 4-consistent and not globally consistent.

4 R-ary constraint networks

In this section we generalize the results of the previous section to networks with constraints of arbitrary arity. We will define m-tightness of r-ary relations, namely relations having r variables. We use the following notations and definitions.

Definition 4 (Relations)
Given a set of variables $X = \{x_1, \ldots, x_n\}$, each associated with a domain of discrete values $D_1, \ldots, D_n$, respectively, a relation *(or, alternatively, a* constraint*) ρ over X is any subset*

$$\rho \subseteq D_1 \times D_2 \times \cdots \times D_n.$$

Given a relation ρ on a set X of variables and a subset $Y \subseteq X$, we denote by $Y = y$ or by y an instantiation of the variables in Y, called a subtuple *and by $\sigma_{Y=y}(\rho)$ the selection of those tuples in ρ that agree with $Y = y$. We denote by $\Pi_Y(\rho)$ the projection of relation ρ on the subset Y. Namely, a tuple over Y appears in $\Pi_Y(\rho)$ if and only if it can be extended to a full tuple in ρ. If Y is not a subset of ρ's variables the projection is over the subset of variables that appear both in Y and in X. The operator $\bowtie$ is the join operator in relational databases.*

Definition 5 (Constraint networks)
A constraint network *R over a set X of variables $\{x_1, x_2, \ldots, x_n\}$, is a set of relations $R_1, \ldots, R_t$, each defined on a subset of variables $S_1, \ldots, S_t$ respectively. A relation in R specified over $Y \subseteq X$ is also denoted R_Y. The set of subsets $S = \{S_1, \ldots, S_t\}$ on which constraints are specified is called the* scheme *of R. The network R represents its set of all consistent solutions over X, denoted $\rho(R)$ or $\rho(X)$, namely,*

$$\rho(R) = \{x = (X_1, \ldots, X_n) \mid \forall S_i \in S, \Pi_{S_i}(x) \in R_i\}.$$

For non-binary networks the notion of *consistency of a subtuple* can be defined in several ways. We will use the following definition. A subtuple over Y is consistent if it satisfies all the constraints defined over Y including all R's constraints obtained by projection over Y.

Definition 6 (Consistency of a subtuple)
A subtuple $Y = y$ is consistent relative to R iff, for all $S_i \in S$,

$$\Pi_{S_i \cap Y}(y) \in \Pi_{S_i \cap Y}(R_i).$$

$\rho(Y)$ is the set of all consistent instantiations of the variables in Y. One can view $\rho(Y)$ as the set of all solutions of the subnetwork defined by Y.

Informally, an r-ary relation is m-tight if every tuple of $r-1$ values can be extended in at most m ways.

Definition 7 *An r-ary relation is m-tight if and only if all of its binary projections (projections on pairs of variables) are m-tight.*

Example 4. We illustrate some of the definitions using the following network, R, over the set of variables, $\{x_1, x_2, x_3, x_4\}$. The relations are given by,

$$R_{S_1} = \{(1,4,2), (2,4,1), (3,1,4), (4,1,3)\},$$
$$R_{S_2} = \{(1,4,2), (2,1,3), (2,1,4), (2,3,1), (3,2,4), (3,4,1), (3,4,2), (4,1,3)\},$$

where $S_1 = \{x_1, x_2, x_3\}$ and $S_2 = \{x_1, x_3, x_4\}$. The set of all solutions of the network is given by,

$$\rho(R) = \{(2,4,1,3), (2,4,1,4), (3,1,4,1), (3,1,4,2)\}.$$

Let $Y = \{x_1, x_3\}$ be a subset of the variables and let the subtuple $y = (2, 1)$ be an instantiation of the variables in Y. Then, $\sigma_{Y=y}(R_{S_2}) = \{(2,1,3), (2,1,4)\}$ and, $\Pi_Y(R_{S_1}) = \{(1,2), (2,1), (3,4), (4,3)\}$. It can be verified that the subtuple $y = (2, 1)$ is consistent relative to R and that the subtuple $y = (1, 2)$ is not consistent relative to R (since $\Pi_{S_2 \cap Y}(y) \notin \Pi_{S_2 \cap Y}(R_{S_2})$). Finally, the network is 3-tight since projecting the relation R_{S_2} onto $\{x_1, x_4\}$ results in a binary relation that is 3-tight, and this is the maximum of all the binary projections.

