(Smart) Look-Ahead Arc Consistency and the Pursuit of CSP Tractability

Hubie Chen[1] and Víctor Dalmau[2]

[1] Department of Computer Science
Cornell University
Ithaca, NY 14853, USA
hubes@cs.cornell.edu

[2] Departament de Tecnologia
Universitat Pompeu Fabra
Barcelona, Spain
victor.dalmau@upf.edu

Abstract. The constraint satisfaction problem (CSP) can be formulated as the problem of deciding, given a pair $(\mathbf{A}, \mathbf{B})$ of relational structures, whether or not there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Although the CSP is in general intractable, it may be restricted by requiring the "target structure" $\mathbf{B}$ to be fixed; denote this restriction by $\mathsf{CSP}(\mathbf{B})$. In recent years, much effort has been directed towards classifying the complexity of all problems $\mathsf{CSP}(\mathbf{B})$. The acquisition of $\mathsf{CSP}(\mathbf{B})$ tractability results has generally proceeded by isolating a class of relational structures $\mathbf{B}$ believed to be tractable, and then demonstrating a polynomial-time algorithm for the class. In this paper, we introduce a new approach to obtaining $\mathsf{CSP}(\mathbf{B})$ tractability results: instead of starting with a class of structures, we start with an algorithm called *look-ahead arc consistency*, and give an algebraic characterization of the structures solvable by our algorithm. This characterization is used both to identify new tractable structures and to give new proofs of known tractable structures.

1 Introduction

1.1 Background

Constraint satisfaction problems arise in a wide variety of domains, such as combinatorics, logic, algebra, and artificial intelligence. An instance of the constraint satisfaction problem (CSP) consists of a set of variables and a set of constraints on those variables; the goal is to decide whether or not there is an assignment to the variables satisfying all of the constraints. It is often convenient to cast the CSP as a relational homomorphism problem, namely, the problem of deciding, given a pair $(\mathbf{A}, \mathbf{B})$ of relational structures, whether or not there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. In this formalization, each relation of $\mathbf{A}$ contains tuples of variables that are constrained together, and the corresponding relation of $\mathbf{B}$ contains the allowable tuples of values that the variable tuples may take.

The CSP is NP-complete in general, motivating the search for polynomial-time tractable cases of the CSP. A particularly useful way to restrict the CSP in order to obtain tractable cases is to restrict the types of constraints that may be expressed, by requiring the "target structure" $\mathbf{B}$ to be fixed; denote this restriction by $\mathsf{CSP}(\mathbf{B})$. This form of restriction can capture and place into a unified framework many particular cases of the CSP that have been independently investigated – for instance, the HORN SATISFIABILITY, 2-SATISFIABILITY, and GRAPH H-COLORABILITY problems. Schaefer was the first to consider the class of problems $\mathsf{CSP}(\mathbf{B})$; he proved a now famous dichotomy theorem, showing that for all relational structures $\mathbf{B}$ over a two-element universe, $\mathsf{CSP}(\mathbf{B})$ is either tractable in polynomial-time, or is NP-complete [17]. In recent years, much effort has been directed towards the program of classifying the complexity of $\mathsf{CSP}(\mathbf{B})$ for all relational structures $\mathbf{B}$ over a finite universe; impressive progress has been made along these lines, including the papers [11, 14, 12, 13, 9, 16, 15, 5, 2–4, 6, 1]. This research program has developed a rich set of tools for studying CSP complexity, which draws on and establishes connections to a diversity of areas, including artificial intelligence, universal algebra, database theory, logic, and group theory. The connection between CSP complexity and universal algebra [14, 12, 5] has been particularly fruitful and has been used heavily to obtain many of the recent results on $\mathsf{CSP}(\mathbf{B})$ complexity. A notion used to establish this connection is that of *invariance of a relational structure under an operation*; roughly speaking, a relational structure is invariant under an operation if the relations of the structure satisfy a certain closure property defined in terms of the operation.

A central component of the $\mathsf{CSP}(\mathbf{B})$ classification program is to identify those structures $\mathbf{B}$ such that $\mathsf{CSP}(\mathbf{B})$ is tractable. Indeed, Bulatov and Jeavons [7] have identified a plausible conjecture as to which structures $\mathbf{B}$ are tractable; in particular, they conjecture that a known necessary condition for tractability is also sufficient. This conjecture has been verified for some large classes of structures $\mathbf{B}$; see [2] and [4].

The acquisition of $\mathsf{CSP}(\mathbf{B})$ tractability results has, almost exclusively, proceeded by isolating an easily describable condition believed to guarantee tractability, and then demonstrating an algorithm that decides $\mathsf{CSP}(\mathbf{B})$ for all $\mathbf{B}$ satisfying the condition [11, 14, 13, 9, 5, 8, 2–4, 6, 1].[3] As an example, in [8], the class of relational structures invariant under a *commutative conservative* operation is demonstrated to be tractable by an algorithm known as *3-minimality*. A simple algorithm can be given for deciding whether or not a relational structure is invariant under a commutative conservative operation; on the other hand, the meta-question of deciding, given a relational structure $\mathbf{B}$, whether or not 3-minimality is a solution procedure for $\mathsf{CSP}(\mathbf{B})$, is not known to be decidable. Put succinctly, this result (and many others) demonstrate that a well-characterized

[3] A notable exception is the paper [9], which gives an algebraic characterization of those relational structures for which *establishing arc consistency* is a solution procedure.

class of relational structures is tractable, via an algorithm that is *not* well-characterized.

1.2 Contributions of this paper

In this paper, we introduce a radically different approach to the acquisition of $\mathsf{CSP}(\mathbf{B})$ tractability results. Instead of taking as our starting point a well-characterized class of relational structures, we begin with an algorithm for constraint satisfaction, and prove that this algorithm is well-characterized. In particular, we introduce a polynomial time algorithm which we call *look-ahead arc consistency* (LAAC), and show that those structures $\mathbf{B}$ for which LAAC decides $\mathsf{CSP}(\mathbf{B})$ are exactly those structures satisfying a simple algebraic criterion; this algebraic characterization can be readily translated to a decision procedure for the meta-question of deciding whether or not LAAC is a solution procedure for a given relational structure.[4] We then use this algebraic characterization to give a new class of tractable relational structures that is described using the algebraic notion of invariance under an operation. We hope that our work will inspire further research devoted to giving algebraic characterizations of algorithms similar to our characterization, and that such research will stimulate an interplay between our new approach of studying well-characterized algorithms, and the classical approach of studying well-characterized problem restrictions.

In addition to containing relational structures that have not been previously observed to be tractable, the new class of relational structures that we identify also contains structures that can be shown to be tractable by known results. The fact that these latter structures are contained in our new class does not yield any new information from the standpoint of classifying $\mathsf{CSP}(\mathbf{B})$ problems as either tractable of NP-complete; nonetheless, we believe this fact to be interesting for two reasons. First, it implies that these relational structures can be solved using a new algorithm (namely, LAAC) which we believe to be conceptually simpler than previously given algorithms for the structures; second, it gives an alternative proof of the tractability of such structures. We mention that the LAAC algorithm is a solution method for the well-known 2-SATISFIABILITY problem; this was, in fact, observed first in [10] for a specialization of the LAAC algorithm to inputs that are boolean formulas in conjunctive normal form.

We study both LAAC and an extension of LAAC which we call *smart look-ahead arc consistency* (SLAAC); like LAAC, the SLAAC algorithm runs in polynomial time. We show that there is an algebraic characterization, similar to that for LAAC, of the relational structures for which SLAAC is a solution procedure. We also demonstrate that SLAAC is a solution procedure for a fragment of the class of relational structures invariant under a commutative conservative operation. As mentioned, this class of structures has previously been shown to be tractable [8]; however, we believe our new demonstration of tractability to

[4] Amusingly, LAAC is itself a decision procedure for this meta-question: the algebraic criterion concerns the existence of a relational homomorphism to $\mathbf{B}$, and can be cast as an instance of $\mathsf{CSP}(\mathbf{B})$.

be interesting for the same two reasons (given above) that our rederivation of known tractability results via LAAC is interesting.

