Solution Equivalent Subquadrangle Reformulations of Constraint Satisfaction Problems

Student: Chris Houghton
Supervisor: Prof. David Cohen

Department of Computer Science,
Royal Holloway, University of London, UK

Abstract. Subquadrangles are a natural way in which to represent constraints as they do not restrict any subset of their scope. There are already known methods for converting any given constraint into a set of subquadrangles, but we require a method by which an entire constraint satisfaction problem instance can be converted as a whole.
In this paper we present two methods by which we might decompose a given instance into a solution equivalent instance whose constraints are all subquadrangles.

1 Introduction

Constraint Satisfaction Problem [2] instances (CSPs) are a natural way to model real life problems such as image processing [2], scheduling [4] and natural language understanding [1]. In this paper we are concerned with the modeling of problems as CSPs and how this can affect the performance of different solution algorithms. In particular we are interested in modeling in the language of subquadrangles.

A Quadrangle is essentially an 'anything-goes' constraint for some Cartesian product of domains. A Subquadrangle [3] is a constraint all of whose projections to proper subsets of the scope are quadrangles; it does not restrict any proper subset of the variables it is constraining.

Subquadrangles are a very 'natural' way in which to represent constraints. This is because they do not place any restrictions on proper subsets of their scope, thus reducing the number of required constraint checks. This leads us to believe that a subquadrangle aware solver could be particularly efficient.

However, in this paper we shall consider another intriguing use of subquadrangle modeling. Subquadrangles delay the failure of constraint checks until you try and assign a value to the last variable in their scope. This makes them ideal for testing constraint solvers as they are 'backtrack nasty'.

We start by giving background definitions in Section 2. We then consider two complete subquadrangle reformulations in Section 3. We draw conclusions and consider future work in Section 4.

2 Definitions

Throughout this paper, we will require the following definitions;

Definition 1. *A* **constraint satisfaction problem [2] instance** *(CSP) is a triple* $\langle V, D, C\rangle$ *where:*

- V *is a finite set of* **variables**.
- D *is a finite* **domain** *of values.*
- C *is a set of* **constraints**.

Each $c \in C$ *is a pair,* $\langle \sigma(c), \rho(c)\rangle$*, where* $\sigma(c)$*, the constraint* **scope***, is any subset of* V *and* $\rho(c)$ *is a set of* **labellings** *each of which is a mapping from* $\sigma(c)$ *to* D.

A **solution** *to a CSP,* $P = \langle V, D, C\rangle$ *is a labelling* s *of* V *such that for any* $\langle \sigma, \rho\rangle \in C, s_{|_\sigma} \in \rho$.

For simplicity, the domain of any instance we define will be $\{0, \ldots, n-1\}$.

Definition 2. *Let* $c_i = \langle \sigma_i, \rho_i\rangle, i = 1, 2$ *be two constraints. For any* $m \subseteq \sigma_1$ *the* **projection** *of* ρ_1 *onto* m*, denoted* $\pi_m \rho_1$*, is simply* $\{f_{|_m} | f \in \rho_1\}$*. The projection of* c_1 *onto* m *is* $\langle m, \pi_m \rho_1\rangle$*. The* **join***,* $c_1 \bowtie c_2$*, is the constraint* $\left\langle \sigma_1 \cup \sigma_2, \{f \mid f_{|_{\sigma_i}} \in \rho_i, i = 1, 2\}\right\rangle$*. The* **union***,* $c_1 \cup c_2$*, of two constraints with the same scope (*$\sigma_1 = \sigma_2$*), is the constraint* $\langle \sigma_1, \{f \mid f \in \rho_1 \text{ or } f \in \rho_2\}\rangle$.

Definition 3. *Let* $P = \langle V, D, C\rangle$ *be a constraint satisfaction problem. Let* $c = \langle \sigma, \rho\rangle$ *be any constraint of* P *and* v *a variable in* σ*. The* **effective domain** *of* v *with respect to* c*,* $E_c(v)$ *is the projection of* c *onto* $\{v\}$.

If the constraint relation $\rho = \{f \mid \forall v \in \sigma, f(v) \in E_c(v)\}$ *we say that* c *is a* **quadrangle***. If for all* $s \subset \sigma$*, the projection of* c *onto* s *is a quadrangle we say that* c *is a* **subquadrangle**.

A quadrangle is a constraint which has the property that if you assign to the variables any arbitrary values from their respective effective domains then the resulting labelling is in the relation. It is essentially 'anything goes' on the effective domains.

