THE POWER OF THE COMBINED BASIC LP AND AFFINE RELAXATION FOR PROMISE CSPS*

JOSHUA BRAKENSIEK†, VENKATESAN GURUSWAMI‡, MARCIN WROCHNA§, AND STANISLAV ŽIVNÝ¶

Abstract. In the field of constraint satisfaction problems (CSP), promise CSPs are an exciting new direction of study. In a promise CSP, each constraint comes in two forms: "strict" and "weak," and in the associated decision problem one must distinguish between being able to satisfy all the strict constraints versus not being able to satisfy all the weak constraints. The most commonly cited example of a promise CSP is the approximate graph coloring problem—which has recently seen exciting progress [12, 24] benefiting from a systematic algebraic approach to promise CSPs based on "polymorphisms," operations that map tuples in the strict form of each constraint to tuples in the corresponding weak form.

In this work, we present a simple algorithm which in polynomial time solves the decision problem for all promise CSPs that admit infinitely many *symmetric* polymorphisms, which are invariant under arbitrary coordinate permutations. This generalizes previous work of the first two authors [7]. We also extend this algorithm to a more general class of *block-symmetric* polymorphisms. As a corollary, this single algorithm solves all polynomial-time tractable Boolean CSPs simultaneously. These results give a new perspective on Schaefer's classic dichotomy theorem and shed further light on how symmetries of polymorphisms enable algorithms. Finally, we show that block symmetric polymorphisms are not only sufficient but also necessary for this algorithm to work, thus establishing its precise power.

Key words. CSP, linear programming, linear equations, polymorphisms, algorithms, promise CSP

AMS subject classifications. 68Q25, 68W25, 68R01, 08A70, 03B70, 90C05

1. Introduction.

A central challenge in the theory of algorithms is to understand the mathematical structure (or lack thereof) that governs the efficient tractability (or intractability) of a computational problem. For the class of constraint satisfaction problems (CSP), a rich algebraic theory culminating in the recent resolution of the Feder-Vardi dichotomy conjecture [14] in [11, 25] has established a striking link between problem structure and its tractability. In particular, a CSP is efficiently solvable if and only if its defining relations admit an "interesting" polymorphism. Informally, a polymorphism is a function whose component-wise action preserves membership in the relations defining the CSP, and "interesting" means that the function obeys some *non-trivial identities*. As an example, for the (efficiently solvable) CSP corresponding to linear equations over a ring R, the 3-ary function $f(x, y, z) = x - y + z$ is a poly-

*An extended abstract of part of this work (by the first two authors) appeared in the *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'20)* [8].

†Stanford University, Stanford, CA 94305, USA. Email: jbrakens@stanford.edu.

Funding: Research supported in part by an REU supplement to NSF CCF-1526092 and a NSF Graduate Research Fellowship.

‡Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213. Email: venkatg@cs.cmu.edu.

Funding: Research supported in part by NSF grants CCF-1814603 and CCF-1908125.

§Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, OX1 3QD Oxford, UK. Email: marcin.wrochna@cs.ox.ac.uk.

Funding: Research supported by funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532).

¶Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, OX1 3QD Oxford, UK. Email: standa.zivny@cs.ox.ac.uk.

Funding: Research supported by a Royal Society University Research Fellowship and by funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532).

morphism (capturing the fact that if v_1, v_2, v_3 are solutions to a linear system, then so is $v_1 - v_2 + v_3$), and it obeys the so-called Mal'tsev identity $f(x, y, y) = f(y, y, x) = x$ for all $x, y \in R$. Indeed, generalizing Gaussian elimination, any CSP with such a Mal'tsev polymorphism is efficiently tractable [10, 9].

Recently, an exciting new direction of study has emerged in the rich backdrop of the complexity dichotomy for CSPs. This concerns a vast generalization of the CSP framework to the class of *promise constraint satisfaction problems* (PCSP). In a promise CSP, each constraint comes in two forms: "strict" and "weak." Given an instance of a PCSP, one must distinguish between being able to satisfy all the strict constraints versus not being able to satisfy all the weak constraints. (This is the *decision* version; in the *search* version, given an instance with a promised assignment satisfying the strong form of the constraints, one seeks an assignment satisfying the weak form of the constraints.) A prime example of a PCSP is the approximate graph coloring problem, where one seeks to color a graph using more colors than its chromatic number.

The formal study of promise CSPs originated in [1] who classified the complexity of a PCSP called $(2+\epsilon)$-SAT. They further defined an extension of polymorphisms to the promise setting and postulated that the structure of those polymorphisms might govern the complexity of a PCSP. (This extension of polymorphisms to the promise setting is quite natural, requiring that the operation map tuples obeying the strict form of a constraint to a tuple satisfying its weak form.) Building on the impetus of [1], Brakensiek and Guruswami systematically studied PCSPs under the polymorphic lens and established promising links to the universal-algebraic framework developed for CSPs [6, 7]. It emerged from these works that a rich enough family of polymorphisms leads to efficient algorithms, whereas severely limited polymorphisms are a prescription for hardness. However, unlike for CSPs, there is no sharp transition between these cases — the significant difficulty being that, unlike for CSPs, polymorphisms for PCSPs are *not* closed under composition and lack the rich algebraic structure of a *clone* (c.f., [4]). This nascent algebraic theory for PCSPs was lifted to a more abstract level in [12, 2] and also led to concrete breakthroughs in approximate graph coloring/homomorphisms [12, 18, 24]. In particular, while previous works [6, 7] focused on the actual *form* of the polymorphisms, the results of [12] reveal that it is not the polymorphisms themselves, but rather solely the *identities* they satisfy, that capture the complexity of the associated PCSP, extending a similar phenomenon known earlier for CSPs [5].

This work concerns the theme of designing algorithms for PCSP based on a rich enough family of polymorphisms. Our main result is that the decision version of an arbitrary PCSP admitting an *infinite family of symmetric polymorphisms* — i.e., polymorphisms which are invariant under any permutation of inputs — is tractable (see Theorem 3.2). Our result also extends to the case of *block-symmetric* polymorphisms (see Theorem 4.1). That is, the coordinates can be partitioned into "blocks" such that the function is invariant under permutations within each block. Notably, in the block-symmetric case the algorithm is identical–only the analysis changes. Furthermore, the number of blocks is irrelevant, the only assumption we need is that the minimum block size can be made arbitrarily large. Our final result (Theorem 5.1) shows that block-symmetry is not only sufficient but also necessary for our algorithm to work. In fact, Theorem 5.1 also establishes that without loss of generality one can assume that there are only two blocks of symmetric coordinates.

Further our algorithm is very simple — it checks if the canonical linear programming (LP) relaxation of the PCSP is feasible, and if so, it further checks if a slight

adaptation of a canonical *affine relaxation* is feasible. The algorithm outputs satisfiable if both these relaxations are feasible. The polymorphisms are not used in the algorithm itself and only enter the analysis. The analysis is short but subtle — if we had symmetric polymorphisms of all arities then it is known that the basic LP relaxation itself correctly decides satisfiability, as one can round the fractional solution to a satisfying assignment using the polymorphism after clearing denominators of the fractional solution [19, 4]. If polymorphisms only exist of certain arities (e.g., all odd majorities), then the LP alone doesn't suffice (e.g., [19]). We solve a linear system over the integers corresponding to the affine relaxation which lets us adjust the LP solution to match the arity at which a polymorphism exists. As a subtle twist, the affine relaxation is not of the original PCSP, but rather a *refinement* of the CSP which results from throwing out assignments to constraints which were ruled out by the basic LP.

It should be pointed out that we only solve the *decision* version of the PCSP, and *not* the search version. Unlike CSPs, for promise CSP there is no known reduction from search to decision, even for special cases like approximate graph coloring. Our work might be indicative of the subtle relationship between the search and decision problems for promise CSPs.

We now compare our result here with the previous work [7] where an algorithm was given to solve (the search version of) any PCSP that admits an infinite family of *structured symmetric polymorphisms*. Examples of such structured families include threshold and threshold-periodic polymorphisms. The value of a threshold polymorphism (for a Boolean PCSP) depends on whether the fraction of 1s in the input belongs in a finite number of intervals. (A basic example consists of Majority functions of odd arities, which are polymorphisms for 2-SAT.) A threshold-periodic polymorphism can have a periodic behavior depending on which interval the Hamming weight belongs to — for example it can be Majority for relative weights in $(1/3, 2/3)$ and parity outside this interval. More generally, one can generalize to the non-Boolean case, as well as for the block-symmetric case, via regional polymorphisms whose value depends on the geometric region in which the vector of frequencies of the inputs to the polymorphisms lies. Due to this geometric interpretation, [7] assumes a fixed number of blocks (corresponding to a fixed dimension), whereas our new algorithm and analysis is independent of the number of blocks. The algorithm was a combination of solving the LP relaxation (albeit over a special ring like $\mathbb{Z}[\sqrt{2}]$ rather than the rationals) and the affine relaxation over a large enough finite ring. The analysis relied on the special *structure* of the polymorphisms (beyond their full symmetry). In contrast, our result here is more general, and only requires the polymorphism to be a symmetric function — its exact specifics or structure do not matter. It is encouraging that our methodology is consistent with the algebraic result in [12] that the symmetries possessed by the polymorphisms capture the complexity of the PCSP.