We now state the general theorem.

Theorem 3 *If an r-ary network, R, is m-tight, and if the network is strongly $((m+1)(r-1)+1)$-consistent, then the network is globally consistent.*

Proof. Let $k = (m+1)(r-1)+1$. We show that any network with relations that are m-tight that is strongly k-consistent is $(k+i)$-consistent for any $i \geq 1$.

Let $X' = (X_1, X_2, \ldots, X_{k+i-1})$ be a consistent instantiation of $k+i-1$ variables[1] and let x_{k+i} be an arbitrary new variable. We will show that there exists an instantiation X_{k+i} of x_{k+i} such that the extended tuple $(X_1, X_2, \ldots, X_{k+i-1}, X_{k+i})$ is consistent. This means that any relation $R_Y \in R$ involving variable x_{k+i}, and a non-empty subset of variables from $\{x_1, \ldots, x_{k+i-1}\}$ should be satisfied. Let X'_Y be the partial tuple of X' that is restricted to the set Y over which R_Y is defined. We call this tuple a *constraint-tuple*. Since all the constraints and their projections are m-tight, constraint R_Y will allow X'_Y to be extended by at most m values of x_{k+i}. Each such constraint-tuple, X'_Y can be regarded as a fan club, with its allowed values in x_{k+i} relative to R_Y as the discussed famous people. Therefore, condition (2) of Lemma 1 is satisfied. Also, condition (3) of Lemma 1 is satisfied, since the length of each constraint-tuple is $r-1$ or less, the requirement of strong $(m+1)(r-1)+1$-consistency, ensures that any set of up to $(m+1)$ constraint-tuples (overlapping or not), has a consistent extension in x_{k+i}. Therefore, from Lemma 1 it follows that there is a common value of x_{k+i} that satisfies all the constraints simultaneously. □

[1]Note that according to the definition of consistency this means that X' satisfies all the constraints defined on its own subset of variables as well as those obtained by projection.

Example 5. Consider again the confused n-queens problem discussed in Example 2. There we saw that, after enforcing path consistency, the networks are 3-tight, for n odd. Enforcing 4-consistency requires 3-ary constraints. Adding the necessary 3-ary constraints does not change the value of m; the networks are still 3-tight. Hence, by Theorem 3, if the networks are strongly 9-consistent, the networks are globally consistent. Kondrak [1993] has shown that recording 3-ary constraints is sufficient to guarantee the networks are strongly 9-consistent for all n, n odd. Hence, independently of n, the networks are globally consistent once strong 4-consistency is enforced.

Example 6. Constraint networks have proven fruitful in representing and reasoning about temporal information. We use an example from Allen's [1983] framework for reasoning about temporal relations between intervals or events to illustrate the application of Theorem 3. Allen identifies thirteen *basic* relations that can hold between two intervals. In order to represent indefinite information, the relation between two intervals is allowed to be a disjunction of the basic relations. For example, the relation {b,bi} between events A and D in Figure 3 represents the disjunction, (A before D) $\vee$ (A after D). Allen provides a transitivity table for propagating the temporal information.

Allen's framework can be formulated as a constraint network with finite domains as follows: there is a variable for each pair of intervals, the domains of the variables are the possible basic relations, and there are ternary constraints defined by the transitivity table. For example, consider the temporal information given by,

A	{oi,m}	B	A	{b,o}	C_i	A	{b,bi}	D_i
B	{b,d}	C_i	B	{bi,o}	D_i	D_i	{b,oi}	C_i

for $i = 1, \ldots, (n-2)/2$. Formulating this temporal information as a constraint network with finite domains, we can show that enforcing strong 4-consistency is sufficient to ensure the network is globally consistent, for all $n \geq 4$. Below we show the analysis for the simple case of $n = 4$. The general case is similar, just notationally more complicated. Figure 3 shows the six