A subquadrangle [3] does not constrain the variables in any proper subset of its scope, in that the projection onto any proper subset of its variables is a quadrangle. This means that you can assign any values from the effective domains of any $|\sigma| - 1$ variables of the constraint before being constrained as to what value you may choose for the last one. There is, however, always a value which can be assigned to this last variable.

Example 1. Any unary constraint is a quadrangle, as is any constraint which allows only one labelling. As all of the projections onto proper subsets are unary, a binary constraint is always a subquadrangle, but is not necessarily a quadrangle.

Definition 4. *Let $\langle V, D, C\rangle$ be a CSP and $\sigma \subseteq V$. We define the following constraint for $m \leq n$ and $i < m$;*

$$A(\sigma, m, i) = \left\langle \sigma, \left\{ f \mid \sum_{v \in \sigma} f(v) \mod m \equiv i \right\} \right\rangle$$

In later sections, we will make strong use of the following proposition;

Proposition 1. *Definition 4 may be used to generate subquadrangles on a given scope in such a way that their join is empty. This will be used in our two subquadrangle decomposition methods as a base with which to reformulate a given constraint on a given scope.*

In order for proposition 1 to hold, we need the following two lemmas.

Lemma 1. *The constraints defined in definition 4 are subquadrangles.*

Proof. Given any partial tuple in the relation, t, over a $|\sigma| - 1$ subset of σ, there must be an extension to the tuple in f as we mod by no more than the size of the domain, n, and $i < n$.

Lemma 2. *For a given scope, σ, and set of unique values W where $\forall w \in W, w < m$ and $|W| > 1$, the join of the constraints, $\bowtie_{w \in W} A(\sigma, m, w)$ is empty.*

Proof. Given any two constraints $A(\sigma, m, p)$ and $A(\sigma, m, q)$ where $p, q \in W$ and $p \neq q$. If a tuple, t, is in the relation of $A(\sigma, m, p)$, then by definition 4 it can not also be in the relation of $A(\sigma, m, q)$. As none of the relations of the constraints share any common tuples, yet are over the same scope, σ, there can be no tuples in the relation of the join.

3 Making Backtrack Nasty

3.1 Decomposing into three Subquadrangles

We can now use the subquadrangles defined in definition 4 to generate a set of subquadrangles which is equivalent to a given single constraint.

Definition 5. *Let $c = \langle \sigma, \rho \rangle$ be a constraint. Let $x, y \notin \sigma, x \neq y$ be additional variables, both with domain $\{0, 1\}$. We can define the following three subquadrangles:*

$$q_0 = A(\sigma(c) \cup \{x\}, 2, 0) \cup [c \bowtie \langle \{x\}, \{\langle 0\rangle, \langle 1\rangle\}\rangle \cap A(\sigma(c) \cup \{x\}, 2, 1)]$$
$$q_1 = A(\sigma(c) \cup \{y\}, 2, 1)$$
$$q_2 = \langle \{x, y\}, \{\langle 0, 0\rangle, \langle 1, 1\rangle\}\rangle$$

Before we go on to propose a theorem based on this definition, we need a small lemma;

Lemma 3. *A subquadrangle, c, combined with any other labellings over the same scope and over subsets of the effective domains is still a subquadrangle.*

Proof. A quadrangle is a Cartesian product of effective domains. From the definition of a subquadrangle, adding a labelling from this Cartesian product still leaves all of the $|\sigma| - 1$ ary projections as quadrangles as you do not increase their number of labellings. The result follows from repeated application of adding single labellings.

Theorem 1. *Given a constraint c, Definition 5 generates 3 subquadrangles,* q_0*,* q_1*,* q_2*, such that* $\pi_{\sigma(c)} \bowtie_{i=0}^{2} q_i = c$*.*

Proof. It is clear that if given a constraint c, definition 5 generates 3 subquadrangles, q_0, q_1, q_2, such that $\pi_{\sigma(c)} \bowtie_{i=0}^{2} q_i = c$, then Method 1 holds by repeated application.

So, we are required to prove that:

1. q_0, q_1 and q_2 are subquadrangles.
2. All solutions in c are in $\pi_{\sigma(c)} (q_0 \bowtie q_1 \bowtie q_2)$
3. There are no solutions in $\pi_{\sigma(c)} (q_0 \bowtie q_1 \bowtie q_2)$ which are not in c.

