SIAM J. DISCRETE MATH.
Vol. 37, No. 3, pp. 2049–2071

PARAMETERIZED COMPLEXITY FOR FINDING A PERFECT PHYLOGENY FROM MIXED TUMOR SAMPLES*

WEN-HORNG SHEU† AND BIING-FENG WANG†

Abstract. Motivated by an application in cancer genomics, Hajirasouliha and Raphael [*Proceedings of the* 14*th International Workshop on Algorithms in Bioinformatics*, 2014, pp. 354–367] proposed the *split-row problem* (SR). 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 $r^{(1)}, r^{(2)}, \ldots, r^{(k)}$ 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. [*ACM Trans. Algorithms*, 14 (2018), 26] proved the APX-hardness of SR and presented efficient exact and approximation algorithms. The parameterized study of SR 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 SR. This result indicates that SR 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 SR, called the *distinct split-row problem* (DSR). Our algorithm can be adapted to this variant as well. In addition, our algorithms can be extended to solve SR and DSR with the following additional constraint: only the rows in a given subset are allowed to be split.

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

MSC codes. 05C05, 68Q27, 68W40, 92D15

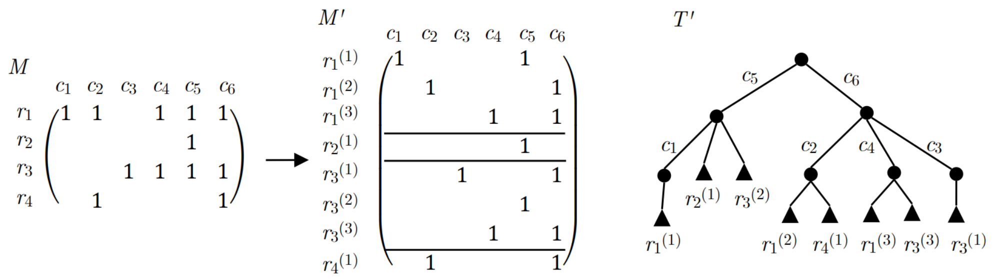
DOI. 10.1137/21M1449269

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 $\{c_1, c_2, \ldots, c_6\}$. 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'. For clarity, in Figure 1, 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 [8, 9, 10, 19, 23]. In this paper, we study a generalization introduced by Hajirasouliha and Raphael [13], called the *split-row problem* (SR). In this problem, an $m \times n$ binary matrix M is given. A *split-row*

*Received by the editors September 28, 2021; accepted for publication (in revised form) May 1, 2023; published electronically September 14, 2023.
https://doi.org/10.1137/21M1449269
Funding: This research was supported by the Ministry of Science and Technology of the Republic of China under grant MOST-109-2221-E-007-080-MY3.

†Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan 30013, Republic of China (whsheu@gapp.nthu.edu.tw, bfwang@cs.nthu.edu.tw).



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

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

FIG. 1. *An illustrative example.*

operation on M is defined as replacing a row r of M by k rows $r^{(1)}, r^{(2)}, \ldots, r^{(k)}$ 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 $2+2=4$.

The study of SR was motivated by an application in cancer genomics [13]. 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 [12, 13, 25]. 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 [15]. This phylogenetic tree can offer a more comprehensive knowledge of tumor progression [3, 22] and help in development of therapies [21]. 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 SR is to minimally decompose the samples and thereby reconstruct the perfect phylogeny. As more datasets of genome sequencing from multiple samples of a tumor become available, there will be increasing need for computational models to infer mutational history of the data [13]. Various other models have been proposed for inferring phylogenetic trees from tumor samples [7, 12, 18, 20, 24, 25]. We refer the reader to [17] for a survey. In the application of phylogeny reconstruction, the problem formulation of SR is simpler than the probabilistic models usually used, because it just does binary classification on the input. In [16], SR was used for phylogeny reconstruction and tested against four popular models. Experimental results showed that due to its simplicity, SR was more efficient in running time. In addition, the phylogenetic trees reconstructed by SR were generally more faithful on real data and were mostly more accurate on simulated data, and SR was more resilient to a certain type of noisy data.

Hujdurović et al. [15] showed the NP-hardness of SR and gave an efficient heuristic algorithm. Later, Hujdurović et al. [14] further proved that SR is APX-hard. A naive method for SR 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 [14]. Hujdurović et al. [14] introduced the *uncovering branching problem* (UB). In this problem, we are given a set family Φ, and we look at its containment digraph, which is a digraph representing the containment relations between pairs of sets in a family. A *branching* is defined to be a subset of arcs which induces a directed spanning forest of the digraph. Each branching is associated with a cost. The goal is to find a branching of the minimum cost. Hujdurović et al. [14] proved that SR is equivalent to UB and called UB the *branching formulation* of SR. Using this formulation, they solved SR more efficiently by a reduction to UB and then enumerating all branchings of the derived containment digraph. 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 SR. 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 the derived containment digraph. Based on the branching formulation, Husić et al. [16] formulated SR as an integer linear program (ILP) and implemented it into a software package called MIPUP, using the CPLEX ILP solver.

The parameterized study of SR was left as a direction for future work in [14]. In this paper, we show that SR can be solved in $O^*(2^{\min(n,2\varepsilon(M)})$ time. In the language of *parameterized complexity,* our result indicates that SR 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 that depends only on k. The fixed-parameter tractability of SR signifies that the combinatorial explosion in the running time of an algorithm can be confined to the parameter $\varepsilon(M)$ and does not necessarily depend on the input size. The reader may refer to the monograph of Downey and Fellows [6] 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, SR 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 SR, called the *distinct split-row problem* (DSR), was also introduced by Hajirasouliha and Raphael in [13]. The task of DSR 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. [15] showed the NP-hardness of DSR. Hujdurović et al. [14] strengthened the NP-hardness result by proving that DSR is APX-hard. In addition, they gave a 2-approximation algorithm, which implies that DSR is APX-complete, and gave an $O^*(n^n)$-time exact algorithm. Our SR algorithm can be modified to solve DSR in $O^*(2^{\min(n,3\varepsilon(M))})$ time. In SR, all rows are allowed to be split. A constrained version of SR mentioned in [14, 15] 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 CSR. In the application of phylogeny reconstruction, CSR corresponds to the case when some samples are known to be not mixtures of distinct cell subpopulation.

Table 1
Exact algorithms for SR, CSR, DSR, and CDSR.

Problem	Results	Source
SR	$O^*(n^n)$	[14]
	ILP	[16]
	$O^*(2^{\min(n,2\varepsilon(M)})$	this paper
CSR	$O^*(2^{\min(n,2\varepsilon(M)})$	this paper
DSR	$O^*(n^n)$	[14]
	ILP	[16]
	$O^*(2^{\min(n,3\varepsilon(M))})$	this paper
CDSR	$O^*(2^{\min(n,3\varepsilon(M))})$	this paper

A constrained version of DSR is defined similarly, denoted by CDSR. To the authors' knowledge, no algorithm for CSR and CDSR has been presented in the literature. Our algorithms for SR and DSR can be extended to solve CSR and CDSR in the same time complexities. Table 1 summarizes exact algorithms for SR, CSR, DSR, and CDSR.

Our new result for SR is designed based on the branching formulation in [14]. More specifically, we give an improved algorithm for finding an optimal branching in the containment digraph of a given set family Φ. An overview is as follows. In the computation of an optimal branching, we find that some vertices, called *trivial vertices,* can be removed without changing the cost of an optimal branching. A branching represents a directed spanning forest 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 *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 with the predetermined one. On the basis of the findings, we give an algorithm consisting of three phases: Phase 1 removes trivial vertices, Phase 2 precomputes the parent of each regular vertex, and Phase 3 finds an optimal branching using dynamic programming. Phases 1 and 2 require polynomial time. Let $t(\Phi)$ be the number of vertices that are nontrivial and nonregular in the containment digraph of Φ. Phase 3 requires $O^*(2^{t(\Phi)})$ time. Let $\beta(\Phi)$ be the minimum cost of a branching and $\delta(\Phi) = \beta(\Phi) - |\Sigma|$, where Σ is the ground set of Φ. To estimate the worst-case complexity of Phase 3, we give an upper bound on $t(\Phi)$. More precisely, we show that $t(\Phi) \leq \min(|\Phi|, 2 \cdot \delta(\Phi) - 1)$. Consequently, our algorithm runs in $O^*(2^{\min(|\Phi|,2\delta(\Phi))})$ time. By applying this result, SR is solved in $O^*(2^{\min(n,2\varepsilon(M)})$ time.

We remark that the ILP-based algorithm in [16] can be speeded up by using our Phases 1 and 2 algorithms to predetermine the values of some variables. A *laminar* set family is a set family in which each pair of sets are either disjoint or related by containment. We also remark that UB is equivalent to the following interesting problem: Transform a given set family Φ to a laminar one by copying the elements of the ground set; it is required that each set in Φ receives at least one copy of each of its original elements; and we want to minimize the number of additional copies. This equivalence will be described in section 2.2.

This paper is organized as follows. Section 2 reviews the branching formulation. Section 3 defines trivial vertices and regular vertices and gives an upper bound on $t(\Phi)$. Section 4 presents our new algorithm for the branching formulation. Section 5 modifies the new algorithm to solve CSR, DSR, and CDSR. Section 6 concludes this paper.

2. Review of the branching formulation.

This section describes Hujdurović et al.'s branching formulation of SR. Let $M \in \{0,1\}^{m\times n}$ be a binary matrix. The entry of M at a row r and a column c is denoted by $M_{r,c}$. 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 $\varepsilon(M)$ [14].

2.1. Connection between conflict-free row splits and perfect phylogenies.

DEFINITION 2.1. *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.*

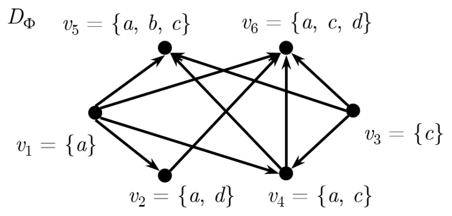
By the perfect phylogeny theorem [10, 11], a binary matrix M exhibits a perfect phylogeny if and only if M is conflict-free. It is not difficult to construct a perfect phylogeny corresponding to a given conflict-free matrix. In particular, [10] showed that the construction can be done in linear time.

A matrix is a *row split* of M if it can be obtained from M by performing split-row operations. Let $\alpha(M)$ be the minimum number of rows of a conflict-free row split of M. 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, $\varepsilon(M) = \alpha(M) - m$ and the computation of $\varepsilon(M)$ can be done by finding a conflict-free row split of M with the minimum number of rows.