We wish to emphasize that, from our viewpoint, the novelty of this paper lies in our ability to provide, for each of the two algorithms, an exact characterization of the problems $\mathsf{CSP}(\mathbf{B})$ for which the algorithm is a solution procedure–and not in the sophistication of the algorithms (which are actually quite simple). Such exact characterizations of algorithms are not in general known, and we regard the ideas in this paper as a starting point for developing exact characterizations for algorithms of higher sophistication.

In what remains of this section, we give a brief overview of the LAAC and SLAAC algorithms. Both of these algorithms make use of *arc consistency*, a localized inference mechanism studied heavily by the constraint satisfaction community. Arc consistency generalizes *unit propagation*, the process of removing unit clauses from boolean formulas. Any instance of the CSP can be efficiently "tightened" into an equivalent instance (that is, an instance having the same satisfying homomorphisms) that is arc consistent. This tightening provides a sort of one-sided satisfiability check: if the second instance has an empty constraint, then the original instance is unsatisfiable; however, the converse does not hold in general. When the second instance does not have any empty constraints, we say that *arc consistency can be established* on the original instance. The *look-ahead arc consistency* algorithm is the following:

- Arbitrarily pick a variable a, and set E to be the empty set.
- For all values b of the target structure $\mathbf{B}$, substitute b for a, and attempt to establish arc consistency; if arc consistency can be established, place b into the set E.
- If E is empty, then output "unsatisfiable"; otherwise, arbitrarily pick a value $b \in E$ and set the variable a to b.
- Repeat.

It can be seen that the algorithm proceeds by picking a variable a, and then constructing a "filtered" set E of possible values for a; it then commits a to any one of the values inside the filtered set (presuming that it is non-empty), and continues. Notice that it never makes sense to set the variable a to a value outside of its "filtered" set E, since if arc consistency cannot be established when value b is substituted for a, then there is no satisfying assignment mapping a to b. This algorithm is quite simple: indeed, up to two arbitrary choices (choice of variable and choice of value) can be made in each iteration, and the only conceptual primitive used other than picking and setting variables is arc consistency. In the "smart" version of look-ahead arc consistency, one of these arbitrary choices is eliminated: the value for a variable a is picked from the set E by applying a set function to E. Despite the simplicity of LAAC and SLAAC, the class of relational structures that are tractable via these algorithms is surprisingly rich.

Note that, due to space limitations, proofs are included in an appendix.

2 Preliminaries

Our definitions and notation are fairly standard, and similar to those used in other papers on constraint satisfaction. A *relation* over a set A is a subset of A^k (for some $k \geq 1$), and is said to have *arity* k. When R is a relation of arity k and $i_1, \ldots, i_l \in \{1, \ldots, k\}$, we use $\mathsf{pr}_{i_1,\ldots,i_l} R$ to denote the arity l relation $\{(a_{i_1}, \ldots, a_{i_l}) : (a_1, \ldots, a_k) \in R\}$.

Relational structures. A *vocabulary* σ is a finite set of relation symbols, each of which has an associated arity. A *relational structure* $\mathbf{A}$ (over vocabulary σ) consists of a universe A and a relation $R^{\mathbf{A}}$ over A for each relation symbol R of σ, such that the arity of $R^{\mathbf{A}}$ matches the arity associated to R by σ. In this paper, we will only be concerned with relational structures having finite universe. Throughout, we will use bold capital letters $\mathbf{A}, \mathbf{B}, \ldots$ to denote relational structures, and the corresponding non-bold capital letters $A, B, \ldots$ to denote their universes.

When $\mathbf{A}$ and $\mathbf{B}$ are relational structures over the same vocabulary σ, we define $\mathbf{A} \times \mathbf{B}$ to be the relational structure with universe $A \times B$ such that for each relation symbol R of σ, it holds that $R^{\mathbf{A} \times \mathbf{B}} = \{((a_1, b_1), \ldots, (a_k, b_k)) : (a_1, \ldots, a_k) \in R^{\mathbf{A}}, (b_1, \ldots, b_k) \in R^{\mathbf{B}}\}$, where k denotes the arity of R. When $\mathbf{A}$ and $\mathbf{B}$ are relational structures over the same vocabulary σ, we define $\mathbf{A} \cup \mathbf{B}$ to be the relational structure with universe $A \cup B$ such that for each relation symbol R of σ, $R^{\mathbf{A} \cup \mathbf{B}} = R^{\mathbf{A}} \cup R^{\mathbf{B}}$.

When $\mathbf{A}$ and $\mathbf{B}$ are relational structures over the same vocabulary σ, a *homomorphism* from $\mathbf{A}$ to $\mathbf{B}$ is a mapping $h : A \to B$ from the universe A of $\mathbf{A}$ to the universe B of $\mathbf{B}$ such that for every relation symbol R of σ and every tuple $(a_1, \ldots, a_k) \in R^{\mathbf{A}}$, it holds that $(h(a_1), \ldots, h(a_k)) \in R^{\mathbf{B}}$. Two relational structures $\mathbf{A}$ and $\mathbf{B}$ over the same vocabulary are said to be *homomorphically equivalent* if there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ and a homomorphism from $\mathbf{B}$ to $\mathbf{A}$.

Constraint satisfaction. The constraint satisfaction problem, denoted by CSP, is defined to be the problem of deciding, given as input a pair of relational structures $(\mathbf{A}, \mathbf{B})$ over the same vocabulary, whether or not there exists a homomorphism from the *source structure* $\mathbf{A}$ to the *target structure* $\mathbf{B}$. When $(\mathbf{A}, \mathbf{B})$ is a CSP instance, we will say that $(\mathbf{A}, \mathbf{B})$ is *satisfiable* if there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$; and, we will at times refer to elements of the universe of the source structure $\mathbf{A}$ as *variables*. The $\mathsf{CSP}(\mathbf{B})$ problem is the restriction of the CSP problem where the target structure must be $\mathbf{B}$.

When $(\mathbf{A}, \mathbf{B})$ is a CSP instance over the vocabulary σ, $a \in A$, and $b \in B$, we let $(\mathbf{A}, \mathbf{B})[a = b]$ denote the CSP instance obtained from $(\mathbf{A}, \mathbf{B})$ in the following way: for each relation symbol R of σ, remove each tuple $(a_1, \ldots, a_k) \in R^{\mathbf{A}}$ such that $a \in \{a_1, \ldots, a_k\}$ and add the tuple $(a_{i_1}, \ldots, a_{i_l})$, where $i_1 < \cdots < i_l$ and $\{i_1, \ldots, i_l\} = \{i : a_i \neq a\}$, to $S^{\mathbf{A}}$, where S a relation symbol such that $S^{\mathbf{B}} = \mathsf{pr}_{i_1,\ldots,i_l}\{(b_1, \ldots, b_k) \in R^{\mathbf{B}} : \forall i \in \{i : a_i = a\}, b_i = b\}$; if necessary, extend the vocabulary σ and the relational structure $\mathbf{B}$ so that there is such a relation symbol S. Intuitively, $(\mathbf{A},\mathbf{B})[a=b]$ is the CSP instance obtained by setting the variable a to the value b; a mapping h is a homomorphism for $(\mathbf{A},\mathbf{B})$ sending a to b if and only if the restriction of h to $A \setminus \{a\}$ is a homomorphism for $(\mathbf{A},\mathbf{B})[a=b]$. We say that a relational structure $\mathbf{B}$ over vocabulary σ *permits constant instantiation for* C if $C \subseteq B$ and the following two properties hold: (1) for every $b \in C$, there exists a relation symbol R_b of σ such that $R_b^{\mathbf{B}} = \{(b)\}$; and, (2) for every relation symbol R of σ, index subset $\{i_1,\ldots,i_l\} \subseteq \{1,\ldots,k\}$ (where k denotes the arity of R and $i_1 < \cdots < i_l$), and $b \in C$, there exists a relation symbol S such that $S^{\mathbf{B}} = \mathsf{pr}_{i_1,\ldots,i_l}\{(b_1,\ldots,b_k) : \forall i \notin \{i_1,\ldots,i_l\}, b_i = b\}$. The key feature of this notion is that if $(\mathbf{A},\mathbf{B})$ is an instance of $\mathsf{CSP}(\mathbf{B})$ and $\mathbf{B}$ permits constant instantiation for C, then for any $a \in A$ and $b \in C$, the instance $(\mathbf{A},\mathbf{B})[a=b]$ is an instance of $\mathsf{CSP}(\mathbf{B})$. If the relational structure $\mathbf{B}$ permits constant instantiation for its entire universe, we simply say that $\mathbf{B}$ *permits constant instantiation*.

