JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 16, Number 11, 2009

Pp. 1529–1537
DOI: 10.1089/cmb.2008.0224

On the Parameterized Complexity of Pooling Design

YONGXI CHENG,[1] DING-ZHU DU,[2] KER-I KO,[3] and GUOHUI LIN[1]

ABSTRACT

Pooling design is a very helpful tool for reducing the number of tests in DNA library screening, which is a key process to obtain high-quality DNA libraries for studying gene functions. Three basic problems in pooling design are, given an $m \times n$ binary matrix and a positive integer d, to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct). The three problems are all known to be coNP-complete. Since in most applications, d is a small integer compared to n, it is interesting to investigate whether there are efficient algorithms solving the above problems when the value of d is small. In this article, we give a negative answer to the above question by studying the parameterized complexity of these three problems, with d as the parameter. We show that the parameterized versions of all the three problems are co-W[2]-complete. An immediate implication of our results is that, given an $m \times n$ binary matrix and a positive integer d, a deterministic algorithm with running time $f(d) \times (mn)^{O(1)}$ (where f is an arbitrary computable function) to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct) should not be expected.

Key words: disjunct matrices, DNA library screening, parameterized complexity, pooling designs, separable matrices.

1. INTRODUCTION

A DNA library consists of thousands of separate DNA *clones*. The basic task of DNA library screening is, for a collection of *probes*, to determine which clone from the library contains which probe. Given a probe, a clone is said to be *positive* if it contains the probe; otherwise, it is said to be *negative*. In practice, to identify all positive clones from a library, clones are often pooled together to be tested against each probe, since checking each clone-probe pair is expensive, and usually only a few clones in the library contain a given probe. An example is when Sequenced-Tagged Site markers (also called STS probes) are used (Olson et al., 1989). There are experimental tests (e.g., the Polymerase Chain Reaction) that can determine in a given pool whether or not there exists at least one clone containing a given probe.

The above application is an instance of the *combinatorial group testing* problem, in which we have n items (each can be either *positive* or *negative)*, and the number of positives is upper bounded by an integer d (usually we assume that $d \ll n$). Suppose there exists some method that can test for any subset of items whether or not it contains at least one positive item. We say that the test outcome is positive if the test result indicates that the subset contains at least one positive item; otherwise, we say that the test outcome is

[1]Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada.
[2]Department of Computer Science, University of Texas at Dallas, Richardson, Texas.
[3]Department of Computer Science, State University of New York at Stony Brook, Stony Brook, New York.

negative. Usually the problem is to identify all positives by using the minimum number of tests. The study of group testing dates back to World War II, at first for economical mass blood testing (Dorfman, 1943). Due to its basic nature, it has found applications in many different areas. Group testing procedures can be *adaptive* or *nonadaptive*. An adaptive procedure conducts the tests one by one, and allows us to design later tests using the outcome information of all previous tests. A nonadaptive procedure specifies all tests at the beginning before knowing the outcomes of any test. The benefit is that all tests can be performed simultaneously. Between fully adaptive and nonadaptive group testing procedures, there are also two-stage procedures, which are of considerable interest (Knill, 1995; Macula, 1999; Berger et al., 2000; De Bonis et al., 2005; Eppstein et al., 2007).

1.1. Pooling design

Probably the most important modern applications of group testing are in the area of computational molecular biology, in which one important subject is clone library screening (Balding et al., 1995; Farach et al., 1997; Du and Hwang, 2006). In applications to molecular biology, a group testing procedure is called a *pooling design*, and the composition of each test is called a *pool*. In such applications, screening one pool at a time is far more expensive and time-consuming than screening many pools in parallel; this strongly encourages the use of nonadaptive procedures.

A nonadaptive group testing procedure can be represented as a 0-1 matrix $M=(m_{ij})$, in which the columns are associated with the items and the rows are associated with the tests, and $m_{ij}=1$ indicates that item j is contained in test i. The test outcomes can be represented by a 0-1 vector, the *outcome vector*, where 0 indicates a negative outcome and 1 indicates a positive outcome. It is not hard to verify that if subset S of columns correspond to all the positive items, then the outcome vector is equal to vector $U(S)$, the *union* (i.e., the componentwise Boolean sum) of all column vectors in S.

In order to identify all positives as long as the number of positives is no more than d, matrix M should satisfy that for any two distinct subsets S_1 and S_2 of columns such that $|S_1|\le d$ and $|S_2|\le d$, $U(S_1)\neq U(S_2)$. A matrix satisfying this property is called $\bar{d}$*-separable*. In the definition, if we replace the condition "$|S_1|\le d$ and $|S_2|\le d$" by "$|S_1|=|S_2|=d$," a matrix satisfying this property is called *d-separable*. If the matrix representing a nonadaptive pooling design is $\bar{d}$*-separable* (or *d-separable*), then theoretically, based on the test outcomes, we can unambiguously identify up to d (or exactly d) positives from the given set. However, the actual process of determining the positives from the outcome vector, that is the *decoding* process, could be very time-consuming. In practice, we can adopt matrices with stronger property to make the decoding process more efficient.