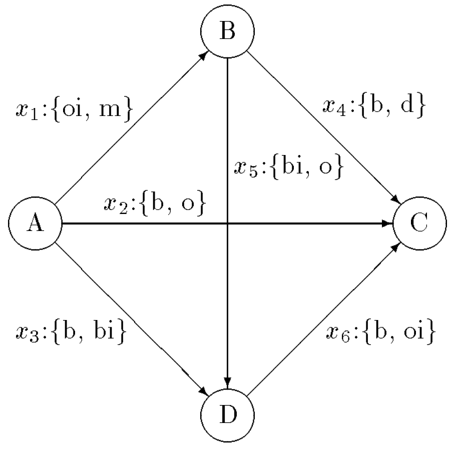


Figure 3: Example temporal network

variables and their associated domains for our example. The ternary constraints for our example are given by,

R_{124} = {(oi,b,b), (oi,o,b), (m,b,b), (m,o,d)},
R_{135} = {(oi,bi,bi), (m,bi,bi), (m,b,o)},
R_{236} = {(b,b,b), (b,b,oi), (b,bi,b), (o,b,oi), (o,bi,b)},
R_{456} = {(b,bi,b), (b,o,b), (d,bi,b), (d,o,oi)}.

It can be shown that the network is 1-tight. Therefore, by Theorem 3, if the network is strongly 5-consistent, then the network is globally consistent. Suppose that we attempt to either verify or achieve this level of strong consistency. The network is strongly 3-consistent, but not 4-consistent. For example, (b,b,oi) is a consistent instantiation of (x_2, x_3, x_6), since it satisfies the constraint R_{236} as well as all the constraints obtained by projection. However, there is no way to extend the instantiation to x_4: (i) $x_4 \leftarrow$ b is inconsistent by the constraint R_{46} obtained by projecting R_{456} on $\{x_4, x_6\}$, and (ii) $x_4 \leftarrow$ d is inconsistent by the constraint R_{24} obtained by projecting R_{124} on $\{x_2, x_4\}$. The modified constraint R'_{236} is given by,

R'_{236} = {(b,b,b), (b,bi,b), (o,b,oi), (o,bi,b)}.

As well, some 3-ary constraints between previously unconstrained triples of variables need to be introduced. For example, (oi,o,oi) is a consistent instantiation of (x_1, x_2, x_6), since it satisfies all the constraints obtained by projection. However, there is no way to extend the instantiation to x_3: (i) $x_3 \leftarrow$ b is inconsistent by the constraint R_{13} obtained by projecting R_{135} on $\{x_1, x_3\}$, and (ii) $x_3 \leftarrow$ bi is inconsistent by the constraint R'_{236}. Once the following 3-ary relations are added, the network is strongly 4-consistent:

R_{126} = {(oi,b,b), (oi,o,b), (m,b,b), (m,o,b), (m,o,oi)},
R_{234} = {(b,b,b), (b,bi,b), (o,b,d), (o,bi,b), (o,bi,d)},
R_{256} = {(b,bi,b), (b,o,b), (o,bi,b), (o,o,oi)},
R_{346} = {(b,b,b), (b,d,oi), (bi,b,b), (bi,d,b)}.

It can now be verified that the network is also strongly 5-consistent. Therefore, by Theorem 3, the network is globally consistent. The network is also minimal. A network of r-ary relations is minimal if each tuple in the relations participates in at least one consistent instantiation of the network. These two properties, global consistency and minimality, ensure that we can efficiently answer some important classes of temporal queries.

4.1 Relational local consistency

In [van Beek and Dechter, 1994] we extended the notion of path-consistency to non-binary relations, and used it to specify an alternative condition under which row-convex non-binary networks of relations are globally consistent. This definition, since it considers the relations rather than the variables as the primitive entities, does not mention the arity of the constraint explicitly. We now extend this definition even further and show how it can be used to alternatively describe Theorem 3.