An equivalent way of formulating the constraint satisfaction problem is as follows. Define a *constraint* to be an expression of the form $R(a_1,\ldots,a_k)$ where R is a relation of arity k over a finite set B, and each a_i is a variable taken from a finite variable set A. Let us say that an assignment $h : A \to B$ *satisfies* the constraint $R(a_1,\ldots,a_k)$ if $(h(a_1),\ldots,h(a_k)) \in R$. With these definitions in hand, we can provide a constraint-based formulation of the CSP: given a finite set of constraints – all of which have relations over the same set B, and variables from the variable set A – decide whether or not there is an assignment $h : A \to B$ satisfying all of the constraints. When $\mathbf{A}$ and $\mathbf{B}$ are relational structures over the same vocabulary σ, the instance $(\mathbf{A},\mathbf{B})$ of the CSP problem as formulated above can be translated into this formulation by creating, for each relation R of σ and for each tuple $(a_1,\ldots,a_k)$ of $R^{\mathbf{A}}$, a constraint $R^{\mathbf{B}}(a_1,\ldots,a_k)$. It is straightforward to verify that an assignment $h : A \to B$ satisfies all such constraints if and only if it is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.

Arc consistency. Suppose that $\mathcal{C}$ is a set of constraints, that is, an instance of the constraint-based CSP. We say that $\mathcal{C}$ is *arc consistent* if for any two constraints $R(a_1,\ldots,a_k)$, $R'(a'_1,\ldots,a'_{k'})$ in $\mathcal{C}$, if $a_i = a'_j$, then $\mathsf{pr}_i R = \mathsf{pr}_j R'$. Any CSP instance $\mathcal{C}$ can be transformed, in polynomial time, into an arc consistent CSP instance that is equivalent in that it has exactly the same satisfying assignments. This is done by continually looking for constraints $R(a_1,\ldots,a_k)$, $R'(a'_1,\ldots,a'_{k'})$ and i,j such that $a_i = a'_j$ and $\mathsf{pr}_i R \neq \mathsf{pr}_j R'$, and replacing R by $\{(a_1,\ldots,a_k) \in R : a_i \in \mathsf{pr}_j R'\}$, and R' by $\{(a'_1,\ldots,a'_{k'}) \in R' : a_j \in \mathsf{pr}_i R\}$.

The procedure of transforming a CSP instance $\mathcal{C}$ into an equivalent arc consistent instance $\mathcal{C}'$ gives a sort of one-sided satisfiability check. Specifically, if the second instance $\mathcal{C}'$ has an empty relation, we can immediately conclude that the original instance $\mathcal{C}$ is unsatisfiable. (This is because no assignment can satisfy a constraint having empty relation.) Note, however, that the converse does not hold: if an instance $\mathcal{C}$ is unsatisfiable, it does not (in general) follow that an equivalent arc-consistent instance $\mathcal{C}'$ contains an empty relation.

When $\mathbf{B}$ is a relational structure over vocabulary σ, define $\mathcal{P}(\mathbf{B})$ to be the relational structure over σ having universe $\wp(B) \setminus \{\emptyset\}$ and such that for every relation symbol R of σ, $R^{\mathcal{P}(\mathbf{B})} = \{(\mathsf{pr}_1 R', \ldots, \mathsf{pr}_k R') : R' \subseteq R^{\mathbf{B}}, R' \neq \emptyset\}$, where k denotes the arity of R. (By $\wp(B)$, we denote the power set of B.) Let $(\mathbf{A}, \mathbf{B})$ be an instance of the CSP, and let $\mathcal{C}$ be the corresponding equivalent instance in the constraint-based CSP formulation. When the above arc consistency procedure can be applied to the instance $\mathcal{C}$ to obtain an equivalent arc consistent instance $\mathcal{C}'$ that does not have an empty relation, there exists a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$.[5] Moreover, when there is a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$, the above arc consistency procedure terminates without introducing an empty relation. In light of these facts, when $(\mathbf{A}, \mathbf{B})$ is a CSP instance, we say that *arc consistency can be established on* $(\mathbf{A}, \mathbf{B})$ if there exists a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$.

Invariance. A powerful algebraic theory for studying the complexity of $\mathsf{CSP}(\mathbf{B})$ problems was introduced in [14, 12]. We briefly recall the key definitions of this theory that will be used in this paper, referring the reader to [14, 12] for a detailed treatment. To every relational structure $\mathbf{B}$, we can associate a class of functions, called the *polymorphisms* of $\mathbf{B}$, and denoted by $\mathsf{Pol}(\mathbf{B})$. A function $f : B^k \to B$ is a polymorphism of $\mathbf{B}$ if f is a homomorphism from $\mathbf{B}^k$ to $\mathbf{B}$; when this holds, we also say that $\mathbf{B}$ is *invariant* under f. An important fact concerning the polymorphisms of a relational structure is that the complexity of $\mathsf{CSP}(\mathbf{B})$ depends only on $\mathsf{Pol}(\mathbf{B})$. Precisely, if $\mathbf{B}_1$ and $\mathbf{B}_2$ are relational structures with the same universe such that $\mathsf{Pol}(\mathbf{B}_1) = \mathsf{Pol}(\mathbf{B}_2)$, then $\mathsf{CSP}(\mathbf{B}_1)$ and $\mathsf{CSP}(\mathbf{B}_2)$ are reducible to each other via polynomial-time many-one reductions [12]. This tight connection between the polymorphisms of $\mathbf{B}$ and the complexity of $\mathsf{CSP}(\mathbf{B})$ permits the use of sophisticated algebraic tools in the quest to understand the complexity of $\mathsf{CSP}(\mathbf{B})$, for all relational structures $\mathbf{B}$ [14, 12, 5].

3 Algorithms

In this section, we present the look-ahead arc consistency algorithm and the smart look-ahead arc consistency algorithm; and, we give purely algebraic characterizations of the class of relational structures solvable by each of the algorithms.

We begin by giving a formal description of look-ahead arc consistency.

Algorithm 1 *Look-Ahead Arc Consistency (*LAAC*).*
Input: a CSP *instance* $(\mathbf{A}, \mathbf{B})$.

- *Initialize h to be an empty mapping, that is, a mapping with empty set as domain.*
- *While the universe of* $\mathbf{A}$ *is non-empty:*

[5] We can define a homomorphism $h : \mathbf{A} \to \mathcal{P}(\mathbf{B})$ as follows. For any element a of A, let $R'(a_1, \ldots, a_{k'})$ be any constraint in $\mathcal{C}'$ such that $a'_i = a$, and define $h(a) = \mathsf{pr}_i R'$. By the definition of arc consistency, the definition of h on a is independent of the choice of constraint. It is straightforward to verify that h is a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$.

- *Arbitrarily pick a variable a from the universe of $\mathbf{A}$.*
- *Compute the set E of values b (from the universe of $\mathbf{B}$) such that arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = b]$.*
- *If the set E is empty:*
  - *Output "unsatisfiable" and terminate.*
- *Else:*
  - *Arbitrarily pick a value b from E, and extend h to map a to b.*
  - *Replace $(\mathbf{A}, \mathbf{B})$ with $(\mathbf{A}, \mathbf{B})[a = b]$.*

– *Output the mapping h and terminate.*

When $\mathbf{B}$ is a relational structure, we will say that LAAC is a solution procedure for CSP($\mathbf{B}$) if for every instance $(\mathbf{A}, \mathbf{B})$ of CSP($\mathbf{B}$), when there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$, the LAAC algorithm outputs such a homomorphism. Notice that if there is no homomorphism from $\mathbf{A}$ to $\mathbf{B}$, the LAAC algorithm will always output "unsatisfiable" on the CSP instance $(\mathbf{A}, \mathbf{B})$.

We now give an algebraic characterization of those relational structures $\mathbf{B}$ for which LAAC solves CSP($\mathbf{B}$). Our algebraic characterization has a particularly simple form, namely, it concerns the existence of a homomorphism from $\mathcal{L}(\mathbf{B})$, a relational structure derivable from $\mathbf{B}$, to $\mathbf{B}$ itself.

