About Neighborhood Substitutability in CSPs

Assef Chmeiss and Lakhdir Saiès

CRIL - Université d’Artois - IUT de Lens
Rue de l’université - SP 16
62307 LENS Cedex, France
{chmeiss, sais}@cril.univ-artois.fr

Abstract. Constraint Satisfaction Problems are known to be NP-Complete. So, many works have been carried out in order to improve the efficiency of backtracking search algorithms. Some used techniques aim to reduce the search space by removing redundant values from domains. One of these techniques is based on the notion of interchangeability and substitutability defined by Freuder[4]. In this work, we propose an extension of the Neighborhood Substitutability. We show that Generalized Neighborhood Substitutability recover the different forms of substitutability.

1 Introduction

Constraint Satisfaction Problems (CSPs) involve the assignment of values to variables which are subject to a set of constraints. CSP is known to be NP-Complete problem. So, many works have been proposed in order to improve classical backtracking algorithms such as constraint propagation, symmetries or intelligent backtracking. The main objective in all cases is to reduce the search space and consequently to deal with harder CSPs.

One of the used techniques is based on the interchangeability and Substitutability introduced by Freuder[4]. Interchangeability can be considered as a way to take into account constraints semantic. Many works have been proposed aiming to exploit such semantical information in solving Constraint Satisfaction Problems. Benson and Freuder [2] showed that interchangeability preprocessing can improve forward checking algorithm on some random CSPs. Also, these techniques have been used to improve filtering algorithms such as Arc-Consistency [5]. Belicha et al[1] studied the notion of Neighborhood substitutability (NS) and defined a partial ordering on domains induced by NS. This ordering is used to remove some values and consequently reduce search space.

In this paper, we propose a weaker form of Neighborhood Substitutability which generalizes the notions of Neighborhood interchangeability and Neighborhood Substitutability. We call this Generalized Neighborhood Substitutability (GNS). Our intuition is that situations verifying GNS might occur more frequently than other constrained notions of Substitutability. The paper is organised as follow. Section 2 recalls some necessary definitions. In section 3, we present the theoretical framework of the GNS. Further work and investigation can be found in section 4.
2 Definitions

Definition 1. A finite binary Constraint Satisfaction Problem (CSP) \( \mathcal{P} \) is defined by the 3-tuple \((X, D, C)\)

- \(X = \{x_1, x_2, \ldots, x_n\}\) a set of \(n\) variables,
- \(D = \{D_1, D_2, \ldots, D_n\}\) a set of finite domains. \(\forall x_i \in X, D_i\) is the set of possible values for the variable \(x_i\),
- \(C = \{C_{i_1,j_1}, C_{i_2,j_2}, \ldots, C_{i_m,j_m}\}\) a set of \(m\) binary constraints. \(C_{ij} \in C, i < j\) defines the constraint between variables \(x_i\) and \(x_j\).

Note that constraints \(C\) define the set of \(m\) relations of compatibility \(R = \{R_{i_1,j_1}, R_{i_2,j_2}, \ldots, R_{i_m,j_m}\}\). \(\forall R_{ij} \in R, i < j, R_{ij} \subseteq D_i \times D_j\). In the sequel, we will use indiscriminately \(C_{ij}\) or \(R_{ij}\) for a constraint between variables \(x_i\) and \(x_j\). Also, variable \(x_i\) is sometimes denoted by its index \(i\). A constraint \(R_{ij}\) is a subset of the cartesian product \((D_i \times D_j)\) and it specifies the pairs of compatible values. In the sequel, we denote a value \(a \in D_i\) by \((i, a)\) (or \(i = a\)). Two values \((i, a)\) and \((j, b)\) are compatible if \((a, b) \in R_{ij}\). \(I_i \{\{R_{ij} \in R\}\) denotes the neighborhood of the variable \(i\).

Definition 2. We define \(N(i, j, a) = \{b \in D_j | (a, b) \in R_{ij}\}\) as the set of values which support \((i, a)\) in \(D_j\), i.e., the set of compatible values with \((i, a)\) in \(D_j\).