Our result and methods have significance even for normal (non-promise) CSPs. For instance, we get a single unified algorithm to solve all non-trivial tractable cases of Boolean CSPs in Schaefer's classic dichotomy theorem [22], namely 2-SAT, Horn-SAT (or its dual), and Mod-2 Linear Equations. The two main techniques to solve CSPs are local propagation based algorithms (which work for the so-called *bounded-width* CSPs [3, 19], etc.) and Gaussian elimination (which is a global algorithm that works for linear equations). The major difficulty in proving the full CSP dichotomy was tackling the complicated ways in which these two very different algorithms might need to be interlaced to solve a general CSP. It is our hope that this work serves as an impetus toward the potential discovery of a more modular CSP algorithm that

incorporates together linear programming or its extensions (like Sherali-Adams, or semidefinite programming) and linear equation solving. In this light, it is encouraging that full symmetry of the polymorphisms, which is indeed a strong assumption, is not the limit of our techniques, which also extend to the block-symmetric case.

To put this work in further context, except for [7] as mentioned previously, nearly all works in the PCSP literature [1, 6, 15] focus primarily on the structure of the *relations*. In particular, [6, 15] characterized the complexity of all *Boolean symmetric relations* (rather than *symmetric polymorphisms*) which encompass many of the known tractable cases of Boolean PCSP. As classified by [15], all the relevant tractable polymorphisms are either symmetric functions or one special case of block-symmetric known as alternative threshold (and variants). Thus, in the context of PCSPs, the algorithm in this paper supersedes these previous works. See Section 4 for further discussion.

2. Notation. We let any finite set A or B denote a *domain*. A *relation* is a subset $R \subseteq A^k$ for any positive integer k; we denote $\mathrm{ar}(R) := k$. We define a *signature* τ to be a set of symbols such that each $R \in \tau$ has a positive integer *arity* $\mathrm{ar}(R)$.

A *relational structure* with signature τ, denoted by $\mathbf{A} := \{R^{\mathbf{A}} \subseteq A^{\mathrm{ar}(R)} : R \in \tau\}$, is an indexed set of relations over A. A *homomorphism* between structures $\mathbf{A} = \{R^{\mathbf{A}} : R \in \tau\}$ and $\mathbf{B} = \{R^{\mathbf{B}} : R \in \tau\}$ with the same signature is a map $\sigma : A \to B$ such that $\sigma(R^{\mathbf{A}}) \subseteq R^{\mathbf{B}}$ for all $R \in \tau$ (where σ is applied to a tuple component-wise).

Two relational structures for which there exists a homomorphism from the first to the second is called a *promise template* and is denoted as $(\mathbf{A}, \mathbf{B})$.

2.1. PCSP: Decision and Search. Consider a promise template $(\mathbf{A}, \mathbf{B})$ with signature τ. An *instance* $\mathbf{X}$ of the *promise constaint satisfaction problem* $\mathrm{PCSP}(\mathbf{A}, \mathbf{B})$ consists of a set of variables $X := \{x_1, \dots, x_n\}$, and a set of constraints $c_1, \dots, c_m$, where $c_j := (R_j, \bar{x}^j)$, where R_j is a symbol in τ and $\bar{x}^j$ is a tuple of arity $\mathrm{ar}(R_j)$. We say that $\mathbf{X}$ is *satisfiable in* $\mathbf{A}$ if one can assign to every variable x_i ($i \in [n]$) a value $\sigma(x_i)$ in the domain A so that for every constraint $c_j = (R_j, \bar{x}^j)$ ($j \in [m]$), the tuple $\sigma(\bar{x}^j)$ (with σ applied component-wise) is in $R_j^{\mathbf{A}}$. Equivalently, $\mathbf{X}$ can be described as a relational structure with domain X and relations $R^{\mathbf{X}} = \{\bar{x} \in X^{\mathrm{ar}(R)} \colon \exists_{j \in [m]}\ c_j = (R, \bar{x})\}$; a satisfying assignment is then the same as a homomorphism from $\mathbf{X}$ to $\mathbf{A}$. If $\mathbf{X}$ is satisfiable in $\mathbf{A}$, then it is satisfiable in $\mathbf{B}$ (because the satisfying assignment can be composed with the homomorphism from $\mathbf{A}$ to $\mathbf{B}$).

We let PCSP-Decision$(\mathbf{A}, \mathbf{B})$ denote the *decision* problem of distinguishing instances satisfiable in $\mathbf{A}$ from those unsatisfiable in $\mathbf{B}$ (with the promise that the input instance falls into one of these two disjoint cases). We let PCSP-Search$(\mathbf{A}, \mathbf{B})$ denote the *search* problem of finding an explicit homomorphism from $\mathbf{X}$ to $\mathbf{B}$, with the promise that a homomorphism from $\mathbf{X}$ to $\mathbf{A}$ exists.

2.2. Polymorphisms. A *polymorphism* of $(\mathbf{A}, \mathbf{B})$ of arity $L \in \mathbb{N}$ is a map $f : A^L \to B$ such that for all $R \in \tau$, $R^{\mathbf{B}} \supseteq f(R^{\mathbf{A}}, \dots, R^{\mathbf{A}})$ where we define the latter to be $\{(f(x_1^{(1)}, \dots, x_1^{(L)}), \dots, f(x_{\mathrm{ar}(R)}^{(1)}, \dots, x_{\mathrm{ar}(R)}^{(L)})) : x^{(1)}, \dots, x^{(L)} \in R^{\mathbf{A}}\}$. In other words, consider any $A^{L \times \mathrm{ar}(R)}$ matrix M, where each *row* is a satisfying assignment in $R^{\mathbf{A}}$. Let $y \in B^{\mathrm{ar}(R)}$ be the result of applying f to each *column* of M. Then, $y \in R^B$. We let $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ denote the set of polymorphisms of $(\mathbf{A}, \mathbf{B})$ (of all arities).

A map $f : A^L \to B$ is said to be *symmetric* if for all $\pi \in S_L$ (the symmetric group on L elements), $f(x_1, \dots, x_L) = f(x_{\pi(1)}, \dots, x_{\pi(L)})$.

2.3. Basic LP and Affine Relaxation. As is well-studied in the CSP literature (e.g., [20, 23]), we consider the canonical linear programming relaxation of a CSP instance, often referred to as the "Basic LP" or "BLP." For our CSP instance $\mathbf{X}$, we represent the assignment $X \to A$ of a variable by a (rational) probability distribution of weights $\{w_i(a)\}_{a \in A}$ summing to 1. We also have a probability distribution over the satisfying assignments to each constraint, which we denote as $p_j(y)$, where $j \in [m]$ is the index of the constraint and $y \in R_j^{\mathbf{A}}$ is the potential assignment. Finally, the marginal distribution of a variable x_i in any constraint has to equal w_i. Explicitly, the linear constraints are as follows.

$$w_i(a) \geq 0 \qquad \text{for all } i \in [n] \text{ and } a \in A \tag{2.1}$$

$$p_j(y) \geq 0 \qquad \text{for all } j \in [m] \text{ and } y \in R_j^{\mathbf{A}} \tag{2.2}$$

$$\sum_{a \in A} w_i(a) = 1 \qquad \text{for all } i \in [n] \tag{2.3}$$

$$\sum_{y \in R_j^{\mathbf{A}}} p_j(y) = 1 \qquad \text{for all } j \in [m] \tag{2.4}$$

$$\sum_{\substack{y \in R_j^{\mathbf{A}} \\ y|_i = a}} p_j(y) = w_i(a) \qquad \begin{array}{l}\text{for all } i \in [n], a \in A, j \in [m] \\ \text{with } x_i \text{ in } \bar{x}^j\end{array}. \tag{2.5}$$

Here $y|_i = a$ denotes that setting $\bar{x}^j = y$ sets $x_i = a$ (that is, if x_i is the k-th variable of the tuple $\bar{x}^j$, then $a = y_k$). We let $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$ denote the rational polytope of solutions. By a theorem of [16] (c.f., [7]), we can efficiently find a *relative interior point* in this polytope. In particular, at such a point, each coordinate is positive if and only if it is positive at some point in the polytope.[1]

In addition to the Basic LP, we also consider the affine relaxation of a Promise CSP. In essence we solve the same linear system, but instead of enforcing each variable to be a nonnegative rational, we enforce that it is an integer (possibly negative). This can be solved in polynomial time via [17] (see also [7] for a more detailed discussion of this approach). We let $r_i(a) \in \mathbb{Z}$ replace $w_i(a)$ for all $a \in A$ and $q_i(y) \in \mathbb{Z}$ replace $p_i(y) \in \mathbb{Q}$ for all $y \in R_j^{\mathbf{A}}$. Explicitly,

$$\sum_{a \in A} r_i(a) = 1 \qquad \text{for all } i \in [n] \tag{2.6}$$

$$\sum_{y \in R_j^{\mathbf{A}}} q_j(y) = 1 \qquad \text{for all } j \in [m] \tag{2.7}$$

$$\sum_{\substack{y \in R_j^{\mathbf{A}} \\ y|_i = a}} q_j(y) = r_i(a) \qquad \begin{array}{l}\text{for all } i \in [n], a \in A, j \in [m] \\ \text{with } x_i \text{ in } \bar{x}^j\end{array}. \tag{2.8}$$

We let $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ denote the integral lattice of solutions.