The relational structure $\mathcal{L}(\mathbf{B})$ is defined as follows. For a relational structure $\mathbf{B}$ with universe B, let $L(B)$ to be the set $(\{S \in \wp(B) : |S| > 1\} \times B) \cup \{(\{s\}, \star) : s \in B\}$, where $\star$ is assumed to not be an element of B. Define $\mathbf{c_B} : (\wp(B) \setminus \{\emptyset\}) \times B \to L(B)$ to be the (surjective) mapping such that $\mathbf{c_B}(S, b) = (S, \star)$ if $|S| = 1$, and $\mathbf{c_B}(S, b) = (S, b)$ if $|S| > 1$. Intuitively, the mapping $\mathbf{c_B}$ "collapses" together elements of $\mathcal{P}(\mathbf{B}) \times \mathbf{B}$ that share the same singleton set in their first coordinate. Define $\mathcal{L}(\mathbf{B})$ to be the relational structure, over the same vocabulary σ as $\mathbf{B}$, having universe $L(B)$ and such that for all relation symbols R of σ, $R^{\mathcal{L}(\mathbf{B})} = \{(\mathbf{c_B}(t_1), \ldots, \mathbf{c_B}(t_k)) : (t_1, \ldots, t_k) \in R^{\mathcal{P}(\mathbf{B}) \times \mathbf{B}}\}$, where k denotes the arity of R. Clearly, $\mathbf{c_B}$ is a homomorphism from $\mathcal{P}(\mathbf{B}) \times \mathbf{B}$ to $\mathcal{L}(\mathbf{B})$; one can think of $\mathcal{L}(\mathbf{B})$ as a "collapsed" version of $\mathcal{P}(\mathbf{B}) \times \mathbf{B}$.

Theorem 1. *Let $\mathbf{B}$ be a relational structure (with universe B). The* LAAC *algorithm is a solution procedure for* CSP($\mathbf{B}$) *if and only if there exists a homomorphism $l : \mathcal{L}(\mathbf{B}) \to \mathbf{B}$ such that $l(\{b\}, \star) = b$ for all $b \in B$.*

Corollary 2. *The class of relational structures $\mathbf{B}$, having the property that the* LAAC *algorithm is a solution procedure for* CSP($\mathbf{B}$)*, is decidable.*

We now turn to the *smart look-ahead arc consistency* (SLAAC) algorithm. The difference between the LAAC and SLAAC algorithms is that the SLAAC algorithm takes as a parameter a set function $f : \wp(B) \setminus \{\emptyset\} \to B$, and when setting a variable, instead of arbitrarily picking from the set E of candidate values, directly sets the variable to $f(E)$. Note that for any relational structure $\mathbf{B}$, if LAAC is a solution procedure for $\mathbf{B}$, then SLAAC is also a solution procedure for $\mathbf{B}$, when parameterized with any set function that is conservative in the sense that $f(S) \in S$ (for all non-empty $S \subseteq B$).

The algebraic characterization for the SLAAC algorithm, as with that for the LAAC algorithm, concerns the existence of certain homomorphisms. Before giving this characterization, it is necessary to introduce the following auxiliary notion. For a relational structure $\mathbf{B}$, we define a pair (U, b), where $U \subseteq B$ and $b \in U$, to be *usable* if there exists a satisfiable CSP instance $(\mathbf{A}, \mathbf{B})$ and there exists a variable $a \in A$ such that

$$U = \{u \in B : \text{ arc consistency can be established on } (\mathbf{A}, \mathbf{B})[a = u]\}$$

and there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ sending a to b.

Theorem 3. *Let $f : \wp(B) \setminus \{\emptyset\} \to B$ be a set function on the set B, and let $\mathbf{B}$ be a relational structure with universe B permitting constant instantiation for the image of f. The* SLAAC *algorithm, parameterized with f, is a solution procedure for* CSP($\mathbf{B}$) *if and only if for every usable pair (U, b) (of $\mathbf{B}$), there exists a homomorphism $h_{U,b} : \mathcal{P}(\mathbf{B})^k \times \mathbf{B} \to \mathbf{B}$ such that $h_{U,b}(\{b_1\}, \ldots, \{b_k\}, b) = f(U)$, where $b_1, \ldots, b_k$ denote the elements of U.*

As for LAAC, we can show that the class of structures that are solvable by SLAAC is decidable. This follows from the algebraic characterization given in the previous theorem, along with the following theorem.

Theorem 4. *Let $\mathbf{B}$ be a relational structure, and let (U, b) be a pair where $U \subseteq B$ and $b \in U$. The problem of determining whether or not (U, b) is usable, is decidable.*

Corollary 5. *The class of relational structures $\mathbf{B}$, having the property that the* SLAAC *algorithm is a solution procedure for* CSP($\mathbf{B}$)*, is decidable.*

We can employ Theorem 3 to demonstrate that the SLAAC algorithm solves CSP($\mathbf{B}$) when the relational structure $\mathbf{B}$ is invariant under a set function. Set functions were studied in the context of CSP complexity in [9]; we say that a relational structure $\mathbf{B}$ is invariant under a set function $f : (\wp(B) \setminus \{\emptyset\}) \to B$ if it is invariant under all of the functions $f_k : B^k \to B$ defined by $f_k(b_1, \ldots, b_k) = f(\{b_1, \ldots, b_k\})$, for $k \geq 1$.

Theorem 6. *Suppose that the relational structure $\mathbf{B}$ (with universe B) is invariant under a set function $f : (\wp(B) \setminus \{\emptyset\}) \to B$. Then, the* SLAAC *algorithm is a solution procedure for* CSP($\mathbf{B}$).

4 Tractability

In this section, we present our primary tractability result, which shows that those relational structures invariant under a certain type of ternary function are tractable via the LAAC algorithm. We also demonstrate that the relational structures for which LAAC is a solution procedure can be combined in certain ways.

Theorem 7. *Suppose that* $t : B^3 \to B$ *is a ternary function satisfying the three identities* $f(x,x,y) = x$, $f(x,y,z) = f(y,x,z)$, *and* $f(f(x,y,w),z,w) = f(x,f(y,z,w),w)$. *For any relational structure* $\mathbf{B}$ *(with universe* B*) invariant under* t, *the* LAAC *algorithm is a solution procedure for* $\mathsf{CSP}(\mathbf{B})$.

One way to view the three identities given in the statement of Theorem 7 is that they require, for each fixed $b \in B$, that the binary operation $g_b : B^2 \to B$ defined by $g_b(x,y) = g(x,y,b)$ is a semilattice operation. The proof of Theorem 7 uses the algebraic characterization of LAAC given by Theorem 1. The relational structures identified by Theorem 7 constitute a new tractable class, described using the notion of invariance, that has not been previously observed.

We can derive the following corollaries from Theorem 7.

Corollary 8. *Let* $d : B^3 \to B$ *be the* dual discriminator *on* B, *that is, the function such that* $d(x,y,z)$ *is equal to* x *if* $x = y$, *and* z *otherwise. For any relational structure* $\mathbf{B}$ *(with universe* B*) invariant under* d, *the* LAAC *algorithm is a solution procedure for* $\mathsf{CSP}(\mathbf{B})$.

The dual discriminator is an example of a *near-unanimity operation*; invariance under a near-unanimity operation has previously been demonstrated to imply $\mathsf{CSP}(\mathbf{B})$ tractability [13]. Another example of a near-unanimity operation that can be shown to imply tractability by Theorem 7 is as follows.

Corollary 9. *Let* $B = \{1,\ldots,k\}$, *and let* $\mathsf{median} : B^3 \to B$ *be the ternary function on* B *which returns the median of its arguments. For any relational structure* $\mathbf{B}$ *(with universe* B*) invariant under* median, *the* LAAC *algorithm is a solution procedure for* $\mathsf{CSP}(\mathbf{B})$.

The next theorem demonstrates that relational structures for which LAAC is a decision procedure can be combined together to give further relational structures also solvable by LAAC.