For two 0-1 vectors u and v with the same number of components, if for any component of u the value is 1, the corresponding component of v is also 1, then we say that u is *covered* by v. A 0-1 matrix is said to be *d-disjunct* if no column is covered by the union of any d other columns. The same structure is also called *cover free family* (Erdös et al., 1985; Ruszinkó, 1994; Füredi, 1996) in combinatorics, and *superimposed code* (Kautz and Singleton, 1964; D'yachkov et al., 1989, 2000) in information theory, and has been extensively studied. Obviously if a matrix is *d-disjunct*, then it is also $\bar{d}$-separable, and thus is d-separable. If the matrix M representing a nonadaptive pooling design is d-disjunct and the number of positives is no more than d, then we have the following efficient decoding procedure with running time linear in the size of M: a column c corresponds to a positive item if and only if c is covered by the outcome vector. d-Disjunct matrix is an important structure in pooling design, and a lot of work has been done on its constructions (Kautz and Singleton, 1964; Erdös et al., 1985; Hwang and Sós, 1987; Macula, 1996; D'yachkov et al., 2000; Ngo and Du, 2002; Park et al., 2003; Du et al., 2006; Fu and Hwang, 2006; Eppstein et al., 2007; Cheng and Du, 2007, 2008).

1.2. Main results

Given an $m\times n$ binary matrix and a positive integer d, to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct) are basic problems in pooling design. They are known to be coNP-complete in classical complexity theory (Du and Ko, 1987). Thus, we shouldn't expect any polynomial time algorithm to solve any of them. However, since in most applications we have that $d\ll n$, an interesting question is whether there are efficient algorithms solving the above decision problems for small values of d.

In this article, by studying the parameterized complexity of the above three problems with d as the parameter, we give a negative answer to the above question. More formally, we study the parameterized

decision problems *p*-DISJUNCTNESS-TEST, *p*-SEPARABILITY-TEST, and *p*-SEPARABILITY*-TEST defined as follows (where $\mathcal{N}$ denotes the set of positive integers).

***p*-DISJUNCTNESS-TEST**
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is d-disjunct.
***p*-SEPARABILITY-TEST**
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is d-separable.
***p*-SEPARABILITY*-TEST**
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is $\bar{d}$-separable.

The main results of our paper are summarized in the following theorem, whose proof will be presented in Section 3.

Theorem 1.1. *p*-DISJUNCTNESS-TEST, *p*-SEPARABILITY*-TEST, *and p*-SEPARABILITY-TEST *are all co-W[2]-complete.*

W[2] is the parameterized complexity class at the second level of the W-hierarchy, and co-W[2] is the class of all parameterized problems whose complements are in W[2]. They will be introduced in detail in the next section. Theorem 1.1 indicates that, given an $m \times n$ binary matrix and a positive integer d, a deterministic algorithm with running time $f(d) \times (mn)^{O(1)}$ (where f is an arbitrary computable function) to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct) does not exist unless the class W[2] collapses to FPT (the class of all fixed-parameter tractable problems), which is commonly conjectured to be false.

2. PRELIMINARIES

Before proving our main results, we first briefly recall the notions of fixed-parameter tractability, relational structures, first-order logic, and the W-hierarchy of parameterized complexity classes.

2.1. Fixed-parameter tractability

The theory of *fixed-parameter tractability* (Downey and Fellows, 1999; Flum and Grohe, 2006) has received considerable attention in recent years, for both theoretical research and practical computation. In this article, we adopt the notations and conventions in Flum and Grohe (2006). Let Σ denote a fixed finite alphabet. A *parameterization* of Σ^* is a polynomial time computable mapping $\kappa : \Sigma^* \to \mathcal{N}$. A *parameterized problem* (over Σ) is a pair (Q,κ) consisting of a set $Q \subseteq \Sigma^*$ and a parameterization κ of Σ^*.

An algorithm A with input alphabet Σ is an *fpt-algorithm with respect to* κ, if for every $x \in \Sigma^*$ the running time of A on input x is at most $f(\kappa(x))|x|^{O(1)}$, for some computable function f. A parameterized problem (Q,κ) is *fixed-parameter tractable* if there is an fpt-algorithm with respect to κ that decides Q. The key point of the definition of fpt-algorithm is that the superpolynomial growth of running time is confined to the parameter $\kappa(x)$, which is usually known to be comparatively small. The class of all fixed-parameter tractable problems is denoted by FPT.