3. BLP+Affine Algorithm and Analysis for Symmetric Polymorphisms. In the BLP+Affine algorithm, given an instance $\mathbf{X}$ of PCSP-Decision$(\mathbf{A}, \mathbf{B})$, we seek

[1] For our specialized LP, we do not need such a hammer. We can instead solve the LP repeatedly, each time maximizing a different variable as the objective function–a similar idea appears in [6]. Averaging the results would then yield a solution such that each variable is positive if and only if it is positive in some LP solution.

to throw out any assignment to a constraint for which the LP determines to have weight 0. That is, given a relative interior point (w, p) of $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$, we refine $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ to $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ by requiring $r_i(a)$ to be zero whenever $w_i(a)$ is, and requiring $q_i(y)$ to be zero whenever $p_i(y)$ is (by adding equations or just removing those variables from equations defining $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$).

The algorithm is presented in Figure 1. Note it does not depend on $\mathbf{B}$; it is only relevant for the correctness proof.

1. Find a relative interior point in $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$. If no solution exists, **Reject**.
2. Refine $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ to $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ by throwing out assignments to constraints which have weight 0 according to the relative interior point.
3. If $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ is empty, **Reject**. Else, **Accept**.

FIG. 1. *BLP+Affine algorithm*

DEFINITION 3.1. *We say the BLP+Affine algorithm* correctly solves PCSP-Decision$(\mathbf{A}, \mathbf{B})$ *if it accepts any instance* $\mathbf{X}$ *satisfiable in* $\mathbf{A}$ *and rejects any instance unsatisfiable in* $\mathbf{B}$.

As stated in the introduction, both the algorithm and the proof are structured similarly to those of [19] and [7]. Like in those works, the weights of the LP solution and affine relaxation are used to construct a list of assignments which are plugged into the relevant polymorphism. The novel contribution here is that a single argument can cover any infinite symmetric family of polymorphisms.

THEOREM 3.2. *Let* $(\mathbf{A}, \mathbf{B})$ *be a promise template (over any finite domain) such that* $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ *has symmetric polymorphisms of arbitrarily large arities. Then, the BLP+Affine algorithm correctly solves* PCSP-Decision$(\mathbf{A}, \mathbf{B})$.

Proof. If an instance $\mathbf{X}$ is satisfiable in $\mathbf{A}$, then the Basic LP relaxation has a solution. The refinement $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ includes every possible assignment which is in the support of some LP solution, including integral solutions. Thus it is non-empty and therefore the algorithm accepts.

Conversely, suppose the algorithm accepts, meaning both $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$ and $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ have solutions (w, p) over $\mathbb{Q}_{\geq 0}$ and (r, q) over $\mathbb{Z}$. The latter is a solution of $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X}, \mathbf{A})$ such that

$$\begin{aligned} w_i(a) = 0 &\implies r_i(a) = 0 \quad \text{for } i \in [n], a \in A \text{ and} \\ p_j(y) = 0 &\implies q_j(y) = 0 \quad \text{for } j \in [m], y \in R_j^{\mathbf{A}}. \end{aligned}$$

We claim $\mathbf{X}$ is satisfiable in $\mathbf{B}$. Among all the coordinates in the LP solution–the w's and p's–let ℓ be the least common denominator of these rational numbers. Let M be the maximum absolute value of any integer which appears in the affine solution (both the variable weights r and the constraint weights q). Let $f : A^L \to B$ be a symmetric polymorphism of arity $L \geq M\ell^2$. Now write $L = u\ell + v$ where $u \in \mathbb{Z}_{\geq 0}$ and $v \in \{0, \ldots, \ell - 1\}$. Note that $u \geq M\ell$.

For each $i \in [n]$ and $a \in A$, let

$$W_i(a) := u\ell w_i(a) + v r_i(a).$$

This is an integer by choice of ℓ. For a fixed $i \in [n]$, note that by Eq. (2.3) and

(2.6)
$$\sum_{a\in A} W_i(a) = \sum_{a\in A} u\ell w_i(a) + vr_i(a) = u\ell + v = L.$$

Also, for fixed $i \in [n]$ and $a \in A$, either $w_i(a) = 0$, which implies that $r_i(a) = 0$ by the refinement, so $W_i(a) = 0$. Otherwise, $w_i(a) \geq 1/\ell$, so

$$W_i(a) \geq u\ell(1/\ell) + v(-M) \geq M\ell - \ell M = 0.$$

That is, $W_i(a)$ for $a \in A$ are non-negative integers which sum to L. We claim that the assignment

$$X_i := f(\ldots, \underbrace{a, \ldots, a}_{W_i(a) \text{ times } \forall a\in A}, \ldots)$$

to x_i defines a satisfying assignment of $\mathbf{X}$ in $\mathbf{B}$. (Since f is symmetric, only the quantity of each $a \in A$ in the input matters.) To verify it is indeed satisfying, consider a constraint in $(R_j, \bar{x}^j)$ (with $j \in [m]$) and assume without loss of generality it is on variables $\bar{x}^j = (x_1, \ldots, x_k)$. We claim $(X_1, \ldots, X_k) \in R_j^{\mathbf{B}}$.

For every valid assignment $y \in R_j^{\mathbf{A}}$ to that constraint in $\mathbf{A}$, define

$$P_j(y) := u\ell p_j(y) + vq_j(y).$$

By similar logic as before, these are non-negative integers that sum to 1. Indeed, by Eqs. (2.4) and (2.7),

$$\sum_{y\in R_j^{\mathbf{A}}} P_j(y) = u\ell \sum_{y\in R_j^{\mathbf{A}}} p_j(y) + v \sum_{y\in R_j^{\mathbf{A}}} q_j(y) = L.$$

Moreover, either $p_j(y) = q_j(y) = 0$, implying $P_j(y) = 0$, or

$$P_j(y) \geq u\ell(1/\ell) + v(-M) \geq M\ell - \ell M = 0.$$

Further note that by Eqs. (2.5) and (2.8),

$$\begin{aligned} W_i(a) &= u\ell w_i(a) + vr_i(a) \\ &= u\ell \sum_{\substack{y\in R_j^{\mathbf{A}} \\ y|_i=a}} p_i(y) + v \sum_{\substack{y\in R_j^{\mathbf{A}} \\ y|_i=a}} q_i(y) \\ &= \sum_{\substack{y\in R_j^{\mathbf{A}} \\ y|_i=a}} P_j(y) \end{aligned} \tag{3.1}$$

For each $j \in [m]$ consider a matrix $M(j) \in A^{L\times k}$, where exactly $P_j(y)$ of the rows are equal to y. For all $i \in [k]$ and $a \in A$, the number of times that a appears in column i is precisely $W_i(a)$ by Eq. (3.1). Thus, f applied to the columns is precisely $(X_1, \ldots, X_k)$. Since f is a polymorphism, this implies $(X_1, \ldots, X_k) \in R_j^{\mathbf{B}}$. This concludes the proof that assigning the value X_i to each variable x_i (for $i \in [n]$) satisfies $\mathbf{X}$ in $\mathbf{B}$ and hence that the algorithm is correct. ∎

Remark. Another algorithm which works is to solve $\mathrm{LP}_{\mathbb{Z}[\sqrt{2}]}(\mathbf{X}, \mathbf{A})$ (that is the constrained variables are over non-negative elements of the ring $\mathbb{Z}[\sqrt{2}]$) using the algorithm from [7], instead of $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$. In this case, Steps 2 and 3 can be omitted.

