2010

A Partial Taxonomy of Substitutability and Interchangeability

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

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

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

Steven D. Prestwich
University College Cork, s.prestwich@4c.ucc.ie

Eugene C. Freuder
University College Cork, e.freuder@cs.ucc.ie

A Partial Taxonomy of Substitutability and Interchangeability

Shant Karakashian[1], Robert Woodward[1], Berthe Y. Choueiry[1], Steven D. Prestwich[2] and Eugene C. Freuder[2]

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

[2] Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Ireland {s.prestwich,e.freuder}@4c.ucc.ie

Abstract. Substitutability, interchangeability and related concepts in Constraint Programming were introduced approximately twenty years ago and have given rise to considerable subsequent research. We survey this work, classify, and relate the different concepts, and indicate directions for future work, in particular with respect to making connections with research into symmetry breaking. This paper is a condensed version of a larger work in progress.

1 Introduction

Many important problems in computer science, engineering and management can be formulated as Constraint Satisfaction Problems (CSPs). A CSP is a triple (V, D, C) where V is a set of variables, D the set of their domain values, and C a set of constraints on the variables that specify the permitted or forbidden combinations of value assignment to variables. A solution to a CSP is an assignment of values to all variables such that all constraints are satisfied. CSPs are usually solved by interleaving backtrack search with some form of constraint propagation, for example forward checking or arc consistency.

Constraint problems often exhibit symmetries. A great deal of research has been devoted to *symmetry breaking* techniques in order to reduce the size of the search space [Various, 1991 present]. The earliest works on symmetry breaking include [Glaisher, 1874; Brown *et al.*, 1988]. In this paper we will not survey the large literature on symmetry breaking, but a recent survey can be found in [Gent *et al.*, 2006].

Interchangeability, proposed in a seminal paper by Freuder [1991], is one of the first forms of symmetry identified for CSPs. Importantly, it is also the first method proposed for *detecting* symmetry as opposed to having a constraint programmer manually specify it. Although there has been since then a steady flow of research papers developing this concept in both theory and practice, it has been relatively neglected compared to other forms of symmetry. This situation is surprising: While in its basic form, interchangeability is a special case of value symmetry, its various extensions (already proposed in the 1991 paper) make

it a more general concept than is sometimes perceived, and anticipate some subsequent developments in symmetry definition and breaking. The comparison with the various types and definitions of symmetry [Benhamou, 1994; Cohen *et al.*, 2006] will be discussed in the longer version of this paper.

The goal of our endeavor is to analyze the research conducted so far on interchangeability, relate it to symmetry, and identify opportunities for future research. This paper is a work in progress and a first step towards our goal. Our survey is partial and far from complete and we welcome the feedback of the readers and workshop participants.

The advantages of detecting and exploiting interchangeability have been established on random problems, benchmarks, and real-world applications. In backtrack search, the advantages are mainly the reduction of the search space and the search effort [1], and the attainment of multiple solutions by bundling. In local search, interchangeability is used to locally repair partial solutions [Petcu and Faltings, 2003]. Real-world applications include nurse scheduling [Weil and Heus, 1998] and resource allocation in hospitals [Choueiry *et al.*, 1995].

This paper is structured as follows. In Section 2, we give the definitions of the basic interchangeability concepts and relate them to each other. In Section 3, we discuss forms of conditional interchangeability. In Section 4, we discuss other forms of interchangeability that have appeared in the literature. In Section 5, we relate the various forms of interchangeability. Finally, in Section 6, we list topics that we plan to cover more fully in the expanded version of this paper.

2 Basic Interchangeability Concepts

In this section, we review the various forms of interchangeability originally introduced in [Freuder, 1991]. We also include a few new interchangeability concepts that directly relate to the original ones. Full interchangeability, the most basic form of interchangeability, is defined as follows.

Full interchangeability (FI) [Freuder, 1991] A value a for variable v is fully interchangeable with value b iff every solution in which $v = a$ remains a solution when b is substituted for a and vice-versa.

If two values are interchangeable then one of them can be removed from the domain, reducing the size of the problem; alternatively they can be bundled together in a Cartesian product representation of solutions. Figures 1, 2 and 3 show examples of two values a and b that are FI (see below for definition of 3-I and NSub). In our figures, a small solid circle denotes a value in the domain of the variable represented by the outline circle, and edges link consistent tuples.

Notice that FI is defined 'at the solution level', which means that in order to find all FI values for a given variable, one must account for *all* constraints and may have to compute all solutions. Thus, FI is a *global* property. In [1994], Benhamou defines the equivalent notion of 'value symmetry in all solutions' as

[1] Most importantly, by factoring out no-goods [Choueiry and Davis, 2002].

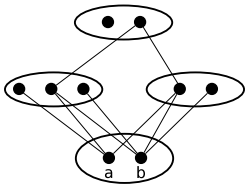

Fig. 1. FI: a and b are FI but not 3-I or NSub.

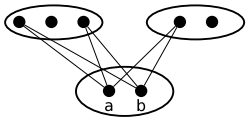

Fig. 2. NI: a and b are NI.

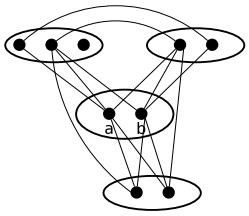

Fig. 3. KI: a and b are 3-I but not NI.

semantic symmetry. Hence, the terms 'semantic' and 'global' are equivalent. Because the detection of global forms of interchangeability is likely to be intractable, Freuder introduced *local* variants, which account only for the constraints defined on a variable, that is, the neighborhood of the variable. In [1994], Benhamou calls such relations *syntactic* symmetries. Section 2.1 discusses local interchangeability. Further, interchangeability is an equivalence relation on the domain of the variable: interchangeable values are equivalent. Such equivalences may be rare in practice. To remedy this situation, Freuder proposed various extensions to the basic concept, which are discussed in Sections 2.2 and 2.3.

In summary, one may think of interchangeability as a core concept characterized as a relation between two values either at the solution level (i.e., global or semantic) or in the neighborhood of the variable (i.e., local or syntactic). Also, the concepts may require that interchangeable values be equivalent (i.e., strong), or not 'perfectly' so (i.e., weak or approximate).