CASE 1:

If we split q_0 into:

$$\begin{aligned} &(\mathrm{A}) = A(\sigma(c) \cup \{x\}, 2, 0) \\ &(\mathrm{B}) = c \bowtie \langle \{x\}, \{\langle 0\rangle, \langle 1\rangle\}\rangle \cap A(\sigma(c) \cup \{x\}, 2, 1) \end{aligned}$$

then

(A) is a subquadrangle from lemma 1.

(A) $\cup$ (B) is a subquadrangle from lemma 3.

It follows that q_0 is a subquadrangle.

q_1 is a subquadrangle from method 1.

q_2 is a subquadrangle as it is binary and by definition all binary constraints are subquadrangles.

CASE 2:

Consider any $t \in \rho(c)$. In q_0, t extended to some value at x, $t_x^{\rightarrow}$, must be in $\rho(A(\sigma(c) \cup \{x\}, 2, 1))$ as it is a subquadrangle. $t_x^{\rightarrow}$ is in $\rho(c \bowtie \langle \{x\}, \{\langle 0\rangle, \langle 1\rangle\}\rangle)$. Since, by definition, $x \notin \sigma(c)$ and so every $t \in \rho(c)$ extends to both 0 and 1 at variable x in the constraint $c \bowtie \langle \{x\}, \{\langle 0\rangle, \langle 1\rangle\}\rangle$. The value of x must be either 0 or 1 depending on which gives $t_x^{\rightarrow} \in \rho(A(\sigma(c) \cup \{x\}, 2, 1))$ odd parity. In q_1, t extended to some value at y, $t_y^{\rightarrow}$, must be in $\rho(A(\sigma(c) \cup \{y\}, 2, 1))$ as it is a subquadrangle. The value of y must be either 0 or 1 depending on which gives $t_y^{\rightarrow} \in \rho(A(\sigma(c) \cup \{x\}, 2, 1))$ odd parity.

As both $t^{\rightarrow}_{x} \in \rho\left(A\left(\sigma\left(c\right) \cup \{x\}, 2, 1\right)\right)$ and $t^{\rightarrow}_{y} \in \rho\left(A\left(\sigma\left(c\right) \cup \{x\}, 2, 1\right)\right)$ must have odd parity, it can be seen that $x = y$.

So, $t^{\rightarrow}_{x,y} \in \rho\left(q_0 \bowtie q_1 \bowtie q_2\right)$, and hence $t \in \rho\left(\pi_{\sigma(c)} q_0 \bowtie q_1 \bowtie q_2\right)$.

CASE 3:

Assume for contradiction that there is a tuple, t, s.t. $t \in \rho\left(\pi_{\sigma(c)} q_0 \bowtie q_1 \bowtie q_2\right)$ but $t \notin \rho\left(c\right)$. If $t \notin \rho\left(c\right)$, then t can not be in $\rho\left(c \bowtie \langle\{x\}, \{\langle 0\rangle, \langle 1\rangle\}\rangle\right)$. Hence, if $t \in \rho\left(\pi_{\sigma(c)} q_0 \bowtie q_1 \bowtie q_2\right)$ then t must be in $\rho\left(A\left(\sigma\left(c\right) \cup \{x\}, 2, 0\right)\right)$, and so $t^{\rightarrow}_{x}$ must have even parity.

However, if $t^{\rightarrow}_{y} \in \rho\left(A\left(\sigma\left(c\right) \cup \{y\}, 2, 1\right)\right)$ then $t^{\rightarrow}_{y}$ must have odd parity. This means that $x \neq y$. By definition, q_3 states that $x = y$ so $t \notin \rho\left(\pi_{\sigma(c)} q_0 \bowtie q_1 \bowtie q_2\right)$.

Method 1 *INPUT: A CSP,* $P = \langle V, D, C\rangle$
OUTPUT: A CSP, $P' = \langle V', D, C'\rangle$.

Foreach $c \in C$ *we replace* c *with the three subquadrangles described in definition 5 and add the extra two variables into* V.

Theorem 2. *Given a CSP,* $P = \langle V, D, C\rangle$, *Method 1 generates a new CSP,* $P' = \langle V', D, C'\rangle$, *such that* $\pi_V \bowtie_{c \in C'} = \pi_V \bowtie_{c \in C}$.

Proof. This method is shown to be true by repeated application of theorem 1.