To sketch why this works, it suffices to justify why solving $\mathrm{LP}_{\mathbb{Z}[\sqrt{2}]}(\mathbf{X},\mathbf{A})$ also solves $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X},\mathbf{A})$ and $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X},\mathbf{A})$. For each assigned value of the form $a+b\sqrt{2}$ in a relative interior solution to $\mathrm{LP}_{\mathbb{Z}[\sqrt{2}]}(\mathbf{X},\mathbf{A})$, consider changing this variable to $a+b\eta$, where η is a sufficiently good rational approximation of $\sqrt{2}$. Such an assignment is in the relative interior of $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X},\mathbf{A})$ as any inequality non-trivially involving η, in particular (2.1), (2.2), is not tight due to $\sqrt{2}$ being irrational. To see why $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X},\mathbf{A})$ is also satisfied, replace each assigned value of $a+b\sqrt{2}$ with a. By inspection, this assignment (when changing w_i's to r_i's and p_j's to q_j's) satisfies $\mathrm{Aff}_{\mathbb{Z}}(\mathbf{X},\mathbf{A})$. It also satisfies $\mathrm{Aff}'_{\mathbb{Z}}(\mathbf{X},\mathbf{A})$ because $a+b\sqrt{2}=0$ with a and b integral implies $a=0$.

4. Extension of Analysis to Block Symmetric Polymorphisms. We say that a map $f : A^L \to B$. is *block-symmetric* if there exists a partition of the coordinates of f into blocks $B_1 \cup \cdots \cup B_k = [L]$ such that f is permutation-invariant within each coordinate block B_i. We define the *width* of f to be the minimum size of any block.[2] A natural example of a block symmetric polymorphism with nontrivial width is *alternating threshold* first studied in [6]

$$AT(x_1,\ldots,x_L) = 1[x_1 - x_2 + x_3 - \cdots \pm x_L \geq 1].$$

In this case, the blocks are the odd and even coordinates. This polymorphism arises in the context of $\mathbf{A}$ corresponding to 1-in-3 SAT and $\mathbf{B}$ corresponding to NAE-SAT. Recent work shows that this PCSP, although tractable and simple to state, is not algebraically reducible (via so-called pp-constructions) to any tractable finite-domain CSP [2].

We now show an analogue of Theorem 3.2 for block-symmetric polymorphisms. Remarkably, the algorithm is identical to the one for ordinary symmetric polymorphisms and is independent of the number of blocks. In particular, it could be that the Promise CSP has finitely many polymorphisms for any particular number of blocks, yet has infinitely many block-symmetric polymorphisms of increasing width.

As discussed in [7, 15], nearly all known tractable Boolean PCSPs have polymorphisms which are either symmetric (such as threshold functions) or block-symmetric (such as alternating threshold). Thus, except for those PCSPs which are "homomorphic relaxations"[3] of a tractable (P)CSP (c.f., [7, 2]), the algorithm presented here supersedes those works in the context of decision PCSP.

THEOREM 4.1. *Let* $(\mathbf{A},\mathbf{B})$ *be a promise template (over any finite domain) such that* $\mathrm{Pol}(\mathbf{A},\mathbf{B})$ *has block-symmetric polymorphisms of arbitrarily large width. Then, the BLP+Affine algorithm correctly solves* PCSP-Decision$(\mathbf{A},\mathbf{B})$.

Proof. The proof proceeds much like that of Theorem 3.2. As before, we know that if $\mathbf{X}$ is satisfiable in $\mathbf{A}$, then the algorithm rejects. We seek to show that if the algorithm accepts, then $\mathbf{X}$ is satisfiable in $\mathbf{B}$.

[2]Note that a function f might have different partitions into symmetric blocks; we define the width to be the maximum width over all such partitions. In particular, every $f : A^L \to B$ is block-symmetric with width at least 1. Finding the exact width or an appropriate partition into blocks is non-trivial. However, we avoid computing or evaluating f altogether by only considering decision problems; see Section 6 for a discussion of search problems.

[3]A homomorphic relaxation of a PCSP$(\mathbf{A},\mathbf{B})$ is another PCSP$(\mathbf{C},\mathbf{D})$ such that $\mathbf{C}$ has a homomorphism to $\mathbf{A}$ and $\mathbf{B}$ to $\mathbf{D}$. In this case, PCSP$(\mathbf{C},\mathbf{D})$ trivially reduces to PCSP$(\mathbf{A},\mathbf{B})$. In general, if $(\mathbf{C},\mathbf{D})$ is a Boolean template that is a homomorphic relaxation of a tractable non-Boolean (P)CSP template, then this is the only algorithm we know for PCSP$(\mathbf{C},\mathbf{D})$. We leave as an open question finding an explicit Boolean PCSP which is a homomorphic relaxation of a non-Boolean CSP but not correctly solvable by our BLP+Affine algorithm.

Again, let ℓ be the least common denominator of all coordinates in the LP solution. Let M be the maximum absolute value of any integer which appears in the affine solution. Let $f : A^{B_1 \cup \cdots \cup B_\kappa} \to B$ be a block-symmetric polymorphism such that each block B_b, with $b \in [\kappa]$, has size at least $M\ell^2$. Let $L_b = |B_b|$. Similar to before, for all $b \in [\kappa]$, write $L_b = u_b\ell + v_b$ where $u_b \in \mathbb{Z}_{\geq 0}$ and $v \in \{0, \ldots, \ell - 1\}$. Note that $u_b \geq M\ell$.

We seek to show there exists *a homomorphism from* $\mathbf{X}$ *to* $\mathbf{B}$. For each $b \in [\kappa]$, $i \in [n]$ and $a \in A$, let

$$W_{b,i}(a) := u_b\ell w_i(a) + v_b r_i(a).$$

For a fixed $b \in [\kappa]$ and $i \in [n]$, by similar logic to the proof of Theorem 3.2, we have that $W_{b,i}(a)$ are non-negative integers for all $a \in A$ and

$$\sum_{a \in A} W_{b,i}(a) = \sum_{a \in A} \big(u_b\ell w_i(a) + v_b r_i(a)\big) = u_b\ell + v_b = L_b.$$

We now claim that the assignment

$$X_i := f(\ldots, \underbrace{\underbrace{a, \ldots, a}_{W_{1,i}(a) \text{ times}}}_{L_1 \text{ total}}, \ldots, \ldots, \ldots, \underbrace{\underbrace{a, \ldots, a}_{W_{\kappa,i}(a) \text{ times}}}_{L_\kappa \text{ total}}, \ldots)$$

to x_i defines a satifying assigment of $\mathbf{X}$ in $\mathbf{B}$. To verify this, consider a constraint in $(R_j, \bar{x}^j)$ (with $j \in [m]$) and assume without loss of generality it is on variables $\bar{x}^j = (x_1, \ldots, x_k)$. We claim $(X_1, \ldots, X_k) \in R_j^{\mathbf{B}}$. For all $b \in [\kappa]$ and assignments $y \in R_j^{\mathbf{A}}$ define

$$P_{b,j}(y) := u_b\ell p_j(y) + v_b q_j(y).$$

By similar logic as previously, $P_{b,j}(y)$ are non-negative integers and by Eqs. (2.4) and (2.7),

$$\sum_{y \in R_j^{\mathbf{A}}} P_{b,j}(y) = u_b\ell \sum_{y \in R_j^{\mathbf{A}}} p_j(y) + v_b \sum_{y \in R_j^{\mathbf{A}}} q_j(y) = L_b.$$

Further note that by Eqs. (2.5) and (2.8) for $i \in [n], a \in A$, and $j \in [m]$

$$\begin{aligned} W_{b,i}(a) &= u_b\ell \sum_{\substack{y \in R_j^{\mathbf{A}} \\ y|_i = a}} p_j(y) + v_b \sum_{\substack{y \in R_j^{\mathbf{A}} \\ y|_i = a}} q_j(y) \\ &= \sum_{\substack{y \in R_j^{\mathbf{A}} \\ y|_i = a}} P_{b,j}(y) \end{aligned} \tag{4.1}$$

For each $j \in [m]$ consider a matrix $M(j) \in A^{L \times k}$, where exactly $P_{b,j}(y)$ of the rows are equal to y in the rows indexed by block B_b. For all $i \in [k]$ and $a \in A$, the number of times that a appears in column i and row-block B_b is precisely $W_{b,i}(a)$ by Eq. (4.1). Thus, f applied to the columns is precisely $(X_1, \ldots, X_k)$. Since f is a polymorphism, this implies $(X_1, \ldots, X_k) \in R_j^{\mathbf{B}}$. This concludes the proof that the algorithm is correct. ∎

5. Characterizing the Algorithm's Power. In this section, we characterize the power of the BLP+Affine algorithm from Figure 1 exactly. Recall, we denote the domains of relational structures $\mathbf{A}, \mathbf{B}, \mathbf{X}$ as A, B, X.