Many NP-hard problems such as the VERTEX COVER problem (Chen et al., 2001) and the ML TYPE-CHECKING problem (Lichtenstein and Pnueli, 1985) have been shown to be fixed-parameter tractable. On the other hand, there is strong theoretical evidence that certain well-known parameterized problems, for instance, the INDEPENDENT SET problem and the DOMINATING SET problem, are not fixed-parameter tractable (Downey and Fellows, 1999). This evidence is provided, similar to the theory of NP-completeness, via a completeness theory based on the following notion of reductions: Let (Q,κ) and (Q',κ') be

parameterized problems over alphabets Σ and Σ', respectively. An *fpt-reduction* from (Q,κ) to (Q',κ') is a mapping $R : \Sigma^* \to (\Sigma')^*$ such that:

1. For all $x \in \Sigma^*$, $x \in Q$ if and only if $R(x) \in Q'$.
2. R is computable by an fpt-algorithm (with respective to κ). That is, there is a computable function f such that $R(x)$ is computable in time $f(\kappa(x))|x|^{O(1)}$.
3. There is a computable function $g : \mathcal{N} \to \mathcal{N}$ such that $\kappa'(R(x)) \leq g(\kappa(x))$, for all $x \in \Sigma*$.

In the above definition, the last requirement is to ensure that class FPT is closed under fpt-reductions, that is, if a parameterized problem (Q,κ) is reducible to another parameterized problem (Q',κ') and $(Q',\kappa') \in$ FPT, then $(Q,\kappa) \in$ FPT.

2.2. Relational structures

In later discussions, we adopt the conventions in descriptive complexity theory, in which instances of decision problems are viewed as structures of some vocabulary instead of languages over some finite alphabet.

A *(relational) vocabulary* τ is a set of relation symbols. Each relation symbol $R \in \tau$ has an *arity* arity$(R) \geq 1$. A *structure* $\mathcal{A}$ of vocabulary τ consists of a set A called the *universe* and an interpretation $R^{\mathcal{A}} \subseteq A^{arity(R)}$ of each relation symbol $R \in \tau$. For a tuple $\bar{a} \in A^{arity(R)}$, we write $R^{\mathcal{A}}\bar{a}$ (or $\bar{a} \in R^{\mathcal{A}}$) to denote that $\bar{a}$ belongs to the relation $R^{\mathcal{A}}$. In this article, we only consider nonempty finite vocabularies and structures with a finite universe.

Recall that a hypergraph is a pair $H=(V,E)$ consisting of a set V of vertices and a set E of hyperedges. Each hyperedge is a subset of V. Graphs are hypergraphs with hyperedges of cardinality two. The following example illustrates how to represent a hypergraph using a relational structure.

Example 2.1. Let τ_{HG} be the vocabulary consisting of the unary relation symbols *VERT* and *EDGE* and the binary relation symbol I. A hypergraph $H=(V,E)$ can be represented by a relational structure $\mathcal{H}$ of vocabulary τ_{HG} as follows:

- The universe of $\mathcal{H}$ is $V \cup E$.
- $VERT^{\mathcal{H}} := V$ and $EDGE^{\mathcal{H}} := E$.
- $I^{\mathcal{H}} := \{(v,e) : v \in V, e \in E, \text{ and } v \in e\}$ is the *incidence relation.*

2.3. First-order logic

We briefly recall the syntax of first-order logic. Let τ be a vocabulary. *Atomic* first-order formulas of vocabulary τ are of the form $x=y$ or $Rx_1 \ldots x_\ell$, where $R \in \tau$ is ℓ-ary (i.e., has arity ℓ) and $x, y, x_1, \ldots, x_\ell$ are variables. First-order formulas of vocabulary τ are built from atomic formulas using Boolean connectives $\wedge$ (and), $\vee$ (or), $\neg$ (negation), together with the existential and universal quantifiers $\exists$ and $\forall$. The connectives $\rightarrow$ (implication) and $\leftrightarrow$ (equivalence) are not part of the language defining first-order formulas, but we use them as abbreviations: $\varphi \rightarrow \psi$ stands for $\neg\varphi\vee\psi$, and $\varphi \leftrightarrow \psi$ stands for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$.

For a variable x, we call x a free variable of φ if x occurs in φ but is not in the scope of a quantifier binding x. We write $\varphi(x_1, \ldots, x_\ell)$ to indicate that all free variables of φ belong to set $\{x_1, \ldots, x_\ell\}$. A formula without free variables is called a *sentence*. Let both Σ_0 and Π_0 denote the class of quantifier-free first-order formulas. For $t \geq 0$, let Σ_{t+1} be the class of all formulas $(\exists x_1 \ldots \exists x_\ell)\varphi$, where $\varphi \in \Pi_t$, and let Π_{t+1} be the class of all formulas $(\forall x_1 \ldots \forall x_\ell)\varphi$, where $\varphi \in \Sigma_t$.

For formulas of second-order logic, in addition to the individual variables, they may also contain *relation variables*, each of the relation variables has a prescribed arity. We use lowercase letters (e.g., x,y,z) to denote individual variables and uppercase letters (e.g., X,Y,Z) to denote relation variables. As in Flum and Grohe (2006), for convenience we allow *free* relation variables to be in first-order formulas, since the crucial difference between first-order and second-order logic is not that second-order formulas can have relation variables, but that second-order formulas can quantify over relations. Therefore, in this paper the syntax of first-order logic is enhanced by including new atomic formulas of the form $Xx_1 \ldots x_\ell$, where X is an ℓ-ary relation variable. The meaning of formula $Xx_1 \ldots x_\ell$ is: The tuple of elements interpreting $(x_1, \ldots, x_\ell)$ is contained in the relation interpreting the relation variable X. We also extend classes such as Σ_t and Π_t to include formulas with free relation variables. It is worth emphasizing again that in first-order logic we do not allow quantification over relation variables.

