A Reciprocity Between Tree Ensemble Optimization and Multilinear Optimization

Jongeun Kim
Department of Industrial and Systems Engineering, University of Minnesota-Twin Cities, MN 55455, kim00623@umn.edu
Jean-Philippe P. Richard
Department of Industrial and Systems Engineering, University of Minnesota-Twin Cities, MN 55455, jrichar@umn.edu
Mohit Tawarmalani
Krannert School of Management, Purdue University, IN 47907, mtawarma@purdue.edu

In this paper, we establish a polynomial equivalence between tree ensemble optimization and optimization of multilinear functions over the Cartesian product of simplices. We use this insight to derive new formulations for tree ensemble optimization problems and to obtain new convex hull results for multilinear polytopes. A computational experiment on multi-commodity transportation problems with costs modeled using tree ensembles shows the practical advantage of our formulation relative to existing formulations of tree ensembles and other piecewise-linear modeling techniques. Motivated by the inability of tree ensembles to model multilinear functions of continuous variables, we then propose a generalization of decision-trees and provide empirical evidence of a better out-of-sample fit for a large collection of randomly perturbed nonlinear functions.

Key words: Tree ensemble optimization, Multilinear polytopes, Convexification, Decision-trees

1. Introduction

Decision-trees and tree ensembles are commonly used predictive models for classification (when the output takes values in a finite discrete set) or for regression (when the output variable takes continuous values). They have been used in a variety of successful applications in medicine (Svetnik et al. 2003), marketing (Ferreira et al. 2016), engineering design (Deepa et al. 2010), drug development (Ma et al. 2015), and demand forecasting (Herrera et al. 2010). Tree ensembles rank among the best predictive model across a variety of datasets (Fernández-Delgado et al. 2014). On the other hand, multilinear optimization problems have been studied extensively in the mathe-

matical programming literature (Tawarmalani and Sahinidis 2002, Luedtke et al. 2012, Crama and Rodríguez-Heck 2017, Del Pia and Khajavirad 2018a,b, Xu et al. 2021, He and Tawarmalani 2021). In particular, factorable programming problems are often modeled as multilinear programs with side constraints modeling univariate functions.

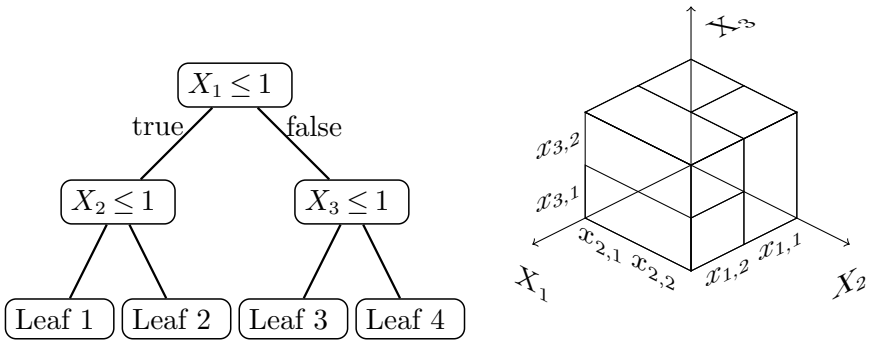
In this paper, we study prescriptive models where the objective and/or constraints are modeled using tree ensembles. The idea of using mixed-integer programming (MIP) representations of pre-trained models is at the forefront of research (Anderson et al. 2020, Mišić 2020). The special case that deals with optimizing an objective expressed as a pre-trained tree ensemble is referred to as the *tree ensemble optimization* (TEO) problem. Let T be the number of trees in the ensemble and let $f_t(X)$, for $t \in \{1, \ldots, T\}$ be the t^{th} decision tree with input variables X. Then, TEO is formulated as

$$\max_X \quad \sum_{t=1}^{T} f_t(X). \tag{1}$$

Mišić (2020) was the first to formalize this problem, to establish its NP-hardness, and to propose MIP-based approaches for its solution. There is now a significant amount of work related to TEO; see our literature review in Section 2. TEO has been used in the design of medicine (Mišić 2020) and assortment planning (Chen and Mišić 2021). In the former, medical effects of candidates and in the latter consumer choices are modeled using random forests.

To the best of our knowledge, TEO and multilinear optimization have not been treated under a common umbrella before. Here, we show that understanding the connections between these problems is useful in extending state-of-the-art results for them. In particular, we develop strong formulations for functions described by pre-trained decision trees and tree ensembles using insights from multilinear optimization. These results follow from the following construction. Assume for simplicity that all variables are numerical as the ideas generalize trivially to the case of categorical variables. When the domain of input variables is a hyper-rectangle and all queries are of the form "`is` $X_v \leq c$`?`", decision trees model piecewise-constant functions defined over a hyper-rectangular partition $\mathcal{S}$ of the original domain. This piecewise-constant function can be modeled by introducing a binary variable for each partition element that indicates whether the input variables belong to

the hyper-rectangle of $\mathcal{S}$ associated with this element. This hyper-rectangle is described using lower and upper bounds on its variables X. Projecting the partition elements along a variable X_v yields a discretization of the line-segment joining its upper and lower bound. For each interval in this discretization, we introduce an *interval indicator* binary variable which takes a value one when X_v belongs to that interval. We refer to the Cartesian product of intervals as a *partitioning atom*. Since each partition element is a union of partitioning atoms and the indicator for each atom is a product of interval indicators, the binary variable for each partition element is a sum of products of interval indicators, *i.e.* it is expressible as a multilinear function of interval indicators. Since the interval indicators for each variable belong to a simplex, $f_t(X)$ is a multilinear function of interval indicators over a Cartesian product of simplices. Therefore, problems involving decision trees are intimately related to multilinear optimization problems over Cartesian product of simplices; see Section 3 for more details. Figure 1 depicts the construction above, starting from a decision tree example defined over $\mathbb{R}^3$, displaying the corresponding hyper-rectangular partition, and producing the multilinear expression of an hyper-rectangle using binary elementary indicator variables.



(a) A decision-tree. (b) Hyper-rectangular partition.

Figure 1 An example of (a) a decision-tree and (b) its corresponding hyper-rectangular partition. Suppose $X \in [0,2]^3$. With binary variables $\boldsymbol{x}$ described in (b) that express the location of X, the indicator of a leaf can be expressed as a multilinear function of $\boldsymbol{x}$. For instance, the indicator of leaf 1 is expressed as $x_{1,1}x_{2,1}$.

Conversely, we will show in Section 3 that every multilinear function can be modeled as a tree ensemble of polynomial size. This establishes a reciprocity between the two problems and allows us to develop various insights into these problems. In particular, a decision-tree view yields new

polyhedral results on convexifying multilinear functions. Reciprocally, insights into convexification techniques for multilinear programs yield strengthened formulations for TEO and provide new ideas about how to use decision trees to develop better approximations of nonlinear functions.

The remainder of this paper is organized as follows. In Section 2, we review related literature. In Section 3, we describe the problems formally and establish a precise reciprocity between them. In Section 4, we introduce a new convexification scheme for multilinear sets over the Cartesian product of simplices using notions of faciality, completability, and decomposition. Faciality (see Section 4.1) identifies multilinear functions with indicators of faces and develops polynomially-sized extended formulations of their convex hull and provides a closed-form projection to the space of original problem variables as well as conditions under which this projection is polynomially-sized. Our discussion on completability (see Section 4.2) expresses multilinear functions in a form that satisfies the faciality condition by introducing additional variables and discusses conditions when this completion is possible with a polynomial increase in variables. Decomposition techniques (see Section 4.3) focus on convexifying a multilinear set by independently convexifying suitable multilinear sets defined over subsets of the original variables. Together, these three tools provide powerful ways of developing convex hulls that generalize a variety of existing results in the literature. In Section 5, we apply the convexification ideas to develop improved formulations for TEO. For example, we give the first polynomial convex-hull formulations for a single decision tree with categorical or numerical variables. We then demonstrate on a multi-commodity flow application how these results can be used to create new stronger formulations for problems with piecewise-constant objectives. Finally, we propose a generalization of decision-trees so they can express general multilinear functions. We conclude the paper in Section 6.

The specific contributions of this paper are as follows:

1. **Equivalence of two problems:** We show that TEO and multilinear optimization over the Cartesian product of simplices are equivalent. This connection reveals new ways of approaching convexification in multilinear optimization and tighter formulations for TEO.

2. **New convexification paradigm:** We introduce a new approach to simultaneously convexify multilinear sets over a Cartesian product of simplices. We derive new polynomially-sized convex hull descriptions for certain multilinear sets and show that many previously derived results follow readily from our approach.
3. **Multi-commodity transportation problem:** We solve, using various formulations, a transportation problem with piecewise-constant costs given via tree ensembles. Our formulation is among the smaller ideal formulations and, on large instances, performs better than piecewise-linear, disjunctive programming, and decision-tree based formulations.
4. **New approximation techniques for nonlinear functions:** We develop a generalized decision tree model to approximate nonlinear functions and their perturbations. Computational results confirm that the new tree structure provides better out-of-sample predictions.

2. Literature Review

TEO is first described and formalized in Mišić (2020), where the author shows that the problem is NP-Hard and gives both a linear and a nonlinear MIP formulation. The paper also develops and computationally evaluates a Benders' decomposition algorithm for solving TEO. Chen and Mišić (2021) introduce an improved formulation for the case where all the independent input variables X are binary. Chen and Mišić (2022) applies these models and techniques to an assortment planning problem. In this context, each tree in the ensemble models the choice of a collection of customers given an offer set.

Mistry et al. (2021) introduces a variant of TEO to design a catalyst mixture, where the objective function is the sum of a tree ensemble function with a convex function of the continuous input variables X that are also used in the training of the tree ensemble. To formulate the problem, the authors introduce variables X in the model and relate them to the indicator variables used in Mišić (2020) through linear *linking constraints*. This formulation is then used inside of an optimization framework for the numerical solution of optimization problems defined through tree ensemble models in Thebelt et al. (2021). Biggs et al. (2017) study a variant of TEO where the branching

decisions use general affine separators instead of threshold values for a single independent variable. In addition to providing an MIP model, the authors prove convergence results regarding the gap of the optimal value of the TEO model over a forest and that over a subset of trees.

Since TEO optimizes a piecewise-constant objective function, it can be viewed as a special case of piecewise-linear optimization. There is significant work on piecewise-linear functions in optimization; see Vielma et al. (2010), Vielma and Nemhauser (2011), Vielma (2015), Misener and Floudas (2012), Huchette and Vielma (2022), Baltean-Lugojan and Misener (2018), among others. We will develop formulations tighter than those in the TEO literature. These formulations also have better computational performance than disjunctive-programming-based formulations for a multi-commodity transportation problem where arc costs are piecewise-constant.

A function $f(x_1,\dots,x_n)$ is *multilinear* if f is linear in each x_i, $i=1,\dots,n$ when x_j, for $j\neq i$, are fixed at some value. The problem of deriving strong and ideal formulations for the graphs/epigraphs of multilinear functions has been widely studied. When the domain of the function is $[0,1]^n$, the convex hull is polyhedral and admits an exponentially-sized extended formulation; see Rikun (1997), Sherali (1997). This is because multilinear functions defined over hyper-rectangles are under-estimated by convex extensions of finite supports; see Tawarmalani and Sahinidis (2002) for a generalization. Crama (1993) identifies conditions under which the standard linearization of the multilinear function describes the envelope. Luedtke et al. (2012) studies McCormick's linearization of mutilinear functions and provide sufficient conditions under which it yields convex and concave envelopes. For bilinear functions whose variables belong to simplices, Bärmann et al. (2020) derive facet-defining inequalities of the bipartite boolean quadratic polytope with multi-choice constraints. Gupte et al. (2020) provide small-sized extended formulations for bilinear functions under certain conditions. The convex hull of permutation-invariant multilinear functions over a permutation-invariant box has been characterized; see Kim et al. (2021), Xu et al. (2021).