THEOREM 5.1. *Let* $(\mathbf{A}, \mathbf{B})$ *be a promise template. The following are equivalent:*
- *BLP+Affine algorithm correctly solves* PCSP-Decision$(\mathbf{A}, \mathbf{B})$.
- $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ *has block-symmetric polymorphisms of arbitrarily high width.*
- *For every* $L \in \mathbb{N}$, $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ *has a block-symmetric polymorphism of arity* $2L+1$ *with two symmetric blocks of variables of size* L *and* $L+1$*, respectively.*

We need a few definitions and fundamental facts from [12, 2]. For an L-ary function $f\colon A^L \to B$ and a function $\pi\colon [L] \to [L']$, the *minor* of f obtained from π is the function $g\colon A^{L'} \to B$ defined as

$$g(x_1, \dots, x_{L'}) := f(x_{\pi(1)}, \dots, x_{\pi(L)}). \tag{5.1}$$

We write $g = f_{/\pi}$. Thus sets of polymorphisms $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ are equipped with an operation $(\cdot)_{/\pi}$ which maps L-ary polymorphisms to L'-ary polymorphisms (for every $\pi\colon [L] \to [L']$). We consider such a structure more abstractly, allowing any objects to play the role of polymorphisms:

DEFINITION 5.2. *A* minion $\mathcal{M}$ *consists of sets* $\mathcal{M}^{(L)}$ *for* $L \in \mathbb{N}$ *and functions* $(\cdot)_{/\pi}\colon \mathcal{M}^{(L)} \to \mathcal{M}^{(L')}$ *for all functions* $\pi\colon [L] \to [L']$*, such that compositions agree:* $(f_{/\pi})_{/\tau} = f_{/\tau\circ\pi}$ *for* $\pi\colon [L] \to [L']$*,* $\tau\colon [L'] \to [L'']$*, and* $f_{/\,\mathrm{id}} = f$*. We write* $\mathcal{M}$ *for the disjoint union of* $\mathcal{M}^{(L)}$*,* $L \in \mathbb{N}$*, and* $\mathrm{ar}(f) = L$ *for* $f \in \mathcal{M}^{(L)}$*. A* minion homomorphism $\xi\colon \mathcal{M} \to \mathcal{N}$ *is a function which preserves arity and minors:* $\mathrm{ar}(\xi(f)) = \mathrm{ar}(f)$ *and* $\xi(f_{/\pi}) = \xi(f)_{/\pi}$ *for all functions* $\pi\colon [L] \to [L']$*.*

Note that the objects in a minion do not have to be functions, and the set $\mathcal{M}^{(L)}$ does not have to be finite, though this is true for minions $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ with finite $\mathbf{A}, \mathbf{B}$. Similarly the operations $(\cdot)_{/\pi}$ are not necessarily defined by Eq. (5.1), though this will always be the case when elements of a minion $f \in \mathcal{M}^{(L)}$ are L-ary function. As an important example, consider the minion $\mathcal{Q}_{conv}$ of convex combination functions, i.e. functions $\mathbb{Q}^L \to \mathbb{Q}$ of the form $w_1x_1 + \dots + w_Lx_L$ for $\sum_1^L w_i = 1$, $w_i \in \mathbb{Q}_{\geq 0}$, with $(\cdot)_{/\pi}$ defined by Eq. (5.1). We can describe the same minion more concisely by identifying a convex L-ary function with its L-tuple of coefficients $(w_1, \dots, w_L)$. That is, the "L-ary objects" of the minion $\mathcal{Q}_{conv}$ can be equivalently defined as distributions on $[L]$:

$$\mathcal{Q}_{conv}^{(L)} = \{w\colon [L] \to \mathbb{Q}_{\geq 0} \ \mid \ \textstyle\sum_{i\in[L]} w(i) = 1\},$$

and for $\pi\colon [L] \to [L']$ and $w \in \mathcal{Q}_{conv}^{(L)}$ one can define $w_{/\pi}$ as

$$w_{/\pi}(i) := w(\pi^{-1}(i)) = \textstyle\sum_{j\in\pi^{-1}(i)} w(j) \quad \text{for } i \in [L'].$$

This minion characterizes the power of the basic linear programming relaxation in the sense that BLP correctly solves PCSP-Decision$(\mathbf{A}, \mathbf{B})$ (i.e. feasibility of $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$ implies $\mathbf{X}$ is satisfiable in $\mathbf{B}$ for all instances $\mathbf{X}$) if and only if $\mathcal{Q}_{conv}$ admits a minion homomorphism to $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$. This was shown by Barto et al. [2, Theorem 7.9]. Our proof straightforwardly extends this part of the argument.

We first define the minion that plays the role of $\mathcal{Q}_{conv}$ for the BLP+Affine relaxation. It assigns two coefficients to every coordinate $i \in [L]$.

DEFINITION 5.3. *The minion* $\mathcal{M}_{\mathrm{BLP+Aff}}$ *is defined as follows: for* $L \in \mathbb{N}$*, its "*L*-ary objects" are*

$$\mathcal{M}_{\mathrm{BLP+Aff}}^{(L)} := \{(w, r) \mid w\colon [L] \to \mathbb{Q}_{\geq 0}, \qquad \textstyle\sum_{i\in[L]} w(i) = 1$$

$$r\colon [L] \to \mathbb{Z}, \qquad \textstyle\sum_{i\in[L]} r(i) = 1$$
$$\forall_{i\in[L]} \quad w(i) = 0 \implies r(i) = 0 \ \}.$$

Equivalently, these could be seen as a function from $[L]$ *to* $\{(a,b) \in \mathbb{Q}_{\geq 0} \times \mathbb{Z} : a = 0 \implies b = 0\}$.

For $\pi\colon [L] \to [L']$ *and* $(w,r) \in \mathcal{M}^{(L)}_{\mathrm{BLP+Aff}}$*, we define the minor* $(w,r)_{/\pi}$ *as* (w',r')*, where*

$$w'(i) := w(\pi^{-1}(i)) = \textstyle\sum_{j\in\pi^{-1}(i)} w(j)$$
$$r'(i) := r(\pi^{-1}(i)), \qquad \textit{for } i \in [L'].$$

It is easy to check this indeed defines a minion (the $w(i) = 0 \implies r(i) = 0$ condition is preserved when taking a minor and composition of minors works as expected). One could also think of a pair $(w,r) \in \mathcal{M}^{(L)}_{\mathrm{BLP+Aff}}$ as an L-ary function on $\mathbb{Q}^2$, $f(\binom{x_1}{y_1}, \ldots, \binom{x_n}{y_n}) = \binom{\sum w(i)x_i}{\sum r(i)x_i}$.

The minion $\mathcal{M}_{\mathrm{BLP+Aff}}$ characterizes the BLP+Affine relaxation as follows.

Lemma 5.4. *Let* $(\mathbf{A},\mathbf{B})$ *be a promise template. The following are equivalent:*
- *BLP+Affine correctly solves* PCSP-Decision$(\mathbf{A},\mathbf{B})$ *(Definition 3.1).*
- $\mathcal{M}_{\mathrm{BLP+Aff}}$ *admits a minion homomorphism to* $\mathrm{Pol}(\mathbf{A},\mathbf{B})$.

As the proof of this lemma directly extends the arguments by Barto et al. [2], we refer the reader to Appendix A for an exposition of it.

We now reinterpret this last condition in terms of concrete polymorphisms. One direction is simple:

Lemma 5.5. *Suppose* $\mathcal{M}_{\mathrm{BLP+Aff}}$ *has a minion homomorphism to some minion* $\mathcal{N} = \mathrm{Pol}(\mathbf{A},\mathbf{B})$*. Then for every* $L \in \mathbb{N}$*,* $\mathcal{N}$ *contains a block-symmetric polymorphism of arity* $2L+1$ *with two blocks of size* L *and* $L+1$*.*

Proof. Given $L \in \mathbb{N}$, consider the following object $(w,r) \in \mathcal{M}^{(2L+1)}_{\mathrm{BLP+Aff}}$: take $w(i) := \frac{1}{2L+1}$ and $r(i) := (-1)^{i+1}$ for $i = 1, \ldots, 2L+1$. For every permutation $\pi\colon [2L+1] \to [2L+1]$ which maps odd coordinates to odd coordinates (and even to even), $(w,r)_{/\pi} = (w,r)$. Thus the image of (w,r) in $\mathcal{N}$ has the same property, i.e. it has arity $2L+1$ and it is symmetric on odd coordinates as well as on even coordinates. ∎

We remark the above lemma in fact applies to any minion $\mathcal{N}$, not only those of the form $\mathrm{Pol}(\mathbf{A},\mathbf{B})$; one can define $f \in \mathcal{N}^{(L)}$ to be block-symmetric with blocks $B_1 \cup \cdots \cup B_k = [L]$ if $f_{/\pi} = f$ holds for all permutations π of $[L]$ that preserve the blocks; the proof then applies without change.