When comparing two forms of interchangeability X and Y, we say that $X \rightarrow Y$ iff any two values a and b that are related by X are also related by Y but the converse does not necessarily hold[2]*,* regardless of whether Y is derived from X by relaxing the conditions of X (i.e., Y is weaker than X) or by 'moving' from the local level to the global level (i.e., syntactic to semantic). Note that when $X \rightarrow Y$, Y leads to greater problem reduction than X.

2.1 Local forms of interchangeability

In general, the identification of a local interchangeability is tractable because it focuses on the neighborhood of the variable. Also, a given local form interchangeability usually implies the corresponding global one.

Neighborhood interchangeability (NI) [Freuder, 1991] A value a for variable v is neighborhood interchangeable with value b iff for every constraint on v, the values compatible with $v = a$ are exactly those compatible with $v = b$. Values a and b are NI in Figure 2 but not in Figures 1 or 3.

Neighborhood interchangeable values for a given variable can be detected by comparing the values in the variable's domain for consistency to all variable-value pairs in the variable's neighborhood and drawing a discrimination tree

[2] a and b are two values or two partial assignments over the same variables.

[Freuder, 1991]. At the end of the process, the leaves of the discrimination tree are annotated with the equivalence NI values for the variable. The complexity of this process is $\mathcal{O}(n^2d^2)$, where n is the number of variables and d is the maximum domain size. Alternatively, one can build a refutation tree, which proceeds by splitting the domain of the variable [Likitvivatanavong and Yap, 2008]. The lower bound of the worst-case complexity of the refutation tree is smaller than that of the discrimination tree. However, it is not clear whether the difference is meaningful in practice. Further, the discrimination tree can be directly used to implement forward checking at no additional cost [Beckwith *et al.*, 2001], but it is not clear yet whether or not the same can be done with the refutation tree.

For non-binary constraints, neighborhood interchangeable values can be detected by constructing non-binary discrimination trees for each variable [Lal *et al.*, 2005]. As described in [Lal *et al.*, 2005], the process also allows the use of forward checking during search. The complexity to build a non-binary discrimination tree for a single variable is $\mathcal{O}(n\ deg\, a^{k+1}(1-t))$, where n is the number of variables, deg is the maximum degree of a variable, a is the maximum domain size, and t is the tightness of the constraints, defined as the ratio of the number of forbidden tuples over the number of all possible tuples.

K-interchangeability (KI) [Freuder, 1991] For $k \geq 2$, two values, a and b for a CSP variable X, are *k-interchangeable* iff a and b are fully interchangeable in any subproblem of the CSP induced by X and $(k-1)$ other variables. Values a and b are 3-I in Figures 2 and 3 but not in Figure 1.

K-interchangeable values can be identified by a modification of the discrimination-tree algorithm for NI. The complexity of the process is $\mathcal{O}(n^k d^k)$ [Freuder, 1991].

Theorem 1. NI $\rightarrow$ KI $\rightarrow$ FI, see [Freuder, 1991].

For $2 < i < j < |V|$, i-interchangeability is a sufficient but not necessary condition for j-interchangeability. NI is 2-interchangeability and FI is $|V|$-interchangeability. Hence, NI $\rightarrow$ KI $\rightarrow$ FI. a and b in Figure 1 are FI but not 3-I, and in Figure 3 they are 3-I but not NI.

2.2 Extended interchangeability: Weak forms

Below, we discuss three weak forms of interchangeability introduced in [Freuder, 1991] (i.e., subproblem interchangeability, partial interchangeability, and substitutability) and a number of other related concepts.

Subproblem interchangeability (SPrI) [Freuder, 1991] Two values are *subproblem interchangeable*, with respect to a subset of variables S, iff they are fully interchangeable with regards to the solutions of the subproblem of the CSP induced by S.

Partial interchangeability (PI) [Freuder, 1991] Two values are *partially interchangeable* with respect to a subset S of variables, iff any solution involving one implies a solution involving the other with possibly different values for variables in S. In Figure 4, a and b are PI *wrt* S, shown with the dotted line.

Theorem 2. FI → PI.

If a and b are FI, they are by definition PI with respect to any subset of V. In Figure 4, a and b are PI *wrt* to the subset S but not FI.

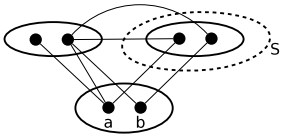

Fig. 4. PI: a and b are PI *wrt* S but not Sub, FI, CtxDepI, NTI, or NPI *wrt* any subset.

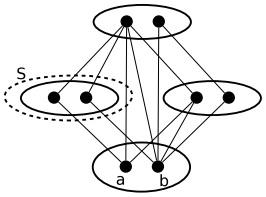

Fig. 5. PI: a and b are PI *wrt* S but not Sub or SPrI *wrt* any subset of variables.

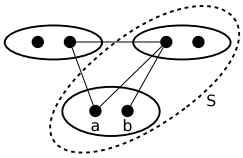

Fig. 6. SPrI: a and b are SPrI *wrt* S but not PI *wrt* any subset of variables.

Theorem 3. SPrI and PI are not comparable[3].

In Figure 5, a and b are PI but not SPrI. In Figure 6, a and b are SPrI but not PI.

Substitutability (Sub) [Freuder, 1991] For two values a and b for variable v, a is substitutable for b iff every solution in which $v = b$ remains a solution when b is replaced by a but not necessarily vice-versa. Figure 7 shows an example.

Note that the concept of substitutability is related to that of dominance [Bellicha *et al.*, 1994], which is used in the literature on symmetry breaking.

Theorem 4. FI → Sub.

If a and b are FI, they are by definition mutually substitutable. In Figure 7, a is substitutable for b, but a and b are not FI.

Again, because substitutable values are expensive to compute, neighborhood substitutability (NSub) (with the obvious definition) is computationally advantageous. In Figure 8, a is NSub for b. In [1994], Bellicha et al. propose NS-Closure, an algorithm to enforce NSub. It removes all of the neighborhood substitutable values from the network. It operates by examining every pair of values (a, b) in a variable's domain, trying to find a *splitter* for the pair. A splitter for (a, b) is a value in the neighborhood of the variable that supports a but not b. If (a, b) does not have a splitter, then a can be removed from the domain. The time complexity of the algorithm is $\mathcal{O}(md^3)$, where m is the number of constraints and d is the maximum domain size. The space complexity of storing the splitters is $\mathcal{O}(nd^2)$, where n is the number of variables.