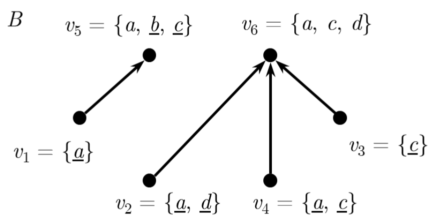
2.2. The branching formulation.

Let Φ be a set family over a set Σ. Without loss of generality, we assume that Φ does not contain the empty set and each element of Σ is contained in at least one set of Φ. The *containment digraph* D_Φ of Φ is the directed acyclic graph with vertex set Φ and arc set $\{(v, v') \mid v, v' \in \Phi, v \subset v'\}$. Figure 2(a) gives an example, in which $\Sigma = \{a, b, c, d\}$ and $\Phi = \{v_1, v_2, \ldots, v_6\}$, where $v_1 = \{a\}, v_2 = \{a, d\}, v_3 = \{c\}, v_4 = \{a, c\}, v_5 = \{a, b, c\}$, and $v_6 = \{a, c, d\}$. 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. A *branching* of D_Φ is a subset B of arcs such that (Φ, 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 (Φ, B) is a directed rooted forest. See Figure 2(b) for an example, in which $B = \{(v_1, v_5), (v_2, v_6), (v_3, v_6), (v_4, v_6)\}$.

Consider a branching B of D_Φ. Let F_B denote the forest (Φ, 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. For an element $e \in \Sigma$ and a vertex $v \in \Phi$, we say that (e, v) is a *target pair* if $e \in v$. Consider a target pair (e, v). We say that e is *B-covered* in v if v has a B-child u such that $e \in u$; otherwise, we say that e is *B-uncovered* in v. A B-uncovered pair is a target pair (e, v) such that e is B-uncovered in v. We denote by $U(B)$ the set of all B-uncovered pairs. For each element $e \in \Sigma$, we denote by $U_B(e)$ the set of all B-uncovered pairs (e, v), where $v \in \Phi$. For the example in Figure 2(b), $U_B(a) = \{(a, v_1), (a, v_2), (a, v_4)\}, U_B(b) = \{(b, v_5)\}, U_B(c) = \{(c, v_3), (c, v_4), (c, v_5)\}, U_B(d) = \{(d, v_2)\}$, and $U(B)$ is the union of $U_B(a), U_B(b), U_B(c)$, and $U_B(d)$. Observe that if a vertex v contains an element e, there is at least one descendant of v in which e is B-uncovered. Thus, $|U_B(e)| \geq 1$



(a) *The containment digraph D_Φ*


B
$v_5 = \{a,\ \underline{b},\ \underline{c}\}$
$v_6 = \{a,\ c,\ d\}$
$v_1 = \{\underline{a}\}$
$v_3 = \{\underline{c}\}$
$v_2 = \{\underline{a},\ \underline{d}\}$
$v_4 = \{\underline{a},\ \underline{c}\}$

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

FIG. 2. *An illustrative example.*

for each element $e \in \Sigma$. The *cost* of a branching B is defined as $|U(B)|$. We denote by $\beta(\Phi)$ the minimum cost of a branching of D_Φ. Define an optimization problem, which is referred as the *branching formulation* of SR, as follows.

DEFINITION 2.2. *The uncovering branching problem:*
Input: A set family Φ.
Task: Find a branching B of D_Φ with $U(B) = \beta(\Phi)$.

Without loss of generality, we assume that Φ contains more than one set; otherwise, D_Φ consists of a single vertex and $B = \emptyset$ is the unique branching. Hujdurović et al. [14] proved that SR is equivalent to UB. An interested reader may refer to [14] for the proof. Given the input matrix M of SR, a set family Φ_M of size n over a set Σ_M of size m is created for the reduction to the branching problem as follows. Let r_i denote the row i of M. Define $\Phi_M = \{v_1, v_2, \ldots, v_n\}$, where $v_j = \{r_i \mid 1 \le i \le m, M_{i,j} = 1\}$. Intuitively, each vertex v_j, $1 \le j \le n$, represents the column j of M. For example, for the matrix M in Figure 1, we have $v_1 = \{r_1\}, v_2 = \{r_1, r_4\}, v_3 = \{r_3\}, v_4 = \{r_1, r_3\}, v_5 = \{r_1, r_2, r_3\}$, and $v_6 = \{r_1, r_3, r_4\}$. The ground set of Φ_M is $\Sigma_M = \{r_1, r_2, \ldots, r_m\}$. Hujdurović et al. [14] showed that $\alpha(M) = \beta(\Phi_M)$ and each minimum cost branching of D_{Φ_M} can be transformed in $O(mn^2)$ time to a minimum cost conflict-free row split of M. Accordingly, SR can be solved by resorting to an algorithm for UB. This result is formally stated as the following theorem.

THEOREM 2.3 ([14]). *$\alpha(M) = \beta(\Phi_M)$, and if there is an $O(T)$-time algorithm for finding a minimum cost branching of D_{Φ_M}, then there is an $O(mn^2 + T)$-time algorithm for SR.*

A *laminar* set family is a set family in which each pair of sets are either disjoint or related by containment. A different interpretation of UB is described as follows. We are given a set family Φ, and we look at its containment digraph. We want to

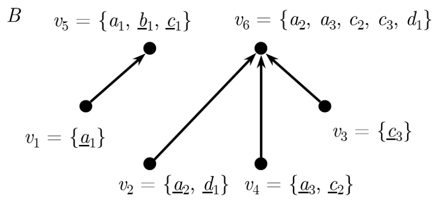

B
$v_5 = \{a_1, \underline{b}_1, \underline{c}_1\}$
$v_6 = \{a_2, a_3, c_2, c_3, d_1\}$
$v_1 = \{\underline{a}_1\}$
$v_3 = \{\underline{c}_3\}$
$v_2 = \{\underline{a}_2, \underline{d}_1\}$
$v_4 = \{\underline{a}_3, \underline{c}_2\}$

FIG. 3. *Make the set family Φ in Figure* 2(a) *laminar by using the branching in Figure* 2(b).

transform Φ to a laminar one by copying the elements of the ground set, and we require that each set in Φ receives at least one copy of the original elements, and the resulting family is laminar with respect to the copies. Here is how a branching B determines the copies: for each element e, $U_B(e)$ corresponds to the set of "copies" of e, and following the branching it determines which copies each set gets (i.e., for a copy corresponding to a B-uncovered pair (e, v), all B-ancestors of v receive this copy). Since B represents a directed forest, the new set family with respect to copies is laminar. For example, consider the set family Φ in Figure 2(a) and the branching B in Figure 2(b). As depicted in Figure 3, by replacing each element $e \in \{a, b, c, d\}$ with $|U_B(e)|$ copies, denoted by $e_1, e_2, \ldots, e_{|U_B(e)|}$, Φ is transformed to a laminar one. Clearly, any feasible "copying" scheme also determines a branching as in every laminar set family, each element is only contained in a chain. Therefore, copying elements to "laminarize" Φ is exactly UB.

Let $\delta(\Phi) = \beta(\Phi) - |\Sigma|$. In this interpretation, $\delta(\Phi)$ is the minimum number of additional copies of the elements needed to make Φ laminar, and our new algorithm for UB in section 4 is an FPT algorithm with respect to the number of new copies needed. Note that since $\alpha(M) = \beta(\Phi_M)$ and $|\Sigma_M| = m$, we have $\varepsilon(M) = \alpha(M) - m = \beta(\Phi_M) - |\Sigma_M| = \delta(\Phi_M)$.

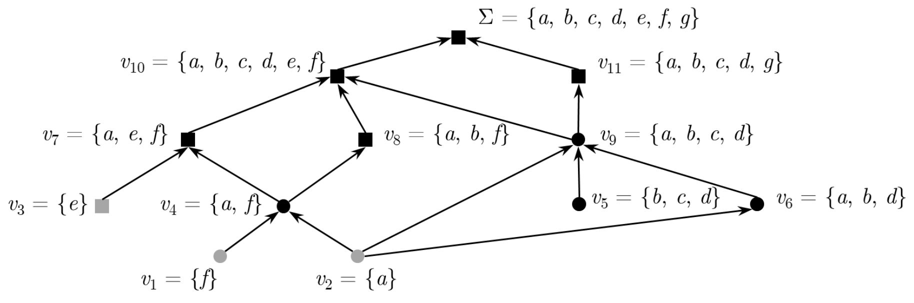
3. An upper bound on $t(\Phi)$.

This section gives an upper bound on $t(\Phi)$, which is the number of vertices of D_Φ that are not trivial and not regular. More precisely, we show that $t(\Phi) \leq 2 \cdot \delta(\Phi) - 1$. Recall that this upper bound is used to estimate the worst-case complexity of our UB algorithm, which requires $O^*(2^{t(\Phi)})$ time. To establish the bound, we assume that D_Φ contains at least one nonregular vertex; for otherwise, $t(\Phi) = 0$ and our algorithm in section 4 solves UB in $O(mn^2)$ time.

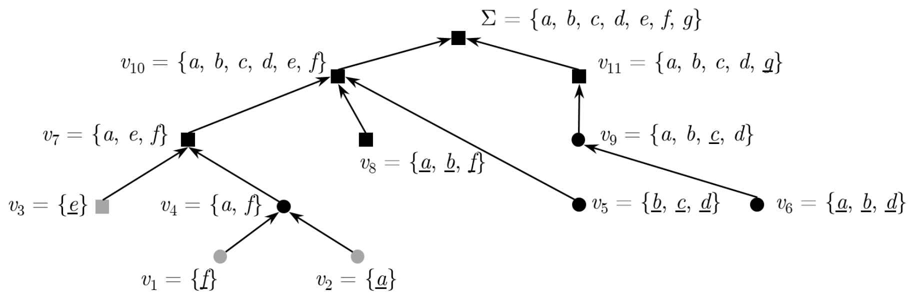
For ease of presentation, in the remainder of this paper, we assume that $\Sigma \in \Phi$. Suppose that this is not true. We simply add Σ into Φ. Denote Φ^+ to be the resulting family. It is easy to observe that given a branching B^+ of D_{Φ^+}, we can obtain a branching B of D_Φ with $U(B^+) \supseteq U(B)$ by deleting all arcs directed to Σ, and given a branching B of D_Φ, we can obtain a branching B^+ of D_{Φ^+} with $U(B^+) = U(B)$ by adding, for each vertex v with $p_B(v) = \text{null}$, an arc from v to Σ. Accordingly, we know that $\beta(\Phi^+) = \beta(\Phi)$.

