Parameterized Complexity for Finding a Perfect Phylogeny from Mixed Tumor Samples

Wen-Horng Sheu and Biing-Feng Wang
Department of Computer Science, National Tsing Hua University
Hsinchu, Taiwan 30013, Republic of China
whsheu@gapp.nthu.edu.tw, bfwang@cs.nthu.edu.tw

Abstract.

Motivated by an application in cancer genomics, Hajirasouliha and Raphael [WABI 2014] proposed the *minimum split-row problem* (MSRP). In this problem, an $m \times n$ binary matrix M is given. A *split-row operation* on M is defined as replacing a row r by $k > 1$ rows $r^{(1)}, r^{(2)}, ..., r^{(k)}$ whose bitwise OR is equal to r. The cost of the operation is the number of additional rows induced, that is, $k - 1$. The goal is to find a sequence of split-row operations that transforms M into a matrix corresponding to a perfect phylogeny and the total cost is minimized. Recently, Hujdurović *et al.* [TALG 2018] proved the APX-hardness of MSRP and presented efficient exact and approximation algorithms. The parameterized study of MSRP was left as a direction for future work. Let $\varepsilon(M)$ denote the minimum total cost. This paper gives an $O^*(2^{\min(n,\, 2\varepsilon(M))})$-time exact algorithm for MSRP. This result implies that MSRP is *fixed-parameter tractable* when parameterized by $\varepsilon(M)$. In addition, in the worst case, our algorithm requires $O^*(2^n)$ time, significantly improving the previous upper bound of $O^*(n^n)$. Hujdurović *et al.*'s exact algorithm can be modified to solve a variant of MSRP, called the *minimum distinct conflict-free row split problem* (MDCRSP). Our algorithm can be modified to solve this variant as well. In addition, our MSRP and MDCRSP algorithms can be extended to solve MSRP and MDCRSP with the following additional constraint: only the rows in a given subset are allowed to be split.

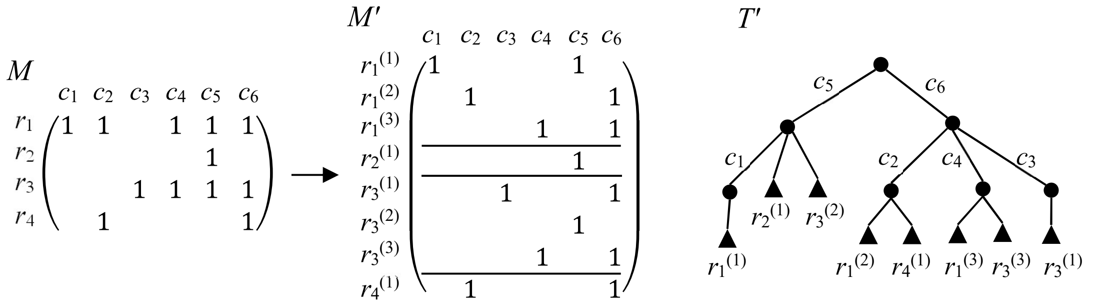
Key words. algorithms, fixed-parameter tractability, minimum split-row problem, minimum distinct conflict-free row split problem, perfect phylogenies

AMS subject classifications. 05C05, 68Q27, 68W40, 92D15

Abbreviated title. Parameterized Complexity for MSRP and MDCRSP

1. Introduction

A *perfect phylogeny* is a rooted tree representing the evolutionary history of m objects in terms of n *characters*. The objects correspond bijectively to the leaves of the tree, and each character labels exactly one edge of the tree. Each object is associated with the set of characters which it exhibits: for an object r and a character c, r has character c if and only if the edge labeled by c is on the unique path from r to the root. Figure 1(b) shows a perfect phylogeny T', where the objects are $\{r_1^{(1)}, r_1^{(2)}, r_1^{(3)}, r_2^{(1)}, r_3^{(1)}, r_3^{(2)}, r_3^{(3)}, r_4^{(1)}\}$ and the characters are $\{c_1, c_2, ..., c_6\}$. The leaf $r_1^{(2)}$ has characters c_2 and c_6. The *matrix representation* of a perfect phylogeny is an $m \times n$ binary matrix in which each row is associated with a leaf, each column is associated with a character, and the entry at row r and column c is 1 if and only if r has character c. In Figure 1, M' is the matrix representation of T'. In Figure 1, for clarity, we omit displaying zeros in a binary matrix. While each perfect phylogeny naturally corresponds to a binary matrix, the opposite may not be true. Given a binary matrix M, the *perfect phylogeny problem* asks if M corresponds to a perfect phylogeny. The perfect phylogeny problem and its various generalizations have received extensive study in computational biology [10, 11, 13, 21, 27]. In this paper, we study a generalization introduced by Hajirasouliha and Raphael [16]. In this problem, an $m \times n$ binary matrix M is given. A *split-row operation* on M is defined as replacing a row r of M by k rows $r^{(1)}, r^{(2)}, ..., r^{(k)}$ whose bitwise OR is equal to r. The cost of the operation is the number of additional rows induced, that is, $k - 1$. The goal is to find a sequence of split-row operations that transforms M into a matrix corresponding to a perfect phylogeny and the total cost is minimized. Consider the example in Figure 1(a). The matrix M is transformed into M' by performing two split-row operations: one replaces r_1 by 3 rows $r_1^{(1)}$, $r_1^{(2)}$ and $r_1^{(3)}$ and the other replaces r_3 by 3 rows $r_3^{(1)}$, $r_3^{(2)}$, and $r_3^{(3)}$. The total cost is $2 + 2 = 4$.



(a) Transforming a binary matrix M into M' by split-row operations

(b) A perfect phylogeny T' corresponding to M'

Figure 1. An illustrative example.

The study of MSRP was motivated by an application in cancer genomics [16]. Cancer is characterized by the uncontrolled growth of mutated cells, which results in the formation of tumors. Recent DNA sequencing technologies have enabled the identification of the somatic mutations involved in a tumor cell subpopulation. This new data has led to much interest in reconstructing the evolutionary history of somatic mutations [15, 16, 29]. Tumor evolution is assumed to admit a perfect phylogeny, in which each object is a tumor cell subpopulation and each character is a somatic mutation [18]. This phylogenetic tree can offer a more comprehensive knowledge of tumor progression [4, 26] and help in development of therapies [25]. In a tumor sample, we may assume that each measured mutation has one of the two possible states: 0 = normal and 1 = mutated. Thus, the sequencing data can be seen as a binary matrix, in which each row is associated with a tumor sample, each column is associated with a mutation, and the entry at row r and column c indicates whether mutation c is observed in sample r. In a perfect condition, the binary matrix shall exhibit a perfect phylogeny. However, the data used in most cancer sequencing studies are obtained from bulk sequencing, in which each sample may be a mixture of several genetically distinct cell subpopulations. Given such data, solving MSRP is to minimally decompose the samples and thereby reconstruct the perfect phylogeny. We refer the reader to [16, 19] for more information regarding the biological aspect of MSRP. Various other models have been proposed for inferring phylogenetic trees from tumor samples [9, 15, 20, 22, 28, 29]. We refer the reader to [24] for a survey.

Hujdurović *et al.* [18] showed the NP-hardness of MSRP and gave an efficient heuristic algorithm. Later, Hujdurović *et al.* [17] further proved that MSRP is APX-hard. A naive method for MSRP is to enumerate all possible ways to split each row into a set of distinct rows. The time complexity of this naive method is $2^{\Omega(mn)}$ in the worst case [17]. Hujdurović *et al.* [17] proposed a new formulation, called the *branching formulation*, of MSRP. In this formulation, the input matrix M is represented by a directed acyclic graph D_M. A *branching* is defined to be a subset of arcs of D_M which induces a directed spanning forest of D_M. Each branching is associated with a cost. Hujdurović *et al.* showed that the branching with minimum cost corresponds to an optimal solution of MSRP. Using this formulation, they solved MSRP more efficiently by enumerating all branchings. Their algorithm requires $O^*(n^n)$ time, where the O^* notation suppresses factors that are bounded by a polynomial in the input size. This result significantly improves the time complexity of the naive approach. Using the branching formulation, Hujdurović *et al.* also presented a polynomial-time approximation algorithm for MSRP. Let $\varepsilon(M)$ denote the minimum cost of transforming a binary matrix M into a matrix corresponding to a perfect phylogeny. Their approximation algorithm outputs a matrix of at most $\min\{h, w\} \times (m + \varepsilon(M))$ rows, where h and w are, respectively, the height and the width of D_M. Based on the branching formulation, Husić *et al.* [19] formulated MSRP as an Integer Linear Program (ILP) and implemented it into a software package called MIPUP, using the CPLEX ILP solver.

The parameterized study of MSRP was left as a direction for future work in [17]. In this paper, we show that MSRP can be solved in $O^*(2^{\min(n,\, 2\varepsilon(M))})$ time. In the language of *parameterized complexity*, our result implies that MSRP is *fixed-parameter tractable* (FPT) when parameterized by $\varepsilon(M)$. A parameterized problem is FPT if it can be solved in $f(k)\cdot I^{O(1)}$ time, where k is the parameter, I denotes the input size, and f is a computable function depends only on k. The fixed-parameter tractability of MSRP signifies that the combinatorial explosion in the running time of an algorithm can be confined to the parameter $\varepsilon(M)$, and not necessarily depends on the input size. The reader may refer to the monograph of Downey and Fellows [7] for more information about the parameterized complexity. In the application of inferring a perfect phylogeny from tumor samples, $\varepsilon(M)$ measures the amount of mixing within the tumor samples. Our result indicates that when the amount of mixing is small, MSRP can be solved efficiently. In addition, in the worst case, our algorithm requires $O^*(2^n)$ time, significantly improving the previous upper bound of $O^*(n^n)$.

A variant of MSRP, called the *minimum distinct conflict-free row split problem* (MDCRSP), was also introduced by Hajirasouliha and Raphael in [16]. The task of MDCRSP is to split the rows of the input matrix such that the resulting matrix corresponds to a perfect phylogeny and has the minimum number of distinct rows. For the example in Figure 1, the number of distinct rows in M' is 5, since $r_1^{(2)} = r_4^{(1)}$, $r_1^{(3)} = r_3^{(3)}$ and $r_2^{(1)} = r_3^{(2)}$. Hujdurović *et al.* [18] showed the NP-hardness of MDCRSP. Hujdurović *et al.* [17] strengthened the NP-hardness result by proving that MDCRSP is APX-hard. In addition, they gave a 2-approximation algorithm, which implies that MDCRSP is APX-complete, and gave an $O^*(n^n)$-time exact algorithm. Our MSRP algorithm can be modified to solve MDCRSP in $O^*(2^{\min(n,\ 3\varepsilon(M))})$ time. In MSRP, all rows are allowed to be split. A constrained version of MSRP mentioned in [17, 18] is as follows: an additional subset of rows is given and only the rows in the given subset are allowed to be split. Denote this constrained version as CMSRP. A constrained version of MDCRSP can be defined similarly, denoted by CMDCRSP. To the authors' knowledge, no algorithm for CMSRP and CMDCRSP has been presented in the literature. Our algorithms for MSRP and MDCRSP can be extended to solve CMSRP and CMDCRSP in the same time complexities. Table 1 summarizes exact algorithms for MSRP, CMSRP, MDCRSP, and CMDCRSP.

Problem	Results	Remark
MSRP	$O^*(n^n)$	[17]
	ILP	[19]
	$O^*(2^{\min(n,\ 2\varepsilon(M))})$	this
CMSRP	$O^*(2^{\min(n,\ 2\varepsilon(M))})$	this
MDCRSP	$O^*(n^n)$	[17]
	ILP	[19]
	$O^*(2^{\min(n,\ 3\varepsilon(M))})$	this
CMDCRSP	$O^*(2^{\min(n,\ 3\varepsilon(M))})$	this

Table 1. Exact algorithms for MSRP, CMSRP, MDCRSP, and CMDCRSP.

Our new algorithm for MSRP is designed based on the branching formulation in [17]. An overview is as follows. In the computation of an optimal branching, we find that some vertices, called the *sibling-compatible vertices*, can be removed from D_M without changing $\varepsilon(M)$. A branching represents a directed spanning forest of D_M and thus can be uniquely specified by the parent of each vertex. In the computation of an optimal branching, we further find that the parents of some vertices, called the *regular vertices*, can be predetermined in a greedy approach such that there exists an optimal branching in which the parent of each regular vertex coincides the predetermined one. On the basis of the findings, we give an algorithm consisting of three phases: Phase 1 removes sibling-compatible vertices; Phase 2 precomputes the parent of each regular vertex; and Phase 3 finds an optimal branching using dynamic programming. Phase 1 requires $O(mn^2 + n^3)$ time. Phase 2 requires $O(n^2)$ time. Let $t(M)$ be the number of vertices that are non-regular and non-sibling-compatible. Phase 3 requires $O^*(2^{t(M)})$ time. To estimate the worst-case complexity of Phase 3, we give an upper bound on $t(M)$. More precisely, we show that $t(M) \le \min(n, 2\cdot\varepsilon(M) - 1)$. Consequently, our algorithm runs in $O^*(2^{\min(n,\ 2\varepsilon(M))})$ time.

This paper is organized as follows. Section 2 reviews the branching formulation. Section 3 defines regular vertices and sibling-compatible vertices and gives an upper bound on $t(M)$, which is the number of vertices that are non-regular and non-sibling-compatible. Section 4 presents our new algorithm for MSRP. Section 5 modifies the new algorithm to solve CMSRP, MDCRSP, and CMDCRSP. Section 6 concludes this paper with some final remarks.

2. Review of the branching formulation

This section describes Hujdurović *et al.*'s branching formulation of MSRP. Let $M \in \{0, 1\}^{m\times n}$ be a binary matrix. Each row of M is a vector in $\{0, 1\}^n$ and each column of M is a vector in $\{0, 1\}^m$. We denote by $R(M) = \{r_1, r_2, ..., r_m\}$ the set of rows of M and by $C(M) = \{c_1, c_2, ..., c_n\}$ the set of columns of M. The entry of M at row r_i and column c_j is denoted by M_{r_i,c_j} or $M_{i,j}$ when appropriate. The input matrix is assumed to contain no rows or columns whose entries are all zeros. Clearly, the removal of such rows or columns does not change $\varepsilon(M)$. We also assume that all columns in the input matrix are distinct. In case this is not true, the removal of duplicates also does not change $\varepsilon(M)$ [17].