2.4. W-hierarchy

We give a brief introduction to the W-hierarchy of parameterized complexity classes, which plays a central role in the theory of parameterized intractability. Roughly speaking, the W-hierarchy classifies problems according to the syntactic form of their definitions, and the definitions are formalized using languages of mathematical logic. The W-hierarchy can be defined in several different ways, we adopt the following definition based on the *weighted Fagin-defined problems*.

Let $\varphi(X)$ be a first-order formula with a free relation variable X of arity s. Define p-WD$_\varphi$ to be the following parameterized decision problem.

p-WD$_\varphi$:
Instance: A *structure* $\mathcal{A}$ and $k \in \mathcal{N}$.
Parameter: κ.
Problem: Decide whether there is a relation $S \subseteq A^s$ of cardinality k such that $\mathcal{A}| = \varphi(S)$.

Here, $\mathcal{A}| = \varphi(S)$ stands for that structure $\mathcal{A}$ satisfies sentence $\varphi(S)$ (or, $\mathcal{A}$ is a model of $\varphi(S)$), and S is called a *solution* for φ in structure$\mathcal{A}$. The readers are referred to Section 4.2 of Flum and Grohe (2006) for more detailed introduction to the semantics of first-order formulas.

For a class Φ of first-order formulas, let p-WD-Φ be the class of all parameterized problems p-WD$_\varphi$ with $\varphi \in \Phi$. For $t \geq 1$, define W[t]:= [p-WD-Π_t]^{fpt}, which is the class of all parameterized problems that are fpt-reducible to some problems in p-WD-Π_t. The classes W[t], for $t \geq 1$, form the W-hierarchy. Thus, the levels of W-hierarchy essentially correspond to the number of alternations between universal and existential quantifiers in the definitions of their complete problems. Problems hard for W[1] or higher class are assumed not to be fixed-parameter tractable. For instance, the INDEPENDENT SET problem is W[1]-complete and the DOMINATING SET problem is W[2]-complete.

For a parameterized problem (Q,κ) over the alphabet Σ, let $(Q,\kappa)^c$ denote its complement, that is the parameterized problem $(\Sigma^* \setminus Q,\kappa)$. Let C be a parameterized complexity class. Then co-C is defined to be the class of all parameterized problems (Q,κ) such that $(Q,\kappa)^c \in C$. Clearly, FPT = co-FPT. From the definition of fpt-reductions, it is easy to see that if class C is closed under fpt-reductions, so is co-C. In particular, each class W[t], $t \geq 1$, gives rise to a new parameterized complexity class co-W[t]. Also, it is easy to prove that if (Q,κ) is complete in parameterized complexity class C under fpt-reductions, then $(Q,\kappa)^c$ is complete in class co-C under fpt-reductions.

3. PROOF OF THEOREM 1.1

We devote this section to the proof of Theorem 1.1. For a binary matrix M, let R_M be the set consisting of all rows in M and let C_M be the set consisting of all columns in M.

Relational structure for a binary matrix. Let τ_{BM} be the vocabulary consisting of the unary relation symbols *ROW* and *COLUMN* and the binary relation symbol I. Then, the binary matrix M can be represented by a structure $\mathcal{M}$ of vocabulary τ_{BM}, where the universe of $\mathcal{M}$ is $R_M \cup C_M$, and the interpretations for the relation symbols in τ_{BM} are as follows:

- $ROW^{\mathcal{M}} := R_M$.
- $COLUMN^{\mathcal{M}} := C_M$.
- $I^{\mathcal{M}} := \{(r,c) : r \in R_M, c \in C_M, \text{ and } M(r,c) = 1\}$, which is the incidence relation.

The proof of Theorem 1.1 is partitioned into the following six lemmas.

Lemma 3.1. p-DISJUNCTNESS-TEST $\in$ co-W[2].

Proof. We consider the following complement problem of p-DISJUNCTNESS-TEST.

***p*-NONDISJUNCTNESS-TEST**
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is NOT d-disjunct.

We will define a Π_2 formula *nondisj*(X), with a free relation variable X of arity 2, and show that p-NONDISJUNCTNESS-TEST is equal to p-WD$_{nondisj(X)}$ (see Section 2.4 for the definition of problem p-WD$_\varphi$). This implies that p-NONDISJUNCTNESS-TEST is in p-WD-Π_2, therefore is in W[2].

A binary matrix is not d-disjunct if and only if there is a set D of d columns and another column $c \notin D$ such that $U(D)$ covers c. Our idea is assuming that the solution S to X is of the form $\{(c_i,c) : c_i \in D\}$. Therefore, X should be a binary relation variable and the solution S to X should have cardinality d.