Theorem 10. *Suppose that* $B_1,\ldots,B_k$ *are sets, none of which contain* $\top$ *as an element, and suppose that* $l_1 : L(B_1) \to B_1, \ldots, l_k : L(B_k) \to B_k$ *are functions such that (for all* $i = 1,\ldots,k$*) it holds that (1)* $l_i(\{b_i\},\star) = b_i$, *for all* $b_i \in B_i$, *and (2) if* $B_i \cap B_j \neq \emptyset$, *then the restrictions of* l_i *and* l_j *to* $L(B_i \cap B_j)$ *are equal, for all* $j = 1,\ldots,k$. *Define* B *to be the set* $(\bigcup_{i=1}^k B_i) \cup \{\top\}$, *and let* $l : L(B) \to B$ *be the function defined by*

- $l(S,b) = s$ *if* $(S,b) = (\{s\},\star)$,
- $l(S,b) = l_i(S,b)$ *if* $|S| > 1$ *and* $S \cup \{b\} \subseteq B_i$ *for some* i, *and*
- $l(S,b) = \top$ *otherwise.*

For any relational structure $\mathbf{B}$ *(with universe* B*) such that* l *is a homomorphism from* $\mathcal{L}(\mathbf{B})$ *to* $\mathbf{B}$, *the* LAAC *algorithm is a solution procedure for* $\mathsf{CSP}(\mathbf{B})$. *Moreover, for any relational structure* $\mathbf{B}_i$ *(with universe* B_i*) such that* l_i *is a homomorphism from* $\mathcal{L}(\mathbf{B}_i)$ *to* $\mathbf{B}_i$, l *is also a homomorphism from* $\mathcal{L}(\mathbf{B}_i)$ *to* $\mathbf{B}_i$.

5 Commutative conservative operations

We say that a binary operation $\bullet$ over the set B is *commutative conservative* if for all $a, b \in B$, it holds that $a \bullet b = b \bullet a$; and, for all $a, b \in B$, it holds that $a \bullet b \in \{a, b\}$. As mentioned in the introduction, relational structures invariant under a commutative conservative operation have been demonstrated to be tractable [8], via an algorithm that is not well-characterized (in the sense discussed in the introduction). In this final section, we utilize the algebraic technology developed in Section 3 to give a precise classification of those commutative conservative operations whose invariant structures are tractable by the SLAAC algorithm, which as we have shown, is well-characterized.

Our classification is stated in terms of a "forbidden configuration". In particular, we show that the commutative conservative groupoids $(B, \bullet)$ tractable via SLAAC are precisely those that do not contain a forbidden subalgebra F.[6] Define F to be the commutative conservative groupoid having universe $\{0, 1, 2, 3\}$ and the following Cayley table.

	0	1	2	3
0	0	1	2	3
1	1	1	2	1
2	2	2	2	3
3	3	1	3	3

The groupoid F can be regarded as the so-called stone-scissors-paper algebra with a bottom element adjoined.

Theorem 11. *Let $(B, \bullet)$ be a commutative conservative groupoid. The following two statements are equivalent:*

- *The groupoid $(B, \bullet)$ does not have a subalgebra isomorphic to F.*
- *For every relational structure $\mathbf{B}$ (with universe B) invariant under $\bullet$, the SLAAC algorithm is a solution procedure for $\mathsf{CSP}(\mathbf{B})$.*

References

1. A. Bulatov. Combinatorial problems raised from 2-semilattices. Manuscript.
2. Andrei A. Bulatov. A Dichotomy Theorem for Constraints on a Three-Element Set. FOCS 2002.
3. A. Bulatov. Malt'sev constraints are tractable. Technical report PRG-RR-02-05, Oxford University, 2002.
4. Andrei A. Bulatov. Tractable conservative Constraint Satisfaction Problems. LICS 2003.
5. Andrei A. Bulatov, Andrei A. Krokhin, and Peter Jeavons. Constraint Satisfaction Problems and Finite Algebras. ICALP 2000.
6. A. Bulatov and P. Jeavons. An Algebraic Approach to Multi-sorted Constraints Proceedings of 9th International Conference on Principles and Practice of Constraint Programming, 2003.

[6] In this paper, a commutative conservative groupoid is simply a set endowed with a commutative conservative binary operation.

7. A. Bulatov, and P. Jeavons. Algebraic structures in combinatorial problems. Technical report MATH-AL-4-2001, Technische Universitat Dresden, 2001.
8. A. Bulatov, and P. Jeavons. Tractable constraints closed under a binary operation. Technical report PRG-TR-12-00, Oxford University, 2000.
9. Victor Dalmau and Justin Pearson. Set Functions and Width 1. Constraint Programming '99.
10. Alvaro del Val. On 2SAT and Renamable Horn. In AAAI'00, Proceedings of the Seventeenth (U.S.) National Conference on Artificial Intelligence, 279-284. Austin, Texas, 2000.
11. Tomás Feder and Moshe Y. Vardi. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory. SIAM J. Comput. 28(1): 57-104, 1998.
12. Peter Jeavons. On the Algebraic Structure of Combinatorial Problems. Theor. Comput. Sci. 200(1-2): 185-204, 1998.
13. P.G.Jeavons, D.A.Cohen and M.Cooper. Constraints, Consistency and Closure. Artificial Intelligence, 1998, 101(1-2), pages 251-265.
14. Peter Jeavons, David A. Cohen, and Marc Gyssens. Closure properties of constraints. J. ACM 44(4): 527-548 (1997).
15. Phokion G. Kolaitis, Moshe Y. Vardi. Conjunctive-Query Containment and Constraint Satisfaction. J. Comput. Syst. Sci. 61(2): 302-332 (2000)
16. Phokion G. Kolaitis and Moshe Y. Vardi. A Game-Theoretic Approach to Constraint Satisfaction. AAAI 2000.
17. T. Schaefer. The complexity of satisfiability problems. Proceedings of the 10th Annual Symposium on Theory of Computing, ACM, 1978.

A Proof of Theorem 1

Proof. Suppose that there exists a homomorphism l of the described form. We assume that $\mathbf{B}$ permits constant instantiation; if it does not, it may be expanded to a relational structure which does and still possesses the homomorphism l. It suffices to show that each time a variable is set by LAAC, satisfiability is preserved. Precisely, we show that when $\mathbf{A}$ is a relational structure such that $(\mathbf{A}, \mathbf{B})$ is satisfiable (via the homomorphism $h : \mathbf{A} \to \mathbf{B}$), then for any $a \in A$, $b \in B$, if arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = b]$, then $(\mathbf{A}, \mathbf{B})[a = b]$ is satisfiable. When arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = b]$, it follows (by the definitions of $(\mathbf{A}, \mathbf{B})[a = b]$ and $\mathcal{P}(\mathbf{B})$) that there is a homomorphism $p : \mathbf{A} \to \mathcal{P}(\mathbf{B})$ such that $p(a) = \{b\}$. Composing the homomorphisms $(p, h) : \mathbf{A} \to \mathcal{P}(\mathbf{B}) \times \mathbf{B}$, $\mathbf{c_B} : \mathcal{P}(\mathbf{B}) \times \mathbf{B} \to \mathcal{L}(\mathbf{B})$, and $l : \mathcal{L}(\mathbf{B}) \to \mathbf{B}$, we obtain a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ sending a to b, as desired.

Suppose that the LAAC algorithm is a solution procedure for $\mathsf{CSP}(\mathbf{B})$. It suffices to show that there is a homomorphism $h : \mathcal{P}(\mathbf{B}) \times \mathbf{B} \to \mathbf{B}$ such that $h(\{b_1\}, b_2) = b_1$ for all $b_1, b_2 \in B$, as such a homomorphism can be viewed as the composition of $\mathbf{c_B}$ and l, for some homomorphism l of the described form. We assume for ease of presentation that $\mathbf{B}$ permits constant instantiation; the same ideas apply when $\mathbf{B}$ does not permit constant instantiation. For every $b \in B$, let R_b be a relation symbol such that $R_b^{\mathbf{B}} = \{(b)\}$. Let $\mathbf{C}_{b_1,b_2}$ be the relational structure over the same vocabulary as $\mathbf{B}$ with universe $(\wp(B) \setminus \{\emptyset\}) \times B$ such that $R_{b_1}^{\mathbf{C}_{b_1,b_2}} = \{(\{b_1\}, b_2)\}$, and $R^{\mathbf{C}_{b_1,b_2}} = \emptyset$ for all other relation symbols R. Let C denote the set $\{\mathbf{C}_{b_1,b_2} : b_1, b_2 \in B\}$. We prove that for every subset $C' \subseteq C$, there is a homomorphism from $(\mathcal{P}(\mathbf{B}) \times \mathbf{B}) \cup (\bigcup_{\mathbf{C} \in C'} \mathbf{C})$ to $\mathbf{B}$; this suffices, as such a homomorphism is precisely a homomorphism h of the form desired. (The presence of a structure $\mathbf{C}_{b_1,b_2}$ as part of the source structure ensures that any homomorphism maps $(\{b_1\}, b_2)$ to b_1.)