Consider two vertices $u, v \in \Phi$. The relation between u and v is classified into three types: u and v are *disjoint* if $u \cap v = \emptyset$; u and v are *nested* if $u \subset v$ or $v \subset u$; and otherwise, u and v are *in conflict*. For convenience, we say that u is *compatible* with v if they are not in conflict. The set Σ contains all the other sets in Φ. We call Σ the *root vertex* of D_Φ. Clearly, in the finding of an optimal branching, we may consider only inclusionwise maximal branchings. In other words, we may consider only branchings B such that F_B is a tree rooted at Σ.

The *size* of a vertex v, denoted by $|v|$, is the number of elements in v. Trivial vertices and regular vertices are formally defined as follows.



(a) *D_Φ of a family Φ over $\Sigma = \{a, b, c, d, e, f, g\}$, in which only elementary arcs are depicted*



$\Sigma = \{a, b, c, d, e, f, g\}$
$v_{10} = \{a, b, c, d, e, f\}$
$v_{11} = \{a, b, c, d, \underline{g}\}$
$v_7 = \{a, e, f\}$
$v_9 = \{a, b, \underline{c}, d\}$
$v_8 = \{\underline{a}, \underline{b}, \underline{f}\}$
$v_3 = \{\underline{e}\}$
$v_4 = \{a, f\}$
$v_5 = \{\underline{b}, \underline{c}, \underline{d}\}$
$v_6 = \{\underline{a}, \underline{b}, \underline{d}\}$
$v_1 = \{\underline{f}\}$
$v_2 = \{\underline{a}\}$

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

FIG. 4. *An illustrative example, in which each regular vertex is represented by a square and each trivial vertex is in gray.*

DEFINITION 3.1. *A vertex $v \in \Phi$ is* trivial *if $|v| = 1$.*

DEFINITION 3.2. *A vertex $v \in \Phi$ is not* regular *(or* nonregular*) if it is contained in both of a pair of conflicting vertices.*

To illustrate these definitions, Figure 4(a) depicts the containment digraph D_Φ of a set family Φ over $\Sigma = \{a,b,c,d,e,f,g\}$. 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_Φ. 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 4(a), D_Φ 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 nonelementary arcs is simply for clarity of illustration and a branching may contain nonelementary arcs. For example, the arc (v_5,v_{10}) in Figure 4(b) is nonelementary. In Figure 4(a), the vertex v_5 is nonregular, since v_{10} and v_{11} both contain it and are in conflict, and the vertices v_1, v_2, and v_3 are trivial. In particular, the root vertex Σ is regular, since no vertex contains Σ. Note that Σ is nontrivial, since it has been assumed that Φ contains more than one set.

For a branching B, let $U_{\rm NR}(B)$ denote the set of B-uncovered pairs (e,v) such that v is nonregular. For example, in Figure 4(b), $(a,v_6) \in U_{\rm NR}(B)$ and $(a,v_8) \notin U_{\rm NR}(B)$. Let B^* be an optimal branching. The upper bound is established in two steps. In the first step, we show that $|U_{\rm NR}(B^*)| \le 2 \cdot \delta(\Phi)$; in the second step, we show that $t(\Phi) \le |U_{\rm NR}(B^*)| - 1$. We proceed to the first step. For a branching B, the *split set* of

B, denoted by $\Sigma_{\mathrm{S}}(B)$, is the set of elements $e \in \Sigma$ such that $|U_B(e)| \geq 2$. For example, in Figure 4(b), $a \in \Sigma_{\mathrm{S}}(B)$, since $|U_B(a)| = |\{(a, v_2), (a, v_6), (a, v_8)\}| \geq 2$. The following lemma says that the number of B^*-uncovered pairs contributed by the elements of $\Sigma_{\mathrm{S}}(B^*)$ is bounded by $2 \cdot \delta(\Phi)$.

LEMMA 3.3. *Let B^* be an optimal branching. Then, $\sum_{e \in \Sigma_{\mathrm{S}}(B^*)} |U_{B^*}(e)| \leq 2 \cdot \delta(\Phi)$.*

Proof. As mentioned, $|U_B(e)| \geq 1$ for each element $e \in \Sigma$. Thus, each element e of Σ has $|U_{B^*}(e)| = 1$ if $e \notin \Sigma_{\mathrm{S}}(B^*)$. Since B^* is optimal, we have $\beta(\Phi) = |U(B^*)| = \sum_{e \in \Sigma} |U_{B^*}(e)| = \sum_{e \in \Sigma}(|U_{B^*}(e)| - 1) + |\Sigma| = \sum_{e \in \Sigma_{\mathrm{S}}(B^*)}(|U_{B^*}(e)| - 1) + |\Sigma|$. Since $\delta(\Phi) = \beta(\Phi) - |\Sigma|$, we have $\delta(\Phi) = \sum_{e \in \Sigma_{\mathrm{S}}(B^*)}(|U_{B^*}(e)| - 1)$. Each $e \in \Sigma_{\mathrm{S}}(B^*)$ contributes at least one to $\delta(\Phi)$. Thus, $|\Sigma_{\mathrm{S}}(B^*)| \leq \delta(\Phi)$. As a result,

$$\begin{aligned} \sum_{e \in \Sigma_{\mathrm{S}}(B^*)} |U_{B^*}(e)| &= \sum_{e \in \Sigma_{\mathrm{S}}(B^*)} (|U_{B^*}(e)| - 1) + \sum_{e \in \Sigma_{\mathrm{S}}(B^*)} 1 \\ &= \delta(\Phi) + |\Sigma_{\mathrm{S}}(B^*)| \leq \delta(\Phi) + \delta(\Phi) = 2 \cdot \delta(\Phi). \end{aligned}$$

Thus, the lemma holds. ∎

LEMMA 3.4. *If an element e is in both of two conflicting vertices, then $|U_B(e)| \geq 2$ for any branching B.*

Proof. Let e be an element that is in both of two conflicting vertices u and w. Consider the rooted forest F_B. Since $e \in u$, u has a descendant u' such that (e, u') is a B-uncovered pair. Similarly, w has a descendant vertex w' such that (e, w') is a B-uncovered pair. Since u and w are in conflict, u is not a B-ancestor of w and vice versa. Thus, they do not have any common descendant. As a result, $(e, u') \neq (e, w')$. Thus, $|U_B(e)| \geq 2$ and the lemma holds. ∎

Let Σ_{NR} be the set of elements that are contained in a nonregular vertex. For example, in Figure 4, we have $\Sigma_{\mathrm{NR}} = v_1 \cup v_2 \cup v_4 \cup v_5 \cup v_6 \cup v_9 = \{a, b, c, d, f\}$. The following lemma shows that $\Sigma_{\mathrm{NR}} \subseteq \Sigma_{\mathrm{S}}(B)$ for any branching B.

LEMMA 3.5. *Let B be a branching. For all elements $e \in \Sigma_{\mathrm{NR}}$, we have $|U_B(e)| \geq 2$.*

Proof. Let $e \in \Sigma_{\mathrm{NR}}$ be an element. Since e is in a nonregular vertex, it is in both of a pair of conflicting vertices. By Lemma 3.4, $|U_B(e)| \geq 2$ and hence the lemma holds. ∎

Recall that $U_{\mathrm{NR}}(B)$ denotes the set $\{(e, v) \in U(B) \mid v \text{ is nonregular}\}$. Lemmas 3.3 and 3.5 lead to the following.

LEMMA 3.6. *For any optimal branching B^*, we have $|U_{\mathrm{NR}}(B^*)| \leq 2 \cdot \delta(\Phi)$.*

Proof. Consider an optimal branching B^*. Each (e, v) in $U_{\mathrm{NR}}(B^*)$ has v being nonregular and e being an element of v. Thus, each (e, v) in $U_{\mathrm{NR}}(B^*)$ has e being an element of a nonregular vertex. As a result, we can equivalently define $U_{\mathrm{NR}}(B^*)$ as the set of uncovered pairs (e, v) such that $e \in \Sigma_{\mathrm{NR}}$ and v is nonregular. Let U' be the set of B^*-uncovered pairs (e, v) with $e \in \Sigma_{\mathrm{NR}}$. By definition, we have $U_{\mathrm{NR}}(B^*) \subseteq U'$. Let U'' be the set of B^*-uncovered pairs (e, v) with $e \in \Sigma_{\mathrm{S}}(B^*)$. By Lemma 3.5, $\Sigma_{\mathrm{NR}} \subseteq \Sigma_{\mathrm{S}}(B^*)$ and thus we have $U' \subseteq U''$. By Lemma 3.3, $|U''| = \sum_{e \in \Sigma_{\mathrm{S}}(B^*)} |U_{B^*}(e)| \leq 2 \cdot \delta(\Phi)$. Consequently, we obtain $|U_{\mathrm{NR}}(B^*)| \leq |U'| \leq |U''| \leq 2 \cdot \delta(\Phi)$. Thus, the lemma holds. ∎

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

LEMMA 3.7. *If a vertex v is regular, all vertices containing v are regular.*

Proof. Let v be a regular vertex. Consider a fixed vertex $u \supset v$. Suppose, for proof by contradiction, that u is nonregular. 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 3.7 is as follows. Let V_{NR} be the set of nonregular vertices and V_{R} be the set of regular vertices. Then, there is no arc of D_Φ which starts from a vertex of V_{R} and ends at a vertex of V_{NR}. Consider a branching B. According to F_B, we partition V_{NR} into the following subsets:

$L_1(B)$: the set of nonregular vertices v with in-degree $=0$ and $|v|=1$;

$L_2(B)$: the set of nonregular vertices v with in-degree $=0$ and $|v|>1$;

$N_1(B)$: the set of nonregular vertices v with in-degree $=1$; and

$N_2(B)$: the set of nonregular vertices v with in-degree >1.

For example, in Figure 4(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 *nonregular leaves* of F_B and vertices in $N_1(B)\cup N_2(B)$ are *nonregular internal nodes* of F_B. The following lemma shows that $|N_2(B)|$, the number of nonregular internal nodes with in-degree more than one, is bounded by the number of nonregular leaves.