Define $\chi_1 := \forall c_1 \forall c_2 (Xc_1c_2 \rightarrow (COLUMNc_1 \wedge COLUMNc_2 \wedge (c_1 \neq c_2)))$, and define $\chi_2 := \forall c_3 \forall c_4 \forall c_5 \forall c_6\ ((Xc_3c_4 \wedge Xc_5c_6) \rightarrow (c_4 = c_6))$. Here χ_1 and χ_2 are to guarantee that the solution S to X of cardinality d has the form $\{(c_1,c),\ldots,(c_d,c)\}$, where $c \in C_M$, $c_i \in C_M$, and $c_i \neq c$, for $1 \leq i \leq d$. Define $nondisj'(X) := \forall r \exists c_7 \exists c_8 (ROWr \rightarrow (Xc_7c_8 \wedge (Irc_8 \rightarrow Irc_7)))$. $nondisj'(X)$ is to guarantee that, the solution S to X satisfies that the union of columns in $\{c_i : (c_i,c) \in S\}$ covers c (so that $\mathcal{M}$ is not d-disjunct). Finally, define $nondisj(X) := \chi_1 \wedge \chi_2 \wedge nondisj'(X)$, which is equivalent to a Π_2 formula.

From the above definition, clearly if there exists a relation $S \subseteq C_M^2$ of cardinality d such that $\mathcal{M}| = nondisj(S)$, then $\mathcal{M}$ is not d-disjunct. On the other hand, if $\mathcal{M}$ is not d-disjunct, then there exist a subset D of d columns $c_1,\ldots,c_d$ and another column $c \notin D$ such that c is covered by $U(D)$. It is not hard to verify that the relation $S = \{(c_1,c),\ldots,(c_d,c)\}$ satisfies $\mathcal{M}| = nondisj(S)$. Therefore, $\mathcal{M}$ is not d-disjunct if and only if there exists a relation S of cardinality d such that $\mathcal{M}| = nondisj(S)$. That is, p-NONDISJUNCTNESS-TEST is p-WD$_{nondisj(X)}$. Thus, p-NONDISJUNCTNESS-TEST $\in$ W[2], and so p-DISJUNCTNESS-TEST $\in$ co-W[2]. ■

Lemma 3.2. p-SEPARABILITY-TEST $\in$ co-W[2].

Proof. We consider the following complement problem of p-SEPARABILITY-TEST.

***p*-NONSEPARABILITY-TEST**
Instance: A binary matrix $\mathcal{M}$ and $d \in \mathcal{N}$.
Parameter: d.
Problem: Decide whether $\mathcal{M}$ is NOT d-separabe.

We will define a formula *nonsep*(Y), with a free relation variable Y of arity 2, and show that p-NONSEPARABILITY-TEST is equal to p-WD$_{nonsep(Y)}$.

A binary matrix is not d-separable if and only if there exist two distinct subsets D_1 and D_2 each contains d columns such that $U(D_1) = U(D_2)$. Assume that $D_1 = \{c_{11}, c_{12}, \ldots, c_{1d}\}$ and $D_2 = \{c_{21}, c_{22}, \ldots, c_{2d}\}$. The idea is to assume that the solution S to Y is of the form $\{(c_{11},c_{21}),(c_{12},c_{22}),\ldots,(c_{1d},c_{2d})\}$, and so Y should be a binary relation variable and the solution S to Y should have cardinality d. We define the formula *nonsep*(Y) satisfying that, there exists a relation $S \subseteq C_M^2$ of cardinality d such that $\mathcal{M}| = nonsep(S)$ if and only if $\mathcal{M}$ is not d-separable.

Define $\chi_3 := \forall c_1 \forall c_2 (Yc_1c_2 \rightarrow (COLUMNc_1 \wedge COLUMNc_2))$, $\chi_4 := \forall c_3 \forall c_4 \forall c_5 \forall c_6 ((Yc_3c_4 \wedge Yc_5c_6) \rightarrow ((c_3 = c_5) \leftrightarrow (c_4 = c_6)))$, and $\chi_5 := \exists c_7 \exists c_8 \forall c_9 ((Yc_7c_8 \wedge \neg Yc_9c_7))$. Here χ_3 is to guarantee that the relation variable $Y \subseteq C_M^2$; χ_4 is to build a bijection between the first component of elements in S and the second component of elements in S, which guarantees that the two subsets $\{c_{1i} : \exists cs.t.(c_{1i},c) \in S\}$ and $\{c_{2j} : \exists cs.t.(c,c_{2j}) \in S\}$ (which intend to be D_1 and D_2 respectively) have the same cardinality; χ_5 is to guarantee that the two subsets $\{c_{1i} : \exists cs.t.(c_{1i},c) \in S\}$ and $\{c_{2j} : \exists cs.t.(c,c_{2j}) \in S\}$ are distinct from each other. Define $nonsep'(Y) := \forall r (ROWr \rightarrow ((\exists c_{10} \exists c_{11} Yc_{10}c_{11} \wedge Irc_{10}) \leftrightarrow (\exists c_{12} \exists c_{13} Yc_{12}c_{13} \wedge Irc_{13})))$, which is to guarantee that, the solution S to Y satisfies that the union of columns in $\{c_{1i} : \exists cs.t.(c_{1i},c) \in S\}$ is equal to the union of columns in $\{c_{2j} : \exists cs.t.(c,c_{2j}) \in S\}$. From basic logic computation it is not hard to verify that $nonsep'(Y)$ is a Π_2 formula with free relation variable Y. Finally, define $nonsep(Y) := \chi_3 \wedge \chi_4 \wedge \chi_5 \wedge nonsep'(Y)$.