Definition 8 (Relational m-consistency) *Let R be a network of relations over a set of variables X, let $R_{S_1}, \ldots, R_{S_{m-1}}$ be $m-1$, $m \geq 3$, relations in R, where $S_i \subseteq X$. We say that $R_{S_1}, \ldots, R_{S_{m-1}}$ are relational m-consistent relative to variable x iff any consistent instantiation of the variables in A, where $A = \bigcup_{i=1}^{m-1} S_i - \{x\}$, has an extension to x that satisfies $R_{S_1}, \ldots, R_{S_{m-1}}$ simultaneously. Namely, if and only if*

$$\rho(A) \subseteq \Pi_A(\bowtie_{i=1}^{m-1} R_{S_i}).$$

(Recall that $\rho(A)$ is the set of all consistent instantiations of the variables in A). A set of relations $R_{S_1}, \ldots, R_{S_{m-1}}$ are relational m-consistent iff they are relational m-consistent relative to each variable in $\bigcap_{i=1}^{m-1} S_i$. A network of relations is said to be relational m-consistent iff every set of $m-1$ relations is relational m-consistent. Relational 3-consistency is also called relational path-consistency. A network is strongly relational m-consistent if it is relational i-consistent for every $i \leq m$.

Note that we do not need to define relational 2-consistency since our definition of consistency of a subtuple, which takes into account all the networks' projections, guarantees that any notion of relational 2-consistency is redundant.

Example 7. Consider the following network of relations. The domains of the variables are all $D = \{0, 1, 2\}$ and the relations are given by,

$$(1)\ R_{fxyz} = \{0000, 1000, 0100, 0010, 0001\},$$

$$(2)\ R_{fzs} = \{011, 122, 021\}.$$

The constraints are not relational path-consistent. For example, the instantiation $f = 0, x = 1, y = 0$ satisfies all the constraints, (namely all the projections of

(1) and (2) on $\{f, x, y\}$ and $\{f\}$ respectively), but it cannot be consistently extended to a legal value of z. If we add the constraint (3)$R_{fxy} = \{000\}$, the first two constraints will become relational path-consistent relative to z since constraint (3) will disallow the partial assignments $f = 0, x = 1, y = 0$. Constraints (1) and (2) are relational path-consistent relative to f since any consistent instantiation of x, y, z will have to satisfy the two constraints $R_{xyz} = \{000, 100, 010, 001\}$ and $R_z = \{1, 2\}$ obtained by projecting constraints (1) and (2) over x, y, z, respectively. Remember that consistency of a subtuple needs to obey all the projected constraints. Once these constraints are obeyed there is an extension to $f = 0$ that satisfies (1) and (2) simultaneously.

We now show that strong relational $(m + 2)$-consistency is sufficient to ensure globally consistency when the relations are m-tight.

Theorem 4 *Let R be a network of relations that is strongly relational $(m + 2)$-consistent. If the relations are m-tight, then the network is globally consistent.*

Proof. Assume that the network is relational $(m+2)$-consistent. Let $X' = (X_1, X_2, \ldots, X_{i-1})$ be a consistent instantiation of $i-1$ variables, $i > m+2$. We will show that for any x_i, there exists an instantiation X_i of x_i such that the extended tuple $(X_1, X_2, \ldots, X_{i-1}, X_i)$ is consistent. This means that any relevant relation $R_Y \in R$ or any of its projections, that are defined over x_i should be satisfied by such an extension. Since all constraints and all their projections are m-tight, all the values of x_i that together with X'_Y are allowed by R_Y do not exceed m. Also, strong relational $(m + 2)$-consistency implies that any subset of $m + 1$ or fewer constraints can be consistently extended by x_i. Consequently, due to Lemma 1 there is a value X_i such that the tuple $(X_1, X_2, \ldots, X_{i-1}, X_i)$ satisfies all the constraints simultaneously. □

When all the constraints are binary, relational m-consistency is identical (up to minor preprocessing) to variable-based m-consistency. Otherwise the conditions are different. In general, the definition of relational m-consistency is similar but not identical to that of m-consistency over the dual representation of the problem in which the constraints are the variables, their allowed tuples are their respective domains and two such constraint-variables are constrained if they have variables in common. The virtue in this new explicit definition (relative to the one based on the dual graph) is that it is simpler to work with, it uses known notations from relational databases, and it immediately translates to consistency enforcing algorithms.