The simultaneous convex hull of multilinear functions over $[0,1]^n$ is also a polytope, which is typically referred to as the *multilinear polytope*. In particular, if $S=\{(\boldsymbol{x},\boldsymbol{y})\in\{0,1\}^{|V|}\times\mathbb{R}^{|E|}\mid y_e=$

$\prod_{v\in e} x_v, \forall e \in E\}$, where $H=(V,E)$ is a hypergraph, then $\text{conv}(S)$ is the multilinear polytope that describes the convex hull of $\prod_{v\in e} x_v, \forall e \in E$ over $[0,1]^n$. Standard linearization describes the convex hull of S when hypergraph H is Berge-acyclic; see Buchheim et al. (2019). Crama and Rodríguez-Heck (2017) introduces a family of valid inequalities called *2-link inequalities* for S that, when used in addition to standard linearization inequalities, are sufficient to describe its convex hull when H is laminar. Del Pia and Khajavirad (2018a) introduces a family of valid inequalities for S called *flower inequalities*. When used in conjunction with standard linearization, these inequalities provide a convex hull description of S when H is γ-acyclic. Finally, Del Pia and Khajavirad (2018b) show that, in some situations, the convex hull of a multilinear set S can be constructed by convexifying smaller sets separately. We borrow an idea from Schrijver (1983) to derive a more general decomposition result that can also be used to produce formulations of 0–1 multilinear sets with constant treewidth given in Bienstock and Munoz (2018).

3. Sources of Reciprocity Between the Two Problems

In this section, we establish an equivalence between TEO and multilinear optimization over the Cartesian product of simplices. We use bold lowercase letters to denote vectors. For positive integers a and b, we define $[a] := \{1,2,\dots,a\}$ and $[a..b] := \{a, a+1, \dots, b\}$. For a positive integer K, we use $\Delta^K := \{\boldsymbol{x} \in \mathbb{R}^K_+ \mid \sum_{j\in[K]} x_j = 1\}$ to denote the simplex having as vertices the K principal vectors of $\mathbb{R}^K$. We refer to the collection of these principal vectors as $\Delta^K_{0,1}$, *i.e.*, $\Delta^K_{0,1} = \Delta^K \cap \{0,1\}^K$.

3.1. TEO as a Multilinear Problem

Consider a tree ensemble model $f(X)$ with T binary decision trees, where, for each $t \in T$, $f_t(X)$ is a function of n independent variables $X=(X_1,\dots,X_n)$. We derive a multilinear formulation for the associated TEO problem (1). We remark that the nonlinear formulation in Mišić (2020) is not multilinear in that variables belonging to the same simplex are multiplied with one another.

Let $\mathcal{N}$ and $\mathcal{C}$ denote the indices of numerical and categorical variables, respectively, and assume that $\mathcal{N} \cup \mathcal{C} = [n]$. The set of leaves (or leaf nodes) of tree t is denoted as $\textbf{leaves}(t)$ and the set of splits (or split/non-leaf nodes) of tree t as $\textbf{splits}(t)$. Each split s is assumed to involve a query

that contains one independent variable, referred to as $\mathbf{v}(s)$. For $v \in \mathcal{N}$, the format of a query is "$X_v \le a$?" where a is a scalar. We refer to this scalar a as the *split value* for independent variable X_v. For $v \in \mathcal{C}$, the format of a query is "$X_v \in A$?" where A is a subset of categories associated with X_v. We denote the set A associated with the query of split s by $\mathbf{c}(s)$.

To streamline the presentation, we make two assumptions that are without loss of generality. First, categorical variables X_v are relabeled to have categories $[K_v]$ for some positive integer K_v. Second, we assume $|\mathcal{N}| = 0$ because numerical variables can be modeled using categorical ones. To do so, for $v \in \mathcal{N}$, let $-\infty < a_{v,1} < a_{v,2} < \cdots < a_{v,K_v-1} < \infty$ denote the split values for X_v. Then, we may replace X_v with a categorical variable $\bar{X}_v$ by defining $(a_{v,j-1}, a_{v,j}] \cap \mathbb{R}$ as category j for $j \in \{1, \ldots, K_v\}$, where $a_{v,0} := -\infty$ and $a_{v,K_v} := \infty$; see Lemma APDX.1 for a formal argument. The split query $X_v \le a_{v,j}$? is the same as $\bar{X}_v \in [j]$? The categories created in this way have a special structure that we exploit in Section 4.1 to develop more compact models.

Let $x_{v,j}$, for $v \in [n]$ and $j \in [K_v]$, be binary variables that indicate if input variable X_v is assigned category j in a solution, *i.e.*, $x_{v,j} = \mathbb{1}[X_v = j]$. Since each input variable X_v is assigned to exactly one category, variables $\{x_{v,j}\}_{j \in [K_v]}$ belong to $\Delta_{0,1}^{K_v}$. With each $\ell \in \mathbf{leaves}(t)$ we associate a binary variable $y_{t,\ell}$. Given X, we define $y_{t,\ell}$ to be one if and only if ℓ is the unique leaf in tree t such that all the queries from ℓ to the root are satisfied by X. Each leaf ℓ can be described by $(J_{t,\ell}^1, \ldots, J_{t,\ell}^n)$, where $J_{t,\ell}^v$ is the set of categories of input variable X_v that satisfy queries on X_v along the path from the root to ℓ. More specifically, $J_{t,\ell}^v = [K_v] \cap \left(\bigcap_{s \in M_{\mathrm{y}}: \mathbf{v}(s)=v} \mathbf{c}(s)\right) \cap \left(\bigcap_{s \in M_{\mathrm{n}}: \mathbf{v}(s)=v} [K_v] \setminus \mathbf{c}(s)\right)$, where $M_{\mathrm{y}} \subseteq \mathbf{split}(t)$ is the set of splits answered "yes" and $M_{\mathrm{n}} \subseteq \mathbf{split}(t)$ is the set of splits answered "no.". Then, $y_{t,\ell} = \prod_{v \in [n]} \sum_{j \in J_{t,\ell}^v} x_{v,j}$. We now can formulate (1) as

$$\max \left\{ \sum_{t \in [T]} \sum_{\ell \in \mathbf{leaves}(t)} p_{t,\ell} y_{t,\ell} \;\middle|\; y_{t,\ell} = \prod_{v \in [n]} \sum_{j \in J_{t,\ell}^v} x_{v,j}, \forall t \in [T],\ \forall \ell \in \mathbf{leaves}(t),\ \boldsymbol{x_v} \in \Delta_{0,1}^{K_v}, \forall v \in [n] \right\} \quad (2)$$

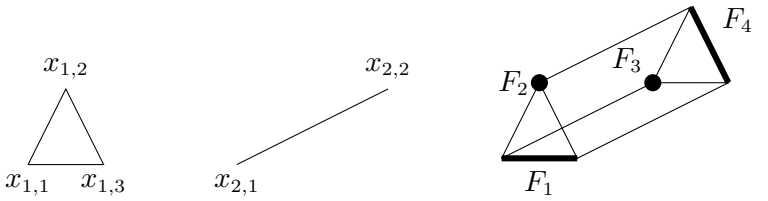
where $p_{t,\ell}$ is the prediction value of leaf ℓ in tree t.

3.2. Multilinear Optimization Over the Cartesian Product of Simplices

We abstract (2) into a family of models that we call *multilinear optimization over the Cartesian products of simplices (MOCPS)*. These models have binary variables $\boldsymbol{x_v} = \{x_{v,j}\}_{j \in [K_v]} \in \Delta_{0,1}^{K_v}$ for

all $v \in [n]$ so that $\boldsymbol{x_v}$ is a binarization of a variable with K_v possible values. We denote the extreme points of $P := \prod_{v\in[n]} \Delta^{K_v}$ by $D := \text{vert}(P)$. For each $(J^1, \dots, J^n)$ such that $J^v \subseteq [K_v]$, we associate the elementary multilinear function $\prod_{v\in[n]} \sum_{j\in J^v} x_{v,j}$. Any multilinear function of $\boldsymbol{x}$ is trivially an affine combination of elementary multilinear terms where $|J^v| = 1$ for all v. We associate each elementary multilinear function $g(\boldsymbol{x})$ with the face F of P it indicates, *i.e.*, $F = \{\boldsymbol{x} \in P \mid g(\boldsymbol{x}) = 1\}$.

EXAMPLE 1. Let $P_1 := \Delta^3$ and $P_2 := \Delta^2$. For $v = 1, 2$, let $x_{v,j}$ be the j^{th} extreme point of P_v; see Figure 2a and Figure 2b. Then, the functions $\mathbb{1}_{F_1}(\boldsymbol{x}) = (x_{1,1} + x_{1,3})x_{2,1}$, $\mathbb{1}_{F_2}(\boldsymbol{x}) = x_{1,2}x_{2,1}$, $\mathbb{1}_{F_3}(\boldsymbol{x}) = x_{1,1}x_{2,2}$, and $\mathbb{1}_{F_4}(\boldsymbol{x}) = (x_{1,2} + x_{1,3})x_{2,2}$ indicate faces $F_1, \dots, F_4$ as shown in Figure 2c. ■



(a) P_1 (b) P_2 (c) Faces $F_1, \dots, F_4$ of P.

Figure 2 Illustration of Example 1.

Given P and a set $\mathcal{F}$ of its faces, we define the multilinear set

$$\text{ML}(D, \mathcal{F}) := \left\{(\boldsymbol{x}, \boldsymbol{y}) \in D \times \mathbb{R}^{|\mathcal{F}|} \,\middle|\, y_F = \mathbb{1}_F(\boldsymbol{x}), \forall F \in \mathcal{F}\right\} \tag{3}$$

where $D = \text{vert}(P)$ and $\mathbb{1}_F(\boldsymbol{x}) := \mathbb{1}[\boldsymbol{x} \in F]$. Given $\boldsymbol{c} \in \mathbb{R}^{|\mathcal{F}|}$, we now define MOCPS as

$$\max \left\{ \sum_{F\in\mathcal{F}} c_F y_F \,\middle|\, (\boldsymbol{x}, \boldsymbol{y}) \in \text{ML}(D, \mathcal{F}) \right\}. \tag{4}$$

TEO, as in (2), is a special case of MOCPS because, for any leaf ℓ in tree t, $y_{t,\ell}$ is an elementary multilinear function determined by $\{J^v_{t,\ell}\}_{v\in[n]}$. Furthermore, MOCPS does not require that each vertex of P belongs to some face of $\mathcal{F}$ while TEO requires that each vertex is contained in one of the leaves for each tree. Clearly, MOCPS also generalizes unconstrained 0–1 multilinear optimization problems $\max_{\boldsymbol{x}\in\{0,1\}^n} \sum_{k\in[m]} c_k \prod_{v\in S_k} x_v$, where $S_k \subseteq [n]$ for $k \in [m]$ by choosing Δ^{K_v} to be Δ^2.

3.3. MOCPS as TEO

In Section 3.2, we showed that TEO is a special case of MOCPS. In this section, we derive the reverse implication by introducing at most a polynomial number of terms, thereby deriving an equivalence between the two problems that is important for the later sections. We assume that:

ASSUMPTION 1. TEO: *Each input variable is used in some branching query. Changing the category of an input variable leads it to a different leaf for some setting of the remaining variables.*

MOCPS: *For all* v, $2 \le K_v \le |\mathcal{F}| + 1$.

By simplifying instances, we can always ensure that Assumption 1 is satisfied. In fact, if TEO does not satisfy Assumption 1, we can either discard an input variable or merge two of its categories into one, without altering the predictions of the ensemble. If $K_v = 1$ in MOCPS, the variable v can be dropped. Further, for each $1 \le i, j \le K_v$ construct an edge between i and j if there exists a face $F \in \mathcal{F}$ with $|J^v \cap \{x_{v,i}, x_{v,j}\}| = 1$. Then, if $|\mathcal{F}| < K_v(K_v - 1)/2$ there is a pair (i, j) that is not connected. Combining these categories does not alter MOCPS. Since $K_v \ge 2$, it follows that $|\mathcal{F}| \ge K_v - 1$.