Theorem 5. NI → NSub → Sub; FI and NSub are not comparable.

[3] This theorem corrects Theorem 5 of [Freuder, 1991].

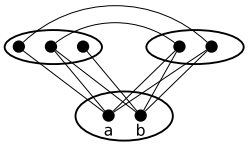

Fig. 7. Sub: a is Sub, but not NSub, for b; a and b are not FI.

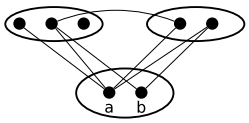

Fig. 8. a is NSub for b but a and b are not NI or FI.

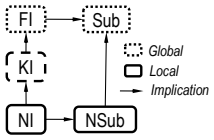

Fig. 9. Illustrating Theorems 1, 4, and 5.

Figure 9 illustrates this situation. First consider NI → NSub. Any NI values are mutually NSub by definition. In Figure 8, a is NSub for b but a and b are not NI. Now, consider NSub → Sub. Given two values a and b for a variable, a is NSub for b, the set of variable-value pairs supporting b is a subset of the one supporting a. By moving to global substitutability, the sets of supports will only lose elements, however, the set of support of b will remain a subset of that of a. Figure 7 shows an example where a is Sub, but not NSub, for b. In Figure 1, a and b are FI but not NSub. In Figure 8, a is NSub for b but a, and b are not FI.

Neighborhood Partial Interchangeability (NPI) [Choueiry and Noubir, 1998] Two values b and c for a variable v are NPI given a boundary of change S (which includes v) iff, for every constraint C defined on the variables (v,w) where $v \in S$, $w \notin S$, we have: $\{j|(b,j)$ satisfies $C\} = \{j|(c,j)$ satisfies $C\}$.

The NPI sets of a variable's domain can be detected by modifying the discrimination tree algorithm of NI to a joint discrimination tree (JDT) by considering the neighborhood of a set of variables instead of the neighborhood of a single variable as done in the discrimination tree [Choueiry and Noubir, 1998]. The complexity of the algorithm to build a JDT for a single variable is $\mathcal{O}(s(n-s)d^2)$ and the space complexity for the tree is $\mathcal{O}((n-s)d)$, where n is the number of variables, s is the size of the given set, and d is the size of the domain.

Theorem 6. NPI and PI are not comparable.[4]

In Figure 4, a and b are PI but not NPI. In Figure 11, they are NPI but not PI.

Theorem 7. NPI → SPrI.

If a and b are NPI outside the boundary of change S, then they are NI in the subproblem induced by $V \setminus S$. If they are NI in the subproblem, then they are also FI, and therefore SPrI in the subproblem induced by $V \setminus S$. Figure 10 shows an example where the converse does not hold.

Directional Interchangeability (DirI) [Naanaa, 2007a] Two values a and b in the domain of a variable X are DirI with respect to a variable ordering of the variables iff they have the same preceding support set: $\{c \mid (a,c) \in C_{XY}$ and $Y \prec X\} = \{c \mid (b,c) \in C_{XY}$ and $Y \prec X\}$.

[4] This theorem corrects [Choueiry and Noubir, 1998], which states that NPI implies PI. This error was mentioned in [Neagu and Faltings, 2005].

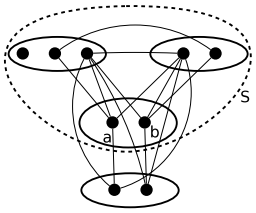

Fig. 10. SPrI: a and b are SPrI *wrt* S but not NPI *wrt* to any subset or Sub.

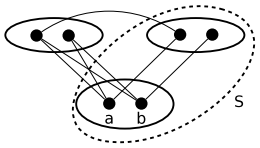

Fig. 11. NPI: a and b are NPI *wrt* S but not PI *wrt* to any subset, SUB, FI or NTI.

Theorem 8. NPI $\equiv$ DirI.

If a and b for a variable X are NPI *wrt* a boundary of change S, then they are DirI *wrt* any variable ordering such that $\forall Y \in S$, $X \prec Y$.

Directional Substitutability (DirSub) [Naanaa, 2007b; 2009] Value a is DirSub for value b for a variable X with respect to a variable ordering of the variables iff the preceding support set of b is a subset of that of b: $\{c \mid (b,c) \in C_{XY}$ and $Y \prec X\} \subseteq \{c \mid (a,c) \in C_{XY}$ and $Y \prec X\}$.

Theorem 9. DirI $\rightarrow$ DirSub.

If a and b of variable X are DirI *wrt* a given variable ordering, then, by definition, the preceding support sets for a and b are the same and consequently subsets of one another.

Neighborhood Interchangeability Relative to a Constraint ($\mathbf{NI}_C$) [Haselböck, 1993] Two values are NI_C relative to a constraint C iff they are NI in the problem induced by the variables in the scope of C.

NI_C values for variable v can be detected by restricting the discrimination tree to the considered constraint. As a result, the time complexity of finding all NI_C sets is $\mathcal{O}(eka^k)$, where e is the number of constraints, k is the maximum arity, and a is the maximum domain size. In [1993], Haselböck modified the usual Revise procedure for lookahead to exploit the (statically computed) NI_C sets during search, yielding a solution bundle. The time complexity of the new Revise procedure is thus reduced to $\mathcal{O}(a'^2)$, where $1 \leq a' \leq a$. In [Beckwith *et al.*, 2001], it was shown that the resulting bundles are never 'thinner' than those obtained in [Benson and Freuder, 1992], and never 'fatter' than those obtained by those obtained in [Hubbe and Freuder, 1992], which in turn are equivalent to those obtained by [Beckwith *et al.*, 2001].

Neighborhood Substitutability Relative to a Constraint ($\mathbf{NSub}_C$) [Bousse-mart *et al.*, 2004] Two values are NSub_C relative to a constraint C iff they are NSub in the problem induced by the variables in the scope of C.

Theorem 10. NPI $\rightarrow \mathrm{NI}_C \rightarrow \mathrm{NS}_C$.

First consider NPI $\rightarrow \mathrm{NI}_C$. If for variable X, a and b are NPI, then for every constraint C between X and a variable outside of the boundary of change, a and b are NI_C. $\mathrm{NI}_C \rightarrow \mathrm{NS}_C$ follows directly from the definition.