LEMMA 3.8. *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 nonregular vertices and the arcs connecting pairs of them. By Lemma 3.7, all children of a nonregular vertex are nonregular. 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 [4]. 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 3.9. *For any branching B, we have $|U_{\mathrm{NR}}(B)| \ge |N_1(B)| + |L_1(B)| + 2\cdot |L_2(B)|$.*

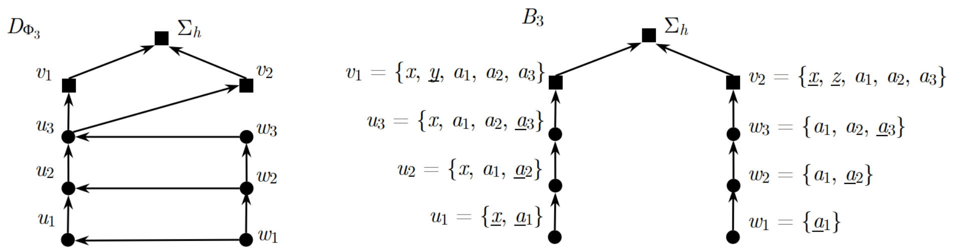
Proof. For any $v \in L_1(B)\cup L_2(B)$, since v has no B-child, all elements of v are B-uncovered. Therefore, there are at least $|L_1(B)|+2\cdot|L_2(B)|$ B-uncovered pairs (e,v) with $v\in L_1(B)\cup L_2(B)$. Consider a nonregular vertex $v\in N_1(B)$. By definition, v has exactly one B-child, say, 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 (e,v) with $v\in N_1(B)$. Consequently, we have $|U_{\mathrm{NR}}(B)| \ge |N_1(B)|+|L_1(B)|+2\cdot|L_2(B)|$ and thus the lemma holds. ∎

Recall that $t(\Phi)$ is the number of vertices of D_Φ that are not trivial and not regular. We are ready to prove that $t(\Phi) \le |U_{\mathrm{NR}}(B^*)|-1$.

LEMMA 3.10. *For any optimal branching B^*, we have $t(\Phi) \le |U_{\mathrm{NR}}(B^*)|-1$.*

Proof. Let B^* be an optimal branching. Since all vertices in $L_1(B^*)$ are trivial, we have

$$\begin{aligned} t(\Phi) &= |N_1(B^*)| + |N_2(B^*)| + |L_2(B^*)| && \text{(by the definition of } t(\Phi)) \\ &\le |N_1(B^*)| + |L_1(B^*)| + 2\cdot|L_2(B^*)| - 1 && \text{(by Lemma 3.8)} \\ &\le |U_{\mathrm{NR}}(B^*)| - 1 && \text{(by Lemma 3.9).} \end{aligned}$$ ∎



D_{Φ_3}
Σ_h
v_1
v_2
u_3
u_2
u_1
w_3
w_2
w_1
B_3
Σ_h
$v_1 = \{x, \underline{y}, a_1, a_2, a_3\}$
$v_2 = \{\underline{x}, \underline{z}, a_1, a_2, a_3\}$
$u_3 = \{x, a_1, a_2, \underline{a_3}\}$
$u_2 = \{x, a_1, \underline{a_2}\}$
$u_1 = \{\underline{x}, \underline{a_1}\}$
$w_3 = \{a_1, a_2, \underline{a_3}\}$
$w_2 = \{a_1, \underline{a_2}\}$
$w_1 = \{\underline{a_1}\}$

FIG. 5. D_{Φ_3} *(only elementary arcs are depicted) and* B_3.

Lemmas 3.6 and 3.10 immediately lead to the following.

THEOREM 3.11. $t(\Phi) \le 2 \cdot \delta(\Phi) - 1$.

The following theorem gives an upper bound on the number of nonregular vertices, which is needed for analyzing the time complexity of our DSR algorithm.

THEOREM 3.12. $|V_{\mathrm{NR}}| \le 3 \cdot \delta(\Phi) - 1$.

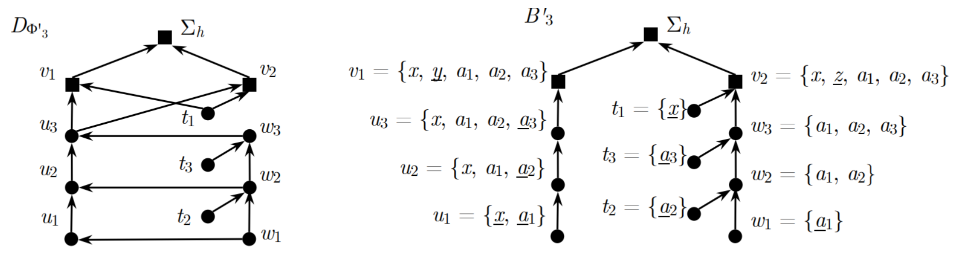
Proof. Let W be the set of nonregular trivial vertices and let $\Sigma_W = \bigcup_{v \in W} v$. Vertices in W are trivial and distinct. Thus, we have $|\Sigma_W| = |W|$. Since $|V_{\mathrm{NR}}| = t(\Phi) + |W| = t(\Phi) + |\Sigma_W|$ and $t(\Phi) \le 2 \cdot \delta(\Phi) - 1$, it suffices to show that $|\Sigma_W| \le \delta(\Phi)$. Consider an optimal branching B^*. By Lemma 3.5, $\Sigma_{\mathrm{NR}} \subseteq \Sigma_{\mathrm{S}}(B^*)$. As shown in the proof of Lemma 3.3, $|\Sigma_{\mathrm{S}}(B^*)| \le \delta(\Phi)$. Since $\Sigma_W \subseteq \Sigma_{\mathrm{NR}} \subseteq \Sigma_{\mathrm{S}}(B^*)$, we have $|\Sigma_W| \le |\Sigma_{\mathrm{S}}(B^*)| \le \delta(\Phi)$ and thus the theorem holds. □

To conclude this section, we remark that the upper bounds in Theorems 3.11 and 3.12 are almost tight. We first show that for any integer $h \ge 1$, a set family Φ_h with $\delta(\Phi_h) = h + 1$ and $t(\Phi_h) = 2 \cdot \delta(\Phi_h) - 3$ can be constructed. Consider a fixed $h \ge 1$. Define Φ_h as a set family over a set Σ_h as follows:

(1) the ground set Σ_h consists of $h + 3$ elements $x, y, z, a_1, a_2, a_3, \ldots, a_h$; and
(2) the set family Φ_h consists of $2h + 3$ sets $\Sigma_h, v_1, v_2, u_1, u_2, u_3, \ldots, u_h, w_1, w_2, w_3, \ldots, w_h$, where $v_1 = \Sigma_h - \{z\}$, $v_2 = \Sigma_h - \{y\}$, $u_i = \{x, a_1, a_2, \ldots, a_i\}$ for $i = 1, 2, \ldots, h$, and $w_i = \{a_1, a_2, \ldots, a_i\}$ for $i = 1, 2, \ldots, h$.

Figure 5 depicts D_{Φ_h} for $h = 3$. As in Figure 4, regular vertices are represented by squares. Note that v_1 and v_2 are in conflict. The vertices Σ_h, v_1, v_2 are regular. All the other vertices are contained in both of v_1 and v_2 and thus are nonregular. Only the vertex w_1 is trivial. Thus, we have $t(\Phi_h) = 2 \cdot h - 1$. Let B_h be the branching such that $p_{B_h}(v_1) = p_{B_h}(v_2) = \Sigma_h$, $p_{B_h}(u_h) = v_1$, $p_{B_h}(w_h) = v_2$, and for $i = 1, 2, \ldots, h - 1$, $p_{B_h}(w_i) = w_{i+1}$ and $p_{B_h}(u_i) = u_{i+1}$. See Figure 5 for B_3. As in Figure 4, uncovered elements are underlined. In B_h, each element in $\Sigma_h - \{y, z\}$ 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 $\Sigma_h - \{y, z\}$ are contained in the nonregular vertex u_h, by Lemma 3.5, each of them contributes at least two B-uncovered pairs. Moreover, each of y and z contributes at least one B-uncovered pair. Thus, $|U(B)|$ is at least $2 \cdot (|\Sigma_h| - 2) + 2 = 2 \cdot h + 4$. Therefore, B_h is optimal, and we have $\delta(\Phi_h) = |U(B_h)| - |\Sigma_h| = h + 1$ and $t(\Phi_h) = 2 \cdot h - 1 = 2 \cdot \delta(\Phi_h) - 3$.

We proceed to show that the upper bound in Theorem 3.12 is almost tight. More specifically, we show that for any integer $h \ge 1$, a set family Φ'_h with $\delta(\Phi'_h) = h + 1$ and $|V_{\mathrm{NR}}| = 3 \cdot \delta(\Phi'_h) - 3$ can be constructed. (See Figure 6 for an example.) Consider



$D_{\Phi'_3}$
Σ_h
v_1
v_2
u_3
t_1
w_3
u_2
t_3
w_2
u_1
t_2
w_1
B'_3
Σ_h
$v_1 = \{x, \underline{y}, a_1, a_2, a_3\}$
$v_2 = \{x, \underline{z}, a_1, a_2, a_3\}$
$t_1 = \{\underline{x}\}$
$u_3 = \{x, a_1, a_2, \underline{a_3}\}$
$w_3 = \{a_1, a_2, a_3\}$
$t_3 = \{\underline{a_3}\}$
$u_2 = \{x, a_1, \underline{a_2}\}$
$w_2 = \{a_1, a_2\}$
$t_2 = \{\underline{a_2}\}$
$u_1 = \{\underline{x}, \underline{a_1}\}$
$w_1 = \{\underline{a_1}\}$

FIG. 6. $D_{\Phi'_3}$ *(only elementary arcs are depicted) and* B'_3.

a fixed $h \geq 1$. Let Φ'_h be the family obtained from Φ_h by adding h sets $t_1, t_2, \ldots, t_h$, where $t_1 = \{x\}$ and $t_i = \{a_i\}$ for $i = 2, 3, \ldots, h$. Since each t_i is contained in both v_1 and v_2, it is nonregular. Thus, for the digraph $D_{\Phi'_h}$, we have $|V_{\text{NR}}| = |\{u_1, u_2, \ldots, u_h, w_1, w_2, \ldots, w_h, t_1, t_2, \ldots, t_h\}| = 3h$. 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 $D_{\Phi'_h}$. 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 $D_{\Phi'_h}$. Thus, $\delta(\Phi'_h) = |U(B'_h)| - |\Sigma_h| = h + 1$. As a result, $|V_{\text{NR}}| = 3h = 3\delta(\Phi'_h) - 3$.

4. An algorithm for the branching formulation.

Let $m = |\Sigma|$ and $n = |\Phi|$. This section presents an $O^*(2^{\min(|\Phi|, 2\delta(\Phi))})$-time algorithm for UB. Recall that our algorithm consists of three phases: Phase 1 removes trivial vertices, Phase 2 predetermines the parent of each regular vertex, and Phase 3 finds an optimal branching using dynamic programming. The containment digraph D_Φ is precomputed, which requires $O(mn^2)$ time [14].