The complexity of TEO can be measured by $L := \sum_{t \in [T]} |\mathbf{leaves}(t)|$, the total number of leaves in the tree ensemble. This is because other values, *i.e.*, the numbers of trees, input variables, categories for each input variable, are bounded above by L. The complexity of MOCPS can be measured using n and $|\mathcal{F}|$, the numbers of simplices and faces, respectively. This is because the other values, *i.e.*, the number of vertices for all simplices, are polynomially bounded in n and $|\mathcal{F}|$. When we wish to differentiate the number of input variables n in TEO from the number of simplices n in MOCPS, we use n_T and n_M, respectively.

Theorem 1, whose proof can be found in APDX.1, establishes a precise constructive polynomial equivalence between the two problems.

THEOREM 1. *Any instance of TEO with L leaves, can be reduced into an instance of MOCPS with at most L simplices and at most L faces. Any instance of MOCPS with n simplices and a set $\mathcal{F}$ of faces can be reduced into an instance of TEO with at most $|\mathcal{F}|(n+1)$ leaves.*

Theorem 1 obviously implies that TEO is at least as hard as MOCPS. In the following sections we use Theorem 1 to transfer polyhedral results from one problem to another.

4. Facial Decomposition and its Convexification Properties

Given a set S and a *partition* $\mathcal{S} = \{S_1, \ldots, S_r\}$ of S, such that $S_i \cap S_j = \emptyset$ for $i \neq j$ and $\bigcup_{i=1}^r S_i = S$, we are interested in the simultaneous convex hull of indicators of S_i, $i = 1, \ldots, r$.

Definition 1. The *partition indicator hull* of a set S, given its partition $\mathcal{S} = \{S_1, \ldots, S_r\}$, is the closed convex hull of $P(S, \mathcal{S})$, where $P(S, \mathcal{S}) := \{(\boldsymbol{x}, \boldsymbol{y}) \mid y_i = \mathbb{1}_{S_i}(\boldsymbol{x}) \text{ for } i \in [r],\ \boldsymbol{x} \in S\}$.

Denote the closed convex hull of S_i as $\operatorname{cl}\operatorname{conv}(S_i)$ and the recession cone of $\operatorname{cl}\operatorname{conv}(S_i)$ as $O^+ \operatorname{cl}\operatorname{conv}(S_i)$. If $O^+ \operatorname{cl}\operatorname{conv}(S_i)$ does not vary with i, the partition indicator hull can be constructed using disjunctive programming techniques (Balas 1985, Stubbs and Mehrotra 1999, Ceria and Soares 1999). In particular, let $X_i = P(S, \mathcal{S}) \cap \{\boldsymbol{y} \mid y_i = 1,\ y_j = 0 \text{ for } j \neq i\}$. Since, $P(S, \mathcal{S}) = \bigcup_{i=1}^r X_i$, it suffices to construct $\operatorname{cl}\operatorname{conv}(\bigcup_{i=1}^r \operatorname{cl}\operatorname{conv}(X_i))$. Since $X_i = \{(\boldsymbol{x}, \boldsymbol{y}) \mid y_i = 1,\ y_j = 0 \text{ for } j \neq i,\ \boldsymbol{x} \in S_i\}$, then $O^+ \operatorname{cl}\operatorname{conv}(X_i) = \{(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{x} \in O^+ \operatorname{cl}\operatorname{conv}(S_i), y_j = 0 \text{ for } j \in [r]\}$. Since $O^+ \operatorname{cl}\operatorname{conv}(S_i) = O^+ \operatorname{cl}\operatorname{conv}(S_{i'})$ for all i, $i' \in [r]$, it follows that $O^+ \operatorname{cl}\operatorname{conv}(X_i) = O^+ \operatorname{cl}\operatorname{conv}(X_{i'})$. Then, Corollary 9.8.1 in Rockafellar (1970) shows that $\operatorname{conv}(\bigcup_{i=1}^r \operatorname{cl}\operatorname{conv}(X_i))$ is closed and has the same recession cone as $O^+ \operatorname{cl}\operatorname{conv}(X_i)$. This convex hull can be computed using Theorem 9.8 in Rockafellar (1970).

Although the construction can be performed in general settings, we will primarily be interested in the following special case. First, we will often choose S to be the set of vertices of a Cartesian product of simplices. Although we restrict attention to vertices as in (3), we remark that the optimal value and the convex hull of the feasible region of MOCPS does not change if D is replaced with P; see Tawarmalani (2010). Second, these vertices are binary valued. Third, we require that $\operatorname{cl}\operatorname{conv}(S_i)$ has a succinct representation. This latter assumption ensures that the disjunctive programming construction described above is tractable. We will be interested in projecting the formulation back to the space of $(\boldsymbol{x}, \boldsymbol{y})$ variables whenever possible.

Given a polytope P with vertices in $\{0,1\}^n$, which we refer to as a *0–1 polytope*, we define specific partitions of interest that will be used to construct convex hulls.

Definition 2. A *vertex decomposition* of a 0–1 polytope P is a partition of $\operatorname{vert}(P)$ into $\mathcal{V} = \{V_1, \ldots, V_r\}$ and, for all i, $\operatorname{conv}(V_i)$ has a tractable $\mathcal{H}$-description.

DEFINITION 3. A *facial decomposition* of a 0–1 polytope P is a vertex decomposition of $\text{vert}(P)$ into $\mathcal{V} = \{V_1, \dots, V_r\}$ such that $\text{conv}(V_i)$ is a face of P, whose defining inequality is available in closed-form. In other words, a facial decomposition may be described as $\mathcal{P} = \{P_1, \dots, P_r\}$ where each P_j is a hyperplane representation of the face $\text{conv}(V_j)$.

The key advantage of these partitions is that the resulting convex hull is integral. In Section 4.1, we derive convex hull representations for (3) when it corresponds to a partition indicator hull of $D := \prod_{v\in[n]} \Delta_{0,1}^{K_v}$. In this case, $\mathcal{F}$ is a facial decomposition of D since F is a face of $\text{conv}(D)$ for all $F \in \mathcal{F}$. In Section 4.2, we discuss ways to construct a facial decomposition that refines a given set of faces. We show that there may not exist such a refined facial decomposition that is polynomially-sized in the number of faces provided. In Section 4.3, we identify conditions under which the convex hull of a given set can be constructed by deriving convex hulls of related sets with fewer variables and constraints. We then show that various results in the literature follow directly from our constructions. The proofs of all upcoming results can be found in APDX.2.

4.1. Partition Indicator Hull of a Facial Decomposition

When the set $\mathcal{F}$ is a facial decomposition of D, we can express (3) as the following lifting of D:

$$\text{ML}(D,\mathcal{F}) := \left\{ (\boldsymbol{x},\boldsymbol{y}) \in \prod_{v\in[n]} \Delta_{0,1}^{K_v} \times \mathbb{R}^{|\mathcal{F}|} \;\middle|\; y_F = \prod_{v\in[n]} \sum_{j\in J_F^v} x_{v,j}, \forall F \in \mathcal{F} \right\} \tag{5}$$

or, by defining $P_F := \{(\boldsymbol{x},\boldsymbol{y}) \mid (\boldsymbol{x},\boldsymbol{y}) \in \text{ML}(D,\mathcal{F}),\ y_F = 1\}$ for all $F \in \mathcal{F}$, as $\text{ML}(D,\mathcal{F}) = \bigvee_{F\in\mathcal{F}} P_F$. The convex hull of P_F can be described by intersecting P with $n+1$ linear constraints as follows:

$$\text{conv}(P_F) = \left\{ (\boldsymbol{x},\boldsymbol{y}) \in \prod_{v\in[n]} \Delta^{K_v} \times \Delta^{|\mathcal{F}|} \;\middle|\; y_F = 1, \sum_{j\in J_F^v} x_{v,j} = 1, \forall v \in [n] \right\}. \tag{6}$$

Then, $\text{conv}(\text{ML}(D,\mathcal{F}))$ is obtained by applying disjunctive programming (see Balas 1985, 1998). The resulting formulation has a polynomial number of variables and constraints.

THEOREM 2. *It holds that* $\text{conv}(ML(D,\mathcal{F}))$ *is the projection in the space of* $(\boldsymbol{x},\boldsymbol{y})$ *of* (7)*:*

$$\sum_{F\in\mathcal{F}: j\in J_F^v} z_{v,j,F} = x_{v,j}, \qquad \forall v \in [n],\ \forall j \in [K_v], \tag{7a}$$

$$\sum_{j\in J_F^v} z_{v,j,F} = y_F, \qquad \forall v\in[n],\ \forall F\in\mathcal{F}, \tag{7b}$$

$$\boldsymbol{x_v}\in\Delta^{K_v}, \qquad \forall v\in[n], \tag{7c}$$

$$\boldsymbol{y}\in\Delta^{|\mathcal{F}|}, \tag{7d}$$

$$z_{v,j,F}\geq 0, \qquad \forall v\in[n],\ \forall F\in\mathcal{F},\ \forall j\in J_F^v, \tag{7e}$$

where (7c) *can be omitted, if desired.*

We next study the formulation of Theorem 2, which is unlike typical formulations constructed using disjunctive programming. In particular, it describes a network flow polytope with one source (resp. one destination) that has a supply (resp. a demand) of 1. The structure of this network flow problem is depicted in Figure 3. For each simplex, it models a transportation problem between nodes associated with the simplex variables $\boldsymbol{x}$ and nodes associated with indicator variables $\boldsymbol{y}$. Since $\boldsymbol{y}\in\Delta^{|\mathcal{F}|}$, which is a simplex, we can project the feasible region to $(\boldsymbol{x},\boldsymbol{y})$ by checking whether it is feasible to transport, using $z_{v,j,F}$, a supply of $\boldsymbol{x_v}$ to meet demands of $\boldsymbol{y}$ independently for each $v\in[n]$. In particular, $(\bar{\boldsymbol{x}},\bar{\boldsymbol{y}})\in\operatorname{conv}(\mathrm{ML}(D,\mathcal{F}))$ if and only if the transportation problems

$$\mathrm{TP}(v):\qquad \sum_{F\in\mathcal{F}:j\in J_F^v} z_{v,j,F} = \bar{x}_{v,j}, \qquad \forall j\in[K_v], \tag{8a}$$

$$\sum_{j\in J_F^v} z_{v,j,F} = \bar{y}_F, \qquad \forall F\in\mathcal{F}, \tag{8b}$$

$$z_{v,j,F}\geq 0, \qquad \forall F\in\mathcal{F},\ \forall j\in J_F^v, \tag{8c}$$

are feasible for all $v\in[n]$. A consequence of Hoffman's circulation theorem (Hoffman (1976)) is that (8) is feasible if for each subset of supply nodes $\bar{N}$, the total demand for nodes that are connected to at least one supply in $\bar{N}$ does not exceed the total supply of $\bar{N}$. Thus, we obtain the following result.

PROPOSITION 1. *Let N be any subset of $[n]$. Then, $\operatorname{conv}(ML(D,\mathcal{F}))$ is described by:*

$$\sum_{F\in\mathcal{F}:J_F^v\subseteq J} y_F \leq \sum_{j\in J} x_{v,j}, \qquad \forall v\in N,\ \forall\emptyset\neq J\subsetneq[K_v], \tag{9a}$$

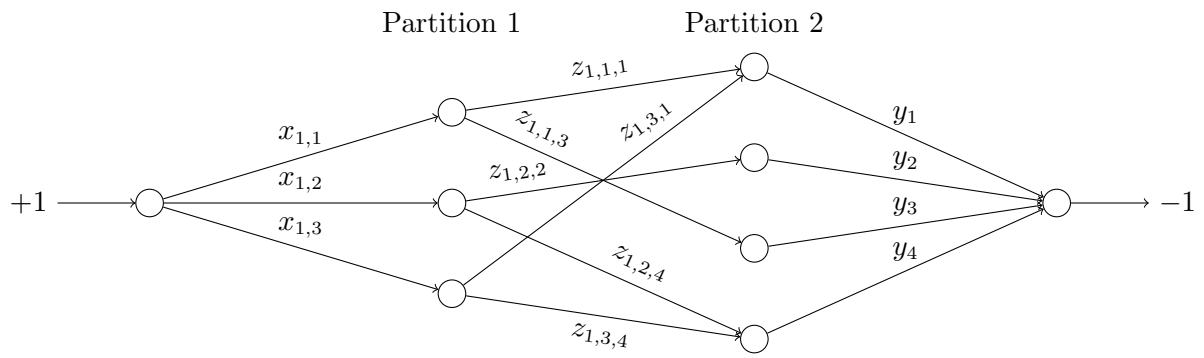


Figure 3 Transportation network corresponding to TP(1) for the example of Figure 2.

$$\sum_{F\in H} y_F \le \sum_{j\in\bigcup_{F\in H} J_F^v} x_{v,j}, \qquad \forall v\in[n]\setminus N,\ \forall \emptyset\neq H\subsetneq\mathcal{F}, \tag{9b}$$

$$(\boldsymbol{x_1},\dots,\boldsymbol{x_n},\boldsymbol{y})\in\left(\prod_{v\in[n]}\Delta^{K_v}\right)\times\Delta^{|\mathcal{F}|}. \tag{9c}$$

There are exponentially many constraints in (9a) and (9b). The associated computational challenge is, however, mitigated by two factors. First, a polynomial time separation algorithm can be constructed by using (7) to generate cuts in the space of original problem variables, as is described in APDX.2.2. Further, not all inequalities (9a) or (9b) are always necessary to describe $\text{conv}(\text{ML}(D,\mathcal{F}))$. Theorem 1 in Kis and Horváth (2021) can be utilized for this purpose, although the main idea in Proposition 2 simply reduces to decomposing (9a) (resp. (9b)) into two constraints of the same type under certain conditions.

