Beyond JWP: A Tractable Class of Binary VCSPs via M-Convex Intersection*

Hiroshi Hirai[1], Yuni Iwamasa[1], Kazuo Murota[2], and Stanislav Živný[3]

1 **Department of Mathematical Informatics, University of Tokyo, Japan.**
`{hirai, yuni_iwamasa}@mist.i.u-tokyo.ac.jp`

2 **Department of Business Administration, Tokyo Metropolitan University, Japan.**
`murota@tmu.ac.jp`

3 **Department of Computer Science, University of Oxford, United Kingdom.**
`standa.zivny@cs.ox.ac.uk`

Abstract

A binary VCSP is a general framework for the minimization problem of a function represented as the sum of unary and binary cost functions. An important line of VCSP research is to investigate what functions can be solved in polynomial time. Cooper–Živný classified the tractability of binary VCSP instances according to the concept of "triangle," and showed that the only interesting tractable case is the one induced by the joint winner property (JWP). Recently, Iwamasa–Murota–Živný made a link between VCSP and discrete convex analysis, showing that a function satisfying the JWP can be transformed into a function represented as the sum of two M-convex functions, which can be minimized in polynomial time via an M-convex intersection algorithm if the value oracle of each M-convex function is given.

In this paper, we give an algorithmic answer to a natural question: What binary finite-valued CSP instances can be solved in polynomial time via an M-convex intersection algorithm? We solve this problem by devising a polynomial-time algorithm for obtaining a concrete form of the representation in the representable case. Our result presents a larger tractable class of binary finite-valued CSPs, which properly contains the JWP class.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases valued constraint satisfaction problems, discrete convex analysis, M-convexity

Digital Object Identifier 10.4230/LIPIcs.STACS.2018.40

1 Introduction

The *valued constraint satisfaction problem (VCSP)* provides a general framework for discrete optimization (see [25] for details). Informally, the VCSP framework deals with the minimiza-

* The first author's research was partially supported by JSPS KAKENHI Grant Numbers 25280004, 26330023, 17K00029. The second author's research was supported by JSPS Research Fellowship for Young Scientists. The third author's research was supported by The Mitsubishi Foundation, CREST, JST, Grant Number JPMJCR14D2, Japan, and JSPS KAKENHI Grant Number 26280004. The last author's research was supported by a Royal Society University Research Fellowship. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532). The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.

tion problem of a function represented as the sum of "small" arity functions, which are called *cost functions*. It is known that various kinds of combinatorial optimization problems can be formulated in the VCSP framework. In general, the VCSP is NP-hard. An important line of research is to investigate what restrictions on classes of VCSP instances ensure polynomial time solvability. Two main types of VCSPs with restrictions are *structure-based VCSPs* and *language-based VCSPs* (see e.g., [4, 25]). Structure-based VCSPs deal with restrictions on the hypergraph structure representing the appearance of variables in a given instance. For example, Gottlob–Greco–Scarcello [7] showed that, if the hypergraph corresponding to a VCSP instance has a bounded hypertree-width, then the instance can be solved in polynomial time. Language-based VCSPs deal with restrictions on cost functions that appear in a VCSP instance. Kolmogorov–Thapper–Živný [13] gave a precise characterization of tractable valued constraint languages via the basic LP relaxation. Kolmogorov–Krokhin–Rolínek [12] gave a dichotomy for all language-based VCSPs (see also [1, 24] for a dichotomy for all language-based CSPs).

Hybrid VCSPs, which deal with a combination of structure-based and language-based restrictions, have emerged recently [4]. Among many kinds of hybrid restrictions, a *binary VCSP*, VCSP with only unary and binary cost functions, is a representative hybrid restriction that includes numerous fundamental optimization problems. Cooper–Živný [2] showed that if a given binary VCSP instance satisfies the *joint winner property (JWP)*, then it can be minimized in polynomial time. The same authors classified in [3] the tractability of binary VCSP instances according to the concept of "triangle," and showed that the only interesting tractable case is the one induced by the JWP (see also [4]). Furthermore, they introduced *cross-free convexity* as a generalization of JWP, and devised a polynomial-time minimization algorithm for cross-free convex instances F, provided a "cross-free representation" of F is given.

In this paper, we introduce a novel tractability principle going beyond triangle and cross-free representation for binary finite-valued CSPs, from now on denoted by VCSPs. A binary VCSP is formulated as follows, where $D_1, D_2, \ldots, D_r$ ($r \geq 2$) are finite sets.

Given: Unary cost functions $F_p : D_p \to \mathbf{R}$ for $p \in \{1, 2, \ldots, r\}$ and binary cost functions $F_{pq} : D_p \times D_q \to \mathbf{R}$ for $1 \leq p < q \leq r$.

Problem: Find a minimizer of $F : D_1 \times D_2 \times \cdots \times D_r \to \mathbf{R}$ defined by

$$F(X_1, X_2, \ldots, X_r) := \sum_{1 \leq p \leq r} F_p(X_p) + \sum_{1 \leq p < q \leq r} F_{pq}(X_p, X_q). \tag{1}$$

Our tractability principle is built on *discrete convex analysis (DCA)* [18, 20], which is a theory of convex functions on discrete structures. In DCA, *L-convexity* and *M-convexity* play primary roles; the former is a generalization of submodularity, and the latter is a generalization of matroids. A variety of polynomially solvable problems in discrete optimization can be understood within the framework of L-convexity/M-convexity (see e.g., [20, 21, 22]). Recently, it has also turned out that discrete convexity is deeply linked to tractable classes of VCSPs. L-convexity is closely related to the tractability of language-based VCSPs. Various kinds of submodularity induce tractable classes of language-based VCSP instances [13], and a larger class of such submodularity can be understood as L-convexity on certain graph structures [9]. On the other hand, Iwamasa–Murota–Živný [11] have pointed out that M-convexity plays a role in hybrid VCSPs. They revealed the reason for the tractability of a VCSP instance satisfying the JWP from a view point of M-convexity. We here continue this line of research, and explore further applications of M-convexity in hybrid VCSPs.

A function $f : \{0,1\}^n \to \mathbf{R} \cup \{+\infty\}$ is called *M-convex* [15, 20] if it satisfies the following generalization of the matroid exchange axiom: for $x = (x_1, x_2, \ldots, x_n), y = (y_1, y_2, \ldots, y_n) \in$

dom f and $i \in \{1,2,\ldots,n\}$ with $x_i > y_i$, there exists $j \in \{1,2,\ldots,n\}$ with $y_j > x_j$ such that $f(x)+f(y) \geq f(x-\chi_i+\chi_j)+f(y+\chi_i-\chi_j)$, where, for a function $f : \mathcal{D} \to \mathbf{R} \cup \{+\infty\}$, the effective domain is denoted as dom $f := \{x \in \mathcal{D} \mid f(x) < +\infty\}$, and χ_i is the ith unit vector.[4] An M-convex function can be minimized in a greedy fashion similarly to the greedy algorithm for matroids. Furthermore, a function $f : \{0,1\}^n \to \mathbf{R} \cup \{+\infty\}$ that is representable as the sum of two M-convex functions is called *M_2-convex*. As a generalization of matroid intersection, the problem of minimizing an M_2-convex function, called the *M-convex intersection problem*, can also be solved in polynomial time if the value oracle of each constituent M-convex function is given [16, 17]; see also [19, Section 5.2]. Our proposed tractable class of VCSPs is based on this result.