From the above definition of *nonsep*(X), if a relation $S \subseteq C_M^2$ of cardinality d satisfies that $\mathcal{M}| = nonsep(S)$, then the two subsets $\{c_{1i} : \exists cs.t.(c_{1i},c) \in S\}$ and $\{c_{2j} : \exists cs.t.(c,c_{2j}) \in S\}$ both contain d columns of $\mathcal{M}$ and are distinct from each other, moreover their unions are identical. This implies that $\mathcal{M}$ is not d-separable. On the other hand, if $\mathcal{M}$ is not d-separable, then there exist two distinct subsets D_1 and D_2 each contains d columns such that $U(D_1) = U(D_2)$. Assume that $D_1 = \{c_{11},\ldots,c_{1d}\}$, and $D_2 = \{c_{21},\ldots,c_{2d}\}$. It is not hard to verify that the relation $S = \{(c_{11},c_{21}),(c_{12},c_{22}),\ldots,(c_{1d},c_{2d})\}$ satisfies $\mathcal{M}| = nonsep(S)$.

From above, there exists a relation $S \subseteq C_M^2$ of cardinality d such that $\mathcal{M}| = nonsep(S)$ if and only if $\mathcal{M}$ is not d-separable, therefore p-NONSEPARABILITY-TEST is p-WD$_{nonsep(Y)}$. χ_3 and χ_4 are Π_1 formulas, χ_5 is a Σ_2 formula, $nonsep'(Y)$ is a Π_2 formula, therefore $nonsep(Y) = \chi_3 \wedge \chi_4 \wedge \chi_5 \wedge nonsep'(Y)$ is equivalent to a Σ_3 formula, which implies that p-NONSEPARABILITY-TEST is in p-WD-Σ_3. Here we apply the fact that p-*WD*-$\Sigma_3 \subseteq p$-*WD*-Π_2.[1] Thus, p-NONSEPARABILITY-TEST $\in$ W[2], and so p-SEPARABILITY-TEST $\in$ co-W[2]. ■

Lemma 3.3. p-SEPARABILITY*-TEST $\in$ co-W[2].

Proof. Since W[2] is closed under fpt-reductions, so is co-W[2]. We will show that p-SEPARABILITY*-TEST is fpt-reducible to p-SEPARABILITY-TEST. Since the latter is in co-W[2] (lemma 3.2), this implies that p-SEPARABILITY*-TEST $\in$ co-W[2]. The reduction can be obtained immediately from the following fact (lemma 2.1.6 in Du and Hwang [2006]): A binary matrix M' containing a zero column is d-separable if and only if the matrix M obtained by removing this zero column from M' is $\bar{d}$-separable.

Let (M,d) be an instance of p-SEPARABILITY*-TEST, where M is a binary matrix and the parameter d is a positive integer. We map (M,d) to (M',d), where M' is obtained by adding a zero column to M. From the above lemma, $(M,d) \in p$-SEPARABILITY*-TEST if and only if $(M',d) \in p$-SEPARABILITY-TEST. It is easy to see that this is an fpt-reduction from p-SEPARABILITY*-TEST to p-SEPARABILITY-TEST. ■

Lemma 3.4. p-DISJUNCTNESS-TEST *is* co-W[2]-*complete.*

Proof. A *hitting set* in a hypergraph $\mathcal{H} = (V, E)$ is a set T of vertices that intersects each hyperedge, that is $T \cap e \neq \emptyset$ for all $e \in E$. The classical HITTING-SET problem is to find a hitting set of a given cardinality k in a given hypergraph $\mathcal{H}$, which is known to be NP-complete. The following parameterized hitting set problem is W[2]-complete (see, Theorem 7.14 in Flum and Grohe [2006]).

***p*-HITTING-SET**
Instance: A hypergraph $\mathcal{H}$ and $k \in \mathcal{N}$.
Parameter: k.
Problem: Decide whether $\mathcal{H}$ has a hitting set of k vertices.

We give an ftp-reduction from p-HITTING-SET to p-NONDISJUNCTNESS-TEST, based on an idea similar to that in Du and Ko (1987). Let $(\mathcal{H},k)$ with $\mathcal{H} = (V, E)$ be an instance of p-HITTING-SET, where $V = \{1, \ldots, n\}, E = \{e_1, \ldots, e_m\}$, and each e_i, $1 \leq i \leq m$, is a subset of V. Define $d = k$, and define an $(n+m) \times (n+1)$ binary matrix M with rows R_i as follows (here we represent each row as a subset of the set of all columns $\{1, 2, \ldots, n+1\}$, in the most natural way).

$$R_i = \{i\}, \qquad i = 1, \ldots, n;$$
$$R_{n+j} = e_j \cup \{n+1\}, \quad j = 1, \ldots, m.$$

First, assume that $\mathcal{H}$ has a hitting set $T \subseteq V$ of size k. Consider the subset $S_1 = T$ of columns of M. Since T is a hitting set of $\mathcal{H}$, $U(S_1)$ covers column $n+1$. Notice that $|S_1| = d$ and column $n+1$ is not in S_1, M is not d-disjunct.