Proposition 2. *Constraint (9a) for indices (v,J) is redundant if there is a non-trivial partition (J_1,J_2) of J (i.e., $J_1\neq\emptyset$ and $J_2\neq\emptyset$), where every $F\in\mathcal{F}$ such that $J_F^v\subseteq J$ satisfies either $J_F^v\subseteq J_1$ or $J_F^v\subseteq J_2$. Similarly, (9b) for indices (v,H) is redundant if there is a non-trivial partition (H_1,H_2) of H such that $(\cup_{F\in H_1}J_F^i)\cap(\cup_{F\in H_2}J_F^i)=\emptyset$.*

Using Proposition 1 and Proposition 2, we obtain the following formulation for $\text{conv}(\text{ML}(D,\mathcal{F}))$ with the set of faces $\mathcal{F}$ presented in Figure 2:

$$y_3\le x_{1,1},\quad y_2\le x_{1,2},\quad y_1+y_3\le x_{1,1}+x_{1,3},\quad y_2+y_4\le x_{1,2}+x_{1,3}, \tag{10a}$$

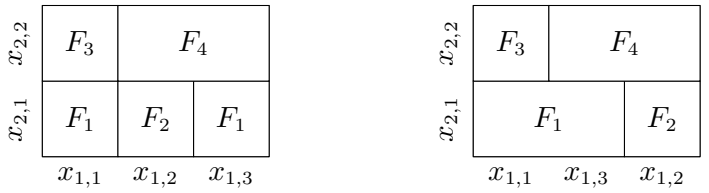
$$y_1+y_2\le x_{2,1},\quad y_3+y_4\le x_{2,2},\quad \boldsymbol{x_1}\in\Delta^3,\quad \boldsymbol{x_2}\in\Delta^2,\quad \boldsymbol{y}\in\Delta^4. \tag{10b}$$

We next describe sufficient conditions under which (9) reduces to a polynomially-sized formulation. These conditions encompass situations where the input variables of a decision tree are numerical, but also apply to certain situations where some variables are categorical.

DEFINITION 4. A collection $\mathcal{F}$ of subsets of vertices of $\text{conv}(D)$ corresponding to faces has the *adjacency property* if for all $F \in \mathcal{F}$ defined using $(J_F^1, \ldots, J_F^n)$, $J_F^v = [a_F^v..b_F^v]$ for some $a_F^v, b_F^v \in [K_v]$.

If $K_v = 2$ for all $v \in [n]$, $\mathcal{F}$ always has the adjacency property because J_F^v is either $\{1\}$, $\{2\}$, or $\{1,2\}$, which are all sets of consecutive integers. We illustrate next that the adjacency property may hold after permuting the indices in $[K_v]$.

EXAMPLE 2. The set of faces in Example 1 does not satisfy the adjacency property. As shown in Figure 4, when we permute $[K_1]$ using $\sigma_1 : [1,2,3] \to [1,3,2]$, the same set of faces satisfies the adjacency property. ■



(a) Original ordering. (b) After switching $x_{1,2}$ and $x_{1,3}$.

Figure 4 Grid representations for Example 1 before and after reordering of variables.

With the adjacency property, only polynomially many inequalities in (9) are non-redundant.

THEOREM 3. *Suppose that (after suitable reordering) $\mathcal{F}$ has the adjacency property. Assume further that, for all $v \in [n]$, $J_F^v = [a_F^v..b_F^v]$ for some $a_F^v, b_F^v \in [K_v]$. Then, (11) describes* $\text{conv}(ML(D,\mathcal{F}))$*:*

$$\sum_{F \in \mathcal{F} \,:\, J_F^v \subseteq [a..b]} y_\ell \le \sum_{j=a}^{b} x_{v,j}, \qquad \forall v \in [n],\ \forall a, b \in [K_v] : a \le b, \tag{11a}$$

$$(\boldsymbol{x_1}, \ldots, \boldsymbol{x_n}, \boldsymbol{y}) \in \left(\prod_{v \in [n]} \Delta^{K_v} \right) \times \Delta^{|\mathcal{F}|}. \tag{11b}$$

EXAMPLE 3. A description of the partition indicator hull of the facial decomposition presented in Example 1 is given in (10). Assume that the variables are first permuted as described in Example 2. Then, a simple inspection of (10) shows that the indices (v, J) for which the constraint (9a) is needed all satisfy the condition that $J = [a..b]$ for some $a, b \in [K_v]$ with $a \leq b$. ■

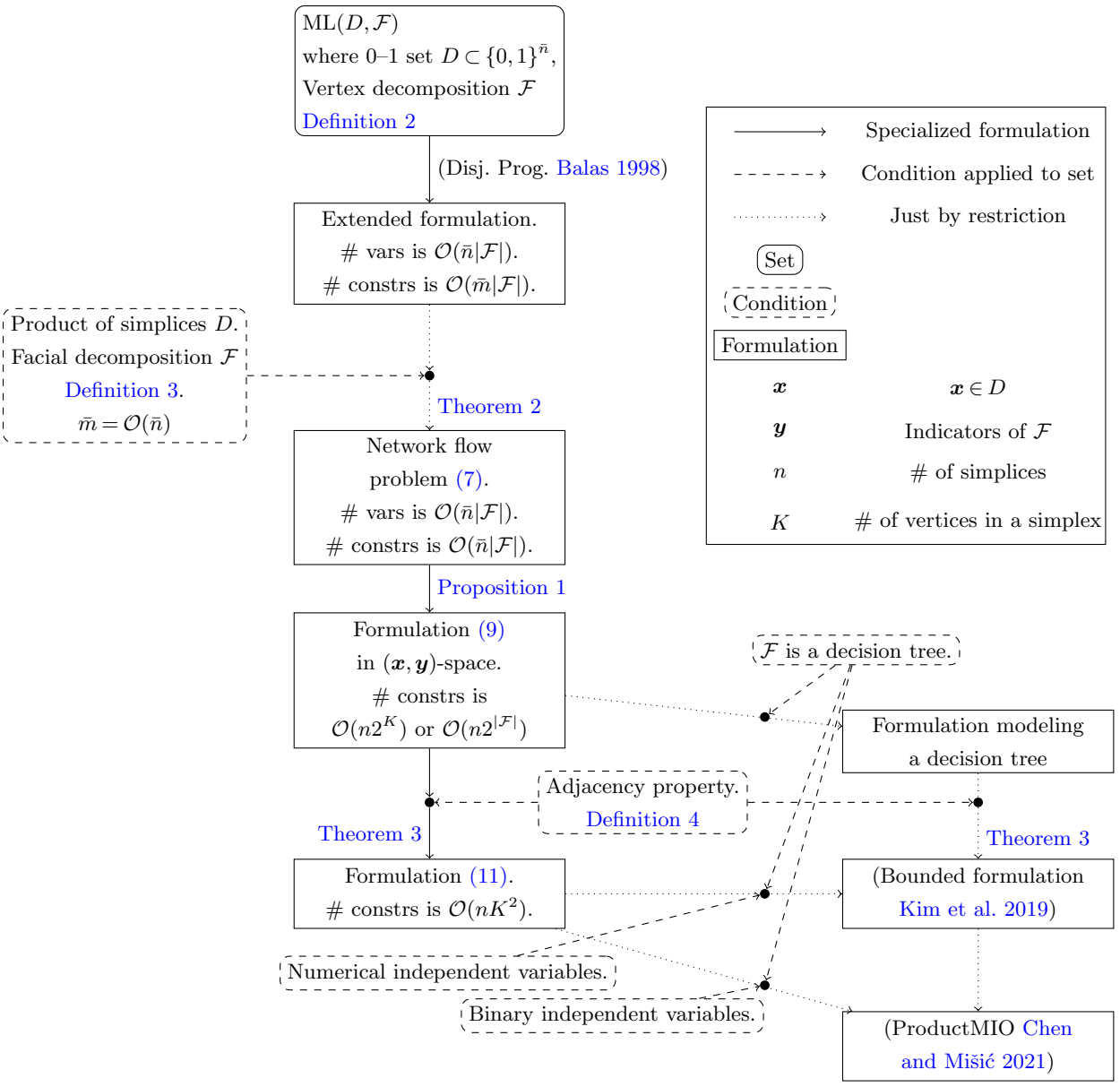


Figure 5 Schematic illustration of relationships between results.

In Figure 5, we summarize the relations between the results obtained in this section and their connections to TEO that we will discuss in Section 5.1. To fix ideas, let $\bar{\boldsymbol{x}} \in \mathcal{H} := \{0,1\}^n$ and consider

a vertex decomposition $\mathcal{F} = \{F_0, \ldots, F_n\}$ where $F_j = \left\{\bar{\boldsymbol{x}} \in \text{conv}(\mathcal{H}) \mid \bar{x}_i = 1,\ \forall i \leq j,\ \bar{x}_{j+1} = 0\right\}$ and where $\bar{x}_{n+1}$ is assumed to be zero. Clearly, $\text{conv}\big(\text{ML}(\mathcal{H}, \mathcal{F})\big)$ is the convex hull of $\left\{(\bar{\boldsymbol{x}}, \boldsymbol{y}) \in \mathcal{H} \times \mathbb{R}^{n+1} \mid y_j = (1 - \bar{x}_{j+1}) \prod_{i=1}^{j} \bar{x}_i,\ \forall j \in [0..n]\right\}$. Each point in $\mathcal{H}$ belongs to the face F_j chosen so that $j+1$ is the smallest index with $\bar{x}_{j+1}$ equal to zero. Moreover since no vertex belongs to two of these faces, it follows that $\mathcal{F}$ is a facial decomposition. As is standard, binary variables x_j can be mapped to Δ^2, and so to the setting of (4), by defining $x_{j,1} = \bar{x}_j$ and $x_{j,2} = 1 - \bar{x}_j$. Then, $\sum_{j \in J} x_{v,j}$ can be written as $0, x_{j,1}, x_{j,2}, 1$ when $J = \emptyset, \{1\}, \{2\}, \{1,2\}$, respectively. As stated earlier, all facial decompositions of $\mathcal{H}$ satisfy the adjacency property. This construction gives a simple way of constructing the convex hull of $\left\{(\bar{\boldsymbol{x}}, \boldsymbol{z}) \in \{0,1\}^n \times \mathbb{R}^n \mid z_j = \prod_{i=1}^{j} \bar{x}_i,\ \forall j \in [n]\right\}$ since $z_j = 1 - \sum_{j'=0}^{j-1} y_{j'}$.