2.3 Other extended forms of interchangeability

Other extended forms that were initially proposed are: meta-interchangeability, dynamic interchangeability, and functional interchangeability.

Meta-interchangeability (MI) [Freuder, 1991] By grouping variables into 'meta-variables', or values into 'meta-values', we can introduce interchangeability into higher level 'meta-problem' representations of the original CSP.

Values may become interchangeable or substitutable during backtrack search after some variables have been instantiated, so even a problem with no interchangeable values may exhibit interchangeability under some search strategy.

Dynamic Neighborhood Interchangeability (DynNI)[5] [Beckwith and Choueiry, 2001] Two values a and b for variable X are DynNI with respect to a set A of variable assignments iff they are NI in the subproblem induced by $A \cup \{X\}$.

Theorem 11. *NI → DynNI.*

Consider values a and b for a variable v that are NI, assume a and b are not DynNI. Then, for an assignment for the subset of variables S, either a and b are not NI in the problem induced by $V \setminus S$, or one of a or b is deleted. The former case is impossible because a and b have the same set of supports in the original problem, and thus must have the same supports after the assignments. The latter case is also impossible because a and b having the same support sets, if a loses all its supports in a neighboring variable, then b also loses all supports because the support sets are the same.

Full Dynamic Interchangeability (FDynI) [Prestwich, 2004a] A value a for variable v is dynamically interchangeable for b with respect to a set A of variable assignments iff they are fully interchangeable in the subproblem induced by A.

Theorem 12. DynNI → FDynI.

If a and b are DynNI, then a and b are consistent with the same set of values in the assignment A. They are also NI relative to the variables in the problem induced by A that are not yet assigned and, consequently, are FI.

Functional interchangeability [Freuder, 1991] Let $S_{a|X}$ be the set of solutions including value a for variable X. Two values a for X and b for Y are *functionally interchangeable* iff there exists functions f and f' such that $f(S_{a|X}) = S_{b|Y}$ and $f(S_{b|Y}) = S_{a|X}$.

Two values a and b for a variable are *isomorphically interchangeable* [Freuder, 1991] iff there exists a 1-1 function f such that $b = f(a)$ and for any solution S involving a, $\{f(v) \mid v \in S\}$ is a solution. Also for any solution S involving b, $\{f^{-1}(v) \mid v \in S\}$ is a solution.

In the longer version of this paper, we compare functional and isomorphic interchangeability with the definitions of symmetry introduced in [Benhamou, 1994; Cohen *et al.*, 2006].

[5] Dynamic Interchangeability (DynI) property was incorrectly characterized as Dynamic Neighborhood Partial Interchangeability (DNPI) in [Beckwith and Choueiry, 2001; Choueiry and Davis, 2002; Lal and Choueiry, 2004; Lal *et al.*, 2005].

3 Conditional Forms of Interchangeability

Conditions can be added to a CSP in the form of constraints that further constrain the problem. In problems with little interchangeability, such conditions can be imposed to increase the interchangeability among the variable values. In [2004], Zhang and Freuder introduced and studied conditional interchangeability, conditional substitutability, conditional neighborhood interchangeability and conditional neighborhood substitutability.

Conditional Interchangeability (ConI) [Zhang and Freuder, 2004] Two values a and b of variable v are ConI under a condition imposed by a set of additional constraints iff they are FI in the problem with the additional constraints.

Similarly Conditional Neighborhood Interchangeability (ConNI), Conditional Substitutability (ConSub), and Conditional Neighborhood Substitutability (ConNSub) are defined by [Zhang and Freuder, 2004] where a problem is NI, Sub and NSub respectively given a set of conditions.

Theorem 13. (ConNI → ConI → ConSub), (ConNI → ConNSub → ConSub), and ConI and ConNSub are not comparable.

For ConNI → ConI and ConNSub → ConSub, see [Zhang and Freuder, 2004]. Consider the local forms: ConNI → ConNSub. For the same set of additional constraints, if a and b are ConNI in the original problem, they are NI in the problem with the additional constraints. Hence, they are also NSub in the problem with the additional constraints, and ConNSub in the original problem. The proof for the global forms (i.e., ConI → ConSub) is similar. Similar to the non-comparability of FI and NSub (see Theorem 5), ConI and ConNSub can be shown to be not comparable.

4 Other Forms of Interchangeability

In this section we review other forms of interchangeability that have appeared in the literature.

Neighborhood Tuple Interchangeability (NTI) [Neagu and Faltings, 1999]. Values a and b for variable v are NTI with respect to a set of variables S if for every consistent tuple t of value assignments to $S \cup \{v\}$ where $x = a$ there is another consistent tuple t' where $v = b$ such that t and t' are consistent with the same value combinations for variables outside of S. Additionally, the same condition must hold when a and b are exchanged. Figure 12 shows an example.

The algorithm proposed in [Neagu and Faltings, 2005] to detect NTI values determines the smallest set S using discrimination trees. The complexity of detecting NTI values is $\mathcal{O}((n^{s_{max}} s_{max}(n - s_{max})d^4)$, where n is the number of variables, d is the maximum domain size, and s_{max} is the maximum size of all possible dependent sets in the neighborhood of the variable.

Theorem 14. NI → NTI → PI and NTI → NPI.

First, consider NI → NTI. Given values a and b that are NI for a variable, for every consistent tuple t with a there is a tuple t' that only differs from t with a replaced with b. Hence t and t' are consistent with the same value combinations. Figure 12 gives an example where the converse does not hold. For (NTI → PI) and (NTI → NPI), see [Neagu and Faltings, 2005]. Figure 4 gives an example where PI ↛ NTI, and Figure 11 gives an example where NPI ↛ NTI.

In [2005], Wilson described a new approach to computation in a semiring-based system based on semiring-labeled decision diagrams (SLDDs). He defines forward neighborhood interchangeability (ForwNI) and uses it for merging nodes in SLDDs, hence compacting the search space. During search, ForwNI takes into account constraints that apply to instantiated and uninstantiated variables.