Conversely, assume that M is not d-disjunct. Then, there exist a subset S_1 of d columns in $\{1, 2, \ldots, n+1\}$ and another column $c \notin S_1$ such that $U(S_1)$ covers c. From the way we define the first n rows of matrix M, c can only be column $n+1$. Thus, column $n+1$ is not in S_1. Set $T = S_1$, then $|T| = k$ and T is a subset of V . Since $U(S_1)$ covers column $n+1$, it is easy to see that T is a hitting set of $\mathcal{H}$.

It is not hard to verify that the above is an fpt-reduction. Therefore, p-NONDISJUNCTNESS-TEST is W[2]-complete, and so p-DISJUNCTNESS-TEST is co-W[2]-complete. ■

Lemma 3.5. p-SEPARABILITY*-TEST *is* co-W[2]-*complete.*

[1]More generally, p-*WD*-$\Sigma_{t+1} \subseteq p$-*WD*-Π_t, for $t \geq 1$. The main idea to prove this conclusion is not complicated; we refer the reader to Proposition 5.4 in Flum and Grohe (2006) for the proof.

Proof. We give an ftp-reduction from p-HITTING-SET to p-NONSEPARABILITY*-TEST. For an instance $(\mathcal{H}, k)$ of p-HITTING-SET, define matrix M in the same way as in the proof of lemma 3.4, and define $d = k + 1$. Next we show the correctness of this reduction.

First, assume that $\mathcal{H}$ has a hitting set $T \subseteq V$ of size k. Consider the following two subsets of columns in M: $S_1 = T$ and $S_2 = T \cup \{n+1\}$. Then, for $1 \le i \le n$, it is obvious that $U(S_1)_i = U(S_2)_i$; for $ni \le n+m$, since T is a hitting set of $\mathcal{H}$, $U(S_1)_i = 1 = U(S_2)_i$. Notice that $|S_1|, |S_2| \le d$ and $S_1 \ne S_2$, M is not $\bar{d}$-separable.

Conversely, assume that M is not $\bar{d}$-separable. Then, there exist two subsets S_1 and S_2 of columns in $\{1, 2, \ldots, n+1\}$ such that $|S_1|, |S_2| \le d$, $S_1 \ne S_2$, and $U(S_1) = U(S_2)$. Since $U(S_1)_i = U(S_2)_i$ for $1 \le i \le n$, we have that $S_1 \cap \{1, \ldots, n\} = S_2 \cap \{1, \ldots, n\}$. To have $S_1 \ne S_2$, column $n+1$ must belong to exactly one of S_1 and S_2. Without loss of generality, assume that $n+1 \notin S_1$, and $n+1 \in S_2$. Set $T = S_1$, then $|T| = |S_1| = |S_2| - 1 \le d - 1 = k$, and T is a subset of V . From $U(S_1)_i = U(S_2)_i = 1$ for $ni \le n+m$, T is a hitting set of $\mathcal{H}$.

Therefore, p-NONSEPARABILITY*-TEST is W[2]-complete, and so p-SEPARABILITY*-TEST is co-W[2]-complete. ■

Lemma 3.6. p-SEPARABILITY-TEST *is* co-W[2]-*complete.*

Proof. Since as proved before in lemma 3.3 that p-SEPARABILITY*-TEST is fpt-reducible to p-SEPARABILITY-TEST, and in lemma 3.5 that p-SEPARABILITY*-TEST is co-W[2]-complete, p-SEPARABILITY-TEST is co-W[2]-hard. Together with lemma 3.2 that p-SEPARABILITY-TEST $\in$ co-W[2], we obtain that p-SEPARABILITY-TEST is co-W[2]-complete. ■

4. DISCUSSION

In this article, we studied the parameterized complexity of three basic problems in pooling design: given an $m \times n$ binary matrix and a positive integer d, to decide whether the matrix is d-separable ($\bar{d}$-separable, or d-disjunct). We proved that these problems are co-W[2]-complete; thus, they do not admit algorithms with running time $f(d) \times (mn)^{O(1)}$ for any computable function f. To the best of our knowledge, in general the best known algorithms for the above problems are simply brute-force. It is interesting to investigate whether these problems admit better algorithms.

ACKNOWLEDGMENTS

Y. Cheng and G. Lin were supported in part by AICML and NSERC. D.-Z. Du was supported in part by the National Science Foundation (grant CCF0621829).

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Balding, D.J., Bruno, W.J., Knill, E., et al. 1996. A comparative survey of non-adaptive pooling designs, 133–154. *Genetic Mapping and DNA Sequencing. IMA Volumes in Mathematics and Its Applications*. Springer-Verlag, New York.