2.1 Connection between conflict-free row splits and perfect phylogenies

Definition 1. *Two columns c and c' of M are in conflict if there exist three rows r, r', r'' such that* (1) $M_{r,c} = 1$ *and* $M_{r,c'} = 1$, (2) $M_{r',c} = 0$ *and* $M_{r',c'} = 1$, *and* (3) $M_{r'',c} = 1$ *and* $M_{r'',c'} = 0$. *We say that M is conflict-free if no two columns of M are in conflict.*

By the Perfect Phylogeny Theorem [13, 14], a binary matrix M exhibits a perfect phylogeny if and only if M is conflict-free. The intuition behind this theorem is as follows. For a column c of M, we define the *support* of c, denoted by $supp_M(c)$, as the set $\{r \mid r \in R(M), M_{r,c} = 1\}$. Note that $supp_M(c)$ is non-empty, since each column in M has at least one non-zero entry. For the example in Figure 1, $supp_M(c_6) = \{r_1, r_3, r_4\}$. Consider two columns $c_i, c_j \in C(M)$, $i \neq j$. Let $u = supp_M(c_i)$ and $v = supp_M(c_j)$. Since all columns in M are distinct, we have $u \neq v$. The relation between u and v can be classified into three types: u and v are *disjoint* if $u \cap v = \varnothing$; u and v are *nested* if $u \subset v$ or $v \subset u$; and otherwise, u and v are *in conflict*. For convenience, for two supports u, v, we say that u is contained in v if $u \subset v$ and say that u is *compatible* with v if they are not in conflict. Observe that two columns of M are in conflict if and only if their supports are in conflict. Suppose that M corresponds to a perfect phylogeny T. By definition, each column of M represents an edge of T and each row of M represents a leaf of T. For each column c of M, its support is just the set of leaves in T having character c, that is, the set of leaves in the subtree under the edge labeled by c. The leaf sets under two edges of T are either disjoint or nested. Thus, M must be conflict-free. On the other hand, suppose that M is conflict-free. By definition, the supports of any two columns are either disjoint or nested. Knowing this fact, it is not difficult to construct a perfect phylogeny corresponding to M. In particular, [13] showed that the construction can be done in linear time. The matrix representation of a perfect phylogeny is unique, while the opposite is not true. For example, Figure 2 gives two other phylogenies corresponding to the matrix M' in Figure 1(a). For ease of discussion, in this paper, we consider only phylogenies in which every internal edge is associated with a label and every other edge does not have a label, where an internal edge is an edge that is not incident to a leaf. Under this consideration, the phylogeny corresponding to a conflict-free matrix is unique. For example, the phylogeny T' in Figure 1(b) is the unique phylogeny corresponding to the matrix M' in Figure 1(a).

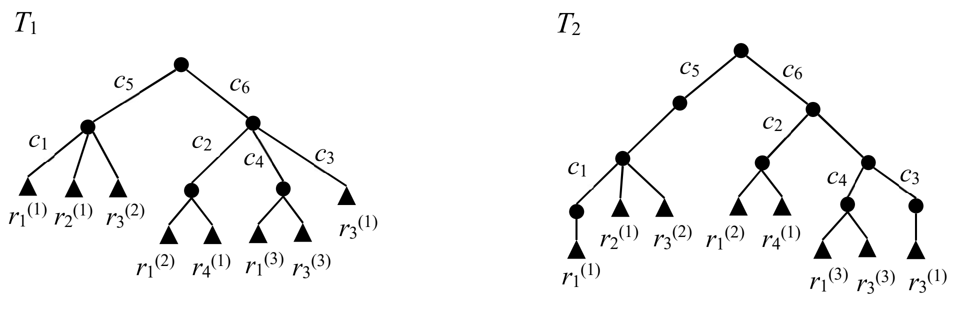


Figure 2. Two phylogenies corresponding to the matrix M' in Figure 1(a).

A matrix is a *row split* of M if it can be obtained from M by performing split-row operations. Let M' be a row split of M. By definition, every row r of M corresponds to one or more rows of M' whose bitwise OR is equal to r. Thus, the set $R(M')$ can be partitioned into m sets $R_1, R_2, ..., R_m$ such that for all $i \in \{1, ..., m\}$, r_i is the bitwise OR of the binary vectors in R_i. For convenience, we call each row in R_i a *split row* of r_i. For example, in Figure 1(a), the set of rows of M' can be partitioned into $R_1 = \{r_1^{(1)}, r_1^{(2)}, r_1^{(3)}\}$, $R_2 = \{r_2^{(1)}\}$, $R_3 = \{r_3^{(1)}, r_3^{(2)}, r_3^{(3)}\}$, and $R_4 = \{r_4^{(1)}\}$; and $r_1^{(1)}$, $r_1^{(2)}$, and $r_1^{(3)}$ are split rows of r_1. As in [17], for ease of discussion, we make a slight abuse of notation by considering any row split M' of M as already equipped with a partition of its rows $R_1, R_2, ..., R_m$ satisfying the above condition. Note that if M' and the partition $\{R_1, R_2, ..., R_m\}$ are known, the corresponding split-row operations that transform M to M' can be easily restored.

Recall that the cost of a split-row operation is the number of additional rows. Thus, any sequence of split-row operations that transforms M into a matrix of m' rows costs $m' - m$. As a result, the computation of $\varepsilon(M)$ can be done by finding a conflict-free row split of M with the minimum number of rows. More specifically, $\varepsilon(M) = m^* - m$, where m^* is the minimum number of rows of a conflict-free row split of M. In the following, we give some properties, which are helpful in understanding the branching formulation of MSRP in [17] and our algorithms. Let M' (equipped with a partition $\{R_1, ..., R_m\}$) be a conflict-free row split of M and let T' be the perfect phylogeny corresponding to M'. Recall that we consider only phylogenies in which an edge is associated with a label if and only if it is internal. By definition, the internal edges of T' are bijectively labeled by the columns of M. Label the root of T' by γ and for each $c \in C(M)$, label the internal node of T' right below the edge c by $supp_M(c)$. (See Figure 3.) Consider a column c_j of M. Let $v_j = supp_M(c_j)$ and let $r_i \in R(M)$ be a row. If $r_i \notin v_j$, since $M_{i,j} = 0$, no row in R_i has character c_j and thus we know that the subtree rooted at v_j in T' contains no split rows of r_i. Assume that $r_i \in v_j$. Then, since $M_{i,j} = 1$, at least one of R_i has value 1 at column c_j and thus at least one row in R_i has character c_j in T'. As a result, we know that the subtree rooted at v_j in T' contains at least one leaf representing a split row of r_i. Accordingly, we have the following.

Property 1. *Let T' be the perfect phylogeny corresponding to a conflict-free row split of M. For each internal node $v = supp_M(c)$, the subtree rooted at v contains a leaf representing a split row of a row $r \in R(M)$ if and only if $r \in v$.*

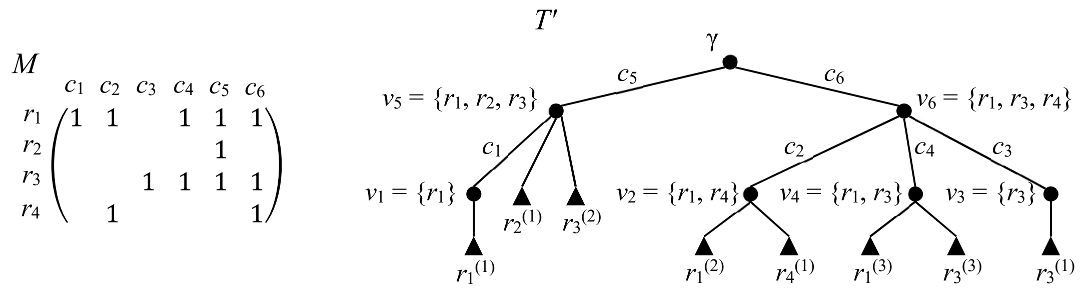


Figure 3. Labeling the internal nodes of the phylogeny in Figure 1(b), where v_i denotes $supp_M(c_i)$.

For example, in Figure 3, we have $v_6 = supp_M(c_6) = \{r_1, r_3, r_4\}$ and the leaves of the subtree rooted at v_6 are split rows of r_1, r_3, and r_4. Since the subtree rooted at a non-root node v is a proper subtree of the subtree rooted at v's parent, Property 1 immediately leads to the following.

Property 2. *Let T' be the perfect phylogeny corresponding to a conflict-free row split of M. Let $v = supp_M(c)$ and $v' = supp_M(c')$ be two internal nodes such that v' is the parent of v. Then, v' contains v.*

For example, in Figure 3, since v_6 is the parent of v_2, we have $v_6 \supset v_2$. Any conflict-free row split of M represents a perfect phylogeny with internal node set $\{\gamma\} \cup \{v \mid v = supp_M(c), c \in C(M)\}$. Therefore, the finding of an optimal row split of M can be done by examining all perfect phylogenies satisfying the following properties:

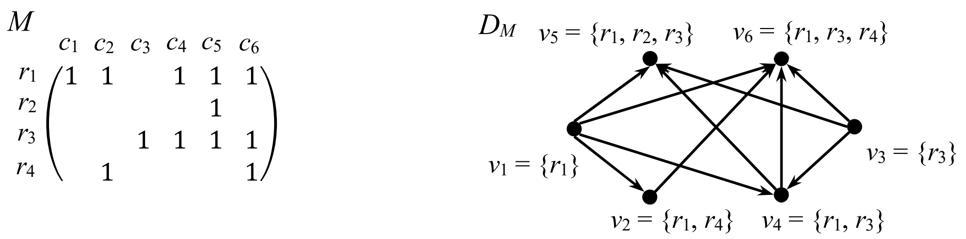
(P0) the internal node set is $\{\gamma\} \cup \{v \mid v = supp_M(c), c \in C(M)\}$;
(P1) the subtree rooted at a node $v = supp_M(c)$ contains a leaf representing a split row of a row $r \in R(M)$ if and only if $r \in v$ (Property 1); and
(P2) for each node $v = supp_M(c)$ that is not a child of the root, v is contained by its parent (Property 2).

2.2 The branching formulation

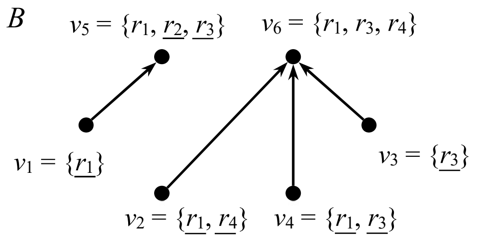
For a digraph D, we use $V(D)$ and $A(D)$, respectively, to denote its vertex set and arc set. A *directed rooted tree* is a rooted tree with its edge directed toward the root. A *directed rooted forest* is a disjoint union of directed rooted trees. The *containment digraph D_M* of M is the directed acyclic graph with vertex set $V(D_M) = \{supp_M(c) \mid c \in C(M)\}$ and arc set $A(D_M) = \{(v, v') \mid v, v' \in V(D_M), v \subset v'\}$. See Figure 4(a) for an example. A *branching* of D_M is a subset B of $A(D_M)$ such that $(V(D_M), B)$ is a digraph in which for each vertex v there is at most one arc leaving v. In other words, a branching B is a subset of arcs such that $(V(D_M), B)$ is a directed rooted forest. See Figure 4(b) for an example, in which $B = \{(v_1, v_5), (v_2, v_6), (v_3, v_6), (v_4, v_6)\}$. Note that $A(D_M)$ contains every internal edge of a phylogeny satisfying Property 2 and thus each branching of D_M represents a possible topology of the non-root internal nodes of a phylogeny satisfying (P0) and Property 2.

Consider a branching B of D_M. Let F_B denote the forest $(V(D_M), B)$. We say that a vertex p is the *B-parent* of a vertex v if p is the parent of v in F_B. The terms B-child and B-ancestor are defined similarly. Note that each vertex v is contained in any of its B-ancestors. The B-parent of a vertex v is denoted by $p_B(v)$, which may be null. Each vertex of D_M is the support of a column in $C(M)$, in which each element is a row in $R(M)$. For a row $r \in R(M)$ and a vertex $v \in V(D_M)$, we say that (r, v) is a *target pair* if $r \in v$. A target pair (r, v) indicates that for a phylogeny to satisfy Property 1, the subtree rooted at v should contain a split row of r.

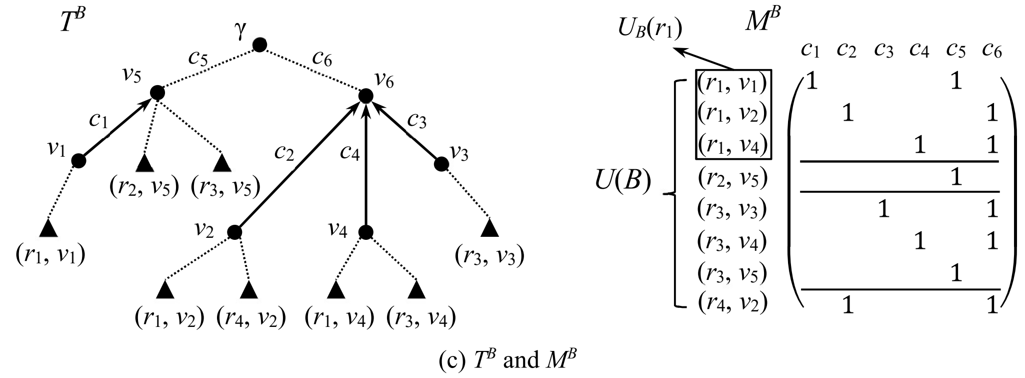
Consider a vertex $v \in V(D_M)$ and a row $r \in v$. By definition, (r, v) is a target pair of M. We say that r is *B-covered* in v if v has a B-child u such that $r \in u$; otherwise, we say that r is *B-uncovered* in v. A B-uncovered pair is a target pair (r, v) such that r is B-uncovered in v. We denote by $U(B)$ the set of all B-uncovered pairs. For each row r of M, we denote by $U_B(r)$ the set of all B-uncovered pairs (r, v), where $v \in V(D_M)$. See Figure 4(b) for an example. In this example, $U_B(r_1) = \{(r_1, v_1), (r_1, v_2), (r_1, v_4)\}$, $U_B(r_2) = \{(r_2, v_5)\}$, $U_B(r_3) = \{(r_3, v_3), (r_3, v_4), (r_3, v_5)\}$, $U_B(r_4) = \{(r_4, v_2)\}$, and $U(B)$ is the union of $U_B(r_1)$, $U_B(r_2)$, $U_B(r_3)$, and $U_B(r_4)$. The *cost* of a branching B is defined as $|U(B)|$. Clearly, to make F_B satisfy Property 1, it is sufficient and necessary to do the following: for each uncovered pair (r, v), add a split row of r as a leaf child of v.



(a) A binary matrix M and its containment digraph D_M



(b) A branching B, in which uncovered elements of each vertex are underlined



(c) T^B and M^B

Figure 4. An illustrative example.

We denote by $\beta(M)$ the minimum cost of a branching of D_M. Ademir Hujdurović *et. al.* showed that MSRP is equivalent to the following optimization problem.

Definition 2. Minimum Uncovering Branching Problem (MUBP):
Input: A binary matrix M.
Task: Find a branching B of D_M with $|U(B)| = \beta(M)$.

A row in a conflict-free row split of M is *redundant* if the matrix obtained by removing it is still a conflict-free row split of M. A conflict-free row split is *compact* if it contains no redundant rows. Clearly, to solve MSRP, only compact conflict-free row splits need to be considered. The equivalence between MSRP and MUBP is formally stated as the following theorem.