Let us return to binary VCSPs. The starting observation for relating VCSP to DCA is that the objective function F on $D_1 \times D_2 \times \cdots \times D_r$ can be regarded as a function f on $\{0,1\}^n$ by the following correspondence between the domains:

$$D_p := \{1,2,\ldots,n_p\} \ni i \;\longleftrightarrow\; (\underbrace{0,\ldots,0,\overset{i}{\check{1}},0,\ldots,0}_{n_p}) \qquad (p \in \{1,2,\ldots,r\}). \tag{2}$$

With this correspondence, the minimization of F can be transformed to that of f. A binary VCSP instance F is said to be *M_2-representable* if the function f obtained from F via the correspondence (2) is M_2-convex.

It is shown in [11] that a binary VCSP instance satisfying the JWP can be transformed to an M_2-representable instance,[5] and two M-convex summands can be obtained in polynomial time. Here the following natural question arises: *What binary VCSP instances are M_2-representable?* In this paper, we give an algorithmic answer to this question by considering the following problem:

Testing M_2-Representability

Given: A binary VCSP instance F.

Problem: Determine whether F is M_2-representable or not. If F is M_2-representable, obtain a decomposition $f = f_1 + f_2$ of the function f into two M-convex functions f_1 and f_2, where f is the function transformed from F via (2).

Our main result is the following:

▶ **Theorem 1.1.** Testing M_2-Representability *can be solved in $O(n^5)$ time.*

An M_2-convex function f can be minimized in polynomial time if such a decomposition can be obtained in polynomial time. Thus we obtain the following corollary of Theorem 1.1.

▶ **Corollary 1.2.** *An M_2-representable binary VCSP instance can be minimized in polynomial time.*

Our result provides us with cross-free representations, and presents a new tractable class of binary VCSPs that goes beyond JWP. A nice feature of our contribution is that the tractability based on M_2-representability is independent of a particular representation (1) of

[4] Although M-convex functions are defined on $\mathbf{Z}^n$ in general, we only need functions on $\{0,1\}^n$ here. M-convex functions on $\{0,1\}^n$ are equivalent to the negative of *valuated matroids* introduced by Dress–Wenzel [5, 6].

[5] In [11], a binary VCSP instance satisfying the JWP was transformed into the sum of two $\mathrm{M}^\natural$-convex functions. It can be easily seen that this function can also be transformed into the sum of two M-convex functions.

a given instance, while the tractability based on JWP or cross-free convexity depends on a representation; see the full version [10] of this paper.

Our approach to a polynomial-time algorithm for Testing M_2-Representability is outlined as follows:

- We establish a unique representation theorem of M_2-convex functions arising from binary VCSP instances (Theorem 2.2).
- With this result, our problem can be separated into two subproblems named Decomposition and Laminarization. The former is the problem of obtaining the unique representation of a given M_2-convex function, and the latter is the problem of making a laminar family from a given family of subsets by means of certain transformations.
- We devise a polynomial-time algorithm for each problem, Decomposition and Laminarization (Theorems 3.4 and 4.8).

The proofs are omitted due to space limitation. The full version [10] of this paper will give the proofs as well as more general results and application to pseudo-Boolean function optimization.

Organization. In Section 2, we introduce the representation theorem (Theorem 2.2) of quadratic M_2-convex functions arising from VCSP instances as well as the subproblems, Decomposition and Laminarization. In Sections 3 and 4, we present polynomial-time algorithms for Decomposition and Laminarization, respectively.

Notation. Let $\mathbf{Z}$, $\mathbf{R}$, $\mathbf{R}_+$, and $\mathbf{R}_{++}$ denote the sets of integers, reals, nonnegative reals, and positive reals, respectively. In this paper, functions can take the infinite value $+\infty$, where $a < +\infty$, $a + \infty = +\infty$ for $a \in \mathbf{R}$, and $0 \cdot (+\infty) = 0$. Let $\overline{\mathbf{R}} := \mathbf{R} \cup \{+\infty\}$. For a positive integer k, we define $[k] := \{1, 2, \ldots, k\}$.

2 Towards testing M_2-representability

2.1 Representation theorem

We introduce a class of quadratic functions on $\{0,1\}^n$ that has a bijective correspondence to binary VCSP instances. Let $\mathcal{A} := \{A_1, A_2, \ldots, A_r\}$ be a partition of $[n]$ with $|A_p| \geq 2$ for $p \in [r]$. We say that $f : \{0,1\}^n \to \overline{\mathbf{R}}$ is a *VCSP-quadratic function of type* $\mathcal{A}$ if f is represented as

$$f(x_1, x_2, \ldots, x_n) := \begin{cases} \displaystyle\sum_{i \in [n]} a_i x_i + \sum_{1 \leq i < j \leq n} a_{ij} x_i x_j & \text{if } \displaystyle\sum_{i \in [n]} x_i = r, \\ +\infty & \text{otherwise,} \end{cases} \tag{3}$$

where $a_i \in \mathbf{R}$ and $a_{ij} \in \overline{\mathbf{R}}$ with $a_{ij} := +\infty \Leftrightarrow i, j \in A_p$ for some $p \in [r]$. We assume $a_{ij} = a_{ji}$ for distinct $i, j \in [n]$.

Suppose that a binary VCSP instance F of the form (1) is given, where we assume $F_{pq} = F_{qp}$ for distinct $p, q \in [r]$. The transformation of F to f based on (2) in Section 1 is formalized as follows. Choose a partition $\mathcal{A} := \{A_1, A_2, \ldots, A_r\}$ of $[n]$ with $|A_p| = n_p (= |D_p|)$ and a bijective correspondence $A_p \to D_p$. Define $a_i := F_p(d)$ if $i \in A_p$ corresponds to $d \in D_p$, $a_{ij} := F_{pq}(d, e)$ if $i \in A_p$ and $j \in A_q$ correspond to $d \in D_p$ and $e \in D_q$, respectively, and $a_{ij} := +\infty$ otherwise. Then the function f in (3) is a VCSP-quadratic function of type $\mathcal{A}$.

The class of M_2-convex VCSP-quadratic functions admits a decomposition into simpler functions ℓ_X. For $X \subseteq [n]$, let $\ell_X : \{0,1\}^n \to \mathbf{R}$ be defined by

$$\ell_X(x) := \sum_{k_-(X)<k<k_+(X)} \left| k - \sum_{i \in X} x_i \right|,$$

where $k_-(X)$ is the number of indices $p \in [r]$ with $X \supseteq A_p$, and $k_+(X)$ is the number of indices $p \in [r]$ with $X \cap A_p \neq \emptyset$. That is, $\ell_X(x)$ is the sum of the distances from $x \in \{0,1\}^n$ to hyperplanes $\{x \in \mathbf{R}^n \mid \sum_{i \in X} x_i = k\}$ for $k_-(X) < k < k_+(X)$. In the following, we consider subsets X with $k_-(X) + 2 \leq k_+(X)$, and denote the family of such subsets X by

$$\Pi = \Pi_{\mathcal{A}} := \{X \subseteq [n] \mid k_-(X) + 2 \leq k_+(X)\}.$$

In other words, $X \in \Pi$ if and only if $\emptyset \neq X \cap A_p \neq A_p$ for more than one $p \in [r]$.