As discussed above, Theorem 2 considers a special case where D, the extreme points of the Cartesian product of simplices, is endowed with a facial decomposition $\mathcal{F}$, and where the extended formulation of the partition indicator hull is a network flow problem. Then, using feasibility conditions, the formulation can be projected to the space of the original problem variables yielding (9). In the special case where the faces satisfy the adjacency property (Definition 4), this formulation is polynomially-sized. When modeling an instance of TEO with a single decision tree as $\text{ML}(D, \mathcal{F})$, the faces in $\mathcal{F}$ form a facial decomposition of $\text{conv}(D)$ since exactly one leaf becomes active for every choice of values for the input variables.

Remark 1. The regions associated with the leaves of a decision tree form a facial decomposition.

We will show in Section 5.1 that, when a decision tree has categorical variables, the network flow formulation of Theorem 2 gives the first polynomially-sized formulation for the convex hull of a single decision tree. For the case of numerical variables, the facial structure satisfies adjacency, and therefore, Theorem 3 provides a polynomially-sized formulation in the space of original problem variables. This yields the *bounded formulation* of Kim et al. (2019). The case where each input variable takes only two levels corresponds to the discussion about the hypercube $\mathcal{H}$ above. In this case, every facial decomposition satisfies the adjacency property. The bounded formulation thus yields the formulation of Chen and Mišić (2021).

As already alluded to, the results above allow us to model a decision tree. To do so, we choose D to be the vertices of the Cartesian product of simplices (obtained by the various levels of each input variable) and $\mathcal{F}$ to be a facial decomposition of D (where each leaf node corresponds to a face of D.) The reverse mapping, however, does not hold as we record in Remark 2 where we say that a facial decomposition $\mathcal{F}$ is *decision-tree-representable* if there exists a decision tree having a leaf for each $F \in \mathcal{F}$.

REMARK 2. Not every facial decomposition is decision-tree-representable.

Figure 6 provides support for Remark 2. In this example, there is no way to partition $\{y_1, \ldots, y_5\}$ into two sets $\mathcal{F}_1$ and $\mathcal{F}_2$ such that $\sum_{F \in \mathcal{F}_i} y_F$ corresponds to a face for $i = 1, 2$. This contrasts with Theorem 1 that establishes that $\mathcal{F}$ can always be represented through a tree ensemble.

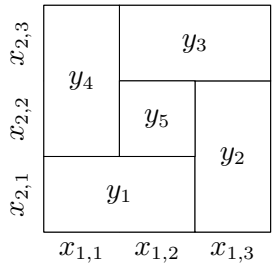


Figure 6 Facial decomposition that is not decision-tree representable.

4.2. Completability: Construction of a Facial Decomposition

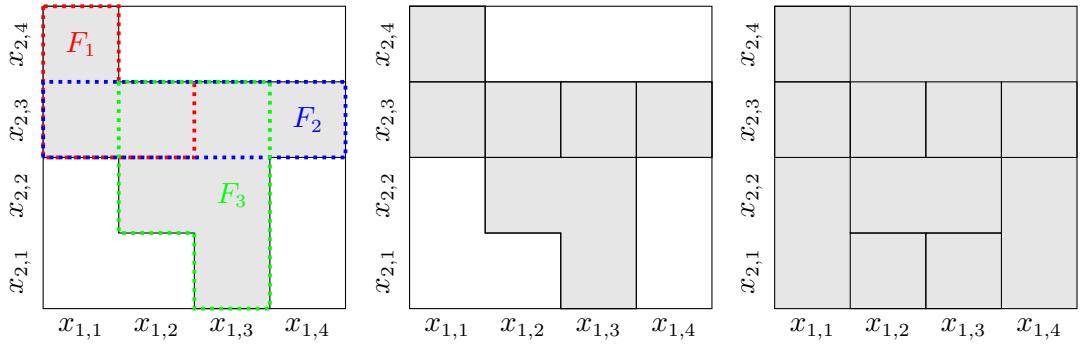
A given set of multilinear terms $\mathcal{F}$ in (4) may not describe a facial decomposition of the Cartesian product of simplices P as the faces may not cover $D := \text{vert}(P)$ and/or may overlap with one another. Given a set of faces $\mathcal{F}$ in P that meet certain technical requirements, we next construct a facial decomposition of P that can be used to derive an extended formulation for $\text{conv}(\text{ML}(D, \mathcal{F}))$. Such a facial decomposition could clearly be that which makes each vertex of P a face of $\mathcal{F}$, although this leads to exponentially-sized formulations. We show in Theorem 4 that, in general, there may not exist such a facial decomposition $\mathcal{F}'$ where the number of faces of $\mathcal{F}'$ is polynomial in n, $K^{\max}$, and $|\mathcal{F}|$ where $K^{\max} := \max_{v \in [n]} K_v$. However, we then identify special cases where such

a polynomially-sized facial decomposition exists and provide algorithms to construct it. We now expand on the relationship between $\mathcal{F}$ and the desired facial decomposition, $\mathcal{F}'$.

DEFINITION 5. Let $\mathcal{F}$ be a set of integral polytopes in an integral polytope P. We say that $\mathcal{F}'$ is a *refinement* of $\mathcal{F}$ if (i) $\mathcal{F}'$ is a set of disjoint integral polytopes in P and (ii) every $F \in \mathcal{F}$ is the convex hull of a union of elements of $\mathcal{F}'$.

DEFINITION 6. For an integral polytope P, we say that $\mathcal{F}'$ is a *P-completion* of a set $\mathcal{F}$ of integral polytopes in P if (i) $\mathcal{F}'$ is a refinement of $\mathcal{F}$ and (ii) $\mathcal{F}'$ is a facial decomposition of P.

Figure 7 shows an example of a refinement (b) and a P-completion (c) where $P = \Delta^4 \times \Delta^4$, $\mathcal{F} = \{F_1, F_2, F_3\}$, $\mathbb{1}_{F_1}(\boldsymbol{x}) = (x_{1,1} + x_{1,2})x_{2,3} + x_{1,1}x_{2,4}$, $\mathbb{1}_{F_2}(\boldsymbol{x}) = x_{2,3}$, and $\mathbb{1}_{F_3}(\boldsymbol{x}) = (x_{1,2} + x_{1,3})(x_{2,2} + x_{2,3}) + x_{1,3}x_{2,1}$.



(a) A set of integral polytopes $\mathcal{F}$ (b) A refinement of $\mathcal{F}$ (c) A P-completion of $\mathcal{F}$

Figure 7 Refinement and completion, using a grid representation.

When $\mathcal{F}'$ is a refinement of $\mathcal{F}$, the variables y_F of $\mathrm{ML}(D, \mathcal{F})$ can be obtained as an affine transformation of the variables $y_{F'}$ of $\mathrm{ML}(D, \mathcal{F}')$. It follows, as we record in Proposition 3, that an extended formulation for $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F}))$ can be obtained from an LP description for $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F}'))$.

PROPOSITION 3. *Let P be the Cartesian product of n simplices and let $D = \mathrm{vert}(P)$. Let $\mathcal{F}$ be a set of integral polytopes in P and $\mathcal{F}'$ be a refinement of $\mathcal{F}$. If there is an LP formulation T for $\mathrm{conv}(ML(D, \mathcal{F}'))$, then there is an LP formulation for $\mathrm{conv}(ML(D, \mathcal{F}))$, which adds $|\mathcal{F}|$ variables and $|\mathcal{F}|$ linear constraints to T.*

The proof of Proposition 3 is constructive. The size of the formulation obtained for $\text{conv}(\text{ML}(D,\mathcal{F}))$ directly relates to that of $\text{conv}(\text{ML}(D,\mathcal{F}'))$. Since Proposition 1 presents an LP formulation for any facial decomposition $\mathcal{F}'$ that is polynomially-sized in n, $K^{\max}$, and $|\mathcal{F}'|$, the construction of Proposition 3 is small if we use a P-completion $\mathcal{F}'$ of $\mathcal{F}$ that has cardinality polynomial in n, $K^{\max}$, and $|\mathcal{F}|$, leading to

DEFINITION 7. A P-completion $\mathcal{F}'$ of a set of faces $\mathcal{F}$ in P is *polynomially-sized* if $|\mathcal{F}'|$ is polynomial in n, $K^{\max}$, and $|\mathcal{F}|$.

Theorem 4 shows that polynomially-sized P-completion might not exist.

THEOREM 4. *There is a family of Cartesian products P_n of n simplices, each with exactly two vertices, and a set $\mathcal{F}_n$ of faces in P_n such that there is no P_n-completion of $\mathcal{F}_n$ whose size is bounded above by a polynomial of n and $|\mathcal{F}_n|$.*

We next introduce cases in which a polynomially-sized LP formulation for $\text{conv}(\text{ML}(D,\mathcal{F}))$ can be derived by constructing a polynomially-sized P-completion of $\mathcal{F}$. We say that a set of integral polytopes $\mathcal{F}$ of P is *P-poly-completable* if there exists a polynomially-sized P-completion $\mathcal{F}'$ of $\mathcal{F}$.

PROPOSITION 4. *Let P be the Cartesian product of n simplices and let $\mathcal{F}$ be a collection of vertices in P. Then, $\mathcal{F}$ is P-poly-completable and there is a completion whose size is at most $n|\mathcal{F}|+1$.*

Proposition 5 shows that $\mathcal{F}$ is P-poly-completable as long as $|\mathcal{F}|$ is constant. The idea of the proof is as follows. First, a single face is P-poly-completable by Proposition 4 because it can be reduced to a vertex in the Cartesian product of simplices after suitable simplification so as to satisfy Assumption 1. Thus, we obtain $|\mathcal{F}|$ facial decompositions where each set has cardinality no more than $n+1$ and contains one of the faces in $\mathcal{F}$. To complete the proof, we introduce the notions of common P-completion and refinement. Given sets of faces $\mathcal{F}_1,\dots,\mathcal{F}_T$ of P, $\mathcal{F}$ is a *common P-completion (resp. refinement) of $\mathcal{F}_1,\dots,\mathcal{F}_T$* if $\mathcal{F}$ is a P-completion (resp. refinement) of $\mathcal{F}_t$ for all $t\in[T]$. We argue that, from the $|\mathcal{F}|$ facial decompositions of P, we can obtain a common P-completion (from their nonempty $|\mathcal{F}|$-wise intersections) that has no more than $(n+1)^{|\mathcal{F}|}$ elements.

PROPOSITION 5. *A collection of a constant number of faces in a Cartesian product of simplices P is P-poly-completable.*

We next consider the case where $\mathcal{F}$ can be expressed by an arborescence, *i.e.*, a directed graph in which, for a node u called the *root* and any other node v, there is exactly one directed path from u to v. We say that $\mathcal{F}$ has the *arborescence property* if either $F_i \subseteq F_j$, $F_j \subseteq F_i$, or $F_i \cap F_j = \emptyset$ holds for all $F_i, F_j \in \mathcal{F}$. When $\mathcal{F}$ has the arborescence property, we can define *the graph of* $\mathcal{F}$ as a directed graph G with $V(G) = \mathcal{F} \cup \{P\}$ and $E(G) = \{(v,w) \in V(G)^2 \mid w \subsetneq v$ and there is no $u \in V(G)$ such that $w \subsetneq u \subsetneq v\}$. We say that *a face F_i (or its corresponding node) covers a face F_j (or its corresponding node)* if there is an arc (F_i, F_j) in the graph of $\mathcal{F}$.

LEMMA 1. *Assume that $\mathcal{F}$ has the arborescence property. Then, the graph of $\mathcal{F}$ is an arborescence whose root is P.*

Consider now the graph G of a set of faces having the arborescence property. We define the set of outneighbors of node v as $\mathcal{N}^+_G(v) := \{w \in V(G) \mid (v,w) \in E(G)\}$. Let $u, w \in \mathcal{N}^+_G(v)$ be distinct. Then, $u \not\subseteq w$ and $w \not\subseteq u$. Therefore, $u \cap w = \emptyset$. Also, if $w \in \mathcal{N}^+_G(v)$ then $w \subsetneq v$. Therefore, for all $v \in V(G)$, $\bigcup_{w \in \mathcal{N}^+_G(v)} w \subseteq v$. If $\bigcup_{w \in \mathcal{N}^+_G(v)} w = v$ holds for all $v \in V(G)$, then the set of leaves of G, $\{v \in V(G) \mid \mathcal{N}^+_G(v) = \emptyset\}$, describes a facial decomposition. In this case, every node v is the union of all the leaves reachable from v. In Theorem 5, we give a sufficient condition for P-poly-completability of $\mathcal{F}$ when $\mathcal{F}$ has the arborescence property. The proof constructs an arborescence G' satisfying $\bigcup_{w \in \mathcal{N}^+_{G'}(v)} w = v$ for all $v \in V(G')$ by adding a polynomial number of nodes to the arborescence of $\mathcal{F}$. We will, with slight abuse of notation, say that a set of nodes in the arborescence of $\mathcal{F}$ is v-poly-completable if there is a v-completion of the faces associated with the set of nodes. We will also say that two nodes in the graph satisfy the arborescence property if one of the corresponding faces is contained in the other or if the faces are disjoint.