Forward Neighborhood Interchangeability (ForwNI) [Wilson, 2005] Given a subset of variables $U \subset V$, two assignments u and u' to a set of variables U are said to be ForwNI if for all constraints $c \in C$ such that $scope(c) \cap (V \setminus U) \neq \emptyset$ and $scope(c) \cap U \neq \emptyset$, $\Pi_{V \setminus U}\{t \in c \,|\, \Pi_U(t) = u\} = \Pi_{V \setminus U}\{t \in c \,|\, \Pi_U(t) = u'\}$.

Theorem 15. NTI → ForwNI.

If a and b for variable X are NTI with respect to set of variables S, then for every assignment t to $S \cup \{X\}$ where $X = a$ there is another consistent tuple t' where $X = b$ such that t and t' are consistent with the same value combinations for variables outside of S. Hence, the set of tuples consistent with t is the same for t' when projected on $V \setminus S$. Therefore, the assignments $\Pi_S(t)$ and $\Pi_S(t')$ are ForwNI.

Tuple substitutability is a global form of ForwNI:

Tuple Substitutability (TupSub) [Jeavons *et al.*, 1994] Two assignments A and B to a set of variables R are TupSub iff $\Pi_{V \setminus R}(\sigma_B(Sol)) \subseteq \Pi_{V \setminus R}(\sigma_A(Sol))$, where Sol is the set of all solutions to the problem.

Theorem 16. ForwNI → TupSub.

If two assignments u and u' are ForwNI, then for every solution in which u participates, u can be substituted with u' because they have the same supports in every constraint that links the scope of u to the rest of the problem. Therefore, the assignments u and u' are interchangeable and consequently substitutable.

Theorem 17. DynNI → ForwNI.

If a and b for variable X is DynNI *wrt* a set A of assignments, then any two assignments u and u' in $A \cup \{X\}$, such that $\Pi_X(u) = a$ and $\Pi_X(u') = b$, have the same set of support tuples because a and b are NI in $V \setminus A$. Therefore, u and u' are ForwNI.

Full Dynamic Substitutability (FDynSub) [Prestwich, 2004b] A value a for variable v is dynamically substitutable with value b with respect to a set A of variable assignments iff a is fully substitutable for b in the subproblem induced by A.

Theorem 18. FDynI → FDynSub, follows directly from the definition.

Theorem 19. Sub → FDynSub.

Consider value a Sub for b for variable v, the set of values supporting b is a subset of the set of values supporting a. Given an assignment of variables in FDynSub, the sets of supports will only lose elements, hence set of values supporting b will remain a subset of the set of values supporting a.

Theorem 20. FDynSub → ConNSub.

If a and b are FDynSub, then a set of constraints can be constructed for the original problem that removes all but the assigned values in the variables that are assigned in FDynSub. In the new problem resulting from adding those constraints, a and b are Sub. Moreover, a and b are NSub because all the values in the neighborhood that are not part of a solution are eliminated by the added constraints.

Theorem 21. TupSub and FDynSub are not comparable.

Context dependent interchangeability (CtxDepI) [Weigel *et al.*, 1996] Values a and b for a CSP variable X are CtxDepI, iff there exists a solution clique in the modified microstructure of the CSP that contain both nodes (X, a) and (X, b). The modified microstructure of the CSP is the original microstructure with edges added between values of the same variable. Figure 13 shows an example.

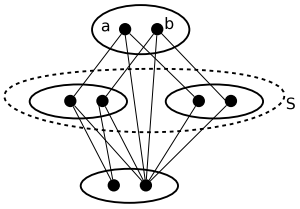

Fig. 12. NTI: a and b are NTI but not FDynNSub.

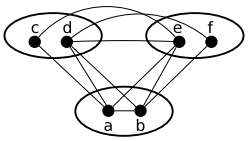

Fig. 13. CtxDepI: a and b are CtxDepI (clique $\{a, b, d, e\}$) but not Sub, FI, or PI.

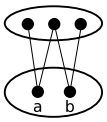

Fig. 14. GNSub: a and b are GNSub but not Sub.

Theorem 22. CtxDepI ≡ FDynI.

Connecting two CtxDepI values a and b for a CSP variable in the micro-structure of the CSP yields a solution clique with a and b. By assigning the values in the clique to the variables, we obtain an assignment set A where a and b are fully interchangeable in the subproblem induced by A. Conversely, if a and b are FDynI with respect to an assignment set A, then there is a solution clique in the modified micro-structure with a, b and all the values in A.

Theorem 23. FI → CtxDepI, FDynI.

If a and b are FI for a variable v, then a solution with $v = a$ yields another solution when replacing a with b. Thus, by connecting a and b in the micro-structure, we obtain a solution clique with a and b. Figure 13 shows an example where the converse does not hold.

Generalized Neighborhood Substitutability (GNSub) [Chmeiss and Sais, 2003] Two values of a variable are GNSub iff they share at least one support with respect to each neighboring variable. Figure 14 shows an example.

Theorem 24. NSub and GNSub are not comparable.

Figures 14 and 15 show counter examples.

Theorem 25. CtxDepI → GNSub

If a and b are CtxDepI, then the variable-value pairs connected to a in the micro-structure of the CSP are also connected to b. Thus, a and b share at least one support and are GNSub. Figure 16 shows an example where the converse does not hold.[6]

Theorem 26. GNSub → ConNI.

If a and b are GNSub for a variable, then we can construct a set of constraints that eliminates all values in the neighboring variables except the ones that are the shared support between a and b in order to make a and b ConNI. In Figure 17, the constraints that eliminate the supports of a make a and b ConNI.

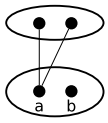

Fig. 15. NSub: a and b are NSub but not ConNI or GNSub.

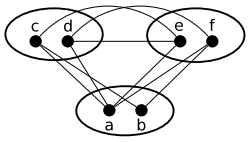

Fig. 16. GNSub: a and b are GNSub but not FDynI or CtxDepI.

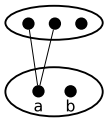

Fig. 17. ConNI: a and b are ConNI but not GNSub.

5 Relationships Between Interchangeability Concepts

The different interchangeability concepts surveyed in the previous sections are related here by the implication relation. Given two interchangeability concepts A and B, $A \rightarrow B$ if every interchangeable pair defined by A, is also defined by B. Hence, B generalizes A. Figures 18 and 19 illustrate those implication relations, and depict a partial ordering because some concepts are not comparable.

[6] Section 6.2 of [Zhang and Freuder, 2004] incorrectly states that CtxDepI and ConI are equivalent.