The proof is by induction on $|C'|$. Suppose $C' = \emptyset$. The mapping from $(\wp(B) \setminus \{\emptyset\}) \times B$ to B that projects onto the second coordinate is a homomorphism from $\mathcal{P}(\mathbf{B}) \times \mathbf{B}$ to $\mathbf{B}$. Now suppose $|C'| \geq 1$. Let $C'' \subseteq C$ and $b_1, b_2 \in B$ be such that $C' = C'' \cup \{\mathbf{C}_{b_1,b_2}\}$ and $|C''| + 1 = |C'|$. The mapping from $(\wp(B) \setminus \{\emptyset\}) \times B$ to $\wp(B) \setminus \{\emptyset\}$ that projects onto the first coordinate is a homomorphism from $(\mathcal{P}(\mathbf{B}) \times \mathbf{B}) \cup (\bigcup_{\mathbf{C} \in C'} \mathbf{C})$ to $\mathcal{P}(\mathbf{B})$, so arc consistency can be established on $((\mathcal{P}(\mathbf{B}) \times \mathbf{B}) \cup (\bigcup_{\mathbf{C} \in C''} \mathbf{C}), B)[(\{b_1\}, b_2), b_1]$. Moreover, the instance $((\mathcal{P}(\mathbf{B}) \times \mathbf{B}) \cup (\bigcup_{\mathbf{C} \in C''} \mathbf{C}), B)$ is satisfiable by induction, so by the assumption that LAAC is an algorithm for $\mathsf{CSP}(\mathbf{B})$, the instance $((\mathcal{P}(\mathbf{B}) \times \mathbf{B}) \cup (\bigcup_{\mathbf{C} \in C'} \mathbf{C}), B)$ is satisfiable. □

B Proof of Theorem 3

Proof. Suppose that there exist homomorphisms $h_{U,b}$ of the described form. As in the proof of Theorem 1, it suffices to show that each time a variable is set by SLAAC, satisfiability is preserved. Precisely, we show that when $\mathbf{A}$ is a relational structure such that $a \in A$, $(\mathbf{A}, \mathbf{B})$ is satisfiable via a homomorphism $h : \mathbf{A} \to \mathbf{B}$

mapping a to b, and $U \subseteq B$ contains exactly those values $c \in B$ such that arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = c]$, then $(\mathbf{A}, \mathbf{B})[a = f(U)]$ is satisfiable. Note that given these assumptions, the pair (U, b) must be usable. Let $b_1, \ldots, b_k$ be the elements of U; since (for each i) arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = b_i]$, there is a homomorphism $p_i : \mathbf{A} \to \mathcal{P}(\mathbf{B})$ such that $p(a) = \{b_i\}$. Composing the homomorphisms $(p_1, \ldots, p_k, h) : A \to \mathcal{P}(\mathbf{B})^k \times \mathbf{B}$ and $h_{U,b} : \mathcal{P}(\mathbf{B})^k \times \mathbf{B} \to \mathbf{B}$, we obtain a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ sending a to $f(U)$, as desired.

Suppose that the SLAAC algorithm is a solution procedure for CSP($\mathbf{B}$). Let (U, b) be a usable pair, and let $b_1, \ldots, b_k$ denote the elements of U. We show that there exists a homomorphism $h : \mathcal{P}(\mathbf{B}) \times \mathbf{B} \to \mathbf{B}$ such that $h(\{b_1\}, \ldots, \{b_k\}, b) = f(U)$. Let $\mathbf{A}$ denote the structure $\mathcal{P}(\mathbf{B})^k \times \mathbf{B}$, and let a denote the element $(\{b_1\}, \ldots, \{b_k\}, b)$ of $\mathbf{A}$. By the homomorphism from $\mathbf{A}$ to $\mathbf{B}$ that projects onto the ith coordinate, there is a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$ such that a is mapped to b_i, implying that arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = b_i]$ for all $b_i \in U$. Also, by the homomorphism from $\mathbf{A}$ to $\mathbf{B}$ that projects onto the last coordinate, there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ such that a is mapped to b.

We extend $\mathbf{A}$ so that the values in U are *exactly* those values c such that arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = c]$, in the following way. Let $(\mathbf{A}_0, \mathbf{B})$ be a satisfiable CSP instance with a variable $a_0 \in A_0$ such that $U = \{c \in B :$ arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a_0 = c]\}$, and there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$ sending a_0 to b. (Such a CSP instance exists by the definition of usable pair.) Identify the variable a_0 with a, assume that $A_0 \setminus \{a_0\}$ and $A \setminus \{a\}$ are disjoint, and consider the CSP instance $(\mathbf{A} \cup \mathbf{A}_0, \mathbf{B})$. The values c in U are exactly those values such that arc consistency can be established on $(\mathbf{A} \cup \mathbf{A}_0, \mathbf{B})[a = c]$. Moreover, $(\mathbf{A} \cup \mathbf{A}_0, \mathbf{B})$ is satisfiable via a homomorphism sending a to b. By the hypothesis that SLAAC solves CSP($\mathbf{B}$), the instance $(\mathbf{A} \cup \mathbf{A}_0, \mathbf{B})$ is satisfiable via a homomorphism h' sending a to $f(U)$. The restriction of h' to A gives the desired homomorphism. □

C Proof of Theorem 4

Proof. Let (U, b) be a pair where $U \subseteq B$ and $b \in U$, and let $b_1, \ldots, b_k$ denote the elements of U. We show that (U, b) is usable if and only if for all homomorphisms $h : \mathcal{P}(\mathbf{B})^k \times \mathbf{B} \to \mathcal{P}(\mathbf{B})$ such that $|h(\{b_1\}, \ldots, \{b_k\}, b)| = 1$, it holds that $h(\{b_1\}, \ldots, \{b_k\}, b) \in U$. This suffices to establish the theorem: there are finitely many homomorphisms from $\mathcal{P}(\mathbf{B})^k \times \mathbf{B}$ to $\mathcal{P}(\mathbf{B})$, so the condition can be effectively checked.

Suppose that the given homomorphism condition holds. Set $\mathbf{A} = \mathcal{P}(\mathbf{B})^k \times \mathbf{B}$ and $a = (\{b_1\}, \ldots, \{b_k\}, b)$. We claim that the instance $(\mathbf{A}, \mathbf{B})$ and the variable a are witness to the usability of (U, b). Observe that arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = u]$ if and only if there is a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$ sending a to $\{u\}$. Hence, by assumption, we cannot establish arc consistency on $(\mathbf{A}, \mathbf{B})[a = u]$ when $u \notin B$. But, we can establish arc consistency on

$(\mathbf{A}, \mathbf{B})[a = b_i]$ for any $b_i \in U$, since the projection onto the ith coordinate is a homomorphism from $\mathcal{P}(\mathbf{B})^k \times \mathbf{B}$ to $\mathcal{P}(\mathbf{B})$ sending a to $\{b_i\}$.

Now suppose that the given homomorphism condition fails. We prove that (U, b) is not usable, by contradiction. Suppose that (U, b) is usable. Let $\mathbf{A}$ and $a \in A$ be witness to the usability of (U, b). It follows that there is a homomorphism $h : \mathbf{A} \to \mathcal{P}(\mathbf{B})^k \times \mathbf{B}$ sending a to $(\{b_1\}, \ldots, \{b_k\}, b)$. Since the homomorphism condition fails, there is a homomorphism $h' : \mathcal{P}(\mathbf{B})^k \times \mathbf{B} \to \mathcal{P}(\mathbf{B})$ such that $h'(\{b_1\}, \ldots, \{b_k\}, b) = c$, where $c \notin U$. Composing h and h', we obtain a homomorphism from $\mathbf{A}$ to $\mathcal{P}(\mathbf{B})$ sending a to c; this implies that arc consistency can be established on $(\mathbf{A}, \mathbf{B})[a = c]$, contradicting that $\mathbf{A}$ and a are witness to the usability of (U, b). □

D Proof of Theorem 6

Proof. We first demonstrate that $\mathbf{B}$ is invariant under a set function h such that the map $b \to h(\{b\})$ acts as the identity on all elements in $\mathsf{im}(h)$. (We use im to denote the image of a function.) Let $c \geq 1$ be sufficiently large so that for all $n > c$, $\mathsf{im}(f_1^c) = \mathsf{im}(f_1^n)$. Define g to be the set function on B such that $g(S) = f_1^c(f(S))$. We have $\mathsf{im}(g_1) = \mathsf{im}(g) = \mathsf{im}(f_1^c)$. Let $d > 1$ be sufficiently high so that g_1^d acts as the identity on all elements in its image, and define h to be the set function on B such that $h(S) = g_1^{d-1}(g(S))$. The map $b \to h(\{b\})$ acts as the identity on all elements in $\mathsf{im}(g_1)$, and hence on all elements in $\mathsf{im}(h)$ (which is a subset of $\mathsf{im}(g_1)$). Moreover, it is straightforward to verify that $\mathbf{B}$ is invariant under all of the functions discussed, in particular, h.