THEOREM 5. *Let P be the Cartesian product of n simplices and $\mathcal{F}$ be a set of faces in P that has the arborescence property. If $\mathcal{N}^+_G(v)$ is v-poly-completable for all $v \in V(G)$ where G is the arborescence of $\mathcal{F}$, then there exists a P-completion $\mathcal{F}'$ of $\mathcal{F}$ where $|\mathcal{F}'|$ is polynomial in n, $K^{\max}$, and $|\mathcal{F}|$.*

Proposition 5 and Theorem 2 can be used to obtain polynomially-sized convex hull formulations for models where the prediction is obtained through boolean decision rules (rule sets in disjunctive normal form); see Su et al. (2016), Dash et al. (2018). This is because the trained model can be expressed as the indicator function of a constant number of faces. Finally, we mention that Angulo et al. (2015) studied a related problem regarding extended formulations for the convex hull of a subset of vertices of a polytope P. The authors show that when P is the unit hypercube $[0,1]^n$, polynomial formulations can be found and provide negative results for the case of general polytopes P. Theorem 5 provides positive results for the case of polytopes P more general than the unit hypercube, and for the case where faces are not restricted to vertices.

4.3. Decomposability

We say a set S is *decomposable into sets* S_1 *and* S_2 if $\operatorname{conv}(S) = \operatorname{conv}(S_1) \cap \operatorname{conv}(S_2)$. We study decomposability for $\mathrm{ML}(D,\mathcal{F})$ where $P_v = \Delta^{K_v}$ for all $v \in [n]$, $P = \prod_{v\in[n]} P_v$, $D = \operatorname{vert}(P)$, and $\mathcal{F}$ is a set of proper faces in P. Since P is a Cartesian product of sets, given a face F of P and $I \subseteq [n]$, we can write $F = F_I \times F_{\bar{I}}$, where F_I (resp. $F_{\bar{I}}$) is a face of $P_I := \prod_{v\in I} P_v$ (resp. $P_{\bar{I}} := \prod_{v\in \bar{I}} P_v$). Therefore, $\mathbb{1}_F(\boldsymbol{x}) = \mathbb{1}_{F_I}(\boldsymbol{x_I})\mathbb{1}_{F_{\bar{I}}}(\boldsymbol{x_{\bar{I}}})$. Since $F \in \mathcal{F}$ is a proper face of P, it is one of three types depending on whether $F_I = P_I$ and/or $F_{\bar{I}} = P_{\bar{I}}$. Using this classification, we define the *partition of* $\mathcal{F}$ *by* I, denoted as $\texttt{partition}(\mathcal{F},I)$, as the triple $(\mathcal{F}^0,\mathcal{F}^1,\mathcal{F}^2)$, where, a face $F \in \mathcal{F}^0$ (resp. $F \in \mathcal{F}^2$) satisfies $F_{\bar{I}} = P_{\bar{I}}$ (resp. $F_I = P_I$) and $\mathcal{F}^1 = \mathcal{F}\backslash(\mathcal{F}^0 \cup \mathcal{F}^2)$ consists of the remaining faces.

EXAMPLE 4. Let $P = (\Delta^2)^3 \times \Delta^3$ and $D = \operatorname{vert}(P)$. Consider

$$T = \left\{ (\boldsymbol{x},\boldsymbol{y}) \in D \times \mathbb{R}^8 \;\middle|\; \begin{array}{ll} y_1 = x_{11}x_{41}, & y_2 = x_{12}x_{21}(x_{42}+x_{43}), \\ y_3 = x_{11}x_{42}, & y_4 = x_{11}x_{43}, \\ y_5 = x_{12}x_{41}, & y_6 = x_{12}x_{22}(x_{42}+x_{43}) \\ y_7 = x_{11}x_{31}, & y_8 = x_{11}x_{32} \end{array} \right\}.$$

For $j \in [8]$, let F_j be the face of P corresponding to y_j. Then, $T = \mathrm{ML}(D,\mathcal{F})$ with $\mathcal{F} = \{F_1,\ldots,F_8\}$. Let $I = \{1,2,3\}$. For F_1, $\mathbb{1}_{(F_1)_I}(\boldsymbol{x}) = x_{11}$ and $\mathbb{1}_{(F_1)_{\bar{I}}}(\boldsymbol{x}) = x_{41}$. For F_7, $\mathbb{1}_{(F_7)_I}(\boldsymbol{x}) =$

$x_{11}x_{31}$ and $\mathbb{1}_{(F_7)_{\bar{I}}}(\boldsymbol{x}) = 1$, *i.e.*, $(F_7)_{\bar{I}} = P_{\bar{I}}$. Proceeding similarly with the other faces, we obtain that $\texttt{partition}(\mathcal{F}, I) = (\mathcal{F}^0, \mathcal{F}^1, \mathcal{F}^2)$ where $\mathcal{F}^0 = \{F_7, F_8\}$, $\mathcal{F}^1 = \{F_1, \ldots, F_6\}$, and $\mathcal{F}^2 = \emptyset$. In short, $\mathcal{F}^0$ (resp. $\mathcal{F}^2$) are the multilinear terms that do not involve variables $x_{4\cdot}$ (resp. $x_{i\cdot}$ for any $i \in I$). ■

In the following result, we show that a multilinear set can sometimes be decomposed into two multilinear sets, by partitioning the variables into $(x_I, x_{\bar{I}})$. Then, we introduce variables for multilinear terms that depend on x_I and are involved in expressions involving both sets of variables. For a concrete illustration, in Example 4, if $I = \{1, 2, 3\}$, in order to separate the dependence of y_6 on $\boldsymbol{x_I} = \{x_1, x_2, x_3\}$ and $\boldsymbol{x_{\bar{I}}} = \{x_4\}$, we introduce a variable for $x_{12}x_{22}$. When it can be arranged that the introduced variables correspond to indicators of disjoint faces, possibly after refinement, Theorem 6 shows that the multilinear set is decomposable.

THEOREM 6. *Consider a set of proper faces $\mathcal{F}$ of $P := \prod_{v\in[n]} P_v$ where $P_v := \Delta^{K_v}$ for all $v \in [n]$ and $I \subseteq [n]$. Express each $F \in \mathcal{F}$ as $F_I \times F_{\bar{I}}$ where F_I (resp. $F_{\bar{I}}$) is a face of P_I (resp. $P_{\bar{I}}$). Let $(\mathcal{F}^0, \mathcal{F}^1, \mathcal{F}^2) = \texttt{partition}(\mathcal{F}, I)$, $\mathcal{F}^1_I := \bigcup_{F\in\mathcal{F}^1}\{F_I\}$, and assume that $\mathcal{F}'_I$ is a refinement of $\mathcal{F}^1_I$. For each $F \in \mathcal{F}^1_I$, let $C(F)$ be the set of faces in $\mathcal{F}'_I$ that refine F, i.e., $C(F) \subseteq \mathcal{F}'_I$ such that $\operatorname{conv}\big(\bigcup_{\hat{F}\in C(F)} \hat{F}\big) = F$. Introduce $\boldsymbol{z} \in \{0,1\}^{|\mathcal{F}'_I|}$ so that $\mathbf{1}^\intercal \boldsymbol{z} \le 1$ and, for every $F \in \mathcal{F}'_I$, let $idx(F)$ be the index of F in $\mathcal{F}'_I$. Then*

$$ML(D, \mathcal{F}) = \operatorname{proj}_{\boldsymbol{x},\boldsymbol{y}} \left\{ (\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \,\middle|\, (\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in ML(D_I, \mathcal{F}^0_I \cup \mathcal{F}'_I), (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in M \right\},$$

where $\mathcal{F}^0_I = \{F_I\}_{F\in\mathcal{F}^0}$, $\boldsymbol{y^0}, \boldsymbol{y^1}, \boldsymbol{y^2}$ correspond to indicators of $\mathcal{F}^0, \mathcal{F}^1, \mathcal{F}^2$, respectively, and

$$M := \left\{ (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y}^1, \boldsymbol{y}^2) \;\middle|\; \begin{array}{ll} \boldsymbol{x_v} \in \Delta^{K_v}_{0,1}, & v \in \bar{I}, \\ \boldsymbol{z} \in \{0,1\}^{|\mathcal{F}'_I|}, \mathbf{1}^\intercal \boldsymbol{z} \le 1, & \\ y^1_F = \mathbb{1}_{F_{\bar{I}}}(\boldsymbol{x}) \displaystyle\sum_{\bar{F}\in C(F_I)} z_{idx(\bar{F})}, & F \in \mathcal{F}^1 \\ y^2_F = \mathbb{1}_{F_{\bar{I}}}(\boldsymbol{x}), & F \in \mathcal{F}^2 \end{array} \right\}.$$

Then, $\operatorname{conv}(ML(D, \mathcal{F})) = \operatorname{proj}_{\boldsymbol{x},\boldsymbol{y}} \big\{ (\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \mid (\boldsymbol{x_I}, \boldsymbol{z}) \in \operatorname{conv}(ML(D_I, \mathcal{F}^0_I \cup \mathcal{F}'_I)), (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y}) \in \operatorname{conv}(M) \big\}.$

In words, Theorem 6 shows that it suffices to convexify $\mathrm{ML}(D_I, \mathcal{F}_I^0 \cup \mathcal{F}_I')$ and M separately in order to convexify $\mathrm{ML}(D, \mathcal{F})$. Further, since M can be expressed equivalently as $M = \mathrm{ML}\left(D_{\bar{I}} \times \Delta_{0,1}^{|\mathcal{F}_I'|+1}, \hat{\mathcal{F}}^2 \cup \hat{\mathcal{F}}^1\right)$, where $\hat{\mathcal{F}}^2 = \bigcup_{F \in \mathcal{F}^2} F_{\bar{I}} \times \Delta^{|\mathcal{F}_I'|+1}$, $\hat{\mathcal{F}}^1 = \bigcup_{F \in \mathcal{F}^1} \{F_{\bar{I}} \times \mathrm{conv}\left(\bigcup_{\bar{F} \in C(F)} \{e_{\mathrm{idx}(\bar{F})}\}\right)\}$, and $e_i \in \{0,1\}^{|\mathcal{F}_I'|+1}$ is the i^{th} principal vector, then M is a multilinear set of the form studied in Section 4.1.