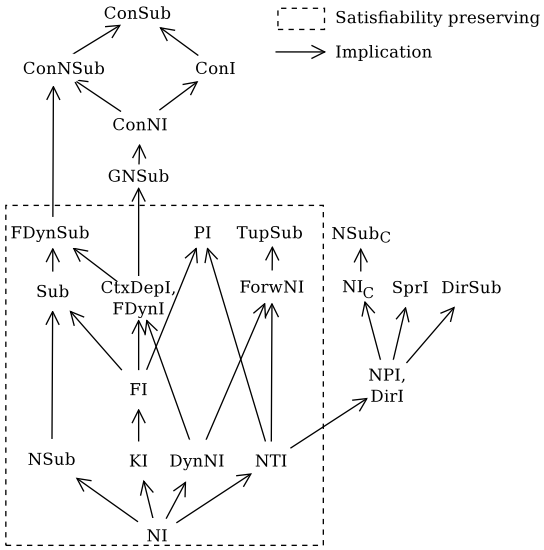

Fig. 18. Hasse Diagram of main interchangeability properties.

An interchangeability concept is *satisfiability preserving* iff when given two values a and b that are either interchangeable or a is substitutable for b, removing b from the problem does not alter the satisfiability of the problem. Not all interchangeability concepts are satisfiability preserving. Only the interchangeability concepts that are inside the dashed rectangle are satisfiability preserving.

In Figure 19, The upper horizontal plane groups concepts defined at the semantic level and thus are likely intractable. The lower horizontal plane groups concepts defined at the syntactic levels (i.e., directly on the constraints), and can likely be efficiently computed.

Interestingly, for a given form of interchangeability, when one moves vertically upward from the lower plane to the higher plane, interchangeability sets do not decrease in size while the interchangeability form is not approximated or compromised. Naturally, this advantage is not free because the computational cost does not decrease. Moving along the directed edges in either horizontal planes does not increase cost or the opportunities for interchangeability (i.e., size of interchangeability sets), but results in an approximation (i.e., weakening) of the 'quality' of the interchangeability. Finally, moving from the higher plane to the lower one allows one to likely avoid intractability and may increase the interchangeability opportunities but also results in approximations that may lose solutions.

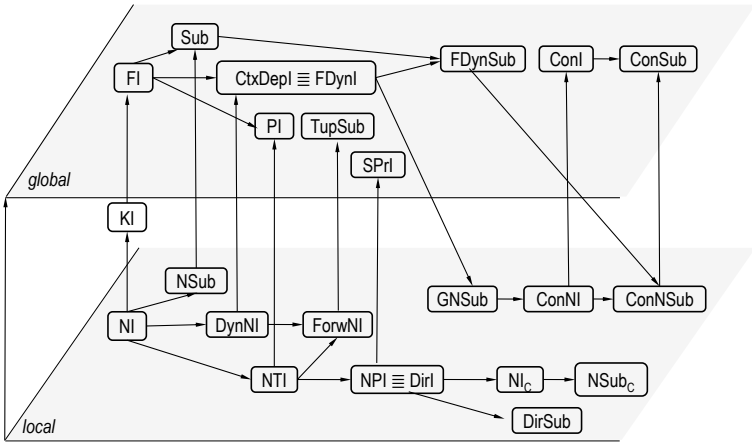

Fig. 19. Depicting qualitative relations between main interchangeability properties.

6 Work in Progress

In this paper we surveyed several forms of interchangeability, presented their definitions, and analyzed their relationship. This document is a work in progress. In the expanded version, we will address in depth the following topics:

More about interchangeability: concepts not mentioned here; missing proofs of incomparability; the satisfiability-preserving property; restrictions to constraint types; algorithms for interchangeability and their complexity; experimental results.

Beyond classical CSPs: soft constraints [Cooper, 2003; Bistarelli *et al.*, 2003; Neagu *et al.*, 2003; Neagu and Faltings, 2003]; distributed CSPs [Petcu and Faltings, 2003; Burke and Brown, 2006; Ezzahir *et al.*, 2007].

Relation to symmetry: various types [Benhamou, 1994; Cohen *et al.*, 2006]; symmetry breaking during search (SBDS) [Roney-Dougal *et al.*, 2004; Backofen and Will, 2002; Gent and Smith, 2000], symmetry breaking by dominance detection (SBDD) [Fahle *et al.*, 2001; Focacci and Milano, 2001], and symmetry breaking by enforcing variable or domain ordering [Bellicha *et al.*, 1994; Yip and Hentenryck, 2009]; restricted classes of symmetries and problems where the symmetry is broken in polynomial time [Hentenryck *et al.*, 2003; Benhamou, 2004].

Relation to search-space compaction: as in CPR [Hubbe and Freuder, 1992], AND/OR graphs [Dechter and Mateescu, 2004], SLDD [Wilson, 2005], and solution robustness [Ginsberg *et al.*, 1998; Hebrard *et al.*, 2004].

Relation to SAT solving.

Acknowledgments

This work was supported in part by Science Foundation Ireland under Grant 00/PI.1/C075. Karakashian and Woodward gratefully acknowledge the support

and hospitality of the Cork Constraint Computation Centre during Summer 2010 when this research was conducted.

References

[Backofen and Will, 2002] Rolf Backofen and Sebastian Will. Excluding Symmetries in Constraint-Based Search. *Constraints*, 7(3/4):333–349, 2002.

[Beckwith and Choueiry, 2001] Amy M. Beckwith and Berthe Y. Choueiry. On the Dynamic Detection of Interchangeability in Finite Constraint Satisfaction Problems. In *Principle and Practice of Constraint Programming (CP 01)*, volume 2239 of *LNCS*, page 760, Paphos, Cyprus, 2001.

[Beckwith *et al.*, 2001] Amy M. Beckwith, Berthe Y. Choueiry, and Hui Zou. How the Level of Interchangeability Embedded in a Finite Constraint Satisfaction Problem Affects the Performance of Search. In *AI 2001: 14th Australian Joint Conference on Artificial Intelligence*, volume 2256 of *LNAI*, pages 50–61, 2001.