We can assume that the structure $\mathbf{B}$ permits constant instantiation on all elements of $\mathsf{im}(h)$; if it does not, we can enlargen $\mathbf{B}$ so that it does, while preserving the invariance of $\mathbf{B}$ under h.

We claim that the SLAAC algorithm, parameterized with h, solves $\mathsf{CSP}(\mathbf{B})$. This follows from Theorem 3: for each usable pair (U, b) with $|U| = k$, we have a homomorphism $h_{U,b} : \mathcal{P}(\mathbf{B})^k \times \mathbf{B} \to \mathbf{B}$ of the desired form, given by $h_{U,b}(S_1, \ldots, S_k, b) = h(\bigcup_{i=1}^k S_i)$. □

E Proof of Theorem 7

Proof. We have that, for each fixed $b \in B$, the binary operation $g_b : B^2 \to B$ defined by $g_b(x, y) = g(x, y, b)$ is a semilattice operation. Define $h : (\wp(B) \setminus \{\emptyset\}) \to B$ to be the mapping defined by

$$h(\{s_1, \ldots, s_k\}, b) = g_b(s_1, g_b(s_2, \ldots, g_b(s_{k-1}, s_k) \ldots)).$$

Note that the right-hand side is well-defined, since g_b is a semilattice operation. It can be verified that h is a homomorphism from $\mathcal{P}(\mathbf{B}) \times \mathbf{B}$ to $\mathbf{B}$ sending any element of the form $(\{s\}, b)$ to s; hence, h can be factored as the composition of the homomorphism $\mathbf{c_B} : \mathcal{P}(\mathbf{B}) \times \mathbf{B} \to \mathcal{L}(\mathbf{B})$ and a homomorphism $l : \mathcal{L}(\mathbf{B}) \to \mathbf{B}$ such that $l(\{s\}, \star) = s$ (for all $s \in B$). It follows from Theorem 1 that the LAAC algorithm is a solution procedure for $\mathsf{CSP}(\mathbf{B})$. □

F Proof of Theorem 11

Proof. Let us start by introducing some notation. We will write $a \leq b$ to indicate that $a \bullet b = b$. By a cycle we mean a sequence $b_1 \leq b_2 \leq \cdots \leq b_n$ such that $b_1 = b_n$. For any subset S of B, we say that S is strongly connected if for all $a, b \in S$, there exists some $b_1, \ldots, b_n \in S$ such that

$$(\ldots((a \bullet b_1) \bullet b_2) \ldots \bullet b_n) = b$$

Let S be any subset of B. If there exists some $b \in S$ such that for all $a \in S$, $a \leq b$ then we say that S has a max, and we define $\max S$ to be b.

We prove the equivalence of the following statements:

(a) The groupoid $(B, \bullet)$ does not have a subalgebra isomorphic to F.
(b) For every $S \subseteq B$, if there exists an element $a \in B$ such that $a \leq b$ for all b in S, then S does not contain any cycle.
(c) For every relational structure $\mathbf{B}$ (with universe B) invariant under $\bullet$, the SLAAC algorithm is a solution procedure for $\mathsf{CSP}(\mathbf{B})$.

(a) $\Rightarrow$ **(b)**

Towards a contradiction, assume that there exists a cycle in S. Henceforth there exists a cycle in S of length 3, say b_1, b_2, b_3. Consequently the subalgebra with universe $\{a, b_1, b_2, b_3\}$ is isomorphic to F.

(b) $\Rightarrow$ **(a)**

Let R be a relation of arity, say k, closed under $\bullet$. We shall see first some properties of R. Let us assume first that for every $1 \leq i \leq k$, $\mathsf{pr}_i R$ is simple.

Let us assume also that $\mathsf{pr}_1 R, \ldots, \mathsf{pr}_k R$ are strongly connected. By the results of [8] R is reduced (in [8] is called almost trivial), which is to say that for each set $I = \{r_1, \ldots, r_m\}$ containing exactly one element from each block of the equivalence $\sim$ on $\{1, \ldots, k\}$ defined by

$$i \sim i' \text{ iff } \mathsf{pr}_{i,i'} R \text{ is a bijective mapping}$$

we have $\mathsf{pr}_I R = \mathsf{pr}_{r_1} R \times \cdots \times \mathsf{pr}_{r_m} R$.

Let us deal now with the general case, that is, commutative conservative groupoids that are not necessarily simple. For every subset S of B, let us fix a congruence, $\equiv_S$ on S. If S has a max then $\equiv_S$ is defined to be the trivial congruence with just one congruence block, otherwise $\equiv_S$ is set to be any maximal congruence with more that one congruence block. We can infer that in this second case (when S has no max) none of the congruence classes has a cycle.

Proof. Let us assume towards a contradiction that one of the congruence classes $\overline{b}$ contains a cycle. Then it must satisfy $\overline{b} \leq \overline{a}$ for all $\overline{a}$ by condition (b). Again by condition (b) we can conclude that $S/\equiv_S$ contains no cycle and hence $S/\equiv_S$ is a total ordering. Take its max $\overline{m}$ which must be different from $\overline{b}$ (since the congruence has more than one congruence block). It is clear than $\overline{m}$ cannot contain a cycle so the set containing all elements in the congruence class of $\overline{m}$ has a max, say m. Clearly m dominates all the elements in S (end of proof).

Similarly, we can infer that $S/\equiv_S$ is strongly connected, since the only simple conservative groupoids which are not strongly connected are those containing precisely two elements. Consequently if $S/\equiv_S$ equals, say $\{\overline{a}, \overline{b}\}$, with $\overline{a} \leq \overline{b}$, then again $\overline{b}$ cannot contain any cycle, and hence it must have a max, which will dominate every element in S. In both cases (S having a max or not) we have that $S/\equiv_S$ is strongly connected. Furthermore, for each congruence class $\overline{b}$ in S we can define the max among the elements in the class, which we shall denote by $\max \overline{b}$. Furthermore, S has a max iff and only iff $\equiv_s$ has only one block.

Let $S_1 \subseteq S_2$ be subsets of B and b be any element in S_2. We define $g(S_1, S_2, b)$ in the following way. Let $\equiv_{S_2}$ be the congruence associated to S_2. If for all s, s' in S_1 we have that $s \equiv_{S_2} s'$ then $g(S_1, S_2, b)$ is defined to be $\max \overline{s}$ for any $s \in S_1$. Otherwise $g(S_1, S_2, b)$ is defined to be $\max \overline{b}$.

Let R_2 be a relation, say of arity k, closed under $\bullet$. Let R_1 be any subset of R_2 and let $(t_1, \ldots, t_k) = \mathbf{t}$ be any tuple in R_2. We shall show that the tuple $(u_1, \ldots, u_k) = \mathbf{u}$ where $u_i = g(\mathsf{pr}_i R_1, \mathsf{pr}_i R_2, t_i)$ belongs to R_2.

Let $\overline{R_2}$ be R_2 modulo $(\equiv_{\mathsf{pr}_1 R}, \ldots, \equiv_{\mathsf{pr}_2 R})$. That is, for every $(b_1, \ldots, b_k)$ in R_2, $\overline{R_2}$ contains the tuple $(\overline{b_1}, \ldots, \overline{b_k})$ where $\overline{b_i}$ is the congruence class of b_i in $\equiv_{\mathsf{pr}_i R}$. Similarly, we define $\overline{R_1}$, $\overline{\mathbf{t}}$, and $\overline{\mathbf{u}}$.