The idea for the other direction is essentially the same as in the proof of Theorem 3.2 and 4.1. We apply it to construct a minion homomorphism from every finite subset of $\mathcal{M}_{\mathrm{BLP+Aff}}$ and use a compactness argument.

Lemma 5.6. *Suppose the minion* $\mathcal{N} = \mathrm{Pol}(\mathbf{A},\mathbf{B})$ *(for* $\mathbf{A},\mathbf{B}$ *finite) contains block-symmetric polymorphisms of arbitrarily high width. Then* $\mathcal{M}_{\mathrm{BLP+Aff}}$ *admits a minion homomorphism to* $\mathcal{N}$*.*

Proof. To avoid cumbersome notation we present the proof only for the case of one block, i.e. we assume that $\mathcal{N}$ contains symmetric polymorphisms of arbitrarily high arity. This extends to more blocks just as Theorem 4.1 extends Theorem 3.2.

We define finite subsets of $\mathcal{M}_{\mathrm{BLP+Aff}}$ as follows. For $L, \ell, M \in \mathbb{N}$, let $\mathcal{M}^{(L)}_{\ell,M}$ be the subset of those $(w,r) \in \mathcal{M}^{(L)}_{\mathrm{BLP+Aff}}$ such that $\ell w(i) \in \mathbb{Z}$ for $i \in [L]$ and

$\sum_i |r(i)| \leq M$. Observe that $\mathcal{M}^{(L)}_{\ell,M}$ is a finite set (since the numbers $\ell w(i)$ are L non-negative integers summing to ℓ and the numbers $r(i)$ are L integers between $-M$ and M). Denote $\mathcal{M}_{\ell,M} := \bigcup_{L\in\mathbb{N}} \mathcal{M}^{(L)}_{\ell,M}$.

For fixed ℓ, M, we define a minion homomorphism from $\mathcal{M}_{\ell,M}$ to $\mathcal{N}$ as follows. Let $f \in \mathcal{N}$ be a function of some arity $L^* \geq M\ell^2$. Let $u, v \in \mathbb{N}$ be numbers such that $L^* = u\ell + v$, $v \in \{0, \dots, \ell - 1\}$. Then $u \geq M\ell$.

Take $L \in \mathbb{N}$ and $(w, r) \in \mathcal{M}^{(L)}_{\ell,M}$. For $i \in [L]$, the number $W_i := u\ell w(i) + vr(i)$ is a non-negative integer. Since $\sum_i W_i = u\ell + v = L^*$, we can map (w, r) to the L-ary minor $g := f(x_1, x_1, x_1, \dots, x_L, x_L)$ of the L^*-ary function f where x_i is repeated W_i times, for $i \in [L]$. We claim that this map is a minion homomorphism from $\mathcal{M}_{\ell,M}$ to $\mathcal{N}$ (in fact to the subminion of minors of f). Indeed, for $\pi\colon [L] \to [L']$, consider the minor $g_{/\pi}$ of g identifying x_j for $j \in \pi^{-1}(i)$ into a single variable z_i (for $i \in [L']$). We have that $g_{/\pi}$ is also a minor of f where z_i is repeated $\sum_{j\in\pi^{-1}(i)} W_j$ times. That is, z_i is repeated $u\ell w(\pi^{-1}(i)) + vr(\pi^{-1}(i))$ times. By symmetry of f the ordering does not matter, thus $g_{/\pi}$ (the minor of the image of f) is the same as the image of the minor $f_{/\pi}$.

We conclude with a compactness argument similar to that of Remark 7.13 in [2]. For $k \in \mathbb{N}$, let $\mathcal{M}_k := \bigcup_{L\leq k} \mathcal{M}^{(L)}_{k!,k}$. Then $\mathcal{M}_k$ is finite, $\mathcal{M}_k \subseteq \mathcal{M}_{k+1}$ (because $k! \cdot w(i) \in \mathbb{Z}$ implies $(k+1)! \cdot w(i) \in \mathbb{Z}$) and $\bigcup_{k\in\mathbb{N}} \mathcal{M}_k = \mathcal{M}_{\text{BLP+Aff}}$. Consider the possible minion homomorphisms from $\mathcal{M}_k$ to $\mathcal{N}$, or more precisely, restrictions of homomorphisms obtained above to $\mathcal{M}_k$ (since $\mathcal{M}_k$ itself is technically not a minion). There are only finitely many possible such restrictions $\mathcal{M}_k \to \mathcal{N}$, because $\mathcal{M}_k$ is finite, the arities of images in $\mathcal{N}$ are bounded, and hence the number of possible images in $\mathcal{N}$ is also finite. Consider an infinite tree with restrictions from any $\mathcal{M}_k$ to $\mathcal{N}$ as nodes, the trivial map from $\mathcal{M}_0 = \emptyset$ being the root, and the parent of a function $\mathcal{M}_{k+1} \to \mathcal{N}$ being its restriction to $\mathcal{M}_k$. This is an infinite tree (because for each k we have some minion homomorphism from a superset of $\mathcal{M}_k$ to $\mathcal{N}$) that is connected (because everyone is connected through its ancestors to the root) and finitely branching (because there are only finitely many restrictions $\mathcal{M}_k \to \mathcal{N}$, for any fixed k). Therefore, by Kőnig's lemma, the tree contains an infinite path $\zeta_k\colon \mathcal{M}_k \to \mathcal{N}$ of homomorphisms that are restrictions of each other. Their union is then a homomorphism from $\bigcup_{k\in\mathbb{N}} \mathcal{M}_k = \mathcal{M}_{\text{BLP+Aff}}$ to $\mathcal{N}$. ∎

(We remark the above proof in fact applies to any minion $\mathcal{N}$, assuming $\mathcal{N}^{(L)}$ is finite for every L.) Lemmas 5.4, 5.5, and 5.6 conclude the proof of Theorem 5.1.

6. Concluding Thoughts. We conclude with a few natural directions of future inquiry raised by this work.

Inspecting the proofs of Theorems 3.2 and 4.1, in order to yield a search algorithm (and not just a decision algorithm), it would suffice to compute:

$$X_i := f(\dots, \underbrace{a, \dots, a}_{W_i(a) \text{ times}}, \dots)$$

for some block-symmetric polymorphism f and a fixed partition into blocks of size at least L, for an integer L which depends polynomially on the least common denominator of rational numbers in the LP solution and the maximum absolute value of integers in the affine solution. In previous work [7], Brakensiek and Guruswami circumvented this problem by assuming that f has special structure (such as being a threshold function, etc.). Even then, we often only assumed that you had oracle access to the structure of f. Thus, except for some simple cases studied in the paper,

truly polynomial-time search algorithms remain elusive. Perhaps one could hope for a search algorithm like the decision algorithm presented in this paper which is oblivious to the underlying polymorphisms (as long as they are symmetric/block-symmetric).

Question. Is there an "oblivious" polynomial-time algorithm for the search version of Promise CSPs with infinitely many symmetric polymorphisms?

We note that an oblivious polynomial-time algorithm is also not known for the search version of Promise CSPs with symmetric polymorphisms of all arities (which capture the power of BLP [2, Theorem 7.9]) and for the search version of Promise CSPs with alternating polymorphisms of all odd arities (which capture the power of the affine relaxation [2, Theorem 7.19]).

Otherwise, one could hope to prove a "structure theorem" that every Promise CSP with infinitely many symmetric polymorphisms also has an infinite threshold-periodic family. As [7] shows, such polymorphisms can get exceedingly complicated, suggesting that such a characterization may only be possible in the Boolean case.

Question. Does every Boolean PCSP with infinitely many symmetric polymorphisms have an infinite threshold-periodic family?

Even without a structure theorem, one could perhaps hope to compute the pertinent values of f "on the fly," but this seems difficult in our current formulation as the arity of f could be exponentially large in the input size!

While Theorem 5.1 characterizes the power of the BLP+Affine algorithm, it is still worthwhile to ask how this compares to other classes of templates, in particular those studied for non-promise CSPs. The following example of a simple template not solved by the BLP+Affine relaxation was communicated to us by Jakub Opršal.

Example 6.1. Let $\mathbf{A}$ be the disjoint union of a directed 2-cycle $\{0, 1\}$ and a directed 3-cycle $\{0', 1', 2'\}$. Then $\mathbf{A}$ is tractable template (i.e. $\mathrm{PCSP}(\mathbf{A}, \mathbf{A})$ is solvable in polynomial time, in fact $\mathrm{Pol}(\mathbf{A}, \mathbf{A})$ has cyclic polymorphisms of every prime arity $p > 3$) but has no non-trivial block-symmetric polymorphisms.