[Bellicha *et al.*, 1994] Amit Bellicha, Christian Capelle, Michel Habib, Tibor Kökény, and Marie-Christine Vilarem. CSP Techniques Using Partial Orders On Domain Values. In *ECAI 1994 Workshop on Constraint Satisfaction Issues Raised by Practical Applications*, 1994.

[Benhamou, 1994] Belaid Benhamou. Study of Symmetry in Constraint Satisfaction Problems. In *Second Workshop on Principles and Practice of Constraint Programming (PPCP 94)*, pages 246–254, 1994.

[Benhamou, 2004] Belaid Benhamou. Symmetry in Not-Equals Binary Constraint Networks. In *Fourth International Workshop on Symmetry in Constraint Satisfaction Problems (SymCon 04)*, pages 2–8, 2004.

[Benson and Freuder, 1992] Brent W. Benson and Eugene C. Freuder. Interchangeability Preprocessing Can Improve Forward Checking Search. In *Tenth European Conference on Artificial Intelligence (ECAI 92)*, pages 28–30, 1992.

[Bistarelli *et al.*, 2003] Stefano Bistarelli, Boi Faltings, and Nicoleta Neagu. Interchangeability in Soft CSPs. In *Recent Advances in Constraints*, volume 2627 of *LNCS*, pages 45–68. Springer, 2003.

[Boussemart *et al.*, 2004] Frederic Boussemart, Fred Hemery, Christophe Lecoutre, and Lakhdar Sais. Support Inference for Generic Filtering. In *Principles and Practice of Constraint Programming (CP 04)*, volume 3258 of *LNCS*, pages 721–725. Springer, 2004.

[Brown *et al.*, 1988] Cynthia A. Brown, Larry Finkelstein, and Paul W. Purdom, Jr. Backtrack Searching in the Presence of Symmetry. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 357 of *LNCS*, pages 99–110. Springer, 1988.

[Burke and Brown, 2006] David A. Burke and Kenneth N. Brown. Applying Interchangeability to Complex Local Problems in Distributed Constraint Reasoning. In *Workshop on Distributed Constraint Reasoning (AAMAS 06)*, pages 1–15, 2006.

[Chmeiss and Sais, 2003] Assef Chmeiss and Lakhdar Sais. About Neighborhood Substitutability In CSPs. In *Third International Workshop on Symmetry in Constraint Satisfaction Problems (SymCon 03)*, pages 41–45, 2003.

[Choueiry and Davis, 2002] Berthe Y. Choueiry and Amy M. Davis. Dynamic Bundling: Less Effort for More Solutions. In *International Symposium on Abstraction, Reformulation and Approximation (SARA 02)*, volume 2371 of *LNAI*, pages 64–82. Springer, 2002.

[Choueiry and Noubir, 1998] Berthe Y. Choueiry and Guevara Noubir. On the Computation of Local Interchangeability in Discrete Constraint Satisfaction Problems. In *Fifteenth National Conference on Artificial Intelligence (AAAI 98)*, pages 326–333, 1998.

[Choueiry *et al.*, 1995] Berthe Y. Choueiry, Boi Faltings, and Rainer Weigel. Abstraction by Interchangeability in Resource Allocation. In *14th International Joint Conference on Artificial Intelligence (IJCAI 95)*, pages 1694–1701, 1995.

[Cohen *et al.*, 2006] David Cohen, Peter Jeavons, Christopher Jefferson, Karen E. Petrie, and Barbara M. Smith. Symmetry Definitions for Constraint Satisfaction Problems. *Constraints*, 11(2):115–137, 2006.

[Cooper, 2003] Martin C. Cooper. Reduction Operations in Fuzzy or Valued Constraint Satisfaction. *Fuzzy Sets and Systems*, 134(3):311–342, 2003.

[Dechter and Mateescu, 2004] Rina Dechter and Robert Mateescu. The Impact of AND/OR Search Spaces on Constraint Satisfaction and Counting. In *Principles and Practice of Constraint Programming (CP 04)*, volume 3258 of *LNCS*, pages 731–736, 2004.

[Ezzahir *et al.*, 2007] Redouane Ezzahir, Mustapha Belaissaoui, Christian Bessiere, and El Houssine Bouyakhf. Compilation Formulation for Asynchronous Backtracking with Complex Local Problems. In *Third International Symposium on Computational Intelligence and Intelligent Informatics (ISCIII 07)*, pages 205–211, 28-30 2007.

[Fahle *et al.*, 2001] Torsten Fahle, Stefan Schamberger, and Meinolf Sellman. Symmetry Breaking. In *Principles and Practices of Constraint Programming (CP 01)*, volume 2239 of *LNCS*, pages 93–107. Springer, 2001.

[Focacci and Milano, 2001] Filippo Focacci and Michela Milano. Global Cut Framework for Removing Symmetries. In *Seventh International Conference on Principles and Practice of Constraint Programming (CP 01)*, volume 2239 of *LNCS*, pages 77–92. Springer, 2001.

[Freuder, 1991] Eugene C. Freuder. Eliminating Interchangeable Values in Constraint Satisfaction Problems. In *National Conference on Artificial Intelligence (AAAI 91)*, pages 227–233, 1991.

[Gent and Smith, 2000] Ian P. Gent and Barbara M. Smith. Symmetry Breaking in Constraint Programming. In *Fourteenth European Conference on Artificial Intelligence (ECAI 00)*, pages 599–603. IOS Press, 2000.

[Gent *et al.*, 2006] Ian Gent, Karen Petrie, and Jean-François Puget. *Handbook of Constraint Programming*, chapter 10, pages 329–376. Elsevier, 2006.

[Ginsberg *et al.*, 1998] Matthew L. Ginsberg, Andrew J. Parkes, and Amitabha Roy. Supermodels and Robustness. In *Fifteenth National Conference on Artificial intelligence (AAAI 98)*, pages 334–339, 1998.

[Glaisher, 1874] J.W.L. Glaisher. On the Problem of the Eight Queens. *Philosophical Magazine, series 4*, 48:457–467, 1874.

[Haselböck, 1993] Alois Haselböck. Exploiting Interchangeabilities in Constraint Satisfaction Problems. In *13th International Joint Conference on Artificial Intelligence (IJCAI 93)*, pages 282–287, 1993.

[Hebrard *et al.*, 2004] Emmanuel Hebrard, Brahim Hnich, and Toby Walsh. Super Solutions in Constraint Programming. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR 04)*, volume 3011 of *LNCS*, pages 157–172. Springer, 2004.