Theorem 2.1. [17] *For a binary matrix $M \in \{0, 1\}^{m \times n}$, the following holds*:
1. *Any branching B of D_M can be transformed to a conflict-free row split M', equipped with a partition $\{R_1, R_2, ..., R_m\}$, of M such that $|R_i| = |U_B(r_i)|$.*
2. *Any compact conflict-free row split M', equipped with a partition $\{R_1, R_2, ..., R_m\}$, of M can be transformed to a branching B of D_M such that $|R_i| = |U_B(r_i)|$.*

From Theorem 2.1, it is easy to conclude that $\beta(M) = m + \varepsilon(M)$. MUBP is referred as the *branching formulation* of MSRP. In the following, we describe how to transform a branching B of D_M to a conflict-free row split of M. (See Figure 4(c).) First, we make F_B a directed rooted tree by creating a pseudo node γ to be the parent of the roots in F_B. Next, for each vertex $v = supp_M(c)$, we label the edge from v to its parent by character c; and for each uncovered pair (r, v), we add a leaf (r, v) as a child of v, which represents a split row of r. After this step, F_B becomes a perfect phylogeny, denoted by T^B, with n characters and $|U(B)|$ leaves. Note that T^B is a phylogeny which satisfies the three properties (P0), (P1), and (P2). We define M^B as the matrix representation of T^B. That is, M^B is a $|U(B)| \times n$ binary matrix in which each row is associated with a leaf of T^B, each column is associated with a character of T^B, and the entry at row (r, v) and column c is 1 if and only if the leaf (r, v) has character c. Note that in the above construction, the set of split rows of a row $r_i \in R(M)$ is $R_i = U_B(r_i)$.

Since M^B corresponds to a perfect phylogeny, it is conflict-free. Recall that T^B satisfies Property 1. Consider a column c_j of M. Let $v_j = supp_M(c_j)$ and let $r_i \in R(M)$ be a row. If $r_i \notin v_j$, by Property 1, the subtree of v_j does not contain any split row of r_i and thus each $(r_i, u) \in R_i$ does not have the character c_j. On the other hand, if $r_i \in v_j$, by Property 1, the subtree of v_j contains at least one split row of r_i and thus R_i contains at least one leaf having the character c_j. Therefore, the bitwise OR of the rows $(r_i, u) \in R_i$ in M^B is equal to r_i and thus M^B is a row split of M.

3. An upper bound on *t*(*M*)

This section gives an upper bound on $t(M)$, which is the number of vertices of D_M that are not regular and not sibling-compatible. More precisely, we show that $t(M) \leq 2 \cdot \varepsilon(M) - 1$. Recall that this upper bound is used to estimate the worst-case complexity of our MSRP algorithm, which requires $O^*(2^{t(M)})$ time.

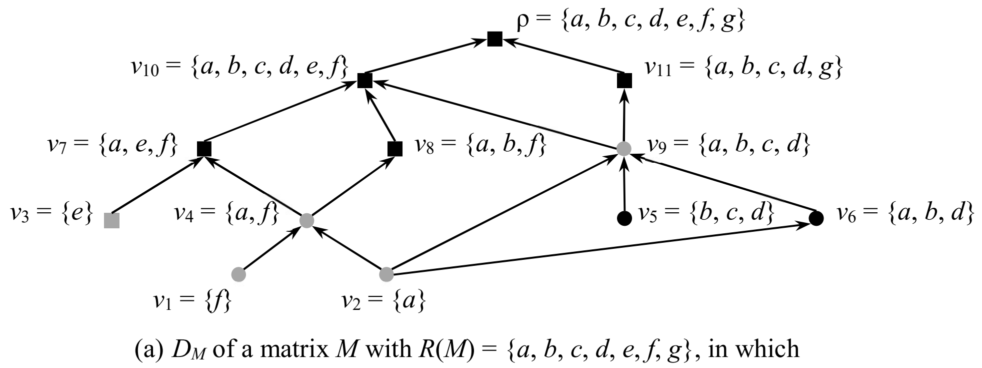
For ease of presentation, in the remainder of this paper, we assume that there is a column of M whose entries are all ones. Suppose that this is not true. We simply add a pseudo column whose entries are all ones. Denote M^+ to be the resulting matrix. Clearly, given a conflict-free row split of M^+, we can obtain a conflict-free row split of M by deleting the pseudo column; and given a conflict-free row split of M, we can obtain a conflict-free row split of M^+ by adding a column whose entries are all ones. Accordingly, it is easy to see that $\varepsilon(M) = \varepsilon(M^+)$.

Let ρ denote the support of the column whose entries are all ones. Note that $v \subset \rho$ for all vertices $v \neq \rho$ in D_M. We call ρ *the root vertex* of D_M. Consider a branching B. If F_B consists of more than one trees, we can make F_B a tree as follows: for each tree not rooted at ρ, assign ρ to be the parent of the root. Clearly, the resulting tree has uncovered pairs no more than those of B. Therefore, in the finding of an optimal branching, we may consider only branchings B such that F_B is a tree rooted at ρ. A vertex $v \in V(D_M)$ is not *regular* (or *non-regular*) if it is contained in both of a pair of conflicting vertices. A vertex v is *sibling-compatible* if there exists a vertex $p \in V(D_M)$ such that:
1. $v \subset p$; and
2. for all vertices $u \neq v$ with $u \subset p$, u and v are compatible.

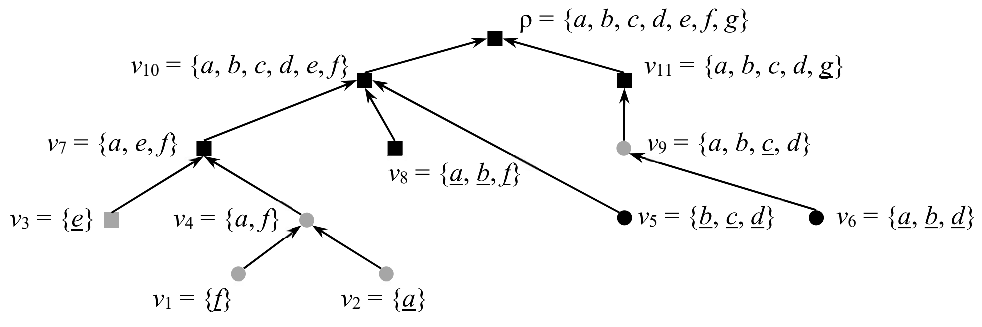
For any two vertices v and p such that the above conditions hold, we say that vertex v is *sibling-compatible at* p, indicating that if p is the parent of v in a branching, v is compatible with all its siblings.

To illustrate the above definitions, Figure 5(a) depicts the containment digraph D_M of a matrix M with $R(M) = \{a, b, c, d, e, f, g\}$. For clarity, in this figure, we display only arcs that are *elementary*, where an arc (u, v) is elementary if there exists no vertex w such that (u, w) and (w, v) are both arcs of D_M. The set of elementary arcs possesses the following property [1]: for any two vertices u, v, there is an arc (u, v) if and only if there is a path consisting of only elementary arcs from u to v. For example, according to

Figure 5(a), D_M has an arc (v_1, v_{10}), since there is a path consisting of only elementary arcs from v_1 to v_{10}. We remark that omitting non-elementary arcs is simply for clarity of illustration and a branching may contain non-elementary arcs. For example, the arc (v_5, v_{10}) in Figure 5(b) is non-elementary. In Figure 5(a), the vertex v_5 is non-regular, since v_{10} and v_{11} both contain it and are in conflict; and the vertex v_9 is sibling-compatible, since $v_9 \subset v_{11}$ and all other vertices contained in v_{11} are compatible with v_9. In particular, the root vertex ρ is regular but not sibling-compatible, since no vertex contains ρ.



(a) D_M of a matrix M with $R(M) = \{a, b, c, d, e, f, g\}$, in which only elementary arcs are depicted



(b) a branching B of D_M, in which uncovered elements of each vertex are underlined

Figure 5. An illustrative example, in which each regular vertex is represented by a square and each sibling-compatible vertex is in grey.

To derive an upper bound on $t(M)$, we focus on a special case of sibling-compatible vertices, called *trivial vertices*. A vertex v is *trivial* if $|v| = 1$, where $|v|$ denotes the size of v. For the example in Figure 5(a), v_1, v_2 and v_3 are trivial vertices. Without loss of generality, we assume that the root vertex ρ is non-trivial; otherwise M is a conflict-free matrix of single row. Since a trivial vertex is compatible with all other vertices and is contained in ρ, it must be sibling-compatible. Let $t'(M)$ be the number of vertices of D_M that are non-regular and non-trivial. Then, $t(M) \le t'(M)$. We prove $t(M) \le 2\cdot\varepsilon(M) - 1$ by proving $t'(M) \le 2\cdot\varepsilon(M) - 1$.

For a branching B, let $U_{\mathrm{NR}}(B)$ denote the set of B-uncovered pairs (r, v) such that v is non-regular. (In the rooted tree T^B, $U_{\mathrm{NR}}(B)$ is the set of leaves whose parent is non-regular.) For example, in Figure 5(b), $(a, v_2) \in U_{\mathrm{NR}}(B)$ and $(a, v_8) \notin U_{\mathrm{NR}}(B)$. Let B^* be an optimal branching. The upper bound is established in two steps. In the first step, we show that $|U_{\mathrm{NR}}(B^*)| \le 2\cdot\varepsilon(M)$; in the second step, we show that $t'(M) \le |U_{\mathrm{NR}}(B^*)| - 1$. We proceed to the first step. For a branching B, the *split set* of B, denoted by $R_{\mathrm{S}}(B)$, is the set of rows $r \in R(M)$ such that $|U_B(r)| \ge 2$. For example, in Figure 5(b), $a \in R_{\mathrm{S}}(B)$, since $|U_B(a)| = |\{(a, v_2), (a, v_6), (a, v_8)\}| \ge 2$. The following lemma says that the number of B^*-uncovered pairs contributed by the elements of $R_{\mathrm{S}}(B^*)$ is bounded by $2\cdot\varepsilon(M)$.

Lemma 3.1. *Let B^* be an optimal branching. Then,* $\sum_{r\in R_{\mathrm{S}}(B^*)} |U_{B^*}(r)| \le 2\cdot\varepsilon(M)$.
Proof. Since B^* is optimal, we have $|U(B^*)| = \sum_{r\in R(M)} |U_{B^*}(r)| = m + \varepsilon(M)$. To obtain M^{B^*}, each row $r \in R_{\mathrm{S}}(B^*)$ is replaced by $|U_{B^*}(r)|$ rows, while each row $r \notin R_{\mathrm{S}}(B^*)$ remains a single row. Thus, $\varepsilon(M) = \sum_{r\in R_{\mathrm{S}}(B^*)} (|U_{B^*}(r)| - 1)$. Since each $r \in R_{\mathrm{S}}(B^*)$ contributes at least one to $\varepsilon(M)$, we have $|R_{\mathrm{S}}(B^*)| \le \varepsilon(M)$. Therefore,

$$\sum_{r\in R_S(B^*)} |U_{B^*}(r)| = \sum_{r\in R_S(B^*)} (|U_{B^*}(r)| - 1) + \sum_{r\in R_S(B^*)} 1 = \varepsilon(M) + |R_S(B^*)| \leq \varepsilon(M) + \varepsilon(M) \leq 2\cdot\varepsilon(M).$$

Thus, the lemma holds. ■

Lemma 3.2. *If a row r is in both of two conflicting vertices, then* $|U_B(r)| \geq 2$ *for any branching B.*
Proof. Let r be a row that is in both of two conflicting vertices u and w. Consider the rooted tree T^B. By Property 1, the subtree of u contains a split row of r, say (r, u'), as a leaf; and the subtree of w also contains a split row of r, say (r, w'), as a leaf. Since u and w are in conflict, u is not a B-ancestor of w and vice versa. Thus, their subtrees are disjoint. As a result, $(r, u') \neq (r, w')$. Thus, $|U_B(r)| \geq 2$ and the lemma holds. ■

Let R_{NR} be the set of all rows which are contained in a non-regular vertex. For example, in Figure 5, we have $R_{\mathrm{NR}} = v_1 \cup v_2 \cup v_4 \cup v_5 \cup v_6 \cup v_9 = \{a, b, c, d, f\}$. The following lemma shows that $R_{\mathrm{NR}} \subseteq R_S(B)$ for any branching B.

Lemma 3.3. *Let B be a branching. For all rows* $r \in R_{\mathrm{NR}}$, *we have* $|U_B(r)| \geq 2$.
Proof. Let $r \in R_{\mathrm{NR}}$ be a row. Since r is in a non-regular vertex, it is in both of a pair of conflicting vertices. By Lemma 3.2, $|U_B(r)| \geq 2$ and hence the lemma holds. ■

Recall that $U_{\mathrm{NR}}(B)$ denotes the set $\{(r, v) \in U(B) \mid v \text{ is non-regular}\}$. Lemmas 3.1 and 3.3 lead to the following.

Lemma 3.4. *For any optimal branching* B^*, *we have* $|U_{\mathrm{NR}}(B^*)| \leq 2\cdot\varepsilon(M)$.
Proof. Consider an optimal branching B^*. Each (r, v) in $U_{\mathrm{NR}}(B^*)$ has v being non-regular and r being an element of v. Thus, each (r, v) in $U_{\mathrm{NR}}(B^*)$ has r being an element of a non-regular vertex. As a result, we can equivalently define $U_{\mathrm{NR}}(B^*)$ as the set of uncovered pairs (r, v) such that $r \in R_{\mathrm{NR}}$ and v is non-regular. Let U' be the set of uncovered pairs (r, v) with $r \in R_{\mathrm{NR}}$. By definition, we have $U_{\mathrm{NR}}(B^*) \subseteq U'$. Let U'' be the set of uncovered pairs (r, v) with $r \in R_S(B^*)$. By Lemma 3.3, $R_{\mathrm{NR}} \subseteq R_S(B^*)$ and thus we have $U' \subseteq U''$. By Lemma 3.1, $|U''| = \sum_{r\in R_S(B^*)} |U_{B^*}(r)| \leq 2\cdot\varepsilon(M)$. Consequently, we obtain $|U_{\mathrm{NR}}(B^*)| \leq |U'| \leq |U''| \leq 2\cdot\varepsilon(M)$. Thus, the lemma holds. ■

We proceed to the second step. First, we show a property of regular vertices.

Lemma 3.5. *If a vertex* $v \in V(D_M)$ *is regular, all vertices containing v are regular.*
Proof. Let $v \in V(D_M)$ be regular vertex. Consider a fixed vertex $u \supset v$. Suppose, for proof by contradiction, that u is non-regular. By definition, u is contained in both of a pair of conflicting vertices. Since $u \supset v$, these two vertices also both contain v, which contradicts the fact that v is regular. Thus, the lemma holds.
■

A consequence of Lemma 3.5 is as follows. Let V_{NR} be the set of non-regular vertices and V_{R} be the set of regular vertices. Then, there is no arc of D_M which starts from a vertex of V_{R} and ends at a vertex of V_{NR}. Consider the directed rooted forest $F_B = (V(D_M), B)$ of a branching B. We partition the non-regular vertices in F_B into the following subsets:

$L_1(B)$: the set of non-regular vertices v with in-degree = 0 and $|v| = 1$;
$L_2(B)$: the set of non-regular vertices v with in-degree = 0 and $|v| > 1$;
$N_1(B)$: the set of non-regular vertices v with in-degree = 1; and
$N_2(B)$: the set of non-regular vertices v with in-degree > 1.