EXAMPLE 5. Consider sets T and I defined in Example 4. Set $\mathcal{F}_I^1$ is the set of faces in P_I corresponding to $\{x_{11}, x_{12}, x_{12}x_{21}, x_{12}x_{22}\}$. The set of faces in P_I corresponding to $\{x_{11}, x_{12}x_{21}, x_{12}x_{22}\}$ is a P_I-completion of $\mathcal{F}_I^1$ that we use as $\mathcal{F}_I'$. This set $\mathcal{F}_I'$ is exactly that which would have been obtained using Theorem 5 since $\mathcal{F}_I^1$ satisfies the arborescence property and the internal nodes of the graph of $\mathcal{F}_I^1$ are completable by Proposition 4. Introducing variables $\boldsymbol{z} \in \Delta_{0,1}^3$, as specified in Theorem 6, to indicate $z_1 = x_{11}$, $z_2 = x_{12}x_{21}$, and $z_3 = x_{12}x_{22}$, we obtain that $\mathrm{conv}(T)$ is described as

$$\mathrm{proj}_{\boldsymbol{x},\boldsymbol{y}} \left\{ (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \;\middle|\; \begin{array}{l} \boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3} \in \Delta^2, \boldsymbol{x_4} \in \Delta^3, \boldsymbol{y} \in \Delta^8, \boldsymbol{z} \in \mathbb{R}^3_+, \mathbf{1}^\intercal \boldsymbol{z} \leq 1 \\ y_7 + y_8 \leq x_{11}, z_2 + z_3 \leq x_{12}, z_2 \leq x_{21}, z_3 \leq x_{22}, y_7 \leq x_{31}, y_8 \leq x_{32}, z_1 = y_7 + y_8 \\ y_1 + y_5 \leq x_{41}, y_3 \leq x_{42}, y_4 \leq x_{43}, y_2 + y_3 + y_4 + y_6 \leq x_{42} + x_{43} \\ y_1 + y_3 + y_4 \leq z_1, y_2 \leq z_2, y_6 \leq z_3, y_2 + y_5 + y_6 \leq z_2 + z_3 \end{array} \right\}.$$

In obtaining this result, we use Theorem 3 twice to construct the convex hull over $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, y_7, y_8, \boldsymbol{z})$ and over $(\boldsymbol{x_4}, \boldsymbol{z}, y_1, \ldots, y_6)$. This is because (y_7, y_8, z_2, z_3) (resp. $(y_1, \ldots, y_6)$) are the indicators of a facial decomposition over $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3})$ (resp. over $(\boldsymbol{x_4}, \boldsymbol{z})$) and both satisfy the adjacency property. The facial decomposition corresponding to (y_7, y_8, z_2, z_3) refines the faces corresponding to $(y_7, y_8, \boldsymbol{z})$ because $z_1 = y_7 + y_8$. ■

More generally, Theorem 6, when combined with the earlier results derived in Theorem 2 and Theorem 5, provides a powerful and systematic framework to construct convex hull descriptions for many multilinear sets. In particular, it is sufficiently general and versatile to provide alternate derivations and extensions of many results in the literature. As is commonly done, we may represent multilinear terms of a problem using a hypergraph where each vertex corresponds to a problem variable and where each hyperedge is incident to the variables nodes occurring in its corresponding

monomial. As a first example, this framework can be used to derive polynomially-sized descriptions of convex hulls of certain multilinear sets whose underlying hypergraphs are laminar. This provides an alternate proof of Theorem 10 in Del Pia and Khajavirad (2018a). As a second example, this framework can be used to derive Theorem 1 in Del Pia and Khajavirad (2018b), a result that states that a convex hull description of the multilinear set associated with a hypergraph H can be obtained from convex hull descriptions of the multilinear sets associated with two smaller sub-hypergraphs H_1 and H_2 if their overlap is a hyperclique. As a third example, Theorem 3.5 in Bienstock and Munoz (2018) can be viewed as a special case of this framework. In particular, it allows for the construction of polynomially-sized formulations for the convex hull of multilinear sets whose structural sparsity is described by a tree-decomposition, when the tree-width is constant. A detailed discussion of these derivations is provided in APDX.2.3. We however stress that the framework is in fact a strict generalization in that it allows for polynomially-sized formulations to be obtained for multilinear sets for which earlier results do not directly apply; see Example APDX.1.

5. Applications

In this section, we apply our results to build improved formulations for specific optimization problems. In Section 5.1, we build an improved model for a standard decision tree using the results in Section 4. In Section 5.2, we perform computational experiments that establish the benefits of our new formulations. In Section 5.3, we propose a new prescriptive model, *multilinear decision trees*, which outperforms classical models in approximating various nonlinear functions.

5.1. Improved Formulations for TEO

In this section, we provide mixed-integer linear programming (MILP) formulations for TEO that are provably tighter than those in the literature. By Remark 1, the leaves in a decision tree correspond to a facial decomposition. We can therefore rewrite TEO as

$$\max \left\{ \sum_{t\in[T]} \sum_{\ell\in\mathbf{leaves}(t)} p_{t,\ell} y_{t,\ell} \;\middle|\; (\boldsymbol{x}, \boldsymbol{y_t}) \in \mathrm{ML}(D, \mathcal{F}_t), \forall t \in [T] \right\},$$

where $D = \prod_{v\in[n]} \Delta_{0,1}^{K_v}$ and for all $t \in [T]$, $\mathcal{F}_t$ is the collection of faces corresponding to $y_{t,\ell}$ for all $\ell \in \mathbf{leaves}(t)$. The MILP formulation we propose uses the constraints of the convex hull of $\mathrm{ML}(D, \mathcal{F}_t)$, known because of Proposition 1, for each $t \in [T]$ to obtain

$$\max \quad \sum_{t\in[T]} \sum_{\ell\in\mathbf{leaves}(t)} p_{t,\ell} y_{t,\ell} \tag{12a}$$

$$\text{s.t.} \quad \sum_{\ell\in\mathbf{leaves}(t):J^v_{t,\ell}\subseteq J} y_{t,\ell} \le \sum_{j\in J} x_{v,j}, \qquad \forall t \in [T],\ \forall v \in [n],\ \forall \emptyset \neq J \subseteq [K_v], \tag{12b}$$

$$\boldsymbol{x_v} \in \Delta_{0,1}^{K_v}, \qquad \forall v \in [n], \tag{12c}$$

$$\boldsymbol{y_t} \in \Delta^{|\mathbf{leaves}(t)|}, \qquad \forall t \in [T]. \tag{12d}$$

The size of this formulation is exponential in $\{K_v\}_{\forall v\in[n]}$. However, as we discussed in Section 4.1, there is a polynomial-time separation algorithm for (12b); see Proposition APDX.1. Alternatively, an extended polynomially-sized formulation follows directly from Theorem 2.

COROLLARY 1. *Formulation (12) is integral and defines the convex hull of the problem when $T = 1$.*

It follows from Corollary 1 that formulation (12) is tightest possible if we focus on each tree separately. Thus, the LP relaxation of (12) is tighter than that of Mišić (2020), which models each $\mathrm{ML}(D, \mathcal{F}_t)$ with a system that does not describe its convex hull. We generate TEOs using various combination of real data sets, ensemble models, and number of trees and compare the LP relaxation objective values of two formulations. To compare the tightness of these formulations, we use *relative LP gap*. Given an optimization problem instance and a formulation f, we denote by δ_f the absolute deviation between the optimal objective value and the LP relaxation objective value. The relative LP gap of f with base formulation f_b is computed as $\frac{\delta_f}{\delta_{f_b}}$. Table 1 displays relative LP gaps of formulation (2) in Mišić (2020) where formulation (12) is used to as the basis of comparison. It shows that the relative LP gaps of formulation (2) in Mišić (2020) are between 1.20 and 4.65. Our formulation also has the advantage of only requiring a collection of faces, and not the decision tree associated with the facial set.

Table 1 Comparison of tightness of (a) formulation (12) and (b) formulation (2) in Mišić (2020) for TEO. Two data sets are considered. For each data set, two kinds of tree ensemble models are obtained with various number of trees (T). We compute the relative LP gap for each formulation where (a) is used as the basis of comparison.

Dataset	Ensemble model	T	Relative LP gap	
			(a)	(b)
Concrete strength (Yeh 1998)	Random forest	100	1.00	3.95
		300	1.00	4.65
		500	1.00	4.44
	Boosted trees	100	1.00	4.57
		300	1.00	1.79
		500	1.00	1.76
Wine quality (Cortez et al. 2009)	Random forest	100	1.00	1.20
		200	1.00	1.20
		300	1.00	1.20
	Boosted trees	100	1.00	1.76
		200	1.00	1.71
		300	1.00	1.64

When every input variable of the tree ensemble is numerical, the faces corresponding to the leaves satisfy the adjacency property. Therefore, by Theorem 3, we can obtain a formulation that is polynomially-sized in the input parameters (T, n, and $K_v, \forall v \in [n]$) by replacing (12b) with

$$\sum_{\ell \in \mathbf{leaves}(t): J^v_{t,\ell} \subseteq [a..b]} y_{t,\ell} \leq \sum_{j=a}^{b} x_{v,j}, \qquad \forall t \in [T],\ \forall v \in [n],\ \forall a, b \subseteq [K_v] : a \leq b. \tag{13}$$

This special case has been studied. In Kim et al. (2019), we described an equivalent formulation when all the input variables are numerical and showed its tightness. Chen and Mišić (2021) considered a special case where all input variables are binary and show the tightness of this formulation. Formulation (12) and Corollary 1 generalize these results.

5.2. Constrained TEO

In this section, we consider TEO with additional constraints and compare the computational performance of the formulations we propose with existing formulations for piecewise-linear functions from the literature. To do so, we consider instances of multi-commodity transportation problems generated in a manner similar to Vielma et al. (2010) except that the cost function is described as a tree ensemble. The transportation problem is defined over the complete bipartite graph $G = (U, V, E)$, where U (resp. V) refers to the set of the supply (resp. demand) nodes and E refers to

the set of arcs. For each commodity $i \in [C]$ where C is the number of commodities, a supply (resp. demand) amount $s_{u,i}$ (resp. $d_{v,i}$) is assigned to each supply (resp. demand) node $u \in U$ (resp. $v \in V$). An individual capacity denoted by $c_{e,i}$ is generated for each arc $e \in E$ and for each commodity $i \in [C]$. We introduce variable $z_{e,i} \in \mathbb{R}$ to represent the flow of commodity $i \in [C]$ on arc $e \in E$. The transportation cost incurred on arc $e \in E$ is given as the sum of the values of pre-trained decision trees defined over common independent variables, $\boldsymbol{z_e} = (z_{e,1}, \ldots, z_{e,C})$, and denoted as $f_e(\boldsymbol{z_e}) = \sum_{t \in [T_e]} f_{e,t}(\boldsymbol{z_e})$ where T_e is the number of trees describing the cost function on arc e, and $f_{e,t}(\boldsymbol{z_e})$ is the t^{th} decision tree function for arc $e \in E$. The objective function is the sum of independent arc costs, $\sum_{e \in E} f_e(\boldsymbol{z_e})$. This problem is a constrained TEO where we minimize the sum of values of pre-trained decision trees under balance and capacity constraints on the input variables $\boldsymbol{z}$.

We compare our formulation (TEO) with four other formulations – (TEOM), (MC), (DCC), and (DLog) – from the literature. Clearly, f_e is modeled as a sum of piecewise constant functions, one for each tree t in the ensemble that models f_e. Such a problem can be formulated using models for piecewise-constant functions. However, our formulations will directly use the tree ensemble representation of f_e. The explicit formulations are given in APDX.3.2. Each formulation models f_e independently of $f_{e'}$ for distinct arcs $e, e' \in E$. From now, we fix e. (TEO), the formulation we propose, is obtained by formulating $(\boldsymbol{x_e}, \boldsymbol{y_{e,t}}) \in \text{ML}(D_e, \mathcal{F}_{e,t})$ for $t \in [T_e]$, where D_e and $\{\mathcal{F}_{e,t}\}_{t \in [T_e]}$ are defined from the pre-trained tree ensemble model for f_e. Variable $\boldsymbol{y_{e,t}}$ indicates whether the corresponding leaf is active for tree $t \in [T_e]$. Variable $\boldsymbol{x_e}$ indicates the hyper-rectangle that contains $\boldsymbol{z_e}$ and is contained in the region corresponding to each active leaf. In (TEO), for each $(\boldsymbol{x_e}, \boldsymbol{y_{e,t}}) \in \text{ML}(D_e, \mathcal{F}_{e,t})$, where $t \in [T]$, we include the constraints in Theorem 3 describing $\text{conv}(\text{ML}(D_e, \mathcal{F}_{e,t}))$ except redundant ones identified in Proposition 2. (TEOM) is a formulation for TEO problems, which extends formulation (2) for a tree ensemble proposed in Mišić (2020) in a manner similar to Mistry et al. (2021), by connecting $\boldsymbol{z}_e$ variables to the variables in the tree ensemble model. The difference between (TEO) and (TEOM) is that (TEO) describes $\text{conv}(\text{ML}(D_e, \mathcal{F}_{e,t}))$ but (TEOM) does not. It follows that the LP relaxation of (TEO) is tighter than that of (TEOM). This is

confirmed by our experiments with the formulations reported in Table 3 where we find that the relative performance of (TEOM) worsens as the depth of trees in the ensemble increases.