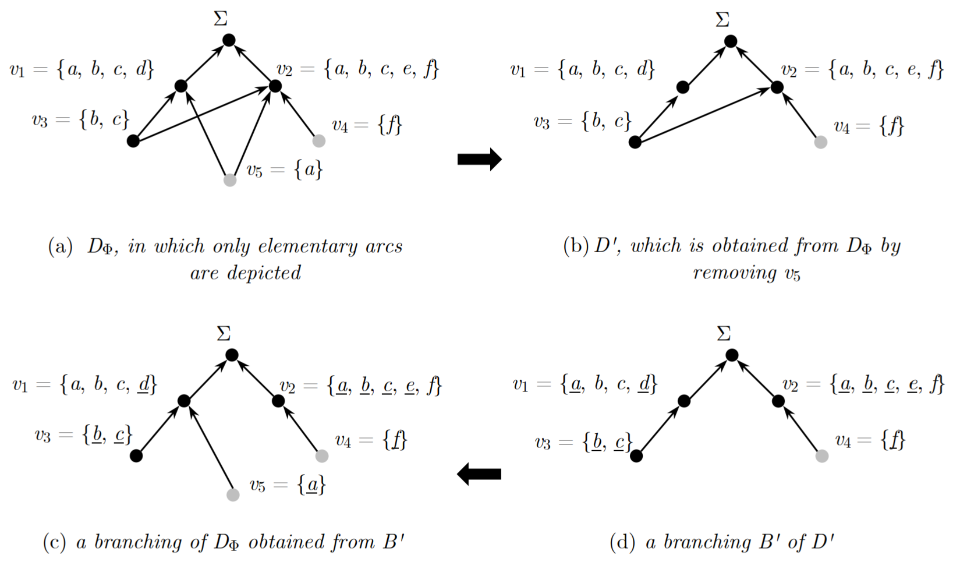
4.1. Phase 1.

For a trivial vertex v, we define the *preferred-parent* of v to be the vertex u such that $u \supset v$ and its size $|u|$ is smallest (with ties broken arbitrarily). In Figure 7(a), v_5 is contained in Σ, v_1, and v_2; and the preferred-parent of v_5 is v_1. The following theorem says that trivial vertices can be removed without changing the cost of an optimal branching. (See Figure 7 for an example.)

THEOREM 4.1. *Let $v = \{e\}$ be a trivial vertex of D_Φ and let D' be the digraph obtained from D_Φ by removing v and all its incident arcs. Let β' be the cost of an optimal branching of D'. Then, $\beta' = \beta(\Phi)$.*

Proof. Let B be an optimal branching of D_Φ. Recall that in the finding of an optimal branching, we may assume that F_B is a tree. Let p be the parent of v in F_B. By deleting the arc (v, p) from B, we obtain a branching B' of D'. Since v is a leaf of F_B, (e, v) is B-uncovered. The child sets of p in B and B' differ only in v; and for every vertex in $\Phi - \{v, p\}$, its child sets in B and B' are the same. As a result, if (e, p) is B'-uncovered, $U(B') = U(B) - \{(e, v)\} \cup \{(e, p)\}$; otherwise, $U(B') = U(B) - \{(e, v)\}$. In either case, $|U(B')| \leq |U(B)|$. Hence, $\beta' \leq |U(B')| \leq |U(B)| = \beta(\Phi)$.

We proceed to show that $\beta(\Phi) \leq \beta'$. Let B' be an optimal branching of D'. Let u be the preferred-parent of v. We obtain from B' a branching B of D_Φ by adding the arc (v, u). (See Figure 7(c), (d).) Since $|v| = 1$ and u is the smallest vertex strictly containing v, we know that (e, u) is B'-uncovered. As a result, it is easy to conclude that $U(B) = U(B') - \{(e, u)\} \cup \{(e, v)\}$. Therefore, $\beta(\Phi) \leq |U(B)| = |U(B')| = \beta'$. This completes the proof. ∎



Σ
$v_1 = \{a, b, c, d\}$
$v_2 = \{a, b, c, e, f\}$
$v_3 = \{b, c\}$
$v_4 = \{f\}$
$v_5 = \{a\}$
(a) D_Φ, in which only elementary arcs are depicted
(b) D', which is obtained from D_Φ by removing v_5
(c) a branching of D_Φ obtained from B'
(d) a branching B' of D'

Fig. 7. *An illustrative example, in which trivial vertices are in gray.*

We remark that the proof of Theorem 4.1 also exhibits that given the preferred-parent of v, we can obtain in $O(1)$ time an optimal branching of D_Φ from an optimal branching of D'.

Our Phase 1 algorithm works as follows. First, find all trivial vertices. Next, for each trivial vertex v, compute its preferred-parent as the vertex u such that (v, u) is an arc of D_Φ and $|u|$ is smallest. The preferred-parent of each trivial vertex is stored so that an optimal solution of the original digraph can be efficiently recovered from an optimal branching of the reduced digraph. Finally, remove all trivial vertices. Since D_Φ contains n vertices and $O(n^2)$ arcs, it is easy to implement Phase 1 in $O(n^2)$ time.

4.2. Phase 2.

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

Lemma 4.2. *Let v be a vertex and $(u_1, u_2, \ldots, u_{s-1}, u_s = \Sigma)$ be the sequence of vertices strictly containing v, in nondecreasing order of their sizes. Then, v is regular if and only if $u_1 \subset u_2 \subset \cdots \subset u_s$.*

Proof. If $u_1 \subset u_2 \subset \cdots \subset u_s$, any pair of vertices strictly containing v are not in conflict and thus v is regular. Assume that v is regular. Consider two vertices u_j and u_{j+1}. Since they both contain v, they are not disjoint. In addition, since v is regular, they are not in conflict. As a result, they are nested. Therefore, the lemma holds. ◻

Lemma 4.2 indicates that no two vertices containing a regular vertex v have the same size. Recall that V_R is the set of regular vertices. For each vertex $v \in V_\mathrm{R}$, the *predetermined parent* of v, denoted by $\pi(v)$, is the smallest vertex strictly containing v. Note that $\pi(v)$ is null if and only if v is the root vertex Σ. Let B_R be the set of arcs $\{(v, \pi(v)) \mid v \in V_\mathrm{R}$ and $v \neq \Sigma\}$. For the example in Figure 4(a), $B_\mathrm{R} = \{(v_7, v_{10}), (v_8, v_{10}), (v_{10}, \Sigma), (v_{11}, \Sigma)\}$. Let T_R be the digraph with vertex set V_R and arc set B_R. By Lemma 3.7, all vertices containing a regular vertex are regular. Thus,

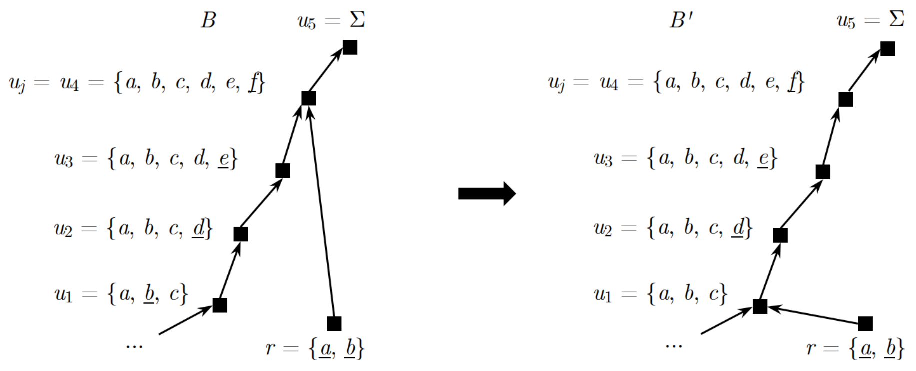

B
$u_5 = \Sigma$
$u_j = u_4 = \{a, b, c, d, e, f\}$
$u_3 = \{a, b, c, d, \underline{e}\}$
$u_2 = \{a, b, c, \underline{d}\}$
$u_1 = \{a, \underline{b}, c\}$
$\cdots$
$r = \{\underline{a}, \underline{b}\}$
B'
$u_5 = \Sigma$
$u_j = u_4 = \{a, b, c, d, e, f\}$
$u_3 = \{a, b, c, d, \underline{e}\}$
$u_2 = \{a, b, c, \underline{d}\}$
$u_1 = \{a, b, c\}$
$\cdots$
$r = \{\underline{a}, \underline{b}\}$

FIG. 8. *An illustration for Case* 1 *in the proof of Lemma* 4.4, *in which* $s = 5$ *and* $j = 4$.

each arc in B_R connects a pair of two regular vertices, from which we conclude that T_R is a tree rooted at Σ. Let $(u_1, u_2, \ldots, u_{s-1}, u_s = \Sigma)$ be the sequence of vertices strictly containing a regular vertex v, in nondecreasing order of their sizes. By definition, $\pi(v) = u_1$. Consider a fixed vertex u_j, $1 \leq j < s$. A vertex containing u_j also contains v_i. Therefore, u_{j+1} is the smallest vertex strictly containing u_j and thus we have $\pi(u_j) = u_{j+1}$. Consequently, we obtain the following.

LEMMA 4.3. *Let v be a regular vertex. The path from v's parent to the root Σ in T_R contains the sequence of vertices strictly containing v, in nondecreasing order of their sizes.*

Lemma 4.3 indicates that a vertex u contains a regular vertex v if and only if u is on the path from v's parent to the root in T_R. Clearly, B_R is a branching of D_Φ. 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 r be the largest regular vertex such that $p_B(r) \neq \pi(r)$ (with ties broken arbitrarily). Since $p_B(\Sigma) = \pi(\Sigma) = $ null, $r \neq \Sigma$ and thus $\pi(r) \neq$ null. A *regularization operation* on B is defined as reassigning $p_B(r)$ to be $\pi(r)$.

LEMMA 4.4. *Let B' be the branching obtained from a branching B by performing a regularization operation. Then, $U(B) \supseteq U(B')$.*

Proof. Let r be the regular vertex whose parent is reassigned by the regularization operation. Two cases are considered.

Case 1: $p_B(r) \neq$ null. (See Figure 8.) Let $W = (u_1, u_2, \ldots, u_{s-1}, u_s = \Sigma)$ be the sequence of vertices strictly containing r, in nondecreasing order of their sizes. By Lemma 4.3, we have $\pi(r) = u_1$ and $p_B(u_j) = \pi(u_j) = u_{j+1}$ for $1 \leq j < s$. Since $p_B(r)$ strictly contains r, $p_B(r) \neq u_1$, and W contains all vertices which strictly contain r, we know that there is a vertex $u_j = p_B(r)$, where $1 < j \leq s$. Since u_{j-1} is a B-child of u_j and $u_{j-1} \supset r$, deleting the arc (r, u_j) from B does not induce any new uncovered element in u_j. As a result, $U(B) \supseteq U(B')$ can be easily concluded.