Relational m-consistency can be enforced on a network that does not possess this level of consistency. Below we present algorithm RC_m, a brute-force algorithm for enforcing strong relational m-consistency on a network R. The algorithm seems to enforce relational m-consistency only (joining every set of $m - 1$ relations), however due to our convention of testing all projections when verifying consistency, strong m-consistency results as well.

```
RC_m(R)
1.  repeat
2.      $Q \leftarrow R$
3.      for every $m-1$ relations $R_{S_1}, \ldots, R_{S_{m-1}} \in Q$
        and every $x \in \bigcap_{i=1}^{m-1} S_i$
4.          do  $A \leftarrow \bigcup_{i=1}^{m-1} S_i - \{x\}$
5.              $R_A \leftarrow R_A \cap \Pi_A(\bowtie_{i=1}^{m-1} R_{S_i})$
6.  until $Q = R$
```

Note that R_Y stands for the current unique constraint specified over a subset of variables Y. If no constraint exists, then R_Y is the universal relation over Y. The algorithm takes any $m - 1$ relations that may or may not be relational m-consistent and enforces relational m-consistency by tightening the relation among the appropriate subsets of variables. We call the operation in Step 5 of the algorithm *extended m-composition*, since it generalizes the composition operation defined on binary relations. Algorithm RC_m computes the closure of R with respect to extended m-composition. We can conclude that:

Theorem 5 *For any network, R, whose closure under extended i-composition, for $i = 3, \ldots, m$, is an $(m-2)$-tight network, $m \geq 3$, algorithm RC_m computes an equivalent globally consistent network.*

Proof. Follows immediately from Theorem 4 and from the fact that RC_m generates a strong relational m-consistent network. □.

While enforcing *variable-based* m-consistency can be done in polynomial time, it is unlikely that relational m-consistency can be achieved tractably, since, as we will shortly see, even for $m = 3$ it solves the NP-complete problem of propositional satisfiability. A more direct argument suggesting an increase in time and space complexity is the fact that the algorithm may need to record relations of arbitrary arity and also that the constraints' tightness may increase.

Example 8. Bi-valued relations are 1-tight and closed under extended 3-composition. Thus, by Theorem 5, bi-valued networks can be solved by algorithm RC_3. In particular, the satisfiability of propositional *CNFs* can be decided by RC_3. Here the extended composition operation (Step 5 of algorithm RC_m) takes the form of pair-wise resolution [Dechter and Rish, 1994]. A different derivation of the same result is already given by [Dechter, 1992b; van Beek and Dechter, 1994].

As with variable-based local-consistency, we can improve the efficiency of enforcing relational consistency by enforcing it only along a certain direction. Below we present algorithm *Directional Relational m-Consistency (DRC_m)* that enforces strong relational m-consistency on a network R, relative to a given ordering, d, of the variables $x_1, x_2, \ldots, x_n$. We denote as $DRC_m(R, d)$, a network that is strongly relational m-consistent relative to an ordering d.

```
DRC_m(R, d)
1. Initialize: generate an ordered partition of the constraints, $bucket_1, ..., bucket_n$, where $bucket_i$ contains
   all the constraints whose highest variable is $x_i$.
2. for $i \leftarrow n$ downto 1
3.    do for every set of $m-1$ relations $R_{S_1}, \ldots, R_{S_{m-1}}$ in $bucket_i$ (if $bucket_i$ contains fewer
          than $m-1$ relations, then take all the relations in the bucket).
4.       do $A \leftarrow \bigcup_{i=1}^{m-1} S_i - \{x_i\}$
5.          $R_A \leftarrow R_A \cap \Pi_A(\bowtie_{i=1}^{m-1} R_{S_i})$
6.          Add $R_A$ to its appropriate bucket.
```

While the algorithm is incomplete for deciding consistency in general, it is complete for $(m-2)$-tight relations that are closed under extended m-composition. In fact, it is sufficient to require directional $(m-2)$-tightness relative to the ordering used. Namely, requiring that if x_i appears before x_j in the ordering then any value of x_i will be $(m-2)$-tight relative to x_j but not vice-versa. For example, functional relations are always 1-tight from input to outputs but not for any ordering.