For example, in Figure 5(b), we have $L_1(B) = \{v_1, v_2\}$, $L_2(B) = \{v_5, v_6\}$, $N_1(B) = \{v_9\}$ and $N_2(B) = \{v_4\}$. Vertices in $L_1(B) \cup L_2(B)$ are *non-regular leaves* of F_B and vertices in $N_1(B) \cup N_2(B)$ are *non-regular internal nodes* of F_B. The following lemma shows that $|N_2(B)|$, the number of non-regular internal nodes with in-degree more than one, is bounded by the number of non-regular leaves.

Lemma 3.6. *For any branching B, we have $|N_2(B)| \le |L_1(B)| + |L_2(B)| - 1$.*
Proof. Let X be the subgraph of F_B induced by V_{NR}, which contains all the non-regular vertices and the arcs connecting pairs of them. By Lemma 3.5, all children of a non-regular vertex are non-regular. As a result, for each vertex in V_{NR}, its in-degree in X is the same as in F_B. Thus, X is a directed rooted forest with leaf set $L_1(B) \cup L_2(B)$ and with internal node set $N_1(B) \cup N_2(B)$. It is known that a rooted tree (or forest) of k leaves contains at most $k - 1$ internal nodes that have more than one child, where $k \ge 1$ is an integer [5]. Therefore, there are at most $|L_1(B) \cup L_2(B)| - 1$ internal nodes with in-degree > 1 in X. That is, $|N_2(B)| \le |L_1(B) \cup L_2(B)| - 1 = |L_1(B)| + |L_2(B)| - 1$. Thus, the lemma holds. ■

Lemma 3.7. *For any branching B, we have $|U_{\text{NR}}(B)| \ge |N_1(B)| + |L_1(B)| + 2\cdot|L_2(B)|$.*
Proof. For any $v \in L_1(B) \cup L_2(B)$, since v has no B-child, all elements of v are B-uncovered. Therefore, there are at least $|L_1(B)| + 2\cdot|L_2(B)|$ B-uncovered pairs (r, v) with $v \in L_1(B) \cup L_2(B)$. Consider a non-regular vertex $v \in N_1(B)$. By definition, v has exactly one B-child, say v'. Since v and v' are distinct, at least one element of v is B-uncovered. Therefore, there are at least $|N_1(B)|$ B-uncovered pairs (r, v) with $v \in N_1(B)$. Consequently, we have $|U_{\text{NR}}(B)| \ge |N_1(B)| + |L_1(B)| + 2\cdot|L_2(B)|$ and thus the lemma holds. ■

We are ready to prove that $t'(M) \le |U_{\text{NR}}(B^*)| - 1$.

Lemma 3.8. *For any optimal branching B^*, we have $t'(M) \le |U_{\text{NR}}(B^*)| - 1$.*
Proof. Let B^* be an optimal branching. Since all vertices in $L_1(B^*)$ are trivial, we have

$$\begin{aligned} t'(M) &= |N_1(B^*)| + |N_2(B^*)| + |L_2(B^*)| && \text{(by the definition of } t'(M)\text{)} \\ &\le |N_1(B^*)| + |L_1(B^*)| + 2\cdot|L_2(B^*)| - 1 && (|N_2(B^*)| \le |L_1(B^*)| + |L_2(B^*)| - 1 \text{ by Lemma 3.6)} \\ &\le |U_{\text{NR}}(B^*)| - 1. && \text{(by Lemma 3.7)} \end{aligned}$$

■

Lemmas 3.4 and 3.8 immediately lead to the following.

Theorem 3.9. $t'(M) \le 2\cdot\varepsilon(M) - 1$.

Recall that $t(M) \le t'(M)$. The following is a consequence of Theorem 3.9.

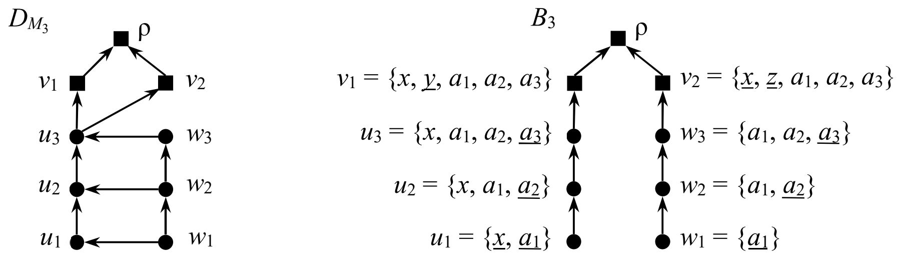
Corollary 3.10. $t(M) \le 2\cdot\varepsilon(M) - 1$.

The following gives an upper bound on the number of non-regular vertices, which is needed for analyzing the time complexity of our MDCRSP algorithm.

Theorem 3.11. $|V_{\text{NR}}| \le 3\cdot\varepsilon(M) - 1$.
Proof. Let W be the set of non-regular trivial vertices and let $R_W = \cup_{v\in W}\, v$. Vertices in W are trivial and distinct. Thus, we have $|R_W| = |W|$. Since $|V_{\text{NR}}| = t'(M) + |W| = t'(M) + |R_W|$ and $t'(M) \le 2\cdot\varepsilon(M) - 1$, it suffices to show that $|R_W| \le \varepsilon(M)$. Consider an optimal branching B^*. Since $R_W \subseteq R_{\text{NR}}$, by Lemma 3.3, $|U_{B^*}(r)| \ge 2$ for each $r \in R_W$. Thus, each $r \in R_W$ contributes at least one to $\varepsilon(M)$. As a result, we have $|R_W| \le \varepsilon(M)$ and thus the theorem holds. ■

To conclude this section, we remark that the upper bounds in Theorem 3.9 and Theorem 3.11 are almost tight. We first show that for any integer $h \ge 1$, a matrix M_h with $\varepsilon(M_h) = h + 1$ and $t'(M_h) = 2\cdot\varepsilon(M_h) - 3$ can be constructed. Consider a fixed $h \ge 1$. Let M_h be an $(h + 3) \times (2\cdot h + 3)$ binary matrix such that the rows are labeled by $x, y, z, a_1, a_2, a_3, ..., a_h$ and the supports of the columns are $\rho, v_1, v_2, u_1, u_2, u_3, ... u_h, w_1, w_2, w_3, ..., w_h$, where $\rho = R(M_h)$, $v_1 = R(M_h) - \{z\}$, $v_2 = R(M_h) - \{y\}$, $u_i = \{x, a_1, a_2, ..., a_i\}$, and $w_i = \{a_1, a_2, ..., a_i\}$ for $i = 1, 2, ..., h$. Figure 6 depicts D_{M_h} for $h = 3$. As in Figure 5, uncovered elements are underlined and regular vertices are represented by squares. Note that v_1 and v_2 are in conflict. The vertices ρ, v_1, v_2 are regular. Each vertex in $V(D_M) - \{\rho, v_1, v_2\}$ is contained in both of v_1 and v_2 and thus is non-regular. Only the vertex w_1 is trivial. Thus, we have $t'(M) = 2\cdot h - 1$. Let B_h be the branching such that $p_{B_h}(v_1) = p_{B_h}(v_2) = \rho$, $p_{B_h}(u_h) = v_1$, $p_{B_h}(w_h) = v_2$, and for $i = 1, 2, ..., h - 1$, $p_{B_h}(w_i) = w_{i+1}$ and $p_{B_h}(u_i) = u_{i+1}$.

(See Figure 6 for B_3.) In B_h, each element in $R(M_h) - \{y, z\}$ contributes two uncovered pairs and each of y and z contributes one uncovered pair. Thus, $|U(B_h)| = 2\cdot h + 4$. In the following, we show that B_h is optimal by showing that any branching has at least $2\cdot h + 4$ uncovered pairs. Consider an arbitrary branching B. Since all elements in $R(M_h) - \{y, z\}$ are contained in the non-regular vertex u_h, by Lemma 3.3, each of them contributes at least two B-uncovered pairs. Moreover, each of y and z contributes at least 1 B-uncovered pair. Thus, $|U(B)|$ is at least $2\cdot(|R(M_h)| - 2) + 2 = 2\cdot h + 4$. Therefore, B_h is optimal and we have $\varepsilon(M_h) = |U(B_h)| - |R(M_h)| = h + 1$ and $t'(M_h) = 2\cdot h - 1 = 2\cdot\varepsilon(M_h) - 3$.



D_{M_3}
ρ
v_1
v_2
u_3
w_3
u_2
w_2
u_1
w_1
B_3
ρ
$v_1 = \{x, \underline{y}, a_1, a_2, a_3\}$
$v_2 = \{\underline{x}, \underline{z}, a_1, a_2, a_3\}$
$u_3 = \{x, a_1, a_2, \underline{a_3}\}$
$w_3 = \{a_1, a_2, \underline{a_3}\}$
$u_2 = \{x, a_1, \underline{a_2}\}$
$w_2 = \{a_1, \underline{a_2}\}$
$u_1 = \{\underline{x}, \underline{a_1}\}$
$w_1 = \{\underline{a_1}\}$

Figure 6. D_{M_3} (only elementary arcs are depicted) and B_3.

In the following, we show that for any integer $h \geq 1$, a matrix N_h with $\varepsilon(N_h) = h + 1$ and $|V_{\mathrm{NR}}| = 3\cdot\varepsilon(N_h) - 3$ can be constructed. Consider a fixed $h \geq 1$. Let N_h be the matrix obtained from M_h by adding h columns whose supports are $t_1, t_2, ..., t_h$, where $t_1 = \{x\}$ and $t_i = \{a_i\}$ for $i = 2, 3, ..., h$. Since each t_i is contained in both v_1 and v_2, it is non-regular. Thus, for the digraph D_{N_h}, we have $|V_{\mathrm{NR}}| = |\{u_1, u_2, ..., u_h, w_1, w_2, ..., w_h, t_1, t_2, ..., t_h\}| = 3h$. Let $B'_h = B_h \cup \{(t_1, v_2)\} \cup \{(t_i, w_i) \mid 2 \leq i \leq h\}$ be a branching of D_{N_h}. Clearly, $|U(B'_h)| = |U(B_h)| = 2h + 4$. Using the same arguments for proving the optimality of B_h, we know that B'_h is optimal for D_{N_h}. Thus, $\varepsilon(N_h) = |U(B'_h)| - |R(N_h)| = h + 1$. As a result, $|V_{\mathrm{NR}}| = 3h = 3\varepsilon(N_h) - 3$.

4. An $O^*(2^{\min(n,\, 2\cdot\varepsilon(M))})$-time algorithm

This section presents an $O^*(2^{\min(n,\, 2\cdot\varepsilon(M))})$-time algorithm for MSRP. Recall that our algorithm consists of three phases: Phase 1 removes sibling-compatible vertices; Phase 2 predetermines the parent of each regular vertex; and Phase 3 finds an optimal branching using dynamic programming. The containment digraph D_M is precomputed, which requires $O(mn^2)$ time [17].

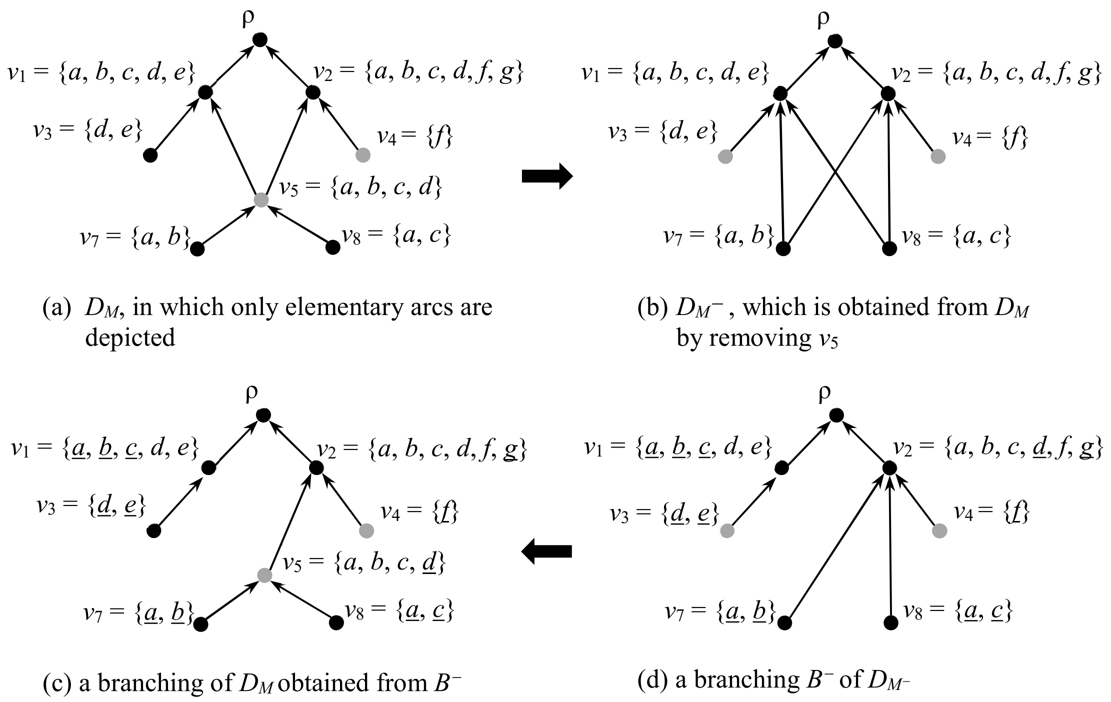
4.1 Phase 1

Let $v_i = supp_M(c_i)$ for $1 \leq i \leq n$. For ease of discussion, a vertex v_j is called a *sub-support* of a vertex v_i if $v_j \subset v_i$ and is called a *super-support* of a vertex v_i if $v_j \supset v_i$. Then, equivalently, we may define sibling-compatible vertices as follows: a vertex v_i is sibling-compatible if it has a super-support v_j whose sub-supports, other than v_i, are all compatible with v_i. If a vertex v_i is sibling-compatible at a vertex v_j, we simply say that v_i prefers v_j. We have the following.