Case 2: $p_B(r) = $ 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 4.4, this transformation does not increase the number of uncovered pairs. Therefore, we have the following.

THEOREM 4.5. *There exists an optimal branching B of D_Φ such that $B \supseteq B_{\mathrm{R}}$.*

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_Φ, in nondecreasing order of their sizes. Next, for each vertex v, we determine whether it is regular by extracting from L the sequence $(u_1, u_2, \ldots, u_{s-1}, u_s = \Sigma)$ of vertices strictly containing v and then check whether there is an arc (u_j, u_{j+1}) for $1 \le j < s$, and if v is regular, we set $\pi(v) = u_1$. The correctness is ensured by Lemma 4.2. 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 $O(mn2^{\min(n,2\delta(\Phi))} + n^2 3^{\min(n,2\delta(\Phi))})$ or $O(mn^5 \lg mn \lg\lg mn 2^{\min(n,2\delta(\Phi))})$, depending on the implementation. Recall that V_{R} and V_{NR} are, respectively, the sets of regular and nonregular vertices. We first topologically sort the vertices of D_Φ into a sequence $(v_1, v_2, \ldots, v_n)$, so that (v_i, v_j) is an arc only if $i < j$. For $1 \le i \le n$, let $V_i = \{v_j \mid j \le i\}$ and $D_\Phi(V_i)$ be the subgraph of D_Φ induced by V_i. Consider a fixed V_i. A *branching on* V_i is a branching of $D_\Phi(V_i)$, in which the parent of each vertex $v \in V_i$ is either null or a vertex in V_i. For a branching B and a vertex $v_j \in \Phi$, define U_B^j to be the set of B-uncovered elements in v_j. The *cost* of a branching B on V_i, denoted by $cost_i(B)$, is the number of uncovered pairs (e, v_j) with $v_j \in V_i$. That is, $cost_i(B) = \sum_{1 \le j \le i} |U_B^j|$.

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 nonregular 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) = $ null. We say that a nonregular vertex v_j is *assigned* in B if $p_B(v_j) \neq$ null.

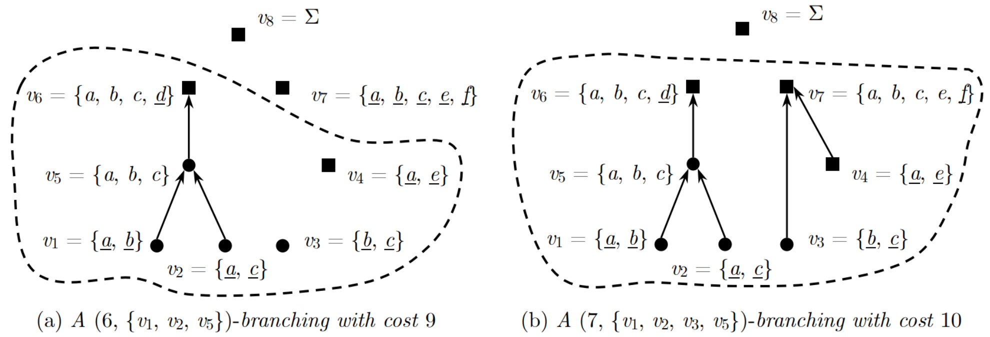
Consider a fixed integer $i \in \{1, 2, \ldots, n\}$. Let S be a subset of V_{NR}. An (i, S)-*branching* is a branching B such that

(1) B is a branching on V_i;
(2) B obeys B_{R}; and
(3) the set of nonregular 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 is the vertex set of D_Φ, a branching on V_n is a branching of D_Φ and thus $f[n, V_{\mathrm{NR}}] = \beta(\Phi)$. We proceed to derive a recurrence for $f[i, S]$. Clearly, if $i = 1$, we have $f[i, \emptyset] = |v_1|$ and $f[i, S] = \infty$ for $S \neq \emptyset$. Consider the case $i \ge 2$. 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 nonregular B-children of v_i. We have the following.

LEMMA 4.6. *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 $\sum_{1 \le j \le i} |U_B^j| = |U_B^i| + \sum_{1 \le j \le i-1} |U_B^j| = |U_B^i| + cost_{i-1}(B_2)$. Note that $|U_B^i| = |v_i| - |\bigcup_{c \in H} c|$, 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



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

FIG. 9. *Two illustrative examples.*

in B, we obtain an (i, S)-branching $X \cup B_1$ with less cost $|U^i_{B_1}| + \sum_{1\le j\le i-1} |U^j_X| = |U^i_B| + cost_{i-1}(X)$, which contradicts the optimality of B. Therefore, B_2 is optimal and the lemma holds. ☐

Let H^i_{R} be the set of B_{R}-children of v_i. Since B obeys B_{R}, we have $H = H^i_{\mathrm{R}} \cup H_{\mathrm{NR}}$ and $|U^i_B| = |v_i| - |\bigcup_{c\in H} c| = |v_i| - |\bigcup_{c\in H^i_{\mathrm{R}}\cup H_{\mathrm{NR}}} c|$. Therefore, according to Lemma 4.6, $f[i, S] = f[i-1, S-H_{\mathrm{NR}}] + |v_i| - |\bigcup_{c\in H^i_{\mathrm{R}}\cup H_{\mathrm{NR}}} c|$. Consequently, we obtain

$$\begin{aligned} f[i,S] &= \min\Bigg\{ f[i-1, S-H_{\mathrm{NR}}] \\ &\qquad + |v_i| - \Bigg|\bigcup_{c\in H^i_{\mathrm{R}}\cup H_{\mathrm{NR}}} c\Bigg| \;\Bigg|\; H_{\mathrm{NR}} \subseteq S \text{ and } \bigcup_{c\in H_{\mathrm{NR}}} c \subseteq v_i \Bigg\} \\ &= \min\{ f[i-1, S-T] + g[i,T] \mid T \subseteq S \}, \end{aligned} \tag{4.1}$$

where $g[i,T] = |v_i| - |\bigcup_{c\in H^i_{\mathrm{R}}\cup T} c|$ if $\bigcup_{c\in T} c \subseteq v_i$ 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 $f[i-1, S-T] + g[i,T] \ne \infty$ for some subset $T \subseteq S$. Since $f[i-1, S-T] \ne \infty$, there exists an $(i-1, S-T)$-branching X. Since $g[i,T] \ne \infty$, we have $\bigcup_{c\in T} c \subseteq v_i$. 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 $v \in H^i_{\mathrm{R}} \cup T$. As a result, $X \cup \{(c, v_i) \mid c \in H^i_{\mathrm{R}} \cup T\}$ 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 given in Algorithm 4.1.

The time complexity is analyzed as follows. Lines 1–3 require $O(n^2 + 2^{|V_{\mathrm{NR}}|})$ time. Note that after Phase 1, all nonregular vertices are not trivial and thus we have $|V_{\mathrm{NR}}| = t(\Phi)$. Consider a fixed iteration of the for-loop in lines 4–17. Each vertex $v \in V_{\mathrm{NR}}$ contains at most m elements. Thus, the computation of $g[i,\cdot]$ in line 6 can be implemented in $O(m2^{|V_{\mathrm{NR}}|})$ 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 $S \subseteq V_{\mathrm{NR}}$. Each set operation on two subsets of V_{NR} takes $O(|V_{\mathrm{NR}}|)$ time. Thus, for a fixed i, the total time spent by lines 13–14 is $O(|V_{\mathrm{NR}}| \times \sum_{S\subseteq V_{\mathrm{NR}}} 2^{|S|}) = O(|V_{\mathrm{NR}}| \times \sum_{0\le k\le |V_{\mathrm{NR}}|} C(|V_{\mathrm{NR}}|, k)2^k)$, which is $O(|V_{\mathrm{NR}}| \times 3^{|V_{\mathrm{NR}}|})$ by the binomial theorem [5]. As a result, the for-loop in lines 4–17 requires $O(mn2^{|V_{\mathrm{NR}}|} + n|V_{\mathrm{NR}}|3^{|V_{\mathrm{NR}}|})$ time. Clearly, lines 19–26 take $O(n|V_{\mathrm{NR}}|)$ time. Consequently, the

Algorithm 4.1 Phase 3—Implementation 1.

```
Input: $D_\Phi$ and $B_R$
Output: an optimal branching $B^*$
 1: topologically sort the vertices in $D_\Phi$
 2: for each $S \subseteq V_{NR}$ do $f[1,S] \leftarrow \infty$
 3: $f[1,\emptyset] \leftarrow |v_1|$
 4: for $i = 2,3,\dots,n$ 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: $B^* \leftarrow \emptyset$
20: $S^* \leftarrow V_{NR}$
21: for $i = n, n-1, \dots, 2$ 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 $B^*$
```

overall time complexity is $O(mn2^{|V_{NR}|} + n|V_{NR}|3^{|V_{NR}|} + n|V_{NR}|)$, which is $O(mn2^{t(\Phi)} + nt(\Phi)3^{t(\Phi)})$ when substituting $t(\Phi)$ for $|V_{NR}|$.

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

LEMMA 4.7 ([2, 5]). *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 $\{-X, -X+1, \dots, X\}$.*

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 $J(S) = f[i-1,S]$ and $K(S) = g[i,S]$ for all $S \subseteq P$. The cost of a branching is at most $\sum_{1\le i\le n} |v_i| \le mn$. To apply Lemma 4.7, we use $N = mn+1$ to represent ∞. That is, when no (i,S)-branching exists, we define $f[i,S] = N$. Then, Lemma 4.7 can be applied by setting $X = N$. Our second implementation is given in Algorithm 4.2.