A family $\mathcal{F} \subseteq \Pi$ is said to be *laminar* if $X \subseteq Y$, $X \supseteq Y$, or $X \cap Y = \emptyset$ holds for all $X, Y \in \mathcal{F}$. Define $\delta_{\mathcal{A}} : \{0,1\}^n \to \overline{\mathbf{R}}$ by $\delta_{\mathcal{A}}(x) := 0$ if $\sum_{i \in A_p} x_i = 1$ for each $A_p \in \mathcal{A}$, and $\delta_{\mathcal{A}}(x) := +\infty$ otherwise. Then the following holds.

▶ **Lemma 2.1.** *For any laminar family $\mathcal{L} \subseteq \Pi$ and any positive weight $c : \mathcal{L} \to \mathbf{R}_{++}$, the function $\sum_{X \in \mathcal{L}} c(X)\ell_X$ on $\{x \in \{0,1\}^n \mid \sum_{i \in [n]} x_i = r\}$ is M-convex.*

Our representation theorem (Theorem 2.2) says that an M_2-convex VCSP-quadratic function is always represented as the sum of $\sum_{X \in \mathcal{L}} c(X)\ell_X$ on $\{x \in \{0,1\}^n \mid \sum_{i \in [n]} x_i = r\}$ and a linear function on dom $\delta_{\mathcal{A}}$. To state it precisely, there are substantial complications to be resolved. In our setting, we are given a VCSP-quadratic function f of type $\mathcal{A}$, which is defined only on dom $f =$ dom $\delta_{\mathcal{A}}$. It can happen that functions ℓ_X and ℓ_Y are identical on dom $\delta_{\mathcal{A}}$ (i.e., $\ell_X + \delta_{\mathcal{A}} = \ell_Y + \delta_{\mathcal{A}}$) even when $X \neq Y$. Thus we have to make a judicious choice between them to demonstrate M_2-representability of f.

To cope with such complications, we define an equivalence relation $\sim$ by: $X \sim Y \Leftrightarrow \ell_X + \delta_{\mathcal{A}} = \ell_Y + \delta_{\mathcal{A}}$. For $\mathcal{F} \subseteq \Pi$, let $\mathcal{F}/\sim$ be the set of representatives (in $\Pi/\sim$) of all elements in $\mathcal{F}$. The equivalence relation is extended to subsets $\mathcal{F}, \mathcal{G}$ of Π by: $\mathcal{F} \sim \mathcal{G} \Leftrightarrow \mathcal{F}/\sim = \mathcal{G}/\sim$. A subset $\mathcal{P}$ of $\Pi/\sim$ is said to be *laminar* if there is a laminar family $\mathcal{L} \subseteq \Pi$ with $\mathcal{P} = \mathcal{L}/\sim$. A family $\mathcal{F} \subseteq \Pi$ is said to be *laminarizable* if $\mathcal{F}/\sim$ is laminar. For simplicity, the equivalence class of $X \in \Pi$ is also denoted by X, and a member of $\Pi/\sim$ is also denoted by its representative X.

Our first result is a representation theorem of M_2-convex functions.

▶ **Theorem 2.2.** *Let f be a VCSP-quadratic function of type $\mathcal{A} = \{A_1, A_2, \ldots, A_r\}$. Then f is M_2-convex if and only if there exist a laminar family $\mathcal{P}_f \subseteq \Pi/\sim$ and a positive weight $c_f : \mathcal{P}_f \to \mathbf{R}_{++}$ such that*

$$f = \sum_{X \in \mathcal{P}_f} c_f(X)\ell_X + \delta_{\mathcal{A}} + (\text{linear function}), \tag{4}$$

where "(linear function)*" means a function $x \mapsto \sum_i p_i x_i + \alpha$ for some $(p_1, p_2, \ldots, p_n) \in \mathbf{R}^n$ and $\alpha \in \mathbf{R}$. In addition, $\mathcal{P}_f$ and c_f in* (4) *are uniquely determined.*

By Theorem 2.2, an M_2-convex function f has the summand $f_1 := \sum_{X \in \mathcal{L}} c_f(X)\ell_X$ on $\{x \in \{0,1\}^n \mid \sum_{i \in [n]} x_i = r\}$, where $\mathcal{L}$ is a laminar family with $\mathcal{L}/\sim = \mathcal{P}_f$.

2.2 Decomposition and Laminarization

To test for M_2-representability by Theorem 2.2, we first solve the following problem Decomposition, which detects non-M_2-convexity of f or obtains decomposition (4).

DECOMPOSITION

Given: A VCSP-quadratic function f of type $\mathcal{A}$

Problem: Either detect the non-M_2-convexity of f, or obtain some $\mathcal{P} \subseteq \Pi/\!\sim$ and $c : \mathcal{P} \to \mathbf{R}_{++}$ satisfying

$$f = \sum_{X \in \mathcal{P}} c(X)\ell_X + \delta_{\mathcal{A}} + (\text{linear function}), \tag{5}$$

where $\mathcal{P}$ is not required to be laminar in general, but in case of M_2-convex f, $\mathcal{P}$ and c should coincide, respectively, with $\mathcal{P}_f$ and c_f in (4).

We emphasize that DECOMPOSITION may possibly output the decomposition (5) even when the input f is not M_2-convex, but if DECOMPOSITION detects the non-M_2-convexity then indeed the input f is not M_2-convex.

Suppose that decomposition (5) is obtained after solving DECOMPOSITION. In this case we have $\mathcal{P}$ at hand. Then we have to check for the laminarizability of an arbitrarily chosen family $\mathcal{F} \subseteq \Pi$ with $\mathcal{F}/\!\sim\, = \mathcal{P}$. This motivates us to consider the following problem.

LAMINARIZATION

Given: $\mathcal{F} \subseteq \Pi$

Problem: Determine whether there exists a laminar family $\mathcal{L}$ with $\mathcal{F} \sim \mathcal{L}$. If it exists, obtain a laminar family $\mathcal{L}$ with $\mathcal{F} \sim \mathcal{L}$.

LAMINARIZATION is a purely combinatorial problem on a set system. Indeed, the equivalence relation $\sim$ can be rephrased in a combinatorial way as follows. For $X \in \Pi$, define $\langle X \rangle := \bigcup\{A_p \in \mathcal{A} \mid \emptyset \neq X \cap A_p \neq A_p\}$, which is the union of A_p contributing to $\ell_X + \delta_{\mathcal{A}}$ nonlinearly. One can see the following.

▶ **Lemma 2.3.** *For $X, Y \in \Pi$, $X \sim Y$ if and only if $\{\langle X \rangle \cap X, \langle X \rangle \setminus X\} = \{\langle Y \rangle \cap Y, \langle Y \rangle \setminus Y\}$.*

LAMINARIZATION can be regarded as the problem of transforming a given family $\mathcal{F}$ to a laminar family by repeating the following operation: replace $X \in \mathcal{F}$ with $[n] \setminus X$, $X \cup A_p$, or $X \setminus A_p$ with some A_p satisfying $\langle X \rangle \cap A_p = \emptyset$.