[Hentenryck *et al.*, 2003] Pascal Van Hentenryck, Pierre Flener, Justin Pearson, and Magnus Ågren. Tractable Symmetry Breaking for CSPs with Interchangeable Values. In *18th International Joint Conference on Artificial Intelligence (IJCAI 03)*, pages 277–282, 2003.

[Hubbe and Freuder, 1992] Paul D. Hubbe and Eugene C. Freuder. An Efficient Cross Product Representation of the Constraint Satisfaction Problem Search Space. In *Tenth National Conference on Artificial Intelligence (AAAI 92)*, pages 421–427, 1992.

[Jeavons *et al.*, 1994] Peter G. Jeavons, David A. Cohen, , and Martin C. Cooper. A Substitution Operation for Constraints. In *Second Workshop on Principles and Practice of Constraint Programming (PPCP 94)*, volume 874 of *LNCS*, pages 18–25. Springer, 1994.

[Lal and Choueiry, 2004] Anagh Lal and Berthe Y. Choueiry. Constraint Processing Techniques for Improving Join Computation: A Proof of Concept. In *Proceedings of the 1^{st} International Symposium on Constraint Databases, CDB'04*, volume 3074 of *LNCS*, pages 149–167. Springer, 2004.

[Lal *et al.*, 2005] Anagh Lal, Berthe Y. Choueiry, and Eugene C. Freuder. Neighborhood Interchangeability and Dynamic Bundling for Non-Binary Finite CSPs. In *20^{th} National Conference on Artificial Intelligence (AAAI 05)*, pages 397–404, 2005.

[Likitvivatanavong and Yap, 2008] Chavalit Likitvivatanavong and Roland H.C. Yap. A Refutation Approach to Neighborhood Interchangeability in CSPs. In *AI 2008: Advances in Artificial Intelligence*, volume 5360 of *LNCS*, pages 93–103. Springer, 2008.

[Naanaa, 2007a] Wady Naanaa. Directional Interchangeability for Enhancing CSP Solving. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR 07)*, volume 4510 of *LNCS*, pages 200–213, 2007.

[Naanaa, 2007b] Wady Naanaa. Substitutability Based Domain Decomposition for Constraint Satisfaction. In *7^{th} International Workshop on Symmetry and Constraint Satisfaction Problems (SymCon 07)*, pages 64–71, 2007.

[Naanaa, 2009] Wady Naanaa. A Domain Decomposition Algorithm for Constraint Satisfaction. *Journal of Experimental Algorithmics (JEA)*, 13(1.13):1–23, 2009.

[Neagu and Faltings, 1999] Nicoleta Neagu and Boi Faltings. Constraint Satisfaction For Case Adaptation. In *Workshop on Formalisation of Adaptation in Case-Based Reasoning of ICCBR 99*, pages 35–41, 1999.

[Neagu and Faltings, 2003] Nicoleta Neagu and Boi Faltings. Soft Interchangeability for Case Adaptation. *Case-Based Reasoning Research and Development*, 2689:1066–1066, 2003.

[Neagu and Faltings, 2005] Nicoleta Neagu and Boi Faltings. Approximating Partial Interchangeability In CSP Solutions. In *International FLAIRS Conference (FLAIRS 05)*, pages 175–181, 2005.

[Neagu *et al.*, 2003] Nicoleta Neagu, Stefano Bistarelli, and Boi Faltings. On the Computation of Local Interchangeability in Soft Constraint Satisfaction Problems. In *International FLAIRS Conference (FLAIRS 03)*, pages 14–18, 2003.

[Petcu and Faltings, 2003] Adrian Petcu and Boi Faltings. Applying Interchangeability Techniques to the Distributed Breakout Algorithm. In *Principles and Practice of Constraint Programming (CP 03)*, volume 2833 of *LNCS*, pages 925–929. Springer, 2003.

[Prestwich, 2004a] Steven Prestwich. Full Dynamic Interchangeability with Forward Checking and Arc Consistency. In *Workshop on Modeling and Solving Problems With Constraints (ECAI 04)*, pages 1–14, 2004.

[Prestwich, 2004b] Steven Prestwich. Full Dynamic Substitutability by SAT Encoding. In *Principles and Practice of Constraint Programming (CP 04)*, volume 3258 of *LNCS*, pages 512–526. Springer, 2004.

[Roney-Dougal *et al.*, 2004] Colva M. Roney-Dougal, Ian P. Gent, Tom Kelsey, and Steve Linton. Tractable Symmetry Breaking Using Restricted Search Trees. In *Sixteenth European Conference on Artificial Intelligence (ECAI 04)*, pages 211–215, 2004.

[Various, 1991 present] Various. Proceedings of the International Workshop on Symmetry and Constraint Satisfaction Problems (SymCon) and of major AI Conferences such as AAAI, IJCAI, and CP, 1991 present.

[Weigel *et al.*, 1996] Rainer Weigel, Boi Faltings, and Berthe Y. Choueiry. Context in Discrete Constraint Satisfaction Problems. In *Twelfth European Conference on Artificial Intelligence (ECAI 96)*, pages 205–209, 1996.

[Weil and Heus, 1998] Georges Weil and Kamel Heus. Eliminating Interchangeable Values in the Nurse Scheduling Problem Formulated as a Constraint Satisfaction Problem. In *Workshop on Constraint-based reasoning in conjunction with FLAIRS'95*, Indianlantic, FL, 1998. Available from www.sci.tamucc.edu/constraint95/kamel.ps.

[Wilson, 2005] Nick Wilson. Decision Diagrams for the Computation of Semiring Valuations. In *International Joint Conference on Artificial Intelligence (IJCAI 05)*, pages 331–336, 2005.

[Yip and Hentenryck, 2009] Justin Yip and Pascal Van Hentenryck. Evaluation of Length-Lex Set Variables. In *Principles and Practice of Constraint Programming (CP 09)*, volume 5732 of *LNCS*, pages 817–832. Springer, 2009.

[Zhang and Freuder, 2004] Yuanlin Zhang and Eugene C. Freuder. Conditional Interchangeability and Substitutability. In *Fourth International Workshop on Symmetry and Constraint Satisfaction Problems (SymCon 04)*, 2004.