Definition 9 (directionally m-tight)
A binary constraint, R_{ij}, is directionally m-tight *with respect to an ordering of the variables, $d = (x_1, \ldots, x_n)$, if x_i appears before x_j in the ordering and every row of the (0,1)-matrix that defines the constraint has at most m ones. An r-ary relation is directionally m-tight with respect to an ordering of the variables if and only if all of its binary projections are directionally m-tight with respect to the ordering.*

The following theorems will be stated without proofs. Their correctness can be verified using similar theorems on directional consistency algorithms reported earlier [Dechter and Pearl, 1989].

Theorem 6 (Completeness)
If a network $DRC_m(R, d)$ is directionally $(m-2)$-tight relative to d, then $DRC_m(R, d)$ is backtrack-free along d.

Like similar algorithms for imposing directional consistency, DRC_m's worst-case complexity can be bounded as a function of the topological structure of the problem via parameters like the *induced width* of the graph [Dechter and Pearl, 1988].

A network of constraints R can be associated with a constraint graph, where each node is a variable and two variables that appear in one constraint are connected. A general graph can be embedded in a *clique-tree* namely, in a graph whose cliques form a tree-structure. The induced width, $W*$, of such an embedding is its maximal clique size and the induced width $W*$ of an arbitrary graph is the minimum induced width over all its tree-embeddings. For more details see [Dechter and Pearl, 1989]. The complexity of DRC_m can be bounded as a function of the $W*$ of its constraint graph.

Theorem 7 (Complexity) *Given a network of relations R, the complexity of algorithm DRC_m along ordering d is $O(exp(mW^*(d)))$ where $W^*(d)$ is the induced width of the constraint graph of R along d.*

Example 9. Crossword puzzles have been used in experimentally evaluating backtracking algorithms for solving constraint networks [Ginsberg *et al.*, 1990]. We use an example puzzle (taken from [Dechter, 1992a]) to illustrate algorithm DRC_m (see Figure 4).

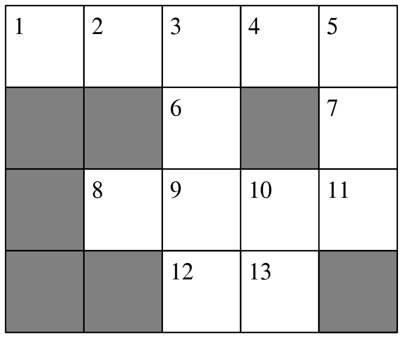


Figure 4: A crossword puzzle

We can formulate this problem as a constraint problem as follows, each possible slot holding a character will be a variable, and the possible words are relations over the variables. Therefore, we have $x_1, \ldots, x_{13}$ variables as marked in the figure. Their domains are the alphabet letters and the constraints are the following relations:

$$\begin{aligned}
R_{1,2,3,4,5} &= \{(H,O,S,E,S), (L,A,S,E,R), (S,H,E,E,T), \\
&\quad (S,N,A,I,L), (S,T,E,E,R)\} \\
R_{3,6,9,12} &= \{(H,I,K,E), (A,R,O,N), (K,E,E,T), \\
&\quad (E,A,R,N), (S,A,M,E)\} \\
R_{8,9,10,11} &= R_{3,6,9,12} \\
R_{5,7,11} &= \{(R,U,N), (S,U,N), (L,E,T), (Y,E,S), \\
&\quad (E,A,T), (T,E,N)\} \\
R_{10,13} &= \{(N,O), (B,E), (U,S), (I,T)\} \\
R_{12,13} &= R_{10,13}
\end{aligned}$$

We see that constraints $R_{10,13}$ and $R_{12,13}$ are 1-tight, however all the rest have higher tightness. For example, the tightness of $R_{5,7,11}$ is 3 due to words like RUN, SUN, and TEN. Constraint $R_{1,2,3,4,5}$ is also 3-tight since its binary projection on $\{x_1, x_5\}$ contains the three pairs {(S,L), (S,T), (S,R)}. For the ordering $d = x_5, x_4, ..., x_1$, however, the constraint is only 2-tight. The tightness of all constraints does not go beyond 3. According to Theorem 6, enforcing relational 5-consistency, if not increasing the tightness, will generate a globally consistent network relative to the ordering used.