Berger, T., Mandell, J.W., and Subrahmanya, P. 2000. Maximally efficient two-stage group testing. *Biometrics* 56, 833–840.

Chen J., Kanj, I.A., and Jia, W. 2001. Vertex cover: further observations and further improvements. *J. Algorithm* 41, 280–301.

Cheng, Y., and Du, D.Z. 2007. Efficient constructions of disjunct matrices with applications to DNA library screening. *J. Comput. Biol.* 14, 1208–1216.

Cheng, Y., and Du, D.Z. 2008. New constructions of one- and two-stage pooling designs. *J. Comput. Biol.* 15, 195–205.

De Bonis, A., Gasieniec, L., and Vaccaro, U. 2005. Optimal two-stage algorithms for group testing problems. *SIAM J. Comput.* 34, 1253–1270.

Dorfman, R. 1943. The detection of defective members of large populations. *Ann. Math. Statist.* 14, 436–440.

Downey, R.G., and Fellows, M.R. 1999. *Parameterized Complexity.* Springer, New York.

Du, D.Z., and Hwang, F.K. 2006. *Pooling Designs and Nonadaptive Group Testing: Important Tools for DNA Sequencing*. World Scientific, New York.

Du, D.Z., Hwang, F.K., Wu, W., et al. 2006. New construction for transversal design. *J. Comput. Biol.* 13, 990–995.

Du, D.Z., and Ko, K.I. 1987. Some completeness results on decision trees and group testing, *SIAM J. Algebra. Discr.* 8, 762–777.

D'yachkov, A.G., Macula, A.J., and Rykov, V.V. 2000. New constructions of superimposed codes. *IEEE Trans. Inform. Theory* 46, 284–290.

D'yachkov, A.G., and Rykov, V.V. 1982. Bounds of the length of disjunct codes. *Probl. Contr. Inform. Theory* 11, 7–13.

D'yachkov, A.G., Rykov, V.V., and Rashad A.M. 1989. Superimposed distance codes. *Probl. Contr. Inform. Theory* 18, 237–250.

Eppstein, D., Goodrich, M.T., and Hirschberg, D.S. 2007. Improved combinatorial group testing algorithms for real-world problem sizes. *SIAM J. Comput.* 36, 1360–1375.

Erdös, P., Frankl, P., and Füredi, Z. 1985. Families of finite sets in which no set is covered by the union of r others. *Israel J. Math.* 51, 79–89.

Farach, M., Kannan, S., Knill, E., et al. 1997. Group testing problems with sequences in experimental molecular biology, In B. Carpentieri et al., eds., 357–367. *Proceedings of the Compression and Complexity of Sequences*. IEEE Press, New York.

Flum, J., and Grohe, M. 2006. *Parameterized Complexity Theory. Volume XIV of Texts in Theoretical Computer Science*. Springer-Verlag, Berlin.

Fu, H.L., and Hwang F.K. 2006. A novel use of t-packings to construct d-disjunct matrices. *Discrete Appl. Math.* 154, 1759–1762.

Füredi, Z. 1996. On *r*-cover-free families. *J. Combin. Theory Ser. A* 73, 172–173.

Hwang, F.K., and Sós, V.T. 1987. Non-adaptive hypergeometric group testing. *Studia Sci. Math. Hungar.* 22, 257–263.

Kautz, W.H., and Singleton, R.C. 1964. Nonrandom binary superimposed codes. *IEEE Trans. Inform. Theory* 10, 363–377.

Knill, E. 1995. Lower bounds for identifying subset members with subset queries. *Proc. 6th ACM-SIAM Symp. Discrete Algorithms (SODA' 95)* 369–377.

Lichtenstein, O., and Pnueli, A. 1985. Checking that finite state concurrent programs satisfy their linear specification. *Proc. 12th ACM Symp. Principles Prog. Lang. (POPL' 85)* 97–107.

Macula, A.J. 1996. A simple construction of d-disjunct matrices with certain constant weights. *Discrete Math.* 162, 311–312.

Macula, A. J. 1999. Probabilistic nonadaptive and two-stage group testing with relatively small pools and DNA library screening. *J. Comb. Optim.* 2, 385–397.

Ngo, H.Q., and Du, D.Z. 2002. New constructions of non-adaptive and error-tolerance pooling designs. *Discrete Math.* 243, 161–170.

Olson, M., Hood, L., Cantor, C., et al. 1989. A common language for physical mapping of the human genome. *Science* 245, 1434–1435.

Park, H., Wu, W., Liu, Z., et al. 2003. DNA screening, pooling design and simplicial complex. *J. Comb. Optim.* 7, 389–394.

Ruszinkó, M. 1994. On the upper bound of the size of the *r*-cover-free families. *J. Combin. Theory Ser. A* 66, 302–310.

Address correspondence to:
Dr. Yongxi Cheng
Department of Computing Science
University of Alberta
Edmonton, Alberta T6G 2E8, Canada

E-mail: chengyx@gmail.com