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 $O(m2^{|V_{NR}|})$ time. By Lemma 4.7, for a fixed i, line 7 requires

Algorithm 4.2 Phase 3—Implementation 2.

```
Input: $D_\Phi$ and $B_R$
Output: an optimal branching $B^*$
 1: topologically sort the vertices in $D_\Phi$
 2: for each $S \subseteq V_{NR}$ do $f[1,S] \leftarrow N$
 3: $f[1,\emptyset] \leftarrow |v_1|$
 4: for $i = 2,3,\ldots,n$ do/* compute $f[i,\cdot]$
 5: begin
 6:     compute $g[i,S]$ for all $S \subseteq V_{NR}$
 7:     compute $f[i,S]$ for all $S \subseteq V_{NR}$ by applying Lemma 4.7
 8: end
 9: /* solution recovery
10: $B^* \leftarrow \emptyset$
11: $S^* \leftarrow V_{NR}$
12: for $i = n, n-1, \ldots, 2$ do/* find the child set of $v_i$
13: begin
14:     $T^* \leftarrow \text{argmin}\{f[i-1, S^*-T] + g[i,T] \mid T \subseteq S^*\}$
15:     $B^* \leftarrow B^* \cup \{(c, v_i) \mid c \in H_R^i \cup T^*\}$
16:     $S^* \leftarrow S^* - T^*$
17: end
18: return $B^*$
```

$O(2^{|P|}|P|^3X\lg(|P|X)\lg\lg(|P|X)) = O(2^{|V_{NR}|}|V_{NR}|^3N\lg(|V_{NR}|N)\lg\lg(|V_{NR}|N))$ time. Thus, lines 1–8 take a total of $O(mn2^{|V_{NR}|}+nN|V_{NR}|^3\lg(|V_{NR}|N)\lg\lg(|V_{NR}|N)2^{|V_{NR}|})$ time. Line 14 can be implemented in $O(|V_{NR}|2^{|V_{NR}|})$ time. Thus, the for-loop in lines 12–17 requires $O(n|V_{NR}|2^{|V_{NR}|})$ time. Consequently, the overall time complexity is $O(mn2^{|V_{NR}|} + nN|V_{NR}|^3\lg(|V_{NR}|N)\lg\lg(|V_{NR}|N)2^{|V_{NR}|})$, which is $O(mn^2t(\Phi)^3\lg mn \lg\lg mn2^{t(\Phi)})$ when substituting $t(\Phi)$ for $|V_{NR}|$ and $mn+1$ for N. Recall that the precomputation of D_Φ takes $O(mn^2)$ time, and Phases 1 and 2 take $O(n^2)$ time. In summary, we obtain the following.

Theorem 4.8. *UB can be solved in time $O(mn^2 + mn2^{t(\Phi)} + nt(\Phi)3^{t(\Phi)})$ or $O(mn^2t(\Phi)^3\lg mn\lg\lg mn2^{t(\Phi)})$, where $m = |\Sigma|$ and $n = |\Phi|$.*

By Theorem 4.8 and the upper bound of $t(\Phi)$ in Theorem 3.11, we obtain the following.

Theorem 4.9. *UB can be solved in time $O(mn^2 + mn2^{\min(n,2\delta(\Phi))} + n^23^{\min(n,2\delta(\Phi))})$ or $O(mn^5\lg mn\lg\lg mn2^{\min(n,2\delta(\Phi))})$, where $m = |\Sigma|$ and $n = |\Phi|$.*

Recall that $\varepsilon(M) = \delta(\Phi_M)$. Since $|\Sigma_M| = m$ and $|\Phi_M| = n$, combining Theorems 2.3 and 4.9 immediately yields the following, which says that SR is fixed-parameter tractable.

Theorem 4.10. *SR can be solved in time $O(mn^2 + mn2^{\min(n,2\varepsilon(M))} + n^23^{\min(n,2\varepsilon(M))})$ or $O(mn^5\lg mn\lg\lg mn2^{\min(n,2\varepsilon(M))})$.*

5. Algorithms for CSR, DSR, and CDSR.

This section gives algorithms for CSR, DSR, and CDSR.

5.1. CSR.

Recall that CSR is a constrained version of SR, in which only the rows in a given subset Q can be split. We denote by $\{r_1, r_2, \ldots, r_m\}$ the set of rows of M.

Let M' be a row split of M. By definition, there is a *valid partition* $\{R_1, R_2, \ldots, R_m\}$ of the set of rows of M' such that for all $i \in \{1, \ldots, m\}$, r_i is the bitwise OR of the rows in R_i. 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 *reduced* if it contains no redundant rows. Clearly, to solve SR (and CSR), only reduced conflict-free row splits need to be considered. Hujdurović et al. gave the following.

THEOREM 5.1 ([14]). *Let $M \in \{0,1\}^{m\times n}$ be a binary matrix. The following holds:*

1. *Any branching B of D_{Φ_M} can be transformed to a conflict-free row split M', equipped with a valid partition $\{R_1, R_2, \ldots, R_m\}$, of M such that $|R_i| = |U_B(r_i)|$ holds for each i.*
2. *Any reduced conflict-free row split M', equipped with a valid partition $\{R_1, R_2, \ldots, R_m\}$, of M can be transformed to a branching B of D_{Φ_M} such that $|R_i| = |U_B(r_i)|$ holds for each i.*

In CSR, a conflict-free row split, equipped with a valid partition $\{R_1, R_2, \ldots, R_m\}$, is *feasible* if $|R_i| = 1$ for each $r_i \notin Q$. By Theorem 5.1, a branching B of D_{Φ_M} with $|U_B(r_i)| = 1$ for each $r_i \notin Q$ corresponds to a feasible conflict-free row split; and each feasible reduced conflict-free row split corresponds to a branching B with $|U_B(r_i)| = 1$ for each $r_i \notin Q$. As a result, CSR is equivalent to the following *constrained uncovering branching problem* (CUB): given a set family Φ and a subset Q of its ground set Σ, find a *feasible* branching of D_Φ with the minimum cost, where a branching B is feasible if $|U_B(r_i)| = 1$ for each $r_i \notin Q$.

We proceed to present an algorithm for CUB. For each element $e \in \Sigma$, let $V|_e$ be the set of vertices containing e in D_Φ. An element $e \in \Sigma$ is *chain-like* if any two vertices $u, v \in V|_e$ are nested. We have the following.

LEMMA 5.2. *If an element e is chain-like, every vertex in $V|_e$ is regular.*

Proof. Suppose, by contradiction, that a vertex $v \in V|_e$ is not regular. By definition, v is contained in both of two conflicting vertices u, w. Since $v \subset u$ and $v \subset w$, we know that u and w are both in $V|_e$. Since u and w are in conflict, e is not chain-like and we have a contradiction. Thus, the lemma holds. ☐

LEMMA 5.3. *An element e is chain-like if and only if there exists a branching B with $|U_B(e)| = 1$.*

Proof. Consider a fixed element e. If $|U_B(e)| = 1$ for some branching B, by Lemma 3.4, there does not exist any pair of conflicting vertices in $V|_e$ and thus e is chain-like. Therefore, the if-part holds. Assume that e is chain-like. Recall that $\pi(v)$ is the smallest vertex strictly containing 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 $|U_{B_{\mathrm{R}}}(e)| = 1$. Let $W = (u_1, u_2, \ldots, u_s = \Sigma)$ be the sequence containing all vertices of $V|_e$ in nondecreasing order of their sizes. Consider a vertex u_i in W, $1 \leq i < s$. By Lemma 5.2, u_i is regular. Since $e \in u_i$, $\pi(u_i)$ contains e and thus is also in W. Furthermore, since any two vertices in W are nested, we know that $\pi(u_i) = u_{i+1}$. Therefore, in the branching B_{R}, the target pairs (e, u_j) are covered for $j = 2, 3, \ldots, s$. As a result, $|U_{B_{\mathrm{R}}}(e)| = |\{(e, u_1)\}| = 1$ and thus the lemma holds. ☐

Let $Q' = \Sigma - Q$. By Lemma 5.3, if any element in Q' is not chain-like, there is no solution. Suppose that all elements in Q' are chain-like. Let B^* be the branching obtained by running our UB algorithm. Since $B^* \supseteq B_{\mathrm{R}}$, by the proof of Lemma 5.3, $|U_{B^*}(e)| = 1$ for all $e \in Q'$. Thus, B^* is feasible. Since B^* is optimal for (the unconstrained) UB, it is optimal for CUB. Consequently, CUB can be solved as follows: determine whether all elements in Q' are chain-like; if not, report that there is no solution, and otherwise run our UB algorithm and return its output.

Let L be the sequence containing all vertices of D_Φ, in nondecreasing order of their sizes. In section 4.2, it has been shown that using L whether a vertex in D_Φ is regular can be determined in $O(n)$ time. Similarly, by the definition of chain-like elements, using L whether an element in Q' is chain-like can be determined in $O(n)$ time. Thus, whether all elements in Q' are chain-like can be determined in $O(n \lg n + mn)$ time. Therefore, the time complexities in Theorem 4.9 hold for CUB as well. Consequently, we obtain the following.

THEOREM 5.4. *CSR can be solved in time* $O(mn^2 + mn2^{\min(n,2\varepsilon(M))} + n^2 3^{\min(n,2\varepsilon(M))})$ *or* $O(mn^5 \lg mn \lg\lg mn 2^{\min(n,2\varepsilon(M))})$.

5.2. DSR. Recall that DSR is to find a conflict-free row split of M with the minimum number of distinct rows. DSR admits a formulation similar to UB. Let Φ be a set family over a set Σ. For a branching B of D_Φ, we say that a vertex $v \in \Phi$ is *B-irreducible* if there exists an element $e \in v$ such that (e, v) is B-uncovered. We denote by $I(B)$ the set of B-irreducible vertices. For the example in Figure 4(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 $\zeta(\Phi)$ be the minimum number of B-irreducible vertices over all branchings B of D_Φ. Hujdurović et al. [14] showed that DSR is equivalent to the following optimization problem.

DEFINITION 5.5. *The irreducing branching problem (IB):*
Input: A set family Φ.
Task: Find a branching B *of* D_Φ *with* $|I(B)| = \zeta(\Phi)$.

THEOREM 5.6 ([14]). *For a binary matrix* $M \in \{0,1\}^{m\times n}$*, the following holds:*
1. *Any branching* B *of* Φ_M *can be transformed to a conflict-free row split* M'*, equipped with a valid partition* $\{R_1, R_2, \ldots, R_m\}$*, of* M *such that* $|I(B)|$ *is equal to the number of distinct rows of* M' *and* $|U_B(r_i)| = |R_i|$ *holds for each* i.
2. *Any reduced conflict-free row split* M'*, equipped with a valid partition* $\{R_1, R_2, \ldots, R_m\}$*, of* M *can be transformed to a branching* B *of* D_{Φ_M} *such that* $|I(B)|$ *is equal to the number of distinct rows of* M' *and* $|U_B(r_i)| = |R_i|$ *holds for each* i.

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

LEMMA 5.7. *There exists an optimal branching* B^* *(with respect to IB) of* D_Φ *such that* $B^* \supseteq B_{\mathrm{R}}$.