Applying DRC_5 to this problem using the ordering $d = x_{13}, x_{12}, x_{11}, x_{10}, x_9, x_5, x_3$ (we disregard the rest of the letters since they appear in just one word), gives the following: Initially the bucket for x_3 contains two relations $R_{3,9,12}$ and $R_{3,5}$ (resulting from projecting away x_6 from $R_{3,6,9,12}$ and x_1, x_2, x_4 from $R_{1,2,3,4,5}$, respectively). Processing variable x_3 adds the relation $R_{5,9,12}$ to the bucket of variable x_5 that is processed next. The relation is:

$$\begin{aligned} R_{5,9,12} &= \Pi_{5,9,12}(R_{3,9,12} \bowtie R_{3,5}) \\ &= \{(\text{S,M,E}), (\text{R,M,E}), (\text{T,R,N}), \\ &\quad (\text{R,R,N}), (\text{L,O,N})\}. \end{aligned}$$

Next, processing of x_5 adds the relation $R_{9,11,12}$ to the bucket of variable x_9. The relation is:

$$\begin{aligned} R_{9,11,12} &= \Pi_{9,11,12}(R_{5,9,12} \bowtie R_{5,11}) \\ &= \{(\text{M,N,E}), (\text{R,N,R}), (\text{O,T,N}), (\text{R,N,N})\}. \end{aligned}$$

Next, processing x_9 adds the relation $R_{10,11,12}$ to the bucket of variable x_{10}. The relation is:

$$\begin{aligned} R_{10,11,12} &= \Pi_{10,11,12}(R_{9,10,11} \bowtie R_{9,11,12}) \\ &= \{(\text{O,N,R})\}. \end{aligned}$$

Next, processing x_{10} adds $R_{11,12,13}$ to the bucket of variable x_{11}. The relation is:

$$\begin{aligned} R_{11,12,13} &= \Pi_{11,12,13}(R_{10,11,12} \bowtie R_{10,13}) \\ &= \{\ \}. \end{aligned}$$

Namely, resulting in an empty relation. At this point the algorithm stops and determines that the problem is inconsistent.

It turns out, however, that cross-word puzzles have a special property that makes them solvable by relational 3-consistency only.

Lemma 2 *When processing a crossword problem by DRC_m for any m, the resulting buckets contain at most two constraints.*

Proof: Let us annotate each variable in a constraint by a + if it appears in a horizontal word and by a − if it appears in a vertical word. Clearly, in the initial specification each variable appears in at most two constraints and each annotated variable appears in just one constraint (with that annotation). We show that this property is maintained throughout the algorithm's performance. The argument can be proved by induction on the processed buckets. Assume that after processing buckets $x_n, ..., x_i$ *all* the constraints appearing in the union of all $bucket_{i-1}$ to $bucket_1$ satisfy that each annotated variable appears in at most one constraint. When processing $bucket_{i-1}$, since it contains only two constraints (otherwise it will contain multiple annotations of variable x_{i-1}), it generates a single new constraint. Assume that the constraint is added to the bucket of x_j.

Clearly, if x_j is annotated positively in the added constraint, $bucket_j$ cannot contain already a constraint with a positive annotation of x_j. Otherwise, it means that before processing bucket $i-1$, there were two constraints with positive annotation of x_j, one in the bucket of x_{i-1} and one in the bucket of x_j, which contradicts the induction hypothesis. Therefore, the rest of the buckets still obey the claimed property. □

Consequently, applying DRC_3 to a cross-word puzzle along any ordering enforces global consistency along that ordering.

Theorem 8 *Given a cross-word puzzle of size n, and for any ordering d, algorithms DRC_3 enforces directional global-consistency along d.*

Note, that it does not mean that cross-word puzzles are tractable. The size of the constraints in the bucket may be exponential. Nevertheless, if the size of the constraints is bounded somehow—by the width, for example—the problem becomes tractable.