A decomposition $f = f_1 + f_2$ into two M-convex functions f_1 and f_2 can be constructed from c_f and $\mathcal{L}$ found by DECOMPOSITION and LAMINARIZATION as $f_1 := \sum_{X \in \mathcal{L}} c_f(X)\ell_X$ on $\{x \in \{0,1\}^n \mid \sum_{i \in [n]} x_i = r\}$ and $f_2 := f - f_1$. By Lemma 2.1, f_1 is an M-convex function, and f_2 is a linear function on dom $\delta_{\mathcal{A}}$.

We devise an $O(n^5)$-time algorithm for DECOMPOSITION in Section 3 and an $O(n^4)$-time algorithm for LAMINARIZATION in Section 4. Thus we obtain Theorem 1.1.

▶ Remark. Our representation theorem (Theorem 2.2) and decomposition algorithm (in Section 3) are inspired by the *polyhedral split decomposition* due to Hirai [8]. This general decomposition principle decomposes, by means of polyhedral geometry, a function on a finite set $\mathcal{D}$ of points of $\mathbf{R}^n$ into a sum of simpler functions, called *split functions*, and a residue term. Actually, (5) can be viewed as a specialization of the polyhedral split decomposition, where $\mathcal{D} = \text{dom } \delta_{\mathcal{A}}$, and $\ell_X + \delta_{\mathcal{A}}$ is a sum of split functions. We refer the reader to [8] for details.

3 Algorithm for DECOMPOSITION

3.1 Outline

To describe our algorithm, we need the concept of *restriction* of a VCSP-quadratic function. Let f be a VCSP-quadratic function of type $\mathcal{A} = \{A_1, A_2, \ldots, A_r\}$. For $Q \subseteq [r]$, let

$\mathcal{A}_Q := \{A_p\}_{p\in Q}$ and $A_Q := \bigcup_{q\in Q} A_q$. For $Q \subseteq [r]$, the *restriction* $f_Q : \{0,1\}^{A_Q} \to \overline{\mathbf{R}}$ *of* f *to* Q is a VCSP-quadratic function of type $\mathcal{A}_Q$ defined by

$$f_Q(x) := \begin{cases} \sum_{i\in A_Q} a_i x_i + \sum_{i,j\in A_Q\ (i<j)} a_{ij}x_ix_j & \text{if } \sum_{i\in A_Q} x_i = |Q|, \\ +\infty & \text{otherwise.} \end{cases}$$

▶ **Lemma 3.1.** *If f is M_2-convex, so is the restriction f_Q for each $Q \subseteq [r]$.*

We abbreviate $\Pi_{\mathcal{A}}$ and $\Pi_{\mathcal{A}_Q}$ to Π and Π_Q, respectively. If f is M_2-convex, then f_Q can also be represented in a form similar to (4), i.e.,

$$f_Q = \sum_{X\in\mathcal{P}_{f_Q}} c_{f_Q}(X)\ell_X + \delta_{\mathcal{A}_Q} + (\text{linear function}),$$

where ℓ_X and $\delta_{\mathcal{A}_Q}$ are defined on $\{0,1\}^{A_Q}$.

Our algorithm to obtain decomposition (5) is outlined as follows, where we abbreviate $\{p,q\}$ and $\{p\}$ to pq and p, respectively, and also $\mathcal{P}_{f_{pq}}$ and $c_{f_{pq}}$ to $\mathcal{P}_{pq}$ and c_{pq}, respectively:

- We obtain a decomposition of the restriction f_Q for $Q = \{1,2\}, \{1,2,3\}, \ldots, \{1,2,3,\ldots,r\}$ in turn:

$$f_Q = \sum_{X\in\mathcal{P}_Q} c_Q(X)\ell_X + \delta_{\mathcal{A}_Q} + (\text{linear function}). \tag{6}$$

- In the initial case for $Q = \{1,2\}$, we can obtain the decomposition (6) by Algorithm 1 (Section 3.2).
- To construct the decomposition (6) for $Q = [r']$ from that for $Q = [r'-1]$, we first compute $(\mathcal{P}_{pr'}, c_{pr'})$ for all $p \in [r'-1]$ by Algorithm 1 and then, with this information, extend $(\mathcal{P}_{[r'-1]}, c_{[r'-1]})$ to $(\mathcal{P}_{[r']}, c_{[r']})$ by Algorithm 2 (Section 3.3).
- We perform the above extension step for $r' = 3$ to $r' = r$, to arrive at the decomposition (5) of f. This is described in Algorithm 3.

3.2 Initial case ($r = 2$)

To compute $\mathcal{P}_{pq}$ and c_{pq} for all distinct $p,q \in [r]$, we consider Decomposition algorithm for the case of $r = 2$. Namely $\mathcal{A} = \{A_1, A_2\}$. Note that $\Pi = \Pi_{\{A_1,A_2\}} = \{X \subseteq [n] \mid \emptyset \neq X \cap A_p \neq A_p \text{ for } p = 1,2\}$. A connected component with at least one edge is said to be *non-isolated*.

Algorithm 1 (for Decomposition in the case of $r = 2$):
Input: A VCSP-quadratic function f of type $\{A_1, A_2\}$.
Step 0: Define $\alpha^* := \min_{i,j\in[n]} a_{ij}$ and $S := \{i \in [n] \mid \min_{j\in[n]} a_{ij} > \alpha^*\}$.
Step 1: For $i \in [n]$ with $b_i := \min_{j\in[n]} a_{ij} - \alpha^* > 0$, update $a_{ij} \leftarrow a_{ij} - b_i$ for $j \in [n] \setminus \{i\}$ in turn.
Step 2: Let the distinct finite values of a_{ij} ($i \in A_1$, $j \in A_2$) be given by $\alpha_1 > \alpha_2 > \cdots > \alpha_m = \alpha^*$. For $\alpha \in \mathbf{R}$, define a graph $G_\alpha := ([n], E_\alpha)$ by $E_\alpha := \{\{i,j\} \mid i \in A_1,\ j \in A_2,\ \alpha \leq a_{ij}\}$. If, for some $\alpha \in \{\alpha_1, \alpha_2, \ldots, \alpha_{m-1}\}$, a (non-isolated) connected component of G_α is not a complete bipartite graph, then output "f is not M_2-convex" and stop.
Step 3: For $s \in [m-1]$, denote by $\mathcal{L}^s$ the set of non-isolated connected components L of G_{α_s}. For $L \in \mathcal{L}^s \setminus \mathcal{L}^{s-1}$ with $s \in [m-1]$, let $\alpha_L := \alpha_s$, where $\mathcal{L}^0 := \emptyset$. Define a laminar family $\mathcal{L}$ by $\mathcal{L} := \bigcup_{s=1}^{m-1} \mathcal{L}^s$. For $L \in \mathcal{L}$, define $c : \mathcal{L} \to \mathbf{R}_{++}$ by $c(L) := (\alpha_L - \alpha_{L^+})/2$, where L^+ is the minimal element in $\mathcal{L}$ properly containing L if L is not maximal, and $\alpha_{L^+} := \alpha^*$ if L is maximal.