Definition 3. \([1]\) We define \(N(i, a) = \cup_{j \in I_i} N(i, j, a)\) as the set of values which support \((i, a)\).

We recall below some definitions about interchangeability and Substitutability introduced by Freuder in \([4]\).

Definition 4 (Substitutability (S)), \([4]\)
Given two values \((i, a)\) and \((i, b)\) from the domain \(D_i\) of a CSP \(\mathcal{P}\). The value \((i, a)\) is substitutable for \((i, b)\) iff substituting \((i, a)\) in any solution involving \((i, b)\) yields another solution.

Definition 5 (Neighborhood Substitutability (NS)), \([1]\)
For two values \((i, a)\) and \((i, b)\) from the domain \(D_i\) of a CSP \(\mathcal{P}\), \((i, a)\) is Neighborhood Substitutable for \((i, b)\) iff \(N(i, b) \subseteq N(i, a)\)

Definition 6 (Neighborhood Interchangeability (NI)), \([1]\)
Two values \((i, a)\) and \((i, b)\), from the domain \(D_i\) of a CSP \(\mathcal{P}\), are Neighborhood Interchangeable iff \(N(i, a) = N(i, b)\).

Proposition 1. Given a CSP \(\mathcal{P} = (X, D, R)\), \((i, a)\) is NS for \((i, b)\), then \(\mathcal{P}\) is satisfiable iff \(\mathcal{P}|_{D_i = \{b\}}\) (\(\mathcal{P}\) for which we delete the value \(b\) from \(D_i\)) is satisfiable.

42
3 Generalized Neighborhood Substitutability

In this section, we propose a generalization of Neighborhood Substitutability and Neighborhood Interchangeability.

Definition 7 (Generalized Neighborhood Substitutability (GNS)).
(i, a) is GNS for (i, b) if and only if \( \forall j \in \Gamma_i, N(i, j, a) \cap N(i, j, b) \neq \emptyset \)

Remark 1. Note that the GNS we propose is a weaker than NS. GNS generalizes both NI and NS. We can easily establish the relationship \( NI < NS < GNS \). In other words, if a value \((i, a)\) is NI for a value \((i, b)\) then \((i, a)\) is also NS for \((i, b)\). If \((i, a)\) is NS for \((i, b)\) then \((i, a)\) is GNS for \((i, b)\) as well. The reverse is false.

Definition 8. An assignment \( I_Y \) of the variables of \( Y = \{i_1, i_2, \ldots, i_k\} \), \( Y \subseteq X \) is an element of the cartesian product \( D_{i_1} \times D_{i_2} \times \cdots \times D_{i_k} \). It can be represented by pairs of variable-value \( \{(i_1, I_Y(i_1)), (i_2, I_Y(i_2)), \ldots, (i_k, I_Y(i_k))\} \)

Definition 9. An assignment \( I_Y \) \((Y \subseteq X)\) satisfies the constraint \( R_{ij} \) if and only if \((i, I_Y(i)), (j, I_Y(j)) \in R_{ij} \).

Definition 10. Let \( P = (X, D, C) \) be a CSP, \( I_Y \) an assignment of \( Y \subseteq X \), we define \( P_{I_Y} = (X, D', C) \) as :
- \( \forall i \in Y, D'_i = \{I_Y(i)\} \)
- \( \forall j \notin Y, D'_j = D_j \)

Proposition 2. Let \( P = (X, D, C) \) be a CSP, \( Y = \{i\} \cup \Gamma_i \), \( I_Y \) and \( I_i \) two assignments of \( Y \) such that:
- \( I_Y(i) = a \) and \( I_i(i) = b \), and
- \( \forall j \in \Gamma_i, I_Y(j) = R_i(j) = v_j \) s.t. \( v_j \in N(i, j, a) \cap N(i, j, b) \),

if \((i, a)\) is GNS for \((i, b)\) then \( P_{I_Y} \) is satisfiable if and only if \( P_{I_i} \) is satisfiable.