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 4.4, 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 5.7, our Phase 2 algorithm predetermines the parent of each regular vertex as done in UB. We proceed to present the modification for Phase 3. Let V_i and an (i, S)-branching be defined the same as in section 4.3. Consider a positive integer

$i \le n$ and a subset $S \subseteq V_{\mathrm{NR}}$. 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 $f'[n, V_{\mathrm{NR}}]$. For $i = 1$, since V_i contains a single vertex, we have $f'[i, \emptyset] = 1$ and $f'[i, S] = \infty$ for $S \ne \emptyset$. Consider the case $i \ge 2$. Recall that H^i_{R} is the set of B_{R}-children of v_i. For a subset $T \subseteq V_{\mathrm{NR}}$, we define $g'[i, T]$ as follows: if $H^i_{\mathrm{R}} \cup T$ is a proper subset of v_i, define $g'[i, T] = 1$, indicating that v_i is irreducible if its child set is $H^i_{\mathrm{R}} \cup T$ in a branching; if $H^i_{\mathrm{R}} \cup T = v_i$, define $g'[i, T] = 0$, indicating that v_i is not irreducible if its child set is $H^i_{\mathrm{R}} \cup T$ in a branching; and otherwise define $g'[i, T] = \infty$, indicating that there is no branching in which the child set of v_i is $H^i_{\mathrm{R}} \cup T$. Then, we have the following:

$$f'[i, S] = \min\{f'[i-1, S-T] + g'[i, T] \mid T \subseteq S\} \text{ for } i > 1 \text{ and } S \subseteq V_{\mathrm{NR}}. \tag{5.1}$$

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_Φ 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 trivial vertices are not removed, all nonregular vertices in the original D_Φ are contained in V_{NR}, which by Theorem 3.12 is of cardinality at most $\min(n, 3\delta(\Phi) - 1)$. It is easy to check that the time complexities of the two implementations are still, respectively, $O(mn2^{|V_{\mathrm{NR}}|} + n|V_{\mathrm{NR}}|3^{|V_{\mathrm{NR}}|} + n|V_{\mathrm{NR}}|)$ and $O(mn2^{|V_{\mathrm{NR}}|} + nN|V_{\mathrm{NR}}|^3 \lg(|V_{\mathrm{NR}}|N) \lg\lg(|V_{\mathrm{NR}}|N)2^{|V_{\mathrm{NR}}|})$. Consider the second implementation. Since $|I(B)|$ is at most n, to apply Lemma 4.7, we can take $N = n+1$ to represent ∞, so that the resulting time complexity is reduced by a factor of m. By substituting $\min(n, 3\delta(\Phi)-1)$ for $|V_{\mathrm{NR}}|$ and $n+1$ for N in the above time complexities, we obtain the following.

THEOREM 5.8. *IB can be solved in time* $O(mn^2 + mn2^{\min(n,3\delta(\Phi))} + n^2 3^{\min(n,3\delta(\Phi))})$ *or* $O(mn^2 + mn2^{\min(n,3\delta(\Phi))} + n^5 \lg n \lg\lg n 2^{\min(n,3\delta(\Phi))})$*, where* $m = |\Sigma|$ *and* $n = |\Phi|$.

Consequently, we obtain the following.

THEOREM 5.9. *DSR can be solved in time* $O(mn^2 + mn2^{\min(n,3\varepsilon(M))} + n^2 3^{\min(n,3\varepsilon(M))})$ *or* $O(mn^2 + mn2^{\min(n,3\varepsilon(M))} + n^5 \lg n \lg\lg n 2^{\min(n,3\varepsilon(M))})$.

5.3. CDSR. Our algorithms for CSR and DSR can be combined to solve CDSR. As in section 5.1, we say that a conflict-free row split, equipped with a valid partition $\{R_1, R_2, \ldots, R_m\}$, is feasible if $|R_i| = 1$ for each $r_i \notin Q$, and a branching B of D_{Φ_M} is feasible if $|U_B(r_i)| = 1$ for each $r_i \notin Q$. CDSR is to find a feasible conflict-free row split of M with a minimum number of distinct rows. By Theorem 5.6, CDSR is equivalent to the following *constrained irreducing branching problem* (CIB): given a set family Φ and a subset Q of its ground set Σ, find a feasible branching B of D_Φ that has the smallest $|I(B)|$. Let $Q' = \Sigma - Q$. By Lemma 5.3, if any element of Q' is not chain-like, no feasible branching exists and thus there is no solution. Assume that each element of Q' is chain-like. Let B^* be the branching obtained by running our IB algorithm. Since $B^* \supseteq B_{\mathrm{R}}$, from the proof of Lemma 5.3, we know that $|U_{B^*}(e)| = 1$ for all $e \in Q'$. Thus, B^* is feasible. Since B^* is optimal for IB, it is optimal for CIB. Therefore, CIB can be solved as efficient as IB. Consequently, we obtained the following.

THEOREM 5.10. *CDSR can be solved in time* $O(mn^2 + mn2^{\min(n,3\varepsilon(M))} + n^2 3^{\min(n,3\varepsilon(M))})$ *or* $O(mn^2 + mn2^{\min(n,3\varepsilon(M))} + n^5 \lg n \lg\lg n 2^{\min(n,3\varepsilon(M))})$.

6. Conclusion and future work.

In this paper, based on the branching formulation in [14], we showed that SR is fixed-parameter tractable when parameterized by $\varepsilon(M)$. The proposed algorithm for finding an optimal branching of the derived digraph was built upon several new structural properties: we showed that the removal of trivial vertices does not change the cost of an optimal branching, proved that the parents of regular vertices can be greedily predetermined, and derived an upper bound on the number of nontrivial nonregular vertices. We also extended our SR algorithm to solve CSR, DSR, and CDSR.

A *kernel* of an UB instance Φ is an UB instance Φ' such that Φ and Φ' are equivalent and the size of Φ' is bounded by a function of $\delta(\Phi)$. We remark that our Phases 1 and 2 algorithms do not imply a kernel for UB. Phase 1 removes some vertices of Φ, but in the worst case, no vertices are removed. Phase 2 predetermines the parent of each regular vertex. This predetermination eases the finding of an optimal branching, but regular vertices are still parts of the input Φ. Therefore, our Phases 1 and 2 algorithms do not imply a kernel of UB. It is well-known that a problem is FPT if and only if it admits a kernelization algorithm. An interested reader may refer to Lemma 2.2 in [5] for the details. We remark that by applying this result, our FPT algorithm for UB can be used to obtain a kernel of UB. However, the size of the kernel is $O(2^{2\delta(\Phi)})$, which is exponential in $\delta(\Phi)$.

Possible directions for future research include to (1) give a polynomial kernel for SR with respect to the parameter $\varepsilon(M)$, (2) improve the $O^*(2^{\min(n,2\varepsilon(M))})$ time SR algorithm, (3) identify and study other meaningful parameters, (4) determine if SR is in APX, and (5) improve the approximation algorithms in [14] for SR and DSR.

Acknowledgment. The authors are grateful to the anonymous referee, who gave valuable comments that greatly improved the presentation of this paper and suggested the interpretation of UB as a problem of copying elements in order to make a set family laminar.

REFERENCES

[1] A. V. AHO, M. R. GAREY, AND J. D. ULLMAN, *The transitive reduction of a directed graph*, SIAM J. Comput., 1 (1972), pp. 131–137.

[2] A. BJÖRKLUND, T. HUSFELDT, P. KASKI, AND M. KOIVISTO, *Fourier meets Möbius: Fast subset convolution*, in Proceedings of the 39th Annual ACM Symposium on Theory of Computing, 2007, pp. 67–74.

[3] 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*, Proc. Natl. Acad. Sci. USA, 105 (2008), pp. 13081–13086.

[4] T. H. CORMEN, C. E. LEISERSON, R. L. RIVEST, AND C. STEIN, *Introduction to Algorithms*, 3rd ed., MIT Press, Cambridge, MA, 2009.

[5] M. CYGAN, F. V. FOMIN, L. KOWALIK, D. LOKSHTANOV, D. MARX, M. PILIPCZUK, M. PILIPCZUK, AND S. SAURABH, *Parameterized Algorithms*, Springer, New York, 2015.

[6] R. G. DOWNEY AND M. R. FELLOWS, *Parameterized Complexity*, Springer, New York, 2012.

[7] 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, 31 (2015), pp. i62–i70.

[8] D. FERNÁNDEZ-BACA, *The perfect phylogeny problem*, in Steiner Trees in Industry, D. D. X. Z. Cheng, ed., Springer, New York, 2001, pp. 203–234.

[9] D. FERNÁNDEZ-BACA AND J. LAGERGREN, *A polynomial-time algorithm for near-perfect phylogeny*, SIAM J. Comput., 32 (2003), pp. 1115–1127.

[10] D. Gusfield, *Efficient algorithms for inferring evolutionary trees*, Networks, 21 (1991), pp. 19–28.

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

[12] I. Hajirasouliha, A. Mahmoody, and B. J. Raphael, *A combinatorial approach for analyzing intra-tumor heterogeneity from high-throughput sequencing data*, Bioinformatics, 30 (2014), pp. i78–i86.

[13] 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, 2014, pp. 354–367.

[14] 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 Trans. Algorithms, 14 (2018), 26.

[15] 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 Trans. Comput. Biol. Bioinform., 15 (2018), pp. 96–108.

[16] 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, 35 (2019), pp. 769–777.

[17] W. M. Ismail, E. Nzabarushimana, and H. Tang, *Algorithmic approaches to clonal reconstruction in heterogeneous cell populations*, Quant. Biol., 7 (2019), pp. 255–265.

[18] W. Jiao, S. Vembu, A. G. Deshwar, L. Stein, and Q. Morris, *Inferring clonal evolution of tumors from single nucleotide somatic mutations*, BMC Bioinform., 15 (2014), 35.

[19] S. Kannan and T. Warnow, *A fast algorithm for the computation and enumeration of perfect phylogenies*, SIAM J. Comput., 26 (1997), pp. 1749–1763.

[20] S. Malikic, A. W. McPherson, N. Donmez, and C. S. Sahinalp, *Clonality inference in multiple tumor samples using phylogeny*, Bioinformatics, 31 (2015), pp. 1349–1356.

[21] 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 Res., 23 (2013), pp. 1097–1108.

[22] S. Nik-Zainal, P. V. 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, 149 (2012), pp. 994–1007.

[23] I. Pe'er, R. Shamir, and R. Sharan, *Incomplete directed perfect phylogeny*, SIAM J. Comput., 33 (2004), pp. 590–607.

[24] 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 Biol., 16 (2015), 91.

[25] F. Strino, F. Parisi, M. Micsinai, and Y. Kluger, *TrAp: A tree approach for fingerprinting subclonal tumor composition*, Nucleic Acids Res., 41 (2013), e165.