Step 4: Turn $c : \mathcal{L} \to \mathbf{R}_{++}$ to $\mathcal{L}/\sim \to \mathbf{R}_{++}$ by defining the value c on an equivalence class as the sum of $c(L)$ over (at most two) members L in the equivalence class. Output $\mathcal{P} := \mathcal{L}/\sim$ and c. ◀

Note that, for distinct $L, L' \in \Pi$, we have $L \sim L' \Leftrightarrow L = [n] \setminus L'$.

▶ **Proposition 3.2.** *Algorithm* 1 *solves* Decomposition *in* $O(n^2)$ *time.*

3.3 General case ($r \geq 3$)

To obtain the decomposition (6) of the restriction f_Q for $Q = \{1,2\}, \{1,2,3\}, \ldots, \{1,2,3,\ldots,r\}$ in turn, we need to extend $(\mathcal{P}_{[r'-1]}, c_{[r'-1]})$ to $(\mathcal{P}_{[r']}, c_{[r']})$ with the use of $(\mathcal{P}_{pr'}, c_{pr'})$ ($p \in [r'-1]$) for $r' = 3, \ldots, r$. Algorithm 2 corresponds to this extension step.

Algorithm 2 (for extending f' to f):

Input: A VCSP-quadratic function f of type $\mathcal{A}$ and restriction $f' := f_{[r-1]}$ given as

$$f' = \sum_{X \in \mathcal{P}'} c'(X)\ell_X + \delta_{\mathcal{A}_{[r-1]}} + (\text{linear function})$$

for a family $\mathcal{P}' \subseteq \Pi_{[r-1]}/\sim$ with $|\mathcal{P}'| \leq 2|A_{[r-1]}|$ and a positive weight c' on $\mathcal{P}'$.

Output: Either detect the non-M_2-convexity of f, or obtain expression

$$f = \sum_{X \in \mathcal{P}} c(X)\ell_X + \delta_{\mathcal{A}} + (\text{linear function}),$$

with $\mathcal{P} \subseteq \Pi/\sim$ satisfying $|\mathcal{P}| \leq 2n = 2|A_{[r]}|$ and a positive weight c on $\mathcal{P}$.

Step 1: For each $p \in [r-1]$, execute Algorithm 1 for f_{pr}. If Algorithm 1 returns "f_{pr} is not M_2-convex" for some $p \in [r-1]$, then output "f is not M_2-convex" and stop. Otherwise, obtain $\mathcal{P}_{pr}$ and c_{pr} for all $p \in [r-1]$. Let $\mathcal{P} := \emptyset$.

Step 2: If $\mathcal{P}' = \emptyset$, go to Step 3. Otherwise, do the following: Let X_0 be an element of $\mathcal{P}'$ such that $\langle X_0 \rangle$ is maximal. Let $\{p_1, p_2, \ldots, p_k\}$ be the set of indices $p \in [r-1]$ with $\langle X_0 \rangle = A_{\{p_1,p_2,\ldots,p_k\}}$. If there exist $X \in \Pi/\sim$ and $X_i \in \mathcal{P}_{p_ir}$ ($i = 1, 2, \ldots, k$) such that $X \sim_{[r-1]} X_0$ and $X \sim_{p_ir} X_i$ for each $i \in [k]$, then go to Step 2-1. Otherwise, go to Step 2-2.

2-1: Update as

$$\begin{aligned}
&\mathcal{P} \leftarrow \mathcal{P} \cup \{X\}, \qquad c(X) \leftarrow \min\{c'(X_0), c_{p_1r}(X_1), c_{p_2r}(X_2), \ldots, c_{p_kr}(X_k)\},\\
&c'(X_0) \leftarrow c'(X_0) - c(X), \qquad c_{p_ir}(X_i) \leftarrow c_{p_ir}(X_i) - c(X) \quad (i \in [k]),\\
&\mathcal{P}' \leftarrow \mathcal{P}' \setminus \{X_0\} \text{ if } c'(X_0) = 0, \qquad \mathcal{P}_{p_ir} \leftarrow \mathcal{P}_{p_ir} \setminus \{X_i\} \text{ if } c_{p_ir}(X_i) = 0 \quad (i \in [k]),
\end{aligned}$$

and go to Step 2.

2-2: Update as $\mathcal{P} \leftarrow \mathcal{P} \cup \{X_0\}$, $\mathcal{P}' \leftarrow \mathcal{P}' \setminus \{X_0\}$, and $c(X_0) \leftarrow c'(X_0)$. Go to Step 2.

Step 3: Update as $\mathcal{P} \leftarrow \mathcal{P} \cup \bigcup_{i \in [k]} \mathcal{P}_{p_ir}$, and $c(X) \leftarrow c_{p_ir}(X)$ for $i \in [k]$ and $X \in \mathcal{P}_{p_ir}$. Then output $\mathcal{P}$ and c. ◀

The following proposition shows that Algorithm 2 works as expected.

▶ **Proposition 3.3.** *If f is M_2-convex, and $\mathcal{P}' = \mathcal{P}_{f'}$ and $c' = c_{f'}$ hold, then $\mathcal{P} = \mathcal{P}_f$ and $c = c_f$ hold. Furthermore, Algorithm* 2 *runs in $O(n^4)$ time.*

Our proposed algorithm for Decomposition can be summarized as follows. It is noted that, if $\mathcal{P}$ is laminar, then $|\mathcal{P}|$ is at most $2n = 2|A_{[r]}|$ (see e.g., [23, Theorem 3.5]).

Algorithm 3 (for Decomposition):

Step 1: Execute Algorithm 1 for the restriction f_{12}. If Algorithm 1 returns "f_{12} is not M_2-convex," then output "f is not M_2-convex" and stop. Otherwise, obtain $\mathcal{P}_{12}$ and c_{12}.

Step 2: For $r' = 3, \ldots, r$, execute Algorithm 2 for $(f_{[r']}, \mathcal{P}_{[r'-1]}, c_{[r'-1]})$. If Algorithm 2 returns "$f_{[r']}$ is not M_2-convex" or $|\mathcal{P}_{[r']}| > 2|A_{[r']}|$ holds for some r', output "f is not M_2-convex" and stop. Otherwise, obtain $c_{[r']}$ and $\mathcal{P}_{[r']}$.

Step 3: Output $\mathcal{P} := \mathcal{P}_{[r]}$ and $c := c_{[r]}$. ◀

▶ **Theorem 3.4.** *Algorithm* 3 *solves* Decomposition *in* $O(n^5)$ *time.*

4 Algorithm for Laminarization

For a VCSP-quadratic function f of type $\mathcal{A}$, suppose that we have obtained $\mathcal{P} \subseteq \Pi/\sim$ by solving Decomposition. The next step for solving Testing M_2-Representability is to check for the laminarity of $\mathcal{P}$. Take $\mathcal{F} \subseteq \Pi$ with $\mathcal{F}/\sim = \mathcal{P}$; such $\mathcal{F}$ can be constructed easily from $\mathcal{P}$. The input of Laminarization is $\mathcal{F}$.