This reformulation has the useful property that it is always strongly $r - 1$ consistent where r is the arity of the smallest constraint (subquadrangle) in C' which contains at least 1 variable from V. (i.e. not counting the q_2 constraints). This is in indication that this formulation is backtrack nasty as this consistency is enough to guarantee that you will always have to go to at least a depth of $r - 1$ in any search branch to see if it leads to a solution, but is not enough to guarantee solutions.

3.2 Obscuring Intersecting Constraints

Method 2 *INPUT: A CSP,* $P = \langle V, D, C\rangle$.
OUTPUT: A CSP, $P' = \langle V', D, C'\rangle$.

Foreach $c \in C$
  Select $c_{i,j} \in C$ *s.t.* $i \neq j$ *and* $\sigma\left(c_i\right) \cap \sigma\left(c_j\right) \neq \emptyset$
  $\hat{c} = c_i \bowtie c_j$
  While $|\sigma\left(\hat{c}\right)| - |\sigma\left(c_i\right)| < 1$
    Select $v \in \sigma\left(c_j\right), v \notin \sigma\left(c_i\right)$
      $c_i \leftarrow c_i \bowtie \langle\langle v\rangle, D\left(v\right)\rangle$
  While $|\sigma\left(\hat{c}\right)| - |\sigma\left(c_j\right)| < 1$
    Select $v \in \sigma\left(c_i\right), v \notin \sigma\left(c_j\right)$
      $c_j \leftarrow c_j \bowtie \langle\langle v\rangle, D\left(v\right)\rangle$
  Generate the following affine constraints;
    $s_i = A\left(\sigma\left(c_i\right) \cup \{y_{i,0}\}, 4, 0\right)$
    $s'_i = A\left(\sigma\left(c_i\right) \cup \{y_{i,1}\}, 4, 1\right)$

$s_j = A\left(\sigma\left(c_j\right) \cup \{y_{i,0}\}, 4, 2\right)$
$s'_j = A\left(\sigma\left(c_j\right) \cup \{y_{i,1}\}, 4, 3\right)$
Extend each tuple of $\rho\left(\pi_{c_i}\hat{c}\right)$ *to* $y_{i,0}$ *s.t.*
the equation $y_{i,0} \equiv 3 - \sum_{j=0}^{|\sigma(c_j)|-1} f\left(v_j\right) \mod 4$ *holds*
and add these extended tuples to $\rho\left(s_i\right)$
Extend each tuple of $\rho\left(\pi_{c_j}\hat{c}\right)$ *to* $y_{i,1}$ *s.t.*
the equation $y_{i,1} \equiv 1 - \sum_{i=0}^{|\sigma(c_i)|-1} f\left(v_i\right) \mod 4$ *holds*
and add these extended tuples to $\rho\left(s_j\right)$

The method has three stages. This approach would not work if the pairs of intersecting constraints chosen have more than one non-intersecting variable each. The first stage of the method is to extend the intersection to originally non-intersecting variables by extending them to any combination of domain values so that each constraint is left with at most one non-intersecting variable.

The second stage creates the four affine subquadrangles whose join is empty. These are used as the base for the third stage which adds in the necessary extended tuples to allow the original solutions to be expressed. This is done by finding the relevant values for the additional y variables. These values are determined by the generated value in the oppositely paired constraint (i.e. s_i and s'_j).

4 Conclusions and Further Work

In this paper we have presented two possible solution equivalent subquadrangle reformulations for constraint satisfaction problems. The motivation behind such decompositions is to provide backtrack nasty reformulations to access the effectiveness of current solution techniques.

We hope to investigate the possibility of a subquadrangle aware solver that understands delayed failure and prevents unnecessary constraint checks.

Our intention is to consider the difference in problem size for different representations of the generated subquadrangles, compared to the size of the original constraints, and see what effects this may have on the complexity of solution.

While Method 1 has been demonstrated to be correct, we are currently relying on empirical evidence to suggest that Method 2 is also correct. We intend to find a proof of this in due course.

References

1. James Allen *Natural Language Understanding — 2nd Edition.* The Benjamin/Cummings Publishing Company, 1995.
2. U. Montanari. Networks of Constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95-132, 1974. 1997.
3. R. Rodošek. *Generation and Comparison of Constraint-Based Heuristics Using the Structure of Constraints.* PhD thesis, Imperial College, University of London, July 1997.
4. P. van Beek. Reasoning about Qualitative Temporal Information. *Artificial Intelligence*, 58:297-326, 1992