Lemma 4.1. *Let v_i and v_j be two vertices such that v_i prefers v_j. If a vertex u satisfies $v_i \subset u \subset v_j$, then v_i prefers u.*

Proof. Since v_i prefers v_j, v_i is compatible with all the other sub-supports of v_j. Since $u \subset v_j$, each sub-support of u is also a sub-support of v_j. Therefore, v_i is compatible with all the other sub-supports of u. Thus, v_i prefers u and the lemma holds. ■

We proceed to show that sibling-compatible vertices can be removed without changing $\varepsilon(M)$. A *sibling-compatible reduction* on M removes a column c_i such that v_i is sibling-compatible. This operation corresponds to removing v_i and all its incident arcs from D_M. (See Figure 7(a)(b) for an example.) Let M^- be the binary matrix obtained from M by a sibling-compatible reduction. Since M^- is a submatrix of M, each conflict-free row split of M induces a conflict-free row split of M^-. Thus, $\varepsilon(M^-) \leq \varepsilon(M)$. In the following, we prove that $\varepsilon(M^-) = \varepsilon(M)$ by showing that $\varepsilon(M^-) \geq \varepsilon(M)$. For a sibling-compatible vertex v_i, its *preferred-parent* is defined to be the vertex v_j such that v_i prefers v_j and $|v_j|$ is minimum (with ties broken arbitrarily). For example, in Figure 7(a), v_5 prefers only v_2 and the preferred-parent of v_5 is v_2; and, v_4 prefers both v_2 and ρ and the preferred-parent of v_4 is v_2.


ρ
$v_1 = \{a, b, c, d, e\}$
$v_2 = \{a, b, c, d, f, g\}$
$v_3 = \{d, e\}$
$v_4 = \{f\}$
$v_5 = \{a, b, c, d\}$
$v_7 = \{a, b\}$
$v_8 = \{a, c\}$
ρ
$v_1 = \{a, b, c, d, e\}$
$v_2 = \{a, b, c, d, f, g\}$
$v_3 = \{d, e\}$
$v_4 = \{f\}$
$v_7 = \{a, b\}$
$v_8 = \{a, c\}$
ρ
$v_1 = \{\underline{a}, \underline{b}, \underline{c}, d, e\}$
$v_2 = \{a, b, c, d, f, g\}$
$v_3 = \{\underline{d}, \underline{e}\}$
$v_4 = \{\underline{f}\}$
$v_5 = \{a, b, c, \underline{d}\}$
$v_7 = \{\underline{a}, \underline{b}\}$
$v_8 = \{\underline{a}, \underline{c}\}$
ρ
$v_1 = \{\underline{a}, \underline{b}, \underline{c}, d, e\}$
$v_2 = \{a, b, c, \underline{d}, f, g\}$
$v_3 = \{\underline{d}, \underline{e}\}$
$v_4 = \{\underline{f}\}$
$v_7 = \{\underline{a}, \underline{b}\}$
$v_8 = \{\underline{a}, \underline{c}\}$

(a) D_M, in which only elementary arcs are depicted

(b) D_{M^-}, which is obtained from D_M by removing v_5

(c) a branching of D_M obtained from B^-

(d) a branching B^- of D_{M^-}

Figure 7. An illustrative example, in which sibling-compatible vertices are in grey.

Lemma 4.2. *Let v_i be a sibling-compatible vertex and M^- be the matrix obtained from M by removing the column c_i. Then, we have $\varepsilon(M^-) \geq \varepsilon(M)$.*

Proof. We prove this lemma by showing that for any branching B^- of D_{M^-}, we can find a branching B of D_M with $|U(B^-)| = |U(B)|$. Consider a fixed branching B^- of D_{M^-}. Let v_j be the preferred-parent of v_i. We obtain from B^- a branching B of D_M as follows:

1. add the vertex v_i and assign v_j to be the parent of v_i; and
2. for each B^--child u of v_j such that $u \subset v_i$, reassign v_i to be its parent.

Figure 7(c)(d) give an illustration. For each vertex $v_a \in V(D_M)$, let $U_B{}^a$ be the set of B-uncovered elements in v_a. Define $U_{B^-}{}^a$ similarly. For each vertex $v_a \notin \{v_i, v_j\}$, since its child sets in B^- and B are the same, we have $|U_B{}^a| = |U_{B^-}{}^a|$. Since $|U(B)| = \sum_{1\leq a\leq n} |U_B{}^a|$ and $|U(B^-)| = \sum_{1\leq a\leq n,\, a\neq i} |U_{B^-}{}^a|$, in the following, we prove $|U(B)| = |U(B^-)|$ by showing that $|U_B{}^i| + |U_B{}^j| = |U_{B^-}{}^j|$.

Let H be the set of B^--children of v_j. We partition the set H into three subsets: $H_1 = \{c \mid c \in H, c \cap v_i = \varnothing\}$, $H_2 = \{c \mid c \in H, c \subset v_i\}$, and $H_3 = \{c \mid c \in H, c \supset v_i\}$. We first show that $H_3 = \varnothing$. Suppose, by contradiction, that H_3 contains a vertex c. Then, by Lemma 4.1, v_i prefers c. That is, c has size smaller than $|v_j|$ and is preferred by v_i, which contradicts the definition of v_j. Thus, we have $H_3 = \varnothing$. Consequently, we have $|U_{B^-}{}^j| = |v_j| - |\bigcup_{c\in H} c| = |v_j| - |\bigcup_{c\in H_1\cup H_2} c|$. Furthermore, since every vertex in H_1 is disjoint with all vertices in H_2, we obtain $|U_{B^-}{}^j| = |v_j| - |\bigcup_{c\in H_1} c| - |\bigcup_{c\in H_2} c|$.

We proceed to determine the sizes of $U_B{}^i$ and $U_B{}^j$. The child set of v_j in B is $\{v_i\} \cup H_1$. Since v_i is disjoint with all vertices in H_1, we have $|U_B{}^j| = |v_j| - |\bigcup_{c\in H_1} c| - |v_i|$. The child set of v_i in B is H_2 and thus we have $|U_B{}^i| = |v_i| - |\bigcup_{c\in H_2} c|$. As a result, we have $|U_B{}^i| + |U_B{}^j| = |v_j| - |\bigcup_{c\in H_1} c| - |\bigcup_{c\in H_2} c| = |U_{B^-}{}^j|$, which completes the proof of this lemma. ■

Recall that $\varepsilon(M^-) \leq \varepsilon(M)$. By combining this and Lemma 4.2, we obtain the following.

Theorem 4.3. $\varepsilon(M^-) = \varepsilon(M)$.

We remark that the proof of Lemma 4.2 also exhibits that given the preferred-parent of v_i, we can obtain in $O(n)$ time an optimal branching of D_M from an optimal branching of D_{M^-}. The digraph D_M may contain more than one sibling-compatible vertex. In the following, we show that after removing a sibling-compatible vertex, all the other sibling-compatible vertices are still sibling-compatible. We need the following.

Lemma 4.4. *If v_i prefers v_j and v_j prefers v_k, then v_i prefers v_k.*
Proof. Let u be a sub-support of v_k such that $u \neq v_i$ and $u \neq v_j$. It suffices to show that u is compatible with v_i. Since v_j prefers v_k, u is compatible with v_j. If $v_j \cap u = \varnothing$, we have $v_i \cap u = \varnothing$; if $v_j \subset u$, we have $v_i \subset u$; and if $v_j \supset u$, since v_i prefers v_j, u is compatible with v_i. As a result, we know that v_i prefers v_k and the lemma holds. ■

Lemma 4.5. *Let* v_i and v_j *be two sibling-compatible vertices of D_M. Let M^- be the matrix obtained from M by removing c_i. Then, v_j is a sibling-compatible vertex of D_{M^-}.*
Proof. Note that the removal of v_i does not make any two compatible vertices not compatible. By definition, D_M contains a vertex v_a preferred by v_i and contains a vertex v_b preferred by v_j. Clearly, if $v_b \neq v_i$, after the removal of v_i, v_j is still sibling-compatible at v_b. Assume that $v_b = v_i$. Then, by Lemma 4.4, v_j also prefers v_a before the removal of v_i. After the removal of v_i, v_j is still sibling-compatible at v_a. Thus, the lemma holds. ■

By Lemma 4.5, removing a sibling-compatible vertex does not invalidate any other sibling-compatible vertex. Furthermore, the removing may produce new sibling-compatible vertices. For example, in Figure 7(a)(b), v_3 becomes sibling-compatible at v_1 after the removal of v_5.

Our Phase 1 algorithm repeatedly removes sibling-compatible vertices from D_M until no more such vertices can be found. Denote the matrix corresponding to the resulting digraph as M_S. An array Q and two integers, *size* and *next*, are used to maintain the vertices to be removed as well as the removed vertices. During the algorithm, $Q[1]$, $Q[2]$, ..., $Q[next - 1]$ are removed vertices and $Q[next]$, $Q[next + 1]$, ..., $Q[size]$ are vertices to be removed. Initially, Q contains all sibling-compatible vertices of D_M, *size* is set to be the number of sibling-compatible vertices of D_M, and *next* is set to be 1, indicating that $Q[1]$ is the next vertex to be removed. In each step, we remove $Q[next]$ from D_M, increase *next* by 1, and find and add new sibling-compatible vertices into Q. In order to do the finding of new sibling-compatible vertices efficiently, we maintain a table P. Initially, $P[i, j]$ stores the number of sub-supports of v_j which is in conflict with v_i, where $1 \leq i, j \leq n$ and $v_i \subset v_j$. Note that a vertex v_i prefers a vertex v_j if and only if $v_i \subset v_j$ and $P[i, j] = 0$. When a vertex v_i is removed, for all pairs (v_j, v_k) such that neither of v_j and v_k is removed, $v_j \subset v_k$, $v_i \subset v_k$, and v_i and v_j are in conflict, we do the following: decrease the entry $P[j, k]$ by 1 and if $P[j, k]$ becomes 0, add v_j into Q (if v_j is not in Q). Phase 1 is formally described as follows.

```
Procedure PHASE1
input: $D_M$
output: $D_{M_S}$
begin
1.  for each pair of vertices $(v_i, v_j)$ do determine whether $v_i$ and $v_j$ are in conflict
2.  $Q \leftarrow$ an array of size $n$; $size \leftarrow 0$; $next \leftarrow 1$
3.  for each pair of vertices $(v_i, v_j)$ with $v_i \subset v_j$ do   /* compute $P[i, j]$ and check if $v_i$ prefers $v_j$
4.  begin
5.      $P[i, j] \leftarrow 0$
6.      for each vertex $v_k$ do
7.          if ($v_k \subset v_j$ and $v_i$ is in conflict with $v_k$) then $P[i, j] \leftarrow P[i, j] + 1$;
8.      if ($P[i, j] = 0$ and $v_i$ is not in $Q$) then begin $size \leftarrow size + 1$; $Q[size] = v_i$ end
9.  end
10. while ($next \leq size$)
11. begin
12.     $v_i \leftarrow Q[next]$; compute the preferred-parent of $v_i$ (with respect to the current $D_M$)
13.     remove $v_i$ from $D_M$; $next \leftarrow next + 1$
14.     for each pair $(v_j, v_k)$ of vertices do /*  update $P$
15.         if (neither of $v_j$ and $v_k$ is removed, $v_j \subset v_k$, $v_i \subset v_k$, and $v_i$ and $v_j$ are in conflict)
```

```
16.     begin
17.         $P[j, k] \leftarrow P[j, k] - 1$;
18.         if ($P[j, k] = 0$ and $v_j$ is not in $Q$) then begin $size \leftarrow size + 1$; $Q[size] = v_j$ end
19      end
20  end
21. return $D_M$
end
```

Since each vertex contains at most m elements, the determination in line 1 requires $O(mn^2)$ time. After line 1, the conditions in lines 7 and 15 can be checked in $O(1)$ time. Note that whether a vertex v is contained in a vertex u can be determined by examining whether there is an arc (v, u) in D_M. Clearly, the for-loop in lines 3-9 takes $O(n^3)$ time. Consider a fixed iteration of the while-loop in lines 10-20. Line 12 finds the preferred-parent of v_i, so that an optimal branching of the original matrix can be obtained from an optimal branching of M_S by repeatedly using the method given in the proof of Lemma 4.2. Here, the preferred-parent of v_i refers to its preferred-parent with respect to the current D_M, in which $Q[1]$, $Q[2]$, ..., $Q[next - 1]$ are removed. Line 12 is done in $O(n)$ time by checking if $P[i, j] = 0$ for all v_j that contain v_i and are not removed. Line 13 takes $O(n)$ time. The for-loop in lines 14-19 takes $O(n^2)$ time. Since each iteration of the while-loop removes one vertex, there are at most n iterations. Therefore, the while-loop in lines 10-20 requires $O(n^3)$ time in total. As a result, the running time of Phase 1 is $O(mn^2 + n^3)$.

4.2 Phase 2

Phase 2 predetermines the parent of each regular vertex. Recall that a vertex of D_M is regular if it is not contained in both of a pair of conflicting vertices.

Lemma 4.6. *Let $(u_1, u_2, ..., u_{s-1}, u_s = \rho)$ be the sequence of super-supports of a vertex v_i, in non-decreasing order of their sizes. Then, v_i is regular if and only if $u_1 \subset u_2 \subset \ ... \subset u_s$.*
Proof. If $u_1 \subset u_2 \subset \ ... \subset u_s$, any pair of super-supports of v_i are not in conflict and thus v_i is regular. Assume that v_i is regular. Consider two supports u_j and u_{j+1}. Since they both contain v_i, they are not disjoint. In addition, since v_i is regular, they are not in conflict. As a result, they are nested. Therefore, the lemma holds. ■

Lemma 4.6 indicates that no two super-supports of a regular vertex v_i have the same size. Recall that V_R is the set of regular vertices. For each vertex $v_i \in V_R$, the *predetermined parent* of v_i, denoted by $\pi(v_i)$, is the smallest super-support of v_i. Note that $\pi(v_i)$ is null if and only if $v_i = \rho$. Let B_R be the set of arcs $\{(v_i, \pi(v_i)) \mid v_i \in V_R \text{ and } v_i \neq \rho\}$. For the example in Figure 5, $B_R = \{(v_7, v_{10}), (v_8, v_{10}), (v_{10}, \rho), (v_{11}, \rho)\}$. Let Q_R be the digraph with vertex set V_R and arc set B_R. By Lemma 3.5, all supper-supports of a regular vertex are regular. Thus, each arc in B_R connects a pair of two regular vertices, from which we conclude that Q_R is a directed tree rooted at ρ. Let $(u_1, u_2, ..., u_{s-1}, u_s = \rho)$ be the sequence of super-supports of a regular vertex v_i, in non-decreasing order of their sizes. By definition, $\pi(v_i) = u_1$. Consider a fixed vertex u_j, $1 \leq j < s$. Its supper-supports are also supper-supports of v_i. Therefore, u_{j+1} is the smallest supper-support of u_j and thus we have $\pi(u_j) = u_{j+1}$. Consequently, we obtain the following.

Lemma 4.7. *Let v_i be a regular vertex. The path from v_i's parent to the root ρ in Q_R contains the sequence of super-supports of v_i, in non-decreasing order of their sizes.*

Lemma 4.7 indicates that a vertex v_j is a supper-support of a regular vertex v_i if and only of v_j is on the path from v_i's parent to the root in Q_R. Clearly, B_R is a branching of D_M. We proceed to show that in the finding of an optimal branching, we can consider only branchings that contain B_R as a subset.