We have seen that for each $1 \leq i \leq k$, $\mathsf{pr}_i \overline{R_2}$ is strongly connected. We shall show that $\overline{\mathbf{u}} \in \overline{R_2}$.

By the definition of g, we have that $\overline{u_i}$ is $\overline{s_1}$ if if $\mathsf{pr}_i \overline{R_1}$ is a singleton, namely $\{s_1\}$ and $\overline{t_i}$ otherwise.

We shall show that $\overline{\mathbf{u}}$ belongs to $\overline{R_2}$. Since $\overline{R_2}$ is reduced it is only necessary to show that if $i \sim i'$ then $(\overline{u_i}, \overline{u_{i'}})$ belongs to $\mathsf{pr}_{i,i'} \overline{R_2}$. Let us do a case analysis: (1) If $|\mathsf{pr}_i \overline{R_2}| > 1$ and $|\mathsf{pr}_{i'} \overline{R_2}| > 1$ then $(\overline{u_i}, \overline{u_{i'}}) = (\overline{t_i}, \overline{t_{i'}})$ and we are done. (2) If $\mathsf{pr}_i \overline{R_2}$ is a singleton, say $\{s_i\}$, and also $\mathsf{pr}_{i'} \overline{R_2}$ is a singleton, say $\{s_{i'}\}$, then we have that $(\overline{u_i}, \overline{u_{i'}}) = (\overline{s_i}, \overline{s_{i'}})$ which is in $\mathsf{pr}_{i,i'} \overline{R_1}$ and hence in $\mathsf{pr}_{i,i'} \overline{R_2}$. (3) Finally if only one of $\mathsf{pr}_i \overline{R_2}$ and $\mathsf{pr}_{i'} \overline{R_2}$ is a singleton then we can conclude that $i \not\sim i'$ and we are done.

We shall show now an stronger statement: for every $1 \leq l \leq k$, there exists some $(u'_1, \ldots, u'_k)$ such that $u'_l = u_l$ and for all $1 \leq i \leq k$, $u_i \equiv_{\mathsf{pr}_i R_2} u'_i$ (*).

Once (*) has been proven then what it remains is an easy task. For every $1 \leq l \leq i$ let $\mathbf{u_l}$ be the tuple given by (*). The tuple $(u_1, \ldots, u_k)$ can be obtained as

$$(\ldots(\mathbf{u_1} \bullet \mathbf{u_2}) \ldots \bullet \mathbf{u_k})$$

and hence it belongs to R_2.

Proof of (*). Let I be a set containing exactly one representative of each block of $\sim$ on $\overline{R_2}$. We can choose $I = \{r_1, \ldots, r_m\}$ such that $i_1 = l$. We known that exists some tuple $(a_{r_1}, \ldots, a_{r_m})$ in $\mathsf{pr}_I R_2$ such that $a_{r_1} = u_l$. In order to simplify the notation we shall denote by $(a_{r_2}, \ldots, a_{r_m})$ by $\mathbf{a}$ and $(u_{r_2}, \ldots, u_{r_m})$ by $\mathbf{u}$. Since $\mathsf{pr}_{r_2} \overline{R_2}, \ldots, \mathsf{pr}_{r_m} \overline{R_n}$ are strongly connected we have that

$$\overline{(\ldots(\mathbf{a} \bullet \mathbf{b_1}) \ldots \bullet \mathbf{b_n})} = \overline{\mathbf{u}}$$

for some $\mathbf{b_1}, \ldots, \mathbf{b_n} \in \mathsf{pr}_{r_2, \ldots, r_m} R_2$. Furthermore, since $\mathsf{pr}_{r_1, \ldots, r_m} \overline{R_2} = \mathsf{pr}_{r_1} \overline{R_2} \times \cdots \times \mathsf{pr}_{r_m} \overline{R_2}$ we have that for for every $1 \leq i \leq n$, $(c_i, \mathbf{b_i}) \in R_2$ for some c_i with

$c_i \equiv_{\mathsf{pr}_i R_2} u_l$. Thus, if we set $(u'_{r_1}, \dots, u'_{r_m})$ to be

$$(\dots((u_l, \mathbf{a}) \bullet (c_1, \mathbf{b_1})) \dots \bullet (c_n, \mathbf{b_n}))$$

then we have that $u_{r_1} = u_l$ and futhermore for all $1 \leq i \leq k$, $u_i \equiv_{\mathsf{pr}_i R_2} u'_i$. (End of the proof of (*)).

We are almost at home. For every $m \geq 0$, every $S_1, \dots, S_m \subseteq B$, and every b in B, define $f(S_1, \dots, S_m, b)$ to be $g(S_1, S_1 \cup \dots \cup S_m \cup \{b\}, b)$. First notice that for every $b_1, \dots, b_m$ and for every $b, b' \in \{b_1, \dots, b_m\}$ we have that $f(b_1, \dots, b_m, b) = f(b_1, \dots, b_m, b')$. Now it only remains to show (by appeal to Theorem 3) that if $\mathbf{B}$ is an structure closed under $\bullet$ then f defines an homomorphism from $\mathcal{P}(\mathbf{B})^m \times \mathbf{B}$ to $\mathbf{B}$. Let R be any relation, say of arity k, in $\mathbf{B}$, let $R_1, \dots, R_m \subseteq R$ and let $(t_1, \dots, t_m) = \mathbf{t} \in R$. Let $R' \subseteq R$ be the closure of $R_1 \cup \dots \cup R_m \cup \{\mathbf{t}\}$ under $\bullet$. We know that the tuple $(u_1, \dots, u_k)$, where $u_i = g(\mathsf{pr}_i R_1, \mathsf{pr}_i R', t_i)$ belongs also to R' and hence to R. Furthermore, since $\bullet$ is conservative we have that $\mathsf{pr}_i R' = \mathsf{pr}_i R_1 \cup \dots \cup \mathsf{pr}_i R_m \cup \{t_i\}$. Consequently we have that $u_i = f(\mathsf{pr}_i R_1, \mathsf{pr}_i R_1 \cup \dots \cup \mathsf{pr}_i R_m \cup \{t_i\}, t_i)$ and we are done.

(c) $\Rightarrow$ **(a)**

Let $(B, \bullet)$ be a conservative commutative groupoid containing having F as a subalgebra. Let us assume, towards a contradiction, that SLAAC parameterized with f can decide every structure $\mathbf{B}$ invariant under $\bullet$. Let $\mathbf{B}$ be any structure containg the relations $R_1^{\mathbf{B}}$ and $R_2^{\mathbf{B}}$, where $R_1^{\mathbf{B}}$ is the relation containing all pairs in $\{0,1,2,3\}^2$ but $(0,0)$, and $R_2^{\mathbf{B}}$ is the relation containing $\{(1,2),(2,3),(3,1),(0,0)\}$. It is immediate to verify that $R_1^{\mathbf{B}}$ and $R_2^{\mathbf{B}}$ are invariant under $\bullet$.

We shall see that there not exists any homomorphism from $\mathcal{P}(\mathbf{B})^4 \times \mathbf{B}$ to $\mathbf{B}$ satisfying the required conditions. Let h be any homomorphism from $\mathcal{P}(\mathbf{B})^4 \times \mathbf{B}$ to $\mathbf{B}$.

We have that $((\{0\},\{1\},\{2\},\{3\},0),(\{1\},\{2\},\{3\},\{0\},1))$ belongs to the relation $R_1^{\mathcal{P}(\mathbf{B})^4 \times \mathbf{B}}$. Consequently, applying h to this tuple we have that the tuple $(f(\{0,1,2,3\}), f(\{0,1,2,3\}))$ belongs to $R_1^{\mathbf{B}}$. Consequently $f(\{0,1,2,3\}) \neq 0$. Similarly, we have that the tuple $((\{0\},\{1\},\{2\},\{3\},0),(\{0\},\{2\},\{3\},\{0\},0))$ belongs to $R_2^{\mathcal{P}(\mathbf{B})^4 \mathbf{B}}$. Again, by applying h to this tuple we obtain that the tuple $(f(\{0,1,2,3\}), f(\{0,1,2,3\}))$ belongs to $R_2^{\mathbf{B}}$, and consequently $f(\{0,1,2,3\}) = 0$, a contradiction. $\square$