Obviously, the proposition 2 shows that two GNS values induce search redundancy. In the following, we show how to avoid such redundancies by adding additional constraints to the original CSP. These additional constraints can be represented in clausal form (Clausal Constraint Satisfaction Problems)[3]. The obtained CSP is equivalent for satisfiability to the original one. We call them Constraints Avoiding Redundancy (CAR).

Definition 11. Let \( P = (X, D, C) \) be a CSP. We define total ordering \( O \) on domains as: \( \forall i \in X, \text{and} \forall i_1, i_2 \in \{1, \ldots, |D_i|\}, \text{if} \ i_1 < i_2 \text{ then} \ v_{i_1} < v_{i_2} \). Note that \( v_{i_1} \) and \( v_{i_2} \) correspond to the values in \( D_i \) with ranks \( i_1 \) and \( i_2 \) respectively.

Definition 12. Let \( P = (X, D, C) \) be a CSP, \( O \) a total ordering on its domains. We define Constraint Avoiding Redundancy as: \( CAR = \bigwedge_{i \in X, v_k \in O, v_l \in O} (i \neq v_k \land \cdots \land i \neq v_l) \rightarrow \bigvee_{k \in \Gamma_i} v_k \notin N(i, k, a) \), \( \bigwedge \) (resp. \( \bigvee \)) is the conjunctive (resp. disjunctive) logical operator. The \( \rightarrow \) symbol designs the implication operator.
Remark 2. Constraints Avoiding Redundancy defined above can be represented in clausal form as follow:

\[ CAR = \bigwedge_{i \in X, k \in \{1, \ldots, |D_i|\}} (\forall i \in I_i (i = v_1 \lor \ldots \lor i = v_l \lor i \notin N(i, k, v_l))) \]

\( CAR \) indicate that if no solution exists for a value \( v_l \in D_i, i \in \{1, \ldots, |D_i|\} \) then for a value \( v_m \in D_i \) with \( v_m \neq v_l \) it is useless to explore tuples of values in subdomains \( N(i, k, v_l) \), for all \( k \in I_i \).

**Proposition 3.** Let \( P = (X, D, C) \) be a CSP, \( CAR \) the set of Constraints Avoiding Redundancy in \( P \). Then \( P \) is consistent iff \( P \land CAR \) is consistent.

Proposition 3 shows that, by adding some constraints to the original CSP, we can avoid redundant search for a solution.

Remark 3. We can easily remark that added redundant constraints to \( P \) allow us to avoid redundancy related to several forms of Substitutability and Interchangeability (i.e., GNS, NS and N1).

Adding these constraints can be performed in polynomial time using polynomial space complexity. The size of \( CAR \) can be bound by \( O(n \times d) \) where \( n \) is the number of variable and \( d \) is the maximum domain size.

In practice, we can avoid adding additional constraints. In fact, when the neighborhood of a variable \( X_i \) is completely assigned, it is sufficient to verify if there exists a value \( v_l \in D_i \) (already checked) compatible with the values of the current assignment in the neighborhood of \( X_i \) and backtrack in such a case.

### 4 Conclusion and further work

In this paper, we have proposed a generalization of neighborhood interchangeability and substitutability. We have shown how redundancy can be avoided thanks to additional constraints which allow us to recover these different forms of substitutability. We have discussed only the theoretical framework. We envisage to investigate several aspects related to the Generalized Neighborhood Substitutability. We give below some future directions to be explored:

- Experimental validation of the GNS as a preprocessing step or during search (without additional constraints). Also, we plan to investigate its usefulness on different kinds of problems such as structured CSPs and random ones.
- Development of specific heuristics to help exploiting GNS. We plan to define heuristics which lead search to assign the neighborhood of the current variable before choosing new one using well known heuristics.
- Handling efficiently the Constraints Avoiding Redundancy.
- Defining new filtering techniques for clausal CSPs.
References