Consider a branching B which does not contain B_R as a subset. Let v_r be the largest regular vertex such that $p_B(v_r) \neq \pi(v_r)$ (with ties broken arbitrarily). Since $p_B(\rho) = \pi(\rho) =$ null, $v_r \neq \rho$ and thus $\pi(v_r) \neq$ null. A *regularization operation* on B is defined as re-assigning $p_B(v_r)$ to be $\pi(v_r)$.

Lemma 4.8. *Let B' be the branching obtained from B by performing a regularization operation. Then, $U(B) \supseteq U(B')$.*
Proof. Let v_r be the regular vertex whose parent is re-assigned by the regularization operation. Two cases are considered.

Case 1: $p_B(v_r) \neq$ null (See Figure 8). Let $W = (u_1, u_2, ..., u_{s-1}, u_s = \rho)$ be the sequence of super-supports of

v_r, in non-decreasing order of their sizes. By definition, $\pi(v_r) = u_1$. Furthermore, by the definition of v_r and Lemma 4.7, we have $p_B(u_j) = \pi(u_j) = u_{j+1}$ for $1 \le j < s$. Since $p_B(v_r)$ is a super-support of v_r and W contains all super-supports of v_r, we know that W contains a vertex $u_j = p_B(v_r)$, where $1 < j \le s$. Since u_{j-1} is a B-child of u_j and $u_{j-1} \supset v_r$, deleting the arc (v_r, u_j) from B does not induce any uncovered element in v_j. Consequently, it is easy to conclude that $U(B) \supseteq U(B')$.

Case 2: $p_B(v_r) = \text{null}$. In this case, we have $B \subseteq B'$ and thus $U(B) \supseteq U(B')$.

Therefore, the lemma holds. ■

Let B^* be an optimal branching not containing B_R. By repeatedly performing a regularization operation, B^* can be transformed into a branching which contains all arcs in B_R. By Lemma 4.8, this transformation does not increase the number of uncovered pairs. Therefore, we have the following.

Theorem 4.9. *There exists an optimal branching B^* of D_M such that $B^* \supseteq B_R$.*

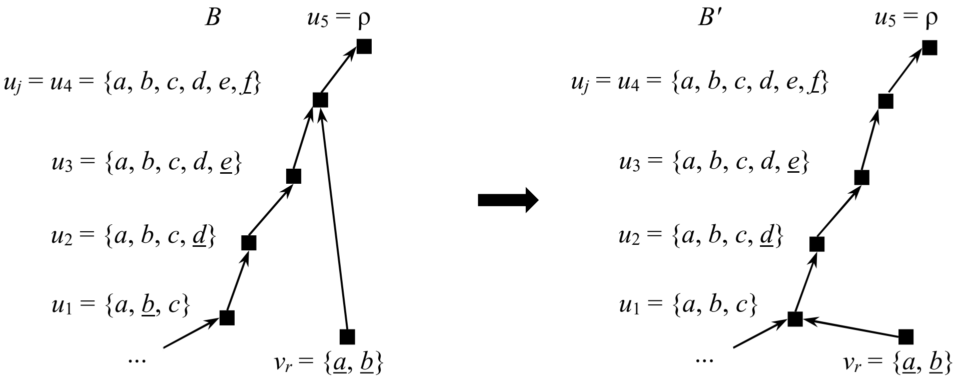

B
$u_5 = \rho$
$u_j = u_4 = \{a, b, c, d, e, \underline{f}\}$
$u_3 = \{a, b, c, d, \underline{e}\}$
$u_2 = \{a, b, c, \underline{d}\}$
$u_1 = \{a, \underline{b}, c\}$
...
$v_r = \{\underline{a}, \underline{b}\}$
B'
$u_5 = \rho$
$u_j = u_4 = \{a, b, c, d, e, \underline{f}\}$
$u_3 = \{a, b, c, d, \underline{e}\}$
$u_2 = \{a, b, c, \underline{d}\}$
$u_1 = \{a, b, c\}$
...
$v_r = \{\underline{a}, \underline{b}\}$

Figure 8. An illustration for Case 1 in the proof of Lemma 4.8, in which $s = 5$ and $j = 4$.

Our Phase 2 algorithm finds all regular vertices and their predetermined parents as follows. First, we create a sequence L containing all vertices of D_M, in non-decreasing order of their sizes. Next, for each vertex v_i, we determine whether it is regular by extracting from L the sequence $(u_1, u_2, ..., u_{s-1}, u_s = \rho)$ of super-supports of v_i and then check whether there is an arc (u_j, u_{j+1}) for $1 \le j < s$; and if v_i is regular, we set $\pi(v_i) = u_1$. The correctness is ensured by Lemma 4.6. The computation of L requires $O(n \lg n)$ time. The determination for each vertex takes $O(n)$ time. Thus, Phase 2 requires $O(n^2)$ time.

4.3 Phase 3

Phase 3 finds an optimal branching using dynamic programming. The finding is done in time $O(mn\, 2^{\min(n, 2\varepsilon(M))} + n^2\, 3^{\min(n, 2\varepsilon(M))})$ or $O(mn^5 \lg mn \lg\lg mn\, 2^{\min(n, 2\varepsilon(M))})$, depending on the implementation. Recall that V_R and V_{NR} are, respectively, the sets of regular and non-regular vertices. We first topologically sort the vertices of D_M into a sequence $(v_1, v_2, ..., v_n)$, so that an arc $(v_i, v_j) \in A(D_M)$ only if $i < j$. For $1 \le i \le n$, let $V_i = \{v_j \mid j \le i\}$ and $D_M(V_i)$ be the subgraph induced by V_i. Consider a fixed V_i. A *branching on* V_i is a branching of $D_M(V_i)$, in which the parent of each vertex $v \in V_i$ is either null or a vertex in V_i. The *cost* of a branching B on V_i, denoted by $cost_i(B)$, is the number of uncovered pairs (r, v_j) with $v_j \in V_i$. That is, $cost_i(B) = \Sigma_{1 \le j \le i} |U_B^j|$. Recall that U_B^j is the set of B-uncovered elements in v_j.

After Phase 2, the parent, $\pi(v)$, of each regular vertex v has been predetermined. Thus, our problem is to optimally assign the parent of each non-regular vertex. Let B be a branching on V_i. We say that B *obeys* B_R if the following holds: for each regular vertex $v_j \in V_i$, if $\pi(v_j) \in V_i$, then $p_B(v_j) = \pi(v_j)$; otherwise, $p_B(v_j) = \text{null}$. We say that a non-regular vertex v_j is *assigned* in B if $p_B(v_j) \ne \text{null}$.

Consider a fixed integer $i \in \{1, 2, ..., n\}$. Let S be a subset of V_{NR}. An (i, S)-*branching* is a branching B such that:

(1) B is a branching on V_i;

(2) B obeys B_{R}; and
(3) the set of non-regular vertices assigned in B is S.

Figure 9 gives two examples. Define $f[i, S]$ to be the minimum cost of an (i, S)-branching and if such a branching does not exist, we define $f[i, S] = \infty$. Note that since $V_n = V(D_M)$, a branching on V_n is a branching of D_M and thus $f[n, V_{\mathrm{NR}}] = \beta(M)$. We proceed to derive a recurrence for $f[i, S]$. Clearly, if $i = 1$, we have $f[i, \varnothing] = |v_1|$ and $f[i, S] = \infty$ for $S \neq \varnothing$. Consider the case $i \geq 2$. Let $S \subseteq V_{\mathrm{NR}}$ be a subset. Assume that an (i, S)-branching exists and let B be an optimal one. Denote by H the child set of v_i in B. The branching B is decomposed into two parts: $B_1 = \{(c, v_i) \mid c \in H\}$ and $B_2 = B - B_1$. Let H_{NR} be the set of non-regular B-children of v_i. We have the following.

Lemma 4.10. *B_2 is an optimal $(i - 1, S - H_{\mathrm{NR}})$-branching.*
Proof. Since B_2 is obtained from B by removing the arcs directed from the vertices in H toward v_i, it is easy to see that it is an $(i - 1, S - H_{\mathrm{NR}})$-branching. The cost of B is equal to $\Sigma_{1\le j\le i} |U_B^j| = |U_B^i| + \Sigma_{1\le j\le i-1} |U_B^j| = |U_B^i| + cost_{i-1}(B_2)$. Note that $|U_B^i| = |v_i| - |\cup_{c\in H} c|$, which is determined by only the arcs in B_1. Let X be an optimal $(i - 1, S - H_{\mathrm{NR}})$-branching. If B_2 is not optimal, by replacing B_2 with X in B, we obtain an (i, S)-branching $X \cup B_1$ with less cost $|U_{B_1}^i| + \Sigma_{1\le j\le i-1} |U_X^j| = |U_B^i| + cost_{i-1}(X)$, which contradicts the optimality of B. Therefore, B_2 is optimal and the lemma holds. ■

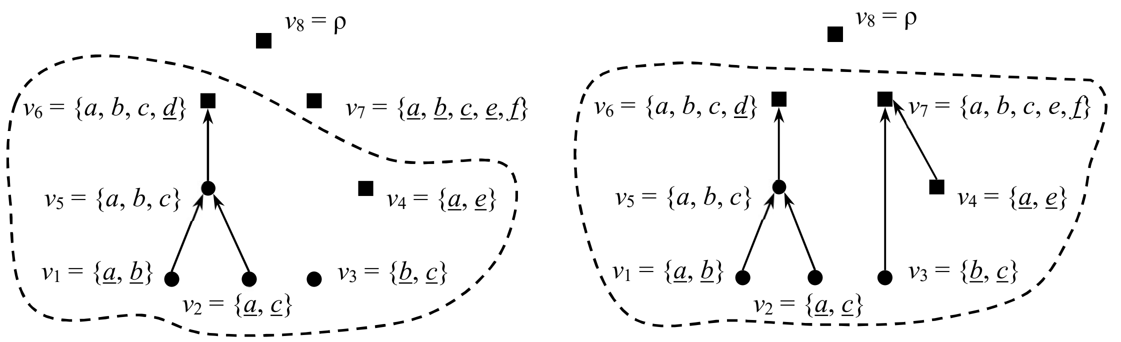

$v_8 = \rho$
$v_6 = \{a, b, c, \underline{d}\}$
$v_7 = \{\underline{a}, \underline{b}, \underline{c}, \underline{e}, \underline{f}\}$
$v_5 = \{a, b, c\}$
$v_4 = \{\underline{a}, \underline{e}\}$
$v_1 = \{\underline{a}, \underline{b}\}$
$v_2 = \{\underline{a}, \underline{c}\}$
$v_3 = \{\underline{b}, \underline{c}\}$
$v_8 = \rho$
$v_6 = \{a, b, c, \underline{d}\}$
$v_7 = \{a, b, c, e, \underline{f}\}$
$v_5 = \{a, b, c\}$
$v_4 = \{\underline{a}, \underline{e}\}$
$v_1 = \{\underline{a}, \underline{b}\}$
$v_2 = \{\underline{a}, \underline{c}\}$
$v_3 = \{\underline{b}, \underline{c}\}$
(a) A $(6, \{v_1, v_2, v_5\})$-branching with cost 9 (b) A $(7, \{v_1, v_2, v_3, v_5\})$-branching with cost 10

Figure 9. Two illustrative examples.

Let H_{R}^i be the set of B_{R}-children of v_i. Since B obeys B_{R}, we have $H = H_{\mathrm{R}}^i \cup H_{\mathrm{NR}}$ and $|U_B^i| = |v_i| - |\cup_{c\in H} c| = |v_i| - |\cup_{c\in H_{\mathrm{R}}^i \cup H_{\mathrm{NR}}} c|$. Therefore, according to Lemma 4.10, $f[i, S] = f[i - 1, S - H_{\mathrm{NR}}] + |v_i| - |\cup_{c\in H_{\mathrm{R}}^i \cup H_{\mathrm{NR}}} c|$. Consequently, we obtain

$$\begin{aligned} f[i, S] &= \min\{f[i-1, S - H_{\mathrm{NR}}] + |v_i| - |\cup_{c\in H_{\mathrm{R}}^i \cup H_{\mathrm{NR}}} c| \mid H_{\mathrm{NR}} \subseteq S \text{ and } \cup_{c\in H_{\mathrm{NR}}} c \subseteq v_i\} \\ &= \min\{f[i-1, S - T] + g[i, T] \mid T \subseteq S\}, \end{aligned} \tag{4-1}$$

where $g[i, T] = |v_i| - |\cup_{c\in H_{\mathrm{R}}^i \cup T} c|$ if $\cup_{c\in T} c \subseteq v_i$ and otherwise $g[i, T] = \infty$. The derivation of (4-1) assumes that an (i, S)-branching exists. Consider the case in which no (i, S)-branching exists. We show that recurrence (4-1) still holds. That is, we show that $f[i - 1, S - T] + g[i, T] = \infty$ for all $T \subseteq S$. Suppose, by contradiction, that $f[i - 1, S - T] + g[i, T] \neq \infty$ for some subset $T \subseteq S$. Since $f[i - 1, S - T] \neq \infty$, there exists an $(i - 1, S - T)$-branching X. Since $g[i, T] \neq \infty$, we have $\cup_{c\in T} c \subseteq v_i$. In addition, since the vertices are arranged in topological order, every vertex $v \in T$ is in V_{i-1}. By definition, $p_X(v) =$ null for each $v \in H_{\mathrm{R}}^i \cup T$. As a result, $X \cup \{(c, v_i) \mid c \in H_{\mathrm{R}}^i \cup T\}$ is an (i, S)-branching and we have a contradiction.