5 Conclusions

In this paper, we have identified a sufficient condition based on the tightness of the constraints, the arity of the constraints, and the level of local consistency, that guarantees that a solution can be found in a backtrack-free manner. The results will be useful in applications where a knowledge base will be queried over and over and the preprocessing costs can be amortized over many queries. As well, we believe our results may have significant explanatory value. In recent computational experiments we discovered that the parameter m, which measures the tightness of the constraints, is a good predictor of the amount of time needed by backtracking algorithms to solve particular constraint networks. A goal in our work is to discover parameters of constraint networks that will allow us to predict how a backtracking algorithm will perform on a given problem.

Acknowledgements

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, by the NSF under grant IRI-9157636, by the Air Force Office of Scientific Research under grant AFOSR 900136, by Toshiba of America, and by a Xerox grant.

References

[Allen, 1983] J. F. Allen. Maintaining knowledge about temporal intervals. *Comm. ACM*, 26:832–843, 1983.

[Beeri *et al.*, 1983] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis. On the desirability of acyclic database schemes. *J. ACM*, 30:479–513, 1983.

[Dechter and Meiri, 1989] R. Dechter and I. Meiri. Experimental evaluation of preprocessing techniques in constraint satisfaction problems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 271–277, Detroit, Mich., 1989.

[Dechter and Pearl, 1988] R. Dechter and J. Pearl. Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence*, 34:1–38, 1988.

[Dechter and Pearl, 1989] R. Dechter and J. Pearl. Tree clustering for constraint networks. *Artificial Intelligence*, 38:353–366, 1989.

[Dechter and Rish, 1994] R. Dechter and I. Rish. Directional resolution: The Davis-Putnam procedure, revisited. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning*, Bonn, Germany, 1994.

[Dechter *et al.*, 1991] R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *Artificial Intelligence*, 49:61–95, 1991.

[Dechter, 1992a] R. Dechter. Constraint networks. In S. C. Shapiro, editor, *Encyclopedia of Artificial Intelligence, Second Edition*, pages 276–285. John Wiley & Sons, 1992.

[Dechter, 1992b] R. Dechter. From local to global consistency. *Artificial Intelligence*, 55:87–107, 1992.

[Freuder, 1978] E. C. Freuder. Synthesizing constraint expressions. *Comm. ACM*, 21:958–966, 1978.

[Freuder, 1982] E. C. Freuder. A sufficient condition for backtrack-free search. *J. ACM*, 29:24–32, 1982.

[Freuder, 1985] E. C. Freuder. A sufficient condition for backtrack-bounded search. *J. ACM*, 32:755–761, 1985.

[Ginsberg *et al.*, 1990] M. L. Ginsberg, M. Frank, M. P. Halpin, and M. C. Torrance. Search lessons learned from crossword puzzles. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 210–215, Boston, Mass., 1990.

[Kondrak, 1993] G. Kondrak, 1993. Personal Communication.

[Mackworth, 1977] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.

[Maruyama, 1990] H. Maruyama. Structural disambiguation with constraint propagation. In *Proceedings of the 28th Conference of the Association for Computational Linguistics*, pages 31–38, Pittsburgh, Pennsylvania, 1990.

[Meiri, 1991] I. Meiri. Combining qualitative and quantitative constraints in temporal reasoning. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 260–267, Anaheim, Calif., 1991. An expanded version is available as: Department of Computer Science Technical Report R-160, University of California, Los Angeles.

[Montanari, 1974] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Inform. Sci.*, 7:95–132, 1974.

[Nadel, 1989] B. A. Nadel. Constraint satisfaction algorithms. *Computational Intelligence*, 5:188–224, 1989.

[van Beek and Dechter, 1994] P. van Beek and R. Dechter. On the minimality and decomposability of row-convex constraint networks. *Accepted for publication in J. ACM*, 1994.

[van Beek, 1992] P. van Beek. Reasoning about qualitative temporal information. *Artificial Intelligence*, 58:297–326, 1992.

[Waltz, 1975] D. Waltz. Understanding line drawings of scenes with shadows. In P. H. Winston, editor, *The Psychology of Computer Vision*, pages 19–91. McGraw-Hill, 1975.