Proof. To see it admits no block-symmetric polymorphisms f of width greater than one, observe that every such width can be represented as $2n + 3n'$ for some $n, n' \in \mathbb{N}$, hence every block can be filled with n copies of values $0, 1$ and n' copies of $0', 1', 2'$, giving some input $\bar{v}$ to f. But f should give the same output on the input $\bar{v}^{\oplus 1}$ consisting of n copies of $1, 0$ and n' copies of $1', 2', 0'$. Since $(v_i, v_i^{\oplus 1})$ is an arc of $\mathbf{A}$ for every i and since f is a polymorphism, $(f(\bar{v}), f(\bar{v}^{\oplus 1}))$ would be a loop in $\mathbf{A}$, a contradiction.

We now observe that $\mathrm{PCSP}(\mathbf{A}, \mathbf{A})$ has a straightforward polynomial time algorithm. For each connected component of constraints, the variables must map to either $\{0, 1\}$ or $\{0', 1', 2'\}$. The first case is equivalent to testing if the graph of constraints is bipartite. The latter can be done by a breath-first search which checks that all directed cycles have length a multiple of 3. ∎

Thus the condition of having block-symmetric polymorphisms of high width is not preserved under disjoint union, even though tractability is. We also know that since $\mathrm{Pol}(\mathbf{A}, \mathbf{A})$ has a majority polymorphism (simply let $f(x, y, z)$ output x if $x = y$ and z otherwise), $\mathrm{PCSP}(\mathbf{A}, \mathbf{A})$ can be solved in polynomial time via the $(2, 3)$-consistency algorithm, 3-rounds of Sherali-Adams, or the canonical SDP relaxation (see also [3, 23, 4]). Informally, these relaxations ensure that there are locally consistent assignments to every (constant-sized) subset of variables. This consistency is quite powerful. For instance, 2-SAT can be solved by the BLP+Affine relaxation or 3 rounds of Sherali-Adams, but not the BLP by itself. This suggests the tantalising possibility that an analogous hierarchy could provide a uniform algorithm for all tractable non-promise

CSPs.

Question. Which (decision) promise CSPs can be solved via constantly many rounds of the Sherali-Adams hierarchy for the BLP+Affine relaxation? Does this capture all tractable non-promise CSPs?

Acknowledgments. We thank Libor Barto, Andrei Krokhin, and Jakub Opršal for useful comments and encouragement. We also thank anonymous reviewers for many helpful comments.

Appendix A. From Relaxations to Minion Homomorphisms. In this appendix, we recall the definition of the minion $\mathcal{Q}_{conv}$ and prove Lemma 5.4 from Section 5. We do this by explaining how free structures relate BLP and Affine relaxations to minions. We carry over the notation from Section 5.

Definition A.1. *The minion $\mathcal{Q}_{conv}$ is defined as follows: for $L \in \mathbb{N}$, the "L-ary object" of the minion are*

$$\mathcal{Q}_{conv}^{(L)} := \{\ w\colon [L] \to \mathbb{Q}_{\geq 0}\ \mid\ \textstyle\sum_{i\in[L]} w(i) = 1\ \};$$

for $\pi\colon [L] \to [L']$ and $w \in \mathcal{Q}_{conv}^{(L)}$, we define the minor $w_{/\pi}$ of w as

$$w_{/\pi}(i) = w(\pi^{-1}(i)) = \textstyle\sum_{j\in\pi^{-1}(i)} w(j).$$

Let us describe how $\mathcal{Q}_{conv}$ characterizes the power of the basic linear programming relaxation; the case of BLP+Affine will be entirely analogous. Recall that for an instance $\mathbf{X}$ of PCSP$(\mathbf{A},\mathbf{B})$, a solution to the BLP relaxation assigns to each variable $i \in X$ a distribution $w_i\colon A \to \mathbb{Q}_{\geq 0}$ with $\sum_{a\in A} w_i(a) = 1$. It also assigns to each constraint j of $\mathbf{X}$ a distribution over satisfying assignments $p_j\colon R^A \to \mathbb{Q}_{\geq 0}$ with sum 1. Finally, the relaxation requires that for a variable i in a constraint j of $\mathbf{X}$, the assignment of $a \in A$ to i has value $w_i(a) = \sum_y p_j(y)$, where the sum runs over all satisfying assignments $y \in R^A$ of the constraint where the variable i takes value a.

In other words, $w_i(a) = p_j(\pi^{-1}(a))$, where $\pi = \pi_{j\to i}\colon R^A \to A$ maps a satisfying assignment y to the value of variable i in constraint j. That is, w_i, as an object of $\mathcal{Q}_{conv}^{|A|}$, is required to be the minor of $p_j \in \mathcal{Q}_{conv}^{|R^A|}$ obtained from π. Thus the BLP relaxation of $\mathbf{X}$ is satisfiable if and only if one can assign some $w_i \in \mathcal{Q}_{conv}^{|A|}$ to each variable $i \in X$ so that the following holds for every constraint j of $\mathbf{X}$: there is a $p_j \in \mathcal{Q}_{conv}^{|R^A|}$ such that for all variables i in j, $w_i = p_j/\pi_{j\to i}$. This can be phrased as the existence of a homomorphism from $\mathbf{X}$ to the free structure $\mathbb{F}_{\mathcal{Q}_{conv}}$, defined as follows.

Definition A.2. *For a relational structure $\mathbf{A}$ and a minion $\mathcal{M}$, the* free structure $\mathbb{F}_{\mathcal{M}}(\mathbf{A})$ *is a template with domain $\mathcal{M}^{|A|}$ (potentially infinite) and with the same signature as $\mathbf{A}$. For each relation R^A of arity k in $\mathbf{A}$, there is a relation $R^{\mathbb{F}}$ of the same arity in $\mathbb{F}_{\mathcal{M}}(\mathbf{A})$ defined as follows: $w_1,\dots,w_k \in \mathcal{M}^{(|A|)}$ are in the relation $R^{\mathbb{F}}$ if there is some $p \in \mathcal{M}^{(|R^A|)}$ such that for each $i \in [k]$, $w_i = p_{/\pi_i}$. Here $\pi_i\colon R^A \to A$ maps $y \in R^A \subseteq A^k$ to its i-th coordinate.*

The above discussion shows that:

Observation A.3. The BLP relaxation of $(\mathbf{X},\mathbf{A})$ has a solution if and only if $\mathbf{X}$ is satisfiable in $\mathbb{F}_{\mathcal{Q}_{conv}}(\mathbf{A})$.

Just as in Definition 3.1, we say that "BLP *correctly solves* PCSP-Decision$(\mathbf{A},\mathbf{B})$" if for every instance $\mathbf{X}$, feasibility of the LP$_{\mathbb{Q}}(\mathbf{X},\mathbf{A})$ implies satisfiability of $\mathbf{X}$ in

$\mathbf{B}$. (Note the other direction is always trivially true: if $\mathbf{X}$ is satisfiable in $\mathbf{A}$, then the relaxation $\mathrm{LP}_{\mathbb{Q}}(\mathbf{X}, \mathbf{A})$ has a solution). Let us write $\mathbf{X} \to \mathbf{A}$ if there exists a homomorphism from $\mathbf{X}$ to $\mathbf{A}$ (i.e. a satisfying assignment); we can now restate the definition.

Observation A.4. Let $(\mathbf{A}, \mathbf{B})$ be a promise template. The following are equivalent:
- BLP correctly solves PCSP-Decision$(\mathbf{A}, \mathbf{B})$;
- for every instance $\mathbf{X}$, $\mathbf{X} \to \mathbb{F}_{\mathcal{Q}_{conv}}(\mathbf{A})$ implies $\mathbf{X} \to \mathbf{B}$.

Entirely analogously, we can restate what it means for BLP+Affine to solve a PCSP (Definition 3.1), by using the minion $\mathcal{M}_{\mathrm{BLP+Aff}}$ (Definition 5.3).

Proposition A.5. *Let $(\mathbf{A}, \mathbf{B})$ be a promise template. The following are equivalent:*
- *BLP+Affine correctly solves* PCSP-Decision$(\mathbf{A}, \mathbf{B})$*;*
- *for every instance $\mathbf{X}$, $\mathbf{X} \to \mathbb{F}_{\mathcal{M}_{\mathrm{BLP+Aff}}}(\mathbf{A})$ implies $\mathbf{X} \to \mathbf{B}$.*

The resulting condition can be simplified by a standard compactness argument. That is, we use the following straightforward generalization of the de Bruijn–Erdős Theorem (see e.g. [13, Theorem 8.1.3] for a discussion and short proofs, [21] for general relational structures).