Based on recurrence (4-1), we give two implementations of Phase 3. The first one is as follows.

```
Procedure PHASE3-IMPLEMENTAION1
input: $D_M$, $B_R$
output: an optimal branching $B^*$
begin
1. topologically sort the vertices in $D_M$
```

```
2.  for each $S \subseteq V_{NR}$ do $f[1, S] \leftarrow \infty$
3.  $f[1, \varnothing] \leftarrow |v_1|$
4.  for $i = 2, 3, ..., n$ do
5.  begin
6.      compute $g[i, S]$ for all $S \subseteq V_{NR}$
7.      for each $S \subseteq V_{NR}$ do /* compute $f[i, S]$
8.      begin
9.        $f[i, S] \leftarrow \infty$
10.       for each $T \subseteq S$ do
11.         if ($f[i-1, S-T] + g[i, T] < f[i, S]$) then
12.         begin
13.            $f[i, S] \leftarrow f[i-1, S-T] + g[i, T]$
14.            $\tau[i, S] \leftarrow T$ /* for backtracking an optimal branching
15.         end
16.     end
17. end
18. /* solution recovery
19. $B^* \leftarrow \varnothing$
20. $S^* \leftarrow V_{NR}$
21. for $i = n, n-1, ..., 2$ do /* find the child set of $v_i$
22.   $T^* \leftarrow \tau[i, S^*]$
23.   $B^* \leftarrow B^* \cup \{(c, v_i) \mid c \in H_R{}^i \cup T^*\}$
24.   $S^* \leftarrow S^* - T^*$
25. end
26. return $B^*$
end
```

The time complexity is analyzed as follows. Lines 1-3 require $O(n^2 + 2^{|V_{NR}|})$ time. Note that after Phase 1, all non-regular vertices are not sibling-compatible and thus we have $|V_{NR}| = t(M)$. Consider a fixed iteration of the for-loop in lines 4-17. Each vertex $v \in V_{NR}$ contains at most m elements. Thus, the computation of $g[i, \cdot]$ in line 6 can be implemented in $O(m\ 2^{|V_{NR}|})$ time. For a fixed S, the for-loop in lines 10-15 performs $O(2^{|S|})$ times, each for a subset T of S. According to line 7, this for-loop runs for all $S \subseteq V_{NR}$. Each set operation on two subsets of V_{NR} takes $O(|V_{NR}|)$ time. Thus, for a fixed i, the total time spent by lines 13-14 is $O(|V_{NR}| \times \Sigma_{S \subseteq V_{NR}} 2^{|S|}) = O(|V_{NR}| \times \Sigma_{0 \le k \le |V_{NR}|} C(|V_{NR}|, k)\ 2^k)$, which is $O(|V_{NR}| \times 3^{|V_{NR}|})$ by the binomial theorem [6]. As a result, the for-loop in lines 4-17 requires $O(mn\ 2^{|V_{NR}|} + n|V_{NR}|\ 3^{|V_{NR}|})$ time. Clearly, lines 19-26 take $O(n|V_{NR}|)$ time. Consequently, the overall time complexity is $O(mn\ 2^{|V_{NR}|} + n|V_{NR}|\ 3^{|V_{NR}|} + n|V_{NR}|)$, which is $O(mn\ 2^{t(M)} + nt(M)\ 3^{t(M)})$ when substituting $t(M)$ for $|V_{NR}|$.

We proceed to present the second implementation. For a set P and two functions F, G: $2^P \rightarrow \mathbb{Z}$, the *subset convolution of F and G over the integer min-sum semiring* is defined for all $S \subseteq P$ by $(F * G)(S) = \min\{F(S-T) + G(T) \mid T \subseteq S\}$. We need the following.

Lemma 4.11. [3, 6] *The subset convolution over the integer min-sum semiring can be computed in time* $O(2^{|P|}|P|^3 X \lg(|P|X) \lg\lg(|P|X))$, *provided that the range of the input functions is* $\{-X, -X+1, \ldots, X\}$.

According to (4-1), the computation of $f[i, \cdot]$ can be seen as a subset convolution within the min-sum semiring of integers by setting $P = V_{NR}$ and defining $F(S) = f[i-1, S]$ and $G(S) = g[i, S]$ for all $S \subseteq P$. The cost of a branching is at most $\Sigma_{1 \le i \le n} |v_i| \le mn$. To applying Lemma 4.11, we use $N = mn + 1$ to represent ∞. That is, when no (i, S)-branching exists, we define $f[i, S] = N$. Then, Lemma 4.11 can be applied by setting $X = N$. Our second implementation is as follows.

```
Procedure PHASE3-IMPLEMENTAION2
input: $D_M$, $B_R$
output: an optimal branching $B^*$
begin
1.  topologically sort the vertices in $D_M$
2.  for each $S \subseteq V_{NR}$ do $f[1, S] \leftarrow N$
3.  $f[1, \varnothing] \leftarrow |v_1|$
4.  for $i = 2, 3, ..., n$ do /* compute $f[i, \cdot]$
```

```
5. begin
6.     compute $g[i, S]$ for all $S \subseteq V_{\mathrm{NR}}$
7.     compute $f[i, S]$ for all $S \subseteq V_{\mathrm{NR}}$ by applying Lemma 4.11
8. end
9. /* solution recovery
10. $B^* \leftarrow \emptyset$
11. $S^* \leftarrow V_{\mathrm{NR}}$
12. for $i = n, n - 1, ..., 2$ do /* find the child set of $v_i$
13. begin
14.   $T^* \leftarrow \mathrm{argmin}\{f[i - 1, S^* - T] + g[i, T] \mid T \subseteq S^*\}$
15.   $B^* \leftarrow B^* \cup \{(c, v_i) \mid c \in H_{\mathrm{R}}^i \cup T^*\}$
16.   $S^* \leftarrow S^* - T^*$
17. end
18. return $B^*$
end
```

The bottleneck of lines 1-8 is the computation of $g[i, \cdot]$ and $f[i, \cdot]$ in lines 6 and 7. Line 6 requires $O(m\ 2^{|V_{\mathrm{NR}}|})$ time. By Lemma 4.11, for a fixed i, line 7 requires $O(2^{|P|}|P|^3X \lg(|P|X) \lg\lg(|P|X)) = O(2^{|V_{\mathrm{NR}}|}|V_{\mathrm{NR}}|^3N \lg(|V_{\mathrm{NR}}|N) \lg\lg(|V_{\mathrm{NR}}|N))$ time. Therefore, lines 1-8 take a total of $O(mn\ 2^{|V_{\mathrm{NR}}|} + nN|V_{\mathrm{NR}}|^3 \lg(|V_{\mathrm{NR}}|N) \lg\lg(|V_{\mathrm{NR}}|N)\ 2^{|V_{\mathrm{NR}}|})$ time. Line 14 can be implemented in $O(|V_{\mathrm{NR}}|\ 2^{|V_{\mathrm{NR}}|})$ time. Thus, the for-loop in lines 12-17 requires $O(n|V_{\mathrm{NR}}|\ 2^{|V_{\mathrm{NR}}|})$ time. Consequently, the overall time complexity is $O(mn\ 2^{|V_{\mathrm{NR}}|} + nN|V_{\mathrm{NR}}|^3 \lg(|V_{\mathrm{NR}}|N) \lg\lg(|V_{\mathrm{NR}}|N)\ 2^{|V_{\mathrm{NR}}|})$, which is $O(mn^2\ t(M)^3 \lg mn \lg\lg mn\ 2^{t(M)})$ when substituting $t(M)$ for $|V_{\mathrm{NR}}|$ and $mn + 1$ for N.

Recall that Phases 1 and 2 take, respectively, $O(mn^2 + n^3)$ and $O(n^2)$ time. In summary, we obtain the following.

Theorem 4.12. *MSRP can be solved in time $O(mn^2 + n^3 + mn\ 2^{t(M)} + nt(M)\ 3^{t(M)})$ or $O(n^3 + mn^2\ t(M)^3 \lg mn \lg\lg mn\ 2^{t(M)})$.*

Combining Theorem 4.12 and Corollary 3.10 immediately yields the following, which says that MSRP is fixed-parameter tractable when parameterized by $\varepsilon(M)$.

Theorem 4.13. *MSRP can be solved in time $O(mn^2 + n^3 + mn\ 2^{\min(n,\ 2\varepsilon(M))} + n^2\ 3^{\min(n,\ 2\varepsilon(M))})$ or $O(mn^5 \lg mn \lg\lg mn\ 2^{\min(n,\ 2\varepsilon(M))})$.*

5. Algorithms for CMSRP, MDCRSP, and CMDCRSP

This section gives algorithms for CMSRP, MDCRSP, and CMDCRSP. As in Section 4, we assume that D_M is precomputed.

5.1 CMSRP

Recall that CMSRP is a constrained version of MSRP, in which only the rows in a given subset S can be split. In CMSRP, a conflict-free row split, equipped with a partition $\{R_1, R_2, ..., R_m\}$, is *feasible* if $|R_i| = 1$ for each $r_i \notin S$. As MSRP, only compact conflict-free row splits need to be considered. By Theorem 2.1, each branching B with $|U_B(r_i)| = 1$ for each $r_i \notin S$ corresponds to a feasible conflict-free row split; and each feasible compact conflict-free row split corresponds to a branching B with $|U_B(r_i)| = 1$ for each $r_i \notin S$. As a result, CMSRP is equivalent to the following: find a *feasible* branching of D_M with the minimum cost, where a branching B is feasible if $|U_B(r_i)| = 1$ for each $r_i \notin S$.

For each row $r \in R(M)$, let $V|_r$ be the set of vertices containing r in $V(D_M)$. A row $r \in R(M)$ is *chain-like* if any two vertices $u, v \in V|_r$ are nested. We have the following.

Lemma 5.1. *If a row r is chain-like, every vertex in $V|_r$ is regular.*
Proof. Suppose, by contradiction, that a vertex $v \in V|_r$ is not regular. By definition, v is contained in both of two conflicting vertices u, w. Since $v \subset u$ and $v \subset w$, u and w are both in $V|_r$. Since u and w are in conflict, r is not chain-like and we have a contradiction. Thus, the lemma holds. ■

Lemma 5.2. *A row r is chain-like if and only if there exists a branching B with $|U_B(r)| = 1$.*
Proof. Consider a fixed row r. If $|U_B(r)| = 1$ for some branching B, by Lemma 3.2, there does not exist

any pair of conflicting vertices in $V|_r$ and thus r is chain-like. Therefore, the if-part holds. Assume that r is chain-like. Recall that $\pi(v)$ is the smallest super-support of a regular vertex v and B_R is the set of arcs $\{(v, \pi(v)) \mid v \text{ is regular}\}$. We complete the proof by showing $|U_{B_R}(r)| = 1$. Let $L = (u_1, u_2, ..., u_{|L|} = \rho)$ be the sequence containing all vertices of $V|_r$ in non-decreasing order of their sizes. Consider a vertex u_i in L, $1 \le i < |L|$. By Lemma 5.1, u_i is regular. Since $r \in u_i$, $\pi(u_i)$ contains r and thus is also in L. Furthermore, since any two vertices in L are nested, we know that $\pi(u_i) = u_{i+1}$. Therefore, in the branching B_R, the target pairs (r, u_j) are covered for $j = 2, 3, ..., |L|$. As a result, $|U_{B_R}(r)| = |\{(r, u_1)\}| = 1$ and thus the lemma holds. ■

Let $S' = R(M) - S$. By Lemma 5.2, if any row in S' is not chain-like, there is no solution. Suppose that all rows in S' are chain-like. Let B^* be the branching obtained by running our MSRP algorithm on M. Since $B^* \supseteq B_R$, by the proof of Lemma 5.2, $|U_{B^*}(r)| = 1$ for all $r \in S'$. Thus, B^* is feasible. Since B^* is optimal for (the unconstrained) MSRP, it is optimal for CMSRP. Consequently, CMSRP can be solved as follows: determine whether all rows in S' are chain-like; if not, report that there is no solution, and otherwise run our MSRP algorithm on M and return its output.

Let L be the sequence containing all vertices of D_M, in non-decreasing order of their sizes. In Section 4.2, it has been showed that using L whether a vertex in D_M is regular can be determined in $O(n)$ time. Similarly, by the definition of chain-like rows, using L whether a row in S' is chain-like can be determined in $O(n)$ time. Thus, whether all rows in S' are chain-like can be determined in $O(n \lg n + mn)$ time. As a result, by Theorem 4.13, we obtain the following.

Theorem 5.3. *CMSRP can be solved in time* $O(mn^2 + n^3 + mn\, 2^{\min(n,\, 2\varepsilon(M))} + n^2\, 3^{\min(n,\, 2\varepsilon(M))})$ *or* $O(mn^5 \lg mn \lg\lg mn\, 2^{\min(n,\, 2\varepsilon(M))})$.

5.2 MDCRSP

Recall that MDCRSP is to find a conflict-free row split of M with the minimum number of distinct rows. MDCRSP admits a formulation similar to MUBP. For a branching B, we say that a vertex $v \in V(D_M)$ is *B-irreducible* if there exists a row $r \in v$ such that (r, v) is B-uncovered. We denote by $I(B)$ the set of B-irreducible vertices. For the example in Figure 5(b), the set of B-irreducible vertices is $\{v_1, v_2, v_3, v_5, v_6, v_8, v_9, v_{11}\}$. Let $\zeta(M)$ be the minimum number of B-irreducible vertices over all branchings B of D_M. Ademir Hujdurović *et. al.* [17] showed that MDCRSP is equivalent to the following optimization problem.

Definition 3. Minimum Irreducing Branching Problem (MIBP):
Input: A binary matrix M.
Task: Find a branching B of D_M with $|I(B)| = \zeta(M)$.

The equivalence between MDCRSP and MIBP is formally stated as the following theorem.

Theorem 5.4. [17] *For a binary matrix* $M \in \{0, 1\}^{m\times n}$, *the following holds*:
1. *For any branching B, the matrix* M^B *is a conflict-free row split of M with* $|I(B)|$ *distinct rows.*
2. *Any compact conflict-free row split M′ of M can be transformed to a branching B of* D_M *such that* $|I(B)|$ *is equal to the number of distinct rows of M′.*

The only difference between MIBP and MUBP is the cost function: In MIBP, the cost of a branching B is $|I(B)|$ instead of $|U(B)|$. In the following, we show how to modify our MSRP algorithm to obtain an algorithm for MDCRSP. Since removing a sibling-compatible vertex may change $\zeta(M)$, Phase 1 is not applied. The following lemma shows that the parent of each regular vertex can still be predetermined in MDCRSP.

Lemma 5.5. *There exists an optimal branching* B^* (*with respect to MIBP*) *of* D_M *such that* $B^* \supseteq B_R$.
Proof. Let B^* be an optimal branching not containing B_R. By repeatedly performing a regularization operation, B^* can be transformed into a branching B' which contains all arcs in B_R. By Lemma 4.8, this transformation does not induce any new uncovered pairs. That is, $U(B^*) \supseteq U(B')$, from which $I(B^*) \supseteq I(B')$ is concluded. Since B^* is optimal, B' is also optimal and the lemma holds. ■

By Lemma 5.5, Phase 2 needs no modification. We proceed to present the modification for Phase 3. Let V_i and (i, S)-branching be defined the same as in Section 4.3. Consider a positive integer $i \leq n$ and a subset $S \subseteq V_{\rm NR}$. We redefine the cost of an (i, S)-branching B as the number of B-irreducible vertices $v \in V_i$. Define $f'[i, S]$ to be the minimum cost of an (i, S)-branching and if such a branching does not exist, $f'[i, S] = \infty$. Then, our problem is to compute $f'[n, V_{\rm NR}]$. For $i = 1$, since V_i contains a single vertex, we have $f'[i, \varnothing] = 1$ and $f'[i, S] = \infty$ for $S \neq \varnothing$. Consider the case $i \geq 2$. Recall that $H_{\rm R}^i$ is the set of $B_{\rm R}$-children of v_i. For a subset $T \subseteq V_{\rm NR}$, we define $g'[i, T]$ as follows: if $H_{\rm R}^i \cup T$ is a proper subset of v_i, define $g'[i, T] = 1$, indicating that v_i is irreducible if its child set is $H_{\rm R}^i \cup T$ in a branching; if $H_{\rm R}^{\rm i} \cup T = v_i$, define $g'[i, T] = 0$, indicating that v_i is not irreducible if its child set is $H_{\rm R}^i \cup T$ in a branching; and if $H_{\rm R}^i \cup T \not\subset v_i$, define $g'[i, T] = \infty$, indicating that there is no branching in which the child set of v_i is $H_{\rm R}^i \cup T$. Then, we have the following:

$$f'[i, S] \;=\; \min\{f'[i-1, S-T] + g'[i, T] \mid T \subseteq S\} \text{ for } i > 1 \text{ and } S \subseteq V_{\rm NR}. \tag{5-1}$$

The proof for (5-1) is analogous to the proof of (4-1) and thus is omitted. The two recurrences in (4-1) and (5-1) are structurally the same. Consequently, the only modification for Phase 3 is to replace f and g by, respectively, f' and g'.