4.1 Outline

For families $\mathcal{G}, \mathcal{H} \subseteq \Pi$, we say that $\mathcal{G}$ is equivalent to $\mathcal{H}$ if $\mathcal{G} \sim \mathcal{H}$. It is easy to see that a laminar family can be constructed easily from a cross-free family $\mathcal{G}$ by switching $X \mapsto [n] \setminus X$ for appropriate $X \in \mathcal{G}$ (see e.g., [14, Section 2.2]); this can be done in $O(|\mathcal{G}|)$ time. Thus, by $X \sim [n] \setminus X$, our goal is to construct a cross-free family equivalent to the input family.

In this section, we devise a polynomial-time algorithm for constructing a desired cross-free family. Our algorithm makes use of weaker notions of cross-freeness, called 2- and 3-local cross-freeness. The existence of a cross-free family is characterized by the existence of a *2-locally cross-free family* (Section 4.2). The existence of such a 2-locally cross-free family can be checked easily by solving a 2-SAT problem. If a 2-locally cross-free family exists, a *3-locally cross-free family* also exists, and can be constructed in polynomial time (Section 4.4). From a 3-locally cross-free family, we can construct a desired cross-free family in polynomial time by the *uncrossing operations* (Section 4.3). Thus we solve Laminarization.

Without loss of generality, we assume that $X \subsetneq \langle X \rangle$ for every X in the input $\mathcal{F}$ and no distinct X, Y with $X \sim Y$ are contained in $\mathcal{F}$, i.e., $|\mathcal{F}| = |\mathcal{F}/\sim|$. For $X \in \mathcal{F}$, let $\overline{X} := \langle X \rangle \setminus X$; note $X \sim \overline{X}$. For $X, Y, Z \in \Pi$, we define $\langle XY \rangle := \langle X \rangle \cap \langle Y \rangle$ and $\langle XYZ \rangle := \langle X \rangle \cap \langle Y \rangle \cap \langle Z \rangle$. For $X \in \mathcal{F}$ and $Q \subseteq [r]$ with $A_Q \subseteq \langle X \rangle$, the *partition line of* X *on* A_Q is a bipartition $\{X \cap A_Q, \overline{X} \cap A_Q\}$ of A_Q. We also assume that $|\mathcal{F}/\sim|$ is at most $2n$ and that $X \subseteq Y$, $X \subseteq \overline{Y}$, $X \supseteq Y$, or $X \supseteq \overline{Y}$ holds on $\langle XY \rangle$ for distinct $X, Y \in \mathcal{F}$ with $\langle XY \rangle \neq \emptyset$, since otherwise $\mathcal{F}$ is not laminarizable.

We can also assume throughout that both $\langle X \rangle \setminus \langle Y \rangle$ and $\langle Y \rangle \setminus \langle X \rangle$ are nonempty for all distinct $X, Y \in \mathcal{F}$. Indeed, for each $X \in \mathcal{F}$, we add a new set A_X with $|A_X| = 2$ to the ground set $[n]$ and to the partition $\mathcal{A}$ of $[n]$; the ground set will be $[n] \cup \bigcup_{X \in \mathcal{F}} A_X$ and the partition will be $\mathcal{A} \cup \{A_X \mid X \in \mathcal{F}\}$. Define $X_+ := X \cup \{x\}$, where x is one of the two elements of A_X and $\mathcal{F}_+ := \{X_+ \mid X \in \mathcal{F}\}$. Note $\langle X_+ \rangle = \langle X \rangle \cup A_X$ and $\langle X_+ \rangle \setminus \langle Y_+ \rangle \neq \emptyset$ for all $X_+, Y_+ \in \mathcal{F}_+$. Then it is easily seen that there exists a cross-free family $\mathcal{L}$ with $\mathcal{L} \sim \mathcal{F}$ if and only if there exists a cross-free family $\mathcal{L}_+$ with $\mathcal{L}_+ \sim \mathcal{F}_+$.

4.2 2-local cross-freeness

For $A \subseteq [n]$, a pair $X, Y \subseteq [n]$ is said to be *crossing on* A if $(X \cap Y) \cap A$, $A \setminus (X \cup Y)$, $(X \setminus Y) \cap A$, and $(Y \setminus X) \cap A$ are all nonempty. A family $\mathcal{G} \subseteq \Pi$ is said to be *cross-free on*

A if there is no crossing pair on A in $\mathcal{G}$. A family $\mathcal{G} \subseteq \Pi$ is called *2-locally cross-free* if no $X, Y \in \mathcal{G}$ is crossing on $\langle X \rangle \cup \langle Y \rangle$. A cross-free family is 2-locally cross-free.

The *LC-graph* $G(\mathcal{F}) = (V(\mathcal{F}), E_\mathrm{f} \cup E_\mathrm{b})$ of the input $\mathcal{F}$ is defined by

$$\begin{aligned} V(\mathcal{F}) &:= \{XY \mid X, Y \in \mathcal{F}, X \neq Y\}, \\ E_\mathrm{f} &:= \{\{XY, XZ\} \mid Y \neq Z,\ (\langle Y \rangle \setminus \langle X \rangle) \cap \langle Z \rangle \neq \emptyset\}, \\ E_\mathrm{b} &:= \{\{XY, YX\} \mid \langle XY \rangle \neq \emptyset\}, \end{aligned}$$

where XY is an abbreviation of ordered pair (X, Y). LC stands for Local Cross-freeness. Note that the structure of LC-graph depends only on $\{\langle X \rangle \mid X \in \mathcal{F}\}$. We call an edge $e \in E_\mathrm{f}$ a *forward edge* and an edge $e \in E_\mathrm{b}$ a *backward edge*. A backward edge $e = \{XY, YX\}$ is said to be *flipping* (resp. *non-flipping*) if $X \subseteq Y$ or $X \supseteq Y$ (resp. $X \subseteq \overline{Y}$ or $X \supseteq \overline{Y}$) holds on $\langle XY \rangle$.

An *LC-labeling* is a function $s : V(\mathcal{F}) \to \{0, 1\}$ such that

$$s(XY) = \begin{cases} s(XZ) & \text{if } \{XY, XZ\} \text{ is a forward edge,} \\ s(YX) & \text{if } \{XY, YX\} \text{ is a non-flipping backward edge,} \\ 1 - s(YX) & \text{if } \{XY, YX\} \text{ is a flipping backward edge, and} \end{cases} \tag{7}$$

$$(s(XY), s(YX)) = \begin{cases} (0,0) & \text{if } X \subsetneq \overline{Y}, \\ (0,1) & \text{if } X \subsetneq Y, \\ (1,0) & \text{if } X \supsetneq Y, \\ (1,1) & \text{if } X \supsetneq \overline{Y}, \end{cases} \quad \text{on } \langle XY \rangle \text{ for backward edge } \{XY, YX\}. \tag{8}$$

Note that (8) imposes no condition if the partition lines of X and Y on $\langle XY \rangle$ are the same. Node $XY \in V(\mathcal{F})$ is said to be *fixed* if the value of an LC-labeling s for XY is determined as (8), that is, if $\langle XY \rangle \neq \emptyset$ and the partition lines of X and Y on $\langle XY \rangle$ are different.