Lemma A.6 (Compactness for structures). *Let $\mathbf{F}, \mathbf{B}$ be relational structures with F infinite and B finite. If every finite induced substructure of $\mathbf{F}$ admits a homomorphism to $\mathbf{B}$, then so does $\mathbf{F}$.*

That is, for a promise template $(\mathbf{A}, \mathbf{B})$ and any minion $\mathcal{M}$, the following are equivalent:
- for every instance $\mathbf{X}$, $\mathbf{X} \to \mathbb{F}_{\mathcal{M}}(\mathbf{A})$ implies $\mathbf{X} \to \mathbf{B}$;
- $\mathbb{F}_{\mathcal{M}}(\mathbf{A}) \to \mathbf{B}$.

A fundamental property of free structures is that the latter condition is equivalent to the existence of a minion homomorphism, as proved by Barto et al. [2, Lemma 4.4].

Lemma A.7 ([2]). *Let $(\mathbf{A}, \mathbf{B})$ be a promise template and let $\mathcal{M}$ be any minion. The following are equivalent:*
- $\mathbb{F}_{\mathcal{M}}(\mathbf{A}) \to \mathbf{B}$*;*
- *there exists a minion homomorphism from $\mathcal{M}$ to* $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$.

Altogether, this shows that BLP+Affine solves PCSP$(\mathbf{A}, \mathbf{B})$ if and only if $\mathcal{M}_{\mathrm{BLP+Aff}}$ admits a minion homomorphism to $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$. This concludes the proof of Lemma 5.4 in Section 5.

We remark that Barto et al. [2, Theorem 7.9] used the same argument to characterize the power of BLP for PCSPs.

Theorem A.8 ([2]). *Let $(\mathbf{A}, \mathbf{B})$ be a promise template. The following are equivalent:*
- *BLP solves* PCSP$(\mathbf{A}, \mathbf{B})$ *(as in Definition 3.1),*
- $\forall_{\mathbf{X}} \;\; \mathbf{X} \to \mathbb{F}_{\mathcal{Q}_{conv}}(\mathbf{A}) \implies \mathbf{X} \to \mathbf{B}$,
- $\mathbb{F}_{\mathcal{Q}_{conv}}(\mathbf{A}) \to \mathbf{B}$,
- $\mathcal{Q}_{conv}$ *admits a minion homomorphism to* $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$,
- $\mathrm{Pol}(\mathbf{A}, \mathbf{B})$ *contains symmetric polymorphisms of every arity.*

Our argument thus only differs in the equivalence of the last two bullets, an analogue of which is proved in Section 5. Finally, let us note that in [2, Theorem 7.19], the power of the Affine relaxation alone was similarly characterized by the minion $\mathcal{Z}_{\mathrm{aff}}$, defined analogously to $\mathcal{Q}_{conv}$, except with integer coefficients (not necessarily non-negative): the L-ary objects are $r\colon [L] \to \mathbb{Z}$ such that $\sum_{i \in [L]} r(i) = 1$.

REFERENCES

[1] P. Austrin, V. Guruswami, and J. Håstad, *(2+(ϵ))-Sat Is NP-hard*, SIAM J. Comput., 46 (2017), pp. 1554–1573, https://doi.org/10.1137/15M1006507.

[2] L. Barto, J. Bulín, A. Krokhin, and J. Opršal, *Algebraic approach to promise constraint satisfaction*, arXiv:1811.00970 [cs, math], (2019).

[3] L. Barto and M. Kozik, *Constraint Satisfaction Problems Solvable by Local Consistency Methods*, Journal of the ACM, 61 (2014), pp. 3:1–3:19, https://doi.org/10.1145/2556646.

[4] L. Barto, A. Krokhin, and R. Willard, *Polymorphisms, and How to Use Them*, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany, (2017), https://doi.org/10.4230/dfu.vol7.15301.1.

[5] L. Barto, J. Opršal, and M. Pinsker, *The wonderland of reflections*, Israel Journal of Mathematics, 223 (2018), pp. 363–398, https://doi.org/10.1007/s11856-017-1621-9.

[6] J. Brakensiek and V. Guruswami, *Promise Constraint Satisfaction: Structure Theory and a Symmetric Boolean Dichotomy*, in Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018, A. Czumaj, ed., SIAM, 2018, pp. 1782–1801, https://doi.org/10.1137/1.9781611975031.117. Full version available as ECCC TR16-183.

[7] J. Brakensiek and V. Guruswami, *An Algorithmic Blend of LPs and Ring Equations for Promise CSPs*, in Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '19, Philadelphia, PA, USA, 2019, Society for Industrial and Applied Mathematics, pp. 436–455.

[8] J. Brakensiek and V. Guruswami, *Symmetric polymorphisms and efficient decidability of PCSPs*, in Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'20), SIAM, 2020, pp. 297–304, https://doi.org/10.1137/1.9781611975994.18, https://arxiv.org/abs/1907.04383. arXiv:1907.04383.

[9] A. Bulatov and V. Dalmau, *A Simple Algorithm for Mal'tsev Constraints*, SIAM J. Comput., 36 (2006), pp. 16–27, https://doi.org/10.1137/050628957.

[10] A. A. Bulatov, *Mal'tsev constraints are tractable*, Electronic Colloquium on Computational Complexity (ECCC), (2002).

[11] A. A. Bulatov, *A Dichotomy Theorem for Nonuniform CSPs*, in 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017, C. Umans, ed., IEEE Computer Society, 2017, pp. 319–330, https://doi.org/10.1109/FOCS.2017.37. arXiv:1703.03021.

[12] J. Bulín, A. Krokhin, and J. Opršal, *Algebraic Approach to Promise Constraint Satisfaction*, in Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, New York, NY, USA, 2019, ACM, pp. 602–613, https://doi.org/10.1145/3313276.3316300.

[13] R. Diestel, *Graph Theory*, Springer-Verlag, Electronic Edition, 2016, http://diestel-graph-theory.com/.

[14] T. Feder and M. Y. Vardi, *The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory*, SIAM J. Comput., 28 (1998), pp. 57–104, https://doi.org/10.1137/S0097539794266766.

[15] M. Ficak, M. Kozik, M. Olšák, and S. Stankiewicz, *Dichotomy for Symmetric Boolean PCSPs*, in 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), C. Baier, I. Chatzigiannakis, P. Flocchini, and S. Leonardi, eds., vol. 132 of Leibniz International Proceedings in Informatics (LIPIcs), Dagstuhl, Germany, 2019, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, pp. 57:1–57:12, https://doi.org/10.4230/LIPIcs.ICALP.2019.57.

[16] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, vol. 2, Springer Science & Business Media, 1993.

[17] R. Kannan and A. Bachem, *Polynomial Algorithms for Computing the Smith and Hermite Normal Forms of an Integer Matrix*, SIAM Journal on Computing, 8 (1979), pp. 499–507, https://doi.org/10.1137/0208040.

[18] A. Krokhin and J. Opršal, *The complexity of 3-colouring H-colourable graphs*, in Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS'19), IEEE, 2019, https://arxiv.org/abs/1904.03214. arXiv:1904.03214.

[19] G. Kun, R. O'Donnell, S. Tamaki, Y. Yoshida, and Y. Zhou, *Linear programming, width-1 CSPs, and robust satisfaction*, in Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012, S. Goldwasser, ed., ACM, 2012, pp. 484–495, https://doi.org/10.1145/2090236.2090274.

[20] P. Raghavendra and D. Steurer, *How to Round Any CSP*, in 2009 50th Annual IEEE

Symposium on Foundations of Computer Science, Oct. 2009, pp. 586–594, https://doi.org/10.1109/FOCS.2009.74.

[21] D. Rorabaugh, C. Tardif, and D. L. Wehlau, *Logical compactness and constraint satisfaction problems*, Log. Methods Comput. Sci., 13 (2017), https://doi.org/10.23638/LMCS-13(1:1)2017.

[22] T. J. Schaefer, *The Complexity of Satisfiability Problems*, in Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, STOC '78, ACM, 1978, pp. 216–226, https://doi.org/10.1145/800133.804350.

[23] J. Thapper and S. Živný, *The Power of Sherali–Adams Relaxations for General-Valued CSPs*, SIAM Journal on Computing, 46 (2017), pp. 1241–1279, https://doi.org/10.1137/16M1079245.

[24] M. Wrochna and S. Živný, *Improved hardness for H-colourings of G-colourable graphs*, in Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'20), SIAM, 2020, pp. 1426–1435, https://doi.org/10.1137/1.9781611975994.86, https://arxiv.org/abs/1907.00872. arXiv:1907.00872.

[25] D. Zhuk, *A Proof of CSP Dichotomy Conjecture*, in 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017, C. Umans, ed., IEEE Computer Society, 2017, pp. 331–342, https://doi.org/10.1109/FOCS.2017.38. arXiv:1704.01914.