Recall that the precomputation of D_M takes $O(mn^2)$ time and Phase 2 takes $O(n^2)$ time. The time complexity of Phase 3 with the above modification is analyzed as follows. Note that since Phase 1 is not applied, all non-regular vertices in the original D_M are contained in $V_{\rm NR}$, which by Theorem 3.11 is of cardinality at most $\min(n, 3\varepsilon(M) - 1)$. It is easy to check that the time complexities of the two implementations are still, respectively, $O(mn\, 2^{|V_{\rm NR}|} + n|V_{\rm NR}|\, 3^{|V_{\rm NR}|} + n|V_{\rm NR}|)$ and $O(mn\, 2^{|V_{\rm NR}|} + nN|V_{\rm NR}|^3 \lg(|V_{\rm NR}|N) \lg\lg(|V_{\rm NR}|N)\, 2^{|V_{\rm NR}|})$. Consider the second implementation. Since $|I(B)|$ is at most n, to apply Lemma 4.11, we can take $N = n + 1$ to represent ∞, so that the resulting time complexity is reduced by a factor of m. By substituting $\min(n, 3\varepsilon(M) - 1)$ for $|V_{\rm NR}|$ and $n + 1$ for N in the above time complexities, we obtain the following.

Theorem 5.6. *MDCRSP can be solved in time* $O(mn^2 + n^3 + mn\, 2^{\min(n,\, 3\varepsilon(M))} + n^2\, 3^{\min(n,\, 3\varepsilon(M))})$ *or* $O(mn^2 + mn\, 2^{\min(n,\, 3\varepsilon(M))} + n^5 \lg n \lg\lg n\, 2^{\min(n,\, 3\varepsilon(M))})$.

5.3 CMDCRSP

Our algorithms for CMSRP and MDCRSP can be combined to solve CMDCRSP. Given a binary matrix M and a subset S of its rows, CMDCRSP is to find a feasible conflict-free row split of M with minimum number of distinct rows, where a conflict-free row split is feasible if $|R_i| = 1$ for each $r_i \notin S$. As in Section 5.1, a branching B is feasible if $|U_B(r_i)| = 1$ for each $r_i \notin S$. By Theorems 2.1 and 5.4, CMDCRSP is equivalent to the following: find a feasible branching B of D_M that has the smallest $|I(B)|$. Let $S' = R(M) - S$. By Lemma 5.2, if any row in S' is not chain-like, no feasible branching exists and thus there is no solution. Assume that each row in S' is chain-like. Let B^* be the branching obtained by running our MDCRSP algorithm on M. Since $B^* \supseteq B_{\rm R}$, from the proof of Lemma 5.2, we know that $|U_{B^*}(r)| = 1$ for all $r \in S'$. Thus, B^* is feasible. Since B^* is optimal for MDCRSP, it is optimal for CMDCRSP. Therefore, CMDCRSP can be solved as efficient as MDCRSP.

Theorem 5.7. *CMDCRSP can be solved in time* $O(mn^2 + n^3 + mn\, 2^{\min(n,\, 3\varepsilon(M))} + n^2\, 3^{\min(n,\, 3\varepsilon(M))})$ *or* $O(mn^2 + mn\, 2^{\min(n,\, 3\varepsilon(M))} + n^5 \lg n \lg\lg n\, 2^{\min(n,\, 3\varepsilon(M))})$.

6. Conclusion and future work

In this paper, we showed that MSRP is fixed-parameter tractable when parameterized by $\varepsilon(M)$. The proposed FPT-time algorithm was built upon several new structural properties of MSRP: we showed that the removal of sibling-compatible vertices does not change $\varepsilon(M)$, proved that the parents of regular vertices can be greedily predetermined, and derived an upper bound on $t(M)$, which is the number of non-regular and non-sibling-compatible vertices. We also extended our MSRP algorithm to solve CMSRP, MDCRSP, and CMDCRSP.

As remarked in Section 3, our upper bounds on $t'(M)$ and $|V_{\rm NR}|$ are almost tight. To analyze the worst-case time complexity of our MSRP algorithm, we use $t'(M)$ as an upper bound on $t(M)$ directly. At this writing, the authors do not aware of any construction of a matrix M with $t(M) = 2\varepsilon(M) - O(1)$. A problem left open in this work is to find a tighter upper bound on $t(M)$. Furthermore, since the removal of a sibling-compatible vertex can make a non-regular and non-sibling-compatible vertex become sibling-compatible, the number of non-regular vertices treated in Phase 3, denoted by $t^*(M)$, may be smaller than $t(M)$. It would also be interesting to derive a tighter upper bound on $t^*(M)$. Other possible directions for future research include: (1) improve the $O^*(2^{\min(n,\ 2\varepsilon(M))})$ time MSRP algorithm; (2) identify and study other meaningful parameters; (3) give a polynomial kernel for MSRP with respect to the parameter $\varepsilon(M)$; (4) determine if MSRP is in APX; and (5) improve the approximation algorithms in [17] for MSRP and MDCRSP.

ACKNOWLEDGMENTS

This research is supported by the Ministry of Science and Technology of the Republic of China under grant MOST-109-2221-E-007-080-MY3.

REFERENCES

[1] A. V. Aho, M. R. Garey, and J. D. Ullman, "The Transitive Reduction of a Directed Graph," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 131–137, 1972.

[2] A. Bashashati, G. Ha, A. Tone, J. Ding, L. M. Prentice, A. Roth, J. Rosner, K. Shumansky, S. Kalloger, J. Senz, W. Yang, M. McConechy, N. Melnyk, M. Anglesio, M. T. Y. Luk, K. Tse, T. Zeng, R. Moore, Y. Zhao, M. A. Marra, B. Gilks, S. Yip, D. G. Huntsman, J. N. McAlpine, and S. P. Shah, "Distinct evolutionary trajectories of primary high-grade serous ovarian cancers revealed through spatial mutational profiling," *The Journal of Pathology*, vol. 231, no. 1, pp. 21–34, 2013.

[3] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, "Fourier Meets MöBius: Fast Subset Convolution," In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing* (*STOC '07*), pp. 67–74, 2007.

[4] P. J. Campbell, E. D. Pleasance, P. J. Stephens, E. Dicks, R. Rance, I. Goodhead, G. A. Follows, A. R. Green, P. A. Futreal, and M. R. Stratton, "Subclonal phylogenetic structures in cancer revealed by ultra-deep sequencing," in *Proceedings of the National Academy of Sciences*, vol. 105, no. 35, pp. 13081–13086, 2008.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms* (3rd ed.), The MIT Press, 2009.

[6] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms* (1st ed.), Springer Publishing Company, Incorporated, 2015.

[7] R. G. Downey and M. R. Fellows, *Parameterized Complexity*, Springer Publishing Company, Incorporated, 2012.

[8] P. Eirew, A. Steif, J. Khattra, G. Ha, D. Yap, H. Farahani, K. Gelmon, S. Chia, C. Mar, A. Wan, E. Laks, J. Biele, K. Shumansky, J. Rosner, A. McPherson, C. Nielsen, A. J. L. Roth, C. Lefebvre, A. Bashashati, C. Souza, C. Siu, R. Aniba, J. Brimhall, A. Oloumi, T. Osako, A. Bruna, J. L. Sandoval, T. Algara, W. Greenwood, K. Leung, H. Cheng, H. Xue, Y. Wang, D. Lin, A. J. Mungall, R. Moore, Y. Zhao, J. Lorette, L. Nguyen, D. Huntsman, C. J. Eaves, C. Hansen, M. A. Marra, C. Caldas, S. P. Shah, and S. Aparicio, "Dynamics of genomic clones in breast cancer patient xenografts at single-cell resolution," *Nature*, vol. 518, no. 7539, pp. 422–426, 2015.

[9] M. El-Kebir, L. Oesper, H. Acheson-Field, and B. J. Raphael, "Reconstruction of clonal trees and tumor composition from multi-sample sequencing data," *Bioinformatics*, vol. 31, no. 12, pp. i62–i70, 2015.

[10] D. Fernández-Baca, "The Perfect Phylogeny Problem," in *Steiner Trees in Industry*, Springer US, pp. 203–234, 2001.

[11] D. Fernández-Baca and J. Lagergren, "A Polynomial-Time Algorithm for Near-Perfect Phylogeny," *SIAM Journal on Computing*, vol. 32, pp. 1115–1127, 2003.

[12] M. Gerlinger, S. Horswell, J. Larkin, A. J. Rowan, M. P. Salm, I. Varela, R. Fisher, N. McGranahan, N. Matthews, C. R. Santos, P. Martinez, B. Phillimore, S. Begum, A. Rabinowitz, B. Spencer-Dene, S. Gulati, P. A. Bates, G. Stamp, L. Pickering, M. Gore, D. L. Nicol, S. Hazell, P. A. Futreal, A. Stewart, and C. Swanton, "Genomic architecture and evolution of clear cell renal cell carcinomas defined by multiregion sequencing," *Nature Genetics*, vol. 46, no. 3, pp. 225–233, 2014.

[13] D. Gusfield, "Efficient algorithms for inferring evolutionary trees," *Networks*, vol. 21, no. 1, pp. 19–28, 1991.

[14] D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*, Cambridge University Press, 1997.

[15] I. Hajirasouliha, A. Mahmoody, and B. J. Raphael, "A combinatorial approach for analyzing intra-tumor heterogeneity from high-throughput sequencing data," *Bioinformatics*, vol. 30, no. 12, pp. i78–i86, 2014.

[16] I. Hajirasouliha and B. J. Raphael, "Reconstructing Mutational History in Multiply Sampled Tumors Using Perfect Phylogeny Mixtures," in *Proceedings of the 14th International Workshop on Algorithms in Bioinformatics*, pp. 354–367, 2014.

[17] A. Hujdurović, E. Husić, M. Milanič, R. Rizzi, and A. I. Tomescu. "Perfect Phylogenies via Branchings in Acyclic Digraphs and a Generalization of Dilworth's Theorem," *ACM Transactions on Algorithms*, vol. 14, no. 2, Article 20, 26 pages, 2018.
[18] A. Hujdurović, U. Kacar, M. Milanič, B. Ries, and A. I.Tomescu, "Complexity and Algorithms for Finding a Perfect Phylogeny from Mixed Tumor Samples," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, no. 1, pp. 96–108, 2018.
[19] E. Husić, X. Li, A. Hujdurović, M. Mehine, R. Rizzi, V. Mäkinen, M. Milanič, and A. I. Tomescu, "MIPUP: minimum perfect unmixed phylogenies for multi-sampled tumors via branchings and ILP," *Bioinformatics*, vol. 35, no. 5, pp. 769–777, 2019.
[20] W. Jiao, S. Vembu, A. G. Deshwar, L. Stein, and Q. Morris, "Inferring clonal evolution of tumors from single nucleotide somatic mutations," *BMC Bioinformatics*, vol. 15, suppl. 1, pp. 35, 2014.
[21] S. Kannan and T. Warnow, "A Fast Algorithm for the Computation and Enumeration of Perfect Phylogenies," *SIAM Journal on Computing*, vol. 26, no. 6, pp. 1749–1763, 1997.
[22] S. Malikic, A. W. McPherson, N. Donmez, and C. S. Sahinalp, "Clonality inference in multiple tumor samples using phylogeny," *Bioinformatics*, vol. 31, no. 9, pp. 1349–1356, 2015.
[23] M. Mehine, H. Heinonen, N. Sarvilinna, E. Pitkänen, N. Mäkinen, R. Katainen, S. Tuupanen, R. Bützow, J. Sjöberg, and L. A. Aaltonen, "Clonally related uterine leiomyomas are common and display branched tumor evolution," *Human Molecular Genetics*, vol. 24, no. 15, pp. 4407–4416, 2015.
[24] W. Mohammed Ismail, E. Nzabarushimana, and H. Tang, "Algorithmic approaches to clonal reconstruction in heterogeneous cell populations," *Quantitative Biology*, vol. 7, no. 4, pp. 255–265, 2019.
[25] D. E. Newburger, D. Kashef-Haghighi, Z. Weng, R. Salari, R. T. Sweeney, A. L. Brunner, S. X. Zhu, X. Guo, S. Varma, M. L. Troxell, R. B. West, S. Batzoglou, and A. Sidow, "Genome evolution during progression to breast cancer," *Genome Research*, vol. 23, no. 7, pp. 1097–1108, 2013.
[26] S. Nik-Zainal, P. Van Loo, D. C. Wedge, L. B. Alexandrov, C. D. Greenman, K. W. Lau, K. Raine, D. Jones, J. Marshall, M. Ramakrishna, A. Shlien, S. L. Cooke, J. Hinton, A. Menzies, L. A. Stebbings, C. Leroy, M. Jia, R. Rance, L. J. Mudie, S. J. Gamble, P. J. Stephens, S. McLaren, P. S. Tarpey, E. Papaemmanuil, H. R. Davies, I. Varela, D. J. McBride, G. R. Bignell, K. Leung, A. P. Butler, J. W. Teague, S. Martin, G. Jönsson, O. Mariani, S. Boyault, P. Miron, A. Fatima, A. Langerod, S. A. J. R. Aparicio, A. Tutt, A. M. Sieuwerts, Å. Borg, G. Thomas, A. V. Salomon, A. L. Richardson, A. L. Borresen-Dale, P. A. Futreal, M. R. Stratton, and P. J.Campbell, "The life history of 21 breast cancers," *Cell*, vol. 149, no. 5, pp. 994–1007, 2012.
[27] I. Pe'er, R. Shamir, and R. Sharan, "*Incomplete Directed Perfect Phylogeny*," *SIAM Journal on Computing*, vol. 33, no. 3, pp. 590–607, 2000.
[28] V. Popic, R. Salari, I. Hajirasouliha, D. Kashef-Haghighi, R. B. West, and S. Batzoglou, "Fast and scalable inference of multi-sample cancer lineages," *Genome Biology*, vol. 16, no. 1, pp. 91, 2015.
[29] F. Strino, F. Parisi, M. Micsinai, and Y. Kluger, "TrAp: a tree approach for fingerprinting subclonal tumor composition," *Nucleic Acids Research*, vol. 41, no. 17, pp. e165, 2013.