An LC-labeling s transforms the family $\mathcal{F}$ to another family $\mathcal{F}^s$ equivalent to $\mathcal{F}$, which is given by $\mathcal{F}^s := \{X^s \mid X \in \mathcal{F}\}$ with $X^s := X \cup \bigcup\{\langle Y \rangle \setminus \langle X \rangle \mid Y \in \mathcal{F} \text{ with } s(XY) = 1\}$. Thanks to condition (7) on forward edges, we have $X^s \cap (\langle Y \rangle \setminus \langle X \rangle) = \emptyset$ for $Y \in \mathcal{F}$ with $s(XY) = 0$.

▶ **Proposition 4.1.** *There exists a 2-locally cross-free family equivalent to $\mathcal{F}$ if and only if there exists an LC-labeling s in $G(\mathcal{F})$. To be specific, $\mathcal{F}^s$ is a 2-locally cross-free family equivalent to $\mathcal{F}$.*

An LC-labeling is nothing but a feasible solution for the 2-SAT problem defined by the constraints (7) and (8). Therefore we can check the existence of an LC-labeling s greedily in $O(|E_\mathrm{f} \cup E_\mathrm{b}|) = O(n^4)$ time as follows, where XY is called a *defined node* if the value of $s(XY)$ has been defined.

1. For each fixed node XY, define $s(XY)$ according to (8).
2. In each connected component of $G(\mathcal{F})$, execute a breadth-first search from a defined node XY, and define $s(ZW)$ for all reached nodes ZW according to (7). If a conflict in value assignment to $s(ZW)$ is detected during this process, output "there is no LC-labeling."
3. If there is an undefined node, choose any undefined node XY, and define $s(XY)$ as 0 or 1 arbitrarily. Then go to 2.

4.3 3-local cross-freeness

A family $\mathcal{G} \subseteq \Pi$ is called *3-locally cross-free* if $\mathcal{G}$ is 2-locally cross-free and $\{X, Y, Z\}$ is cross-free on $\langle X \rangle \cup \langle Y \rangle \cup \langle Z \rangle$ for all $X, Y, Z \in \mathcal{G}$ with $\langle XYZ \rangle \neq \emptyset$. A cross-free family is 3-locally cross-free, and a 3-locally cross-free family is 2-locally cross-free, whereas the converse is not true. We write $X \subseteq^* Y$ to mean $X \subseteq Y$ on $\langle X \rangle \cup \langle Y \rangle$.

Our objective of this subsection is to give an algorithm for constructing a desired cross-free family from a 3-locally cross-free family equivalent to the input $\mathcal{F}$. The algorithm consists of repeated applications of an elementary operation that preserves 3-local cross-freeness. The operation is defined by (9) below, and is referred to as the *uncrossing operation* to X, Y.

▶ **Proposition 4.2.** *Suppose that $\mathcal{G}$ is 3-locally cross-free. For $X, Y \in \mathcal{G}$, define*

$$\mathcal{G}' := \begin{cases} \mathcal{G} \setminus \{X, Y\} \cup \{X \cap Y, X \cup Y\} & \text{if } X \subseteq^* Y \text{ or } Y \subseteq^* X, \\ \mathcal{G} \setminus \{X, Y\} \cup \{X \setminus Y, Y \setminus X\} & \text{if } X \subseteq^* [n] \setminus Y \text{ or } [n] \setminus Y \subseteq^* X. \end{cases} \tag{9}$$

Then $\mathcal{G}'$ is a 3-locally cross-free family equivalent to $\mathcal{G}$.

Note, by the 2-local cross-freeness of $\mathcal{G}$, that all $X, Y \in \mathcal{G}$ satisfy $X \subseteq^* Y$, $Y \subseteq^* X$, $X \subseteq^* [n] \setminus Y$, or $[n] \setminus Y \subseteq^* X$. It is worth mentioning that the uncrossing operation does not preserve 2-local cross-freeness.

Algorithm 4 (for constructing a cross-free family):
Input: A 3-locally cross-free family $\mathcal{G}$.
Step 1: While there is a crossing pair X, Y in $\mathcal{G}$, apply the uncrossing operation to X, Y and modify $\mathcal{G}$ accordingly.
Step 2: Output $\mathcal{G}$. ◀

▶ **Proposition 4.3.** *Algorithm 4 runs in $O(n^2)$ time, and the output $\mathcal{G}$ is cross-free.*

4.4 Constructing 3-locally cross-free family

Our final goal is to show that, for the input $\mathcal{F}$ equivalent to a 2-locally cross-free family, we can always construct, in polynomial time, an LC-labeling s such that $\mathcal{F}^s$ is 3-locally cross-free. In the following, we assume the existence of an LC-labeling.

The following Lemma 4.4 indicates that, more often than not, a triple X, Y, Z in any 2-locally cross-free family is cross-free on $\langle X \rangle \cup \langle Y \rangle \cup \langle Z \rangle$.

▶ **Lemma 4.4.** *Let $\mathcal{G}$ be a 2-locally cross-free family. A triple $\{X, Y, Z\} \subseteq \mathcal{G}$ is cross-free on $\langle X \rangle \cup \langle Y \rangle \cup \langle Z \rangle$ if one of the following conditions holds:*
(1) *$\langle XY \rangle \neq \emptyset$, and $\{X, Y\}$ is cross-free on $\langle X \rangle \cup \langle Y \rangle \cup \langle Z \rangle$.*
(2) *$\langle XY \rangle \not\subseteq \langle Z \rangle$, and $\langle XZ \rangle$ or $\langle YZ \rangle$ is nonempty.*
(3) *The partition lines of X, Y, Z on $\langle XYZ \rangle$ are not the same.*
(4) *$\langle XY \rangle = \langle ZY \rangle \neq \emptyset$, and there is a path $(XY, XY_1, \dots, XY_k)$ in $G(\mathcal{G})$ such that $\{X, Y_k, Z\}$ is cross-free on $\langle X \rangle \cup \langle Y_k \rangle \cup \langle Z \rangle$.*

To construct a 3-locally cross-free family, a particular care is needed for those triples X, Y, Z with $\langle XY \rangle = \langle YZ \rangle = \langle ZX \rangle \neq \emptyset$ for which there exists no path $(XY, XY_1, \dots, XY_k)$ satisfying $\langle XY \rangle \neq \langle XY_k \rangle \neq \emptyset$. This motivates the notion of *special nodes* and *special connected components* in the LC-graph $G(\mathcal{F})$ defined in Section 4.2. For distinct $X, Y \in \mathcal{F}$, define

$$R(XY) := \{Z \in \mathcal{F} \mid \text{There is a path } (XY, XY_1, \dots, XZ)\},$$
$$R^*(XY) := \{Z \in R(XY) \mid \langle XZ \rangle \neq \emptyset\}.$$

We say that XY with $\langle XY \rangle \neq \emptyset$ is *special* if $\langle XZ \rangle = \langle XY \rangle$ holds for all $Z \in R^*(XY)$. For $X, Y \in \mathcal{F}$ such that both XY and YX are special, let $v(XY)$ denote the connected component (as a set of nodes) containing XY or YX in $G(\mathcal{F})$. We call such a component *special*. Let $v^*(XY)$ denote the set of nodes ZW in $v(XY)$ with $\langle ZW \rangle \neq \emptyset$. A special component has an intriguing structure.

▶ **Proposition 4.5.** *If both XY and YX are special, then the following hold.*
(i) $v(XY) = (R^*(XY) \times R(YX)) \cup (R^*(YX) \times R(XY))$.
(ii) $v^*(XY) = (R^*(XY) \times R^*(YX)) \cup (R^*(YX) \times R^*(XY))$.
(iii) *If $ZW \in v^*(XY)$, then ZW is special and $\langle ZW \rangle = \langle XY \rangle$.*

For a special component $v = v(XY)$, we call $\langle XY \rangle$ the *center* of v; this is well-defined by (iii) of Proposition 4.5. For $Q \subseteq [r]$, the *Q-flower* is the nonempty set with size at least two of all special components having center A_Q.

▶ **Proposition 4.6.** *The Q-flower is given as $\{v(X_iX_j) \mid 1 \leq i < j \leq p\}$ for some $p \geq 3$ and distinct $X_1, X_2, \ldots, X_p \in \mathcal{F}$ such that $R(X_iX_j) = R(X_{i'}X_j)$ for all $i, i' < j$, and $R(X_iX_j) \cap R(X_{i'}X_{j'}) = \emptyset$ for all distinct $j, j' \in [p]$, $i < j$, and $i' < j'$.*

The above $X_1, X_2, \ldots, X_p$ are called the *representatives* of the Q-flower.

A component v is said to be *fixed* if v contains a fixed node, and said to be *free* otherwise. A special component $v(XY)$ in the Q-flower is free if and only if the partition lines of X' and Y' on A_Q are the same for all $X' \in R^*(YX)$ and $Y' \in R^*(XY)$. A *free Q-flower* is a maximal set of free components in the Q-flower such that the partition lines on A_Q is the same. Now the set of free components of the Q-flower is partitioned to free Q-flowers each of which is represented as $\{v(X_{i_s}X_{i_t}) \mid 1 \leq s < t \leq q\}$ with a subset $\{X_{i_1}X_{i_2}, \ldots, X_{i_q}\}$ of the representatives. A free Q-flower (for some $Q \subseteq [r]$) is also called a *free flower*.

We now provide a polynomial-time algorithm to construct a 3-locally cross-free family $\mathcal{F}^s$ by defining an appropriate LC-labeling s.

Algorithm 5 (for constructing a 3-locally cross-free family):

Step 0: Determine whether there exists a 2-locally cross-free family equivalent to $\mathcal{F}$. If not, then output "$\mathcal{F}$ is not laminarizable" and stop.

Step 1: For all fixed nodes XY, define $s(XY)$ according to (8). By a breath-first search, define s on all other nodes in fixed components appropriately.

Step 2: For each component v which is free and not special, take any node XY in v. Define $s(XY)$ as 0 or 1 arbitrarily, and define $s(ZW)$ appropriately for all nodes ZW in v. Then all the remaining (undefined) components are special and free.

Step 3: For each free flower, which is assumed to be represented as $\{v(X_iX_j) \mid 1 \leq i < j \leq q\}$, do the following:

- **3-1:** Define the value of $s(X_iX_j)$ for distinct $i, j \in [q]$ so that $\{X_1^s, X_2^s, \ldots, X_q^s\}$ is cross-free on $\bigcup_{i \in [q]} \langle X_i \rangle$.
- **3-2:** Define $s(ZW)$ appropriately for all $ZW \in v(X_iX_j)$.

Step 4: Output $\mathcal{F}^s$. ◀

▶ **Proposition 4.7.** *The output $\mathcal{F}^s$ is 3-locally cross-free, and Algorithm* 5 *runs in $O(n^4)$ time.*

By Propositions 4.3 and 4.7, we obtain the following theorem.

▶ **Theorem 4.8.** *Algorithms* 4 *and* 5 *solve* LAMINARIZATION *in $O(n^4)$ time.*

References

1 A. A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17)*, pages 319–330, 2017.

2 M. C. Cooper and S. Živný. Hybrid tractability of valued constraint problems. *Artificial Intelligence*, 175:1555–1569, 2011.

3 M. C. Cooper and S. Živný. Tractable triangles and cross-free convexity in discrete optimisation. *Journal of Artificial Intelligence Research*, 44:455–490, 2012.

4 M. C. Cooper and S. Živný. *Hybrid tractable classes of constraint problems*, volume 7 of *Dagstuhl Follow-Ups Series*, chapter 4, pages 113–135. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.

5 A. W. M. Dress and W. Wenzel. Valuated matroids: A new look at the greedy algorithm. *Applied Mathematics Letters*, 3(2):33–35, 1990.

6 A. W. M. Dress and W. Wenzel. Valuated matroids. *Advances in Mathematics*, 93:214–250, 1992.

7 G. Gottlob, G. Greco, and F. Scarcello. Tractable optimization problems through hypergraph-based structural restrictions. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP'09, Part II)*, pages 16–30, 2009.

8 H. Hirai. A geometric study of the split decomposition. *Discrete and Computational Geometry*, 36:331–361, 2006.

9 H. Hirai. Discrete convexity and polynomial solvability in minimum 0-extension problems. *Mathematical Programming, Series A*, 155:1–55, 2016.

10 H. Hirai, Y. Iwamasa, K. Murota, and S. Živný. A tractable class of binary VCSPs via M-convex intersection. In preparation.

11 Y. Iwamasa, K. Murota, and S. Živný. Discrete convexity in joint winner property. *Discrete Optimization*. To appear.

12 V. Kolmogorov, A. Krokhin, and M. Rolínek. The complexity of general-valued CSPs. *SIAM Journal on Computing*, 46(3):1087–1110, 2017.

13 V. Kolmogorov, J. Thapper, and S. Živný. The power of linear programming for general-valued CSPs. *SIAM Journal on Computing*, 44(1):1–36, 2015.

14 B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, Heidelberg, 5th edition, 2010.

15 K. Murota. Convexity and Steinitz's exchange property. *Advances in Mathematics*, 124:272–311, 1996.

16 K. Murota. Valuated matroid intersection, I: optimality criteria. *SIAM Journal on Discrete Mathematics*, 9:545–561, 1996.

17 K. Murota. Valuated matroid intersection, II: algorithms. *SIAM Journal on Discrete Mathematics*, 9:562–576, 1996.

18 K. Murota. Discrete convex analysis. *Mathematical Programming*, 83:313–371, 1998.

19 K. Murota. *Matrices and Matroids for Systems Analysis*. Springer, Heidelberg, 2000.

20 K. Murota. *Discrete Convex Analysis*. SIAM, Philadelphia, 2003.

21 K. Murota. Recent developments in discrete convex analysis. In W. Cook, L. Lovász, and J. Vygen, editors, *Research Trends in Combinatorial Optimization*, chapter 11, pages 219–260. Springer, Heidelberg, 2009.

22 K. Murota. Discrete convex analysis: A tool for economics and game theory. *Journal of Mechanism and Institution Design*, 1(1):151–273, 2016.

23 A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, Heidelberg, 2003.

24 D. Zhuk. A proof of CSP dichotomy conjecture. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17)*, pages 331–342, 2017.

25 S. Živný. *The Complexity of Valued Constraint Satisfaction Problems*. Springer, Heidelberg, 2012.