The next three formulations –(MC), (DCC), and (DLog)– are based on general modeling techniques for piecewise-linear functions. Let $m_{e,t}$ be the number of leaves in the t^{th} tree in the ensemble model corresponding to arc e. Both (MC) and (DCC) (see Vielma (2015) for details) include a disjunctive constraint $\boldsymbol{y} \in \Delta_{0,1}^{m_{e,t}}$, where y_j indicates whether point $(z_{e,1}, \ldots, z_{e,C})$ is contained in the hyper-rectangle corresponding to the j^{th} leaf. They differ in the way they describe pieces: (MC) uses an $\mathcal{H}$-representation of the pieces whereas (DCC) uses their $\mathcal{V}$-representation. Formulation (DLog) (as in Ibaraki (1976), Vielma and Nemhauser (2011)) uses a $\mathcal{V}$-representation similar to DCC except that it only requires $\lceil \log_2 m_{e,t} \rceil$ binary variables to express the choice among the $m_{e,t}$ pieces.

Next we compare, in Table 2, the size and tightness of formulations for a constrained TEO with a single decision tree with n independent variables and m leaves. (TEO), (TEOM), and (MC) are polynomial-sized formulations, whereas the number of continuous variables in (DCC) and (DLog) is exponential in n. We remark that (TEO), (MC), (DCC), and (DLog) capture the convex hull of a single decision tree. For problems involving many trees, the bound from (TEO) is the tightest and we use it as the basis of comparison in Table 3.

Table 2 Size and tightness of formulations for modeling a single decision tree with n independent variables and m leaves. The first column represents formulations. Columns `nBinVar`, `nConVar`, and `nConstr` display the number of binary variables, continuous variables, and constraints, respectively. Column `Integrality` indicates whether every extreme point of the LP relaxation is integral or not.

Formulations	`nBinVar`	`nConVar`	`nConstr`	`Integrality`
(TEO)	$m-1$	$\mathcal{O}(nm)$	$\mathcal{O}(nm^2)$	yes
(TEOM)	$m-1$	$\mathcal{O}(nm)$	$\mathcal{O}(nm)$	no
(MC)	$m-1$	$\mathcal{O}(nm)$	$\mathcal{O}(nm)$	yes
(DCC)	$m-1$	$\mathcal{O}(m2^n)$	$\mathcal{O}(m)$	yes
(DLog)	$\lceil \log_2 m \rceil$	$\mathcal{O}(m2^n)$	$\mathcal{O}(m)$	yes

We generate two- and three-commodity transportation problem instances, *i.e.*, $C \in \{2, 3\}$ on a complete bipartite graph with five supply nodes and two demand nodes. We construct a collection of instances by varying the tree ensemble training parameters T and D, where T is the number of

Table 3 Averages of relative LP gaps for multi-commodity transportation problem instances. (C, D, T) are the parameters used in generating instances and N is the number of randomly generated instances. The 5^{th} to 9^{th} columns display the average of relative LP gaps for N randomly generated instances with parameters (C, D, T) where (TEO) is used to as the basis of comparison. The smallest average value(s) are in bold for each row.

C	D	T	N	(TEO)	(TEOM)	(MC)	(DCC)	(DLog)
2	4	1	10	**1.00**	7.47	**1.00**	**1.00**	**1.00**
2	4	3	10	**1.00**	8.28	5.18	5.18	5.18
2	4	5	10	**1.00**	9.86	8.19	8.19	8.19
2	4	7	10	**1.00**	11.99	11.51	11.51	11.51
2	4	9	10	**1.00**	13.42	13.85	13.85	13.85
2	5	1	10	**1.00**	11.67	**1.00**	**1.00**	**1.00**
2	5	3	10	**1.00**	18.07	5.95	5.95	5.95
2	5	5	10	**1.00**	17.23	7.70	7.70	7.70
2	5	7	10	**1.00**	15.47	8.13	8.13	8.13
2	5	9	10	**1.00**	16.81	9.53	9.53	9.53
3	4	1	10	**1.00**	10.38	**1.00**	**1.00**	**1.00**
3	4	3	10	**1.00**	7.78	1.62	1.62	1.62
3	4	5	10	**1.00**	8.40	1.98	1.98	1.98
3	4	7	10	**1.00**	8.52	2.18	2.18	2.18
3	4	9	10	**1.00**	9.30	2.46	2.46	2.46
3	5	1	10	**1.00**	22.97	**1.00**	**1.00**	**1.00**
3	5	3	10	**1.00**	22.51	2.57	2.57	2.57
3	5	5	10	**1.00**	30.66	4.08	4.08	4.08
3	5	7	10	**1.00**	26.64	4.05	4.05	4.05
3	5	9	10	**1.00**	27.03	4.33	4.33	4.33
Total average			200	**1.00**	15.22	4.87	4.87	4.87

* C: Number of commodities.
* D: Maximum depth of trees.
* T: Number of trees used to describe a single arc cost.

trees used to describe the cost on an arc $e \in E$ and D is the maximum depth of those trees. It follows that each instance corresponds to a constrained TEO with $10T$ trees. Given (C, D, T), we construct an instance as follows. First, we randomly generate supply, demand, and capacities. Second, for each arc $e \in E$, we define a nonlinear hidden cost function g_e, generate sample data points describing g_e, and train a random forest model using those data points. Details of the procedure used to generate the instances are provided in APDX.3.1. For each (C, T, D), we randomly construct 10 instances. All five formulations are tested on 200 instances with $C \in \{2, 3\}$, $D \in \{4, 5\}$, and $T \in \{1, 3, 5, 7, 9\}$. Since (TEO) and (TEOM) outperform the remaining formulations, we generate 20 hard instances with $(C, D, T) \in \{(2, 6, 20), (3, 6, 20)\}$ and solve them using these formulations.

Table 4 Averages of solution times for multi-commodity transportation problem instances. (C, D, T) are the parameters used in generating instances and N is the number of randomly generated instances. The 5^{th} and 9^{th} columns display the average of solution times for N randomly generated instances with parameters (C, D, T). The number of instances that reach one hour time limit is given as superscript. The smallest average value(s) are in bold for each row.

C	D	T	N	(TEO)	(TEOM)	(MC)	(DCC)	(DLog)
2	4	1	10	0.7	**0.7**	**0.7**	0.8	1.2
2	4	3	10	2.5	**2.4**	46.1	429.5	1880.8^{3}
2	4	5	10	**3.3**	3.5	608.4	3218.8^{7}	3600.0^{10}
2	4	7	10	**4.5**	4.8	2959.2^{3}	3600.0^{10}	3600.0^{10}
2	4	9	10	**4.1**	5.3	3536.3^{8}	3600.0^{10}	3600.0^{10}
2	5	1	10	2.6	2.5	**2.2**	2.5	6.0
2	5	3	10	**4.6**	6.9	578.5	3098.2^{6}	3600.0^{10}
2	5	5	10	**10.0**	14.1	3555.9^{9}	3600.0^{10}	3600.0^{10}
2	5	7	10	**17.7**	34.7	3600.0^{10}	3600.0^{10}	3600.0^{10}
2	5	9	10	**31.8**	73.6	3600.0^{10}	3600.0^{10}	3600.0^{10}
3	4	1	10	**0.8**	1.0	1.1	1.5	2.3
3	4	3	10	**3.0**	3.6	24.0	146.1	432.0
3	4	5	10	**3.8**	5.8	145.5	1681.1^{1}	3140.1^{5}
3	4	7	10	**7.1**	7.3	657.4	2861.1^{3}	3600.0^{10}
3	4	9	10	9.0	**8.0**	1457.6	3600.0^{10}	3600.0^{10}
3	5	1	10	2.3	**2.0**	2.1	4.7	14.4
3	5	3	10	**6.0**	8.2	483.7	1662.8	3051.3^{7}
3	5	5	10	**10.5**	12.5	2446.2^{3}	3523.4^{9}	3600.0^{10}
3	5	7	10	**13.6**	19.3	3600.0^{10}	3600.0^{10}	3600.0^{10}
3	5	9	10	**13.4**	19.8	3600.0^{10}	3600.0^{10}	3600.0^{10}
Total average			200	**7.6**	11.8	1545.2	2271.5	2586.4

* C: Number of commodities.
* D: Maximum depth of trees.
* T: Number of trees used to describe a single arc cost.

We perform our computational experiments on a computer running Linux Mint 19.3 with Intel i7-6700K CPU cores running at 4.00GHz and 48GB of memory. The code is written in Julia v1.6.3 with Gurobi v9.0.3 (Gurobi Optimization, LLC 2021) as an MIP solver and JuMP package v0.21.10 (Dunning et al. 2017). We set a time limit of one hour for the 200 smaller instances and three hours for the remaining instances. Table 3 compares the tightness of formulations. When $T = 1$, the LP relaxation values of (TEO), (MC), (DCC), and (DLog) are the same and tighter than (TEOM) because those formulations describe the convex hull for a single tree. However, when $T > 1$, the LP relaxation values of (TEO) are tighter than those of other formulations. This is because, in (TEO), the formulations for various trees that describe the cost on the same arc are connected

Table 5 Averages of relative LP gaps and solution times for multi-commodity transportation problem instances. (C, D, T) are the parameters used in generating instances and N is the number of randomly generated instances. The 5^{th}–6^{th} (resp. 7^{th}–8^{th}) columns display the average of relative LP gaps (resp. solution times) for N randomly generated instances with parameters (C, D, T). The number of instances that reach the three hours time limit is given as superscript. The smallest relative LP gap and average value are in bold for each row.

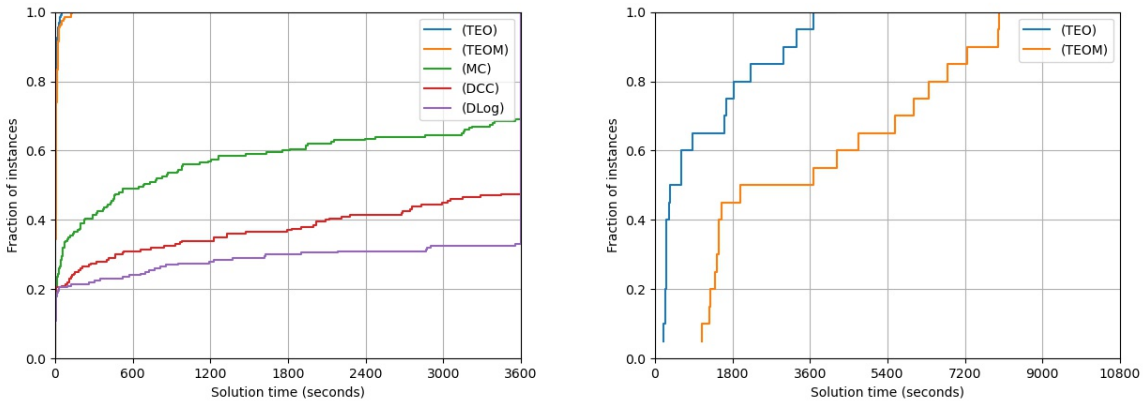
C	D	T	N	Relative LP gap		Solution time	
				(TEO)	(TEOM)	(TEO)	(TEOM)
2	6	20	10	**1.00**	25.03	**2293.8**	6538.5^{1}
3	6	20	10	**1.00**	68.59	**300.7**	1896.5

* C: Number of commodities.
* D: Maximum depth of trees.
* T: Number of trees used to describe a single arc cost.

by $\boldsymbol{x}$ variables, while these variables are not present in (MC), (DCC), and (DLog). The instances are classified using (C, D, T) and the average time taken by each formulation over 10 instances of each problem type is reported in Table 4. Clearly, (TEO) and (TEOM) outperform the other three formulations when $T > 1$. Moreover, (TEO) performs better than (TEOM) as T increases. We perform additional experiments only with (TEO) and (TEOM) with larger maximum depth $(D = 6)$ and number of trees $(T = 20)$. The results are summarized in Table 5, which shows that (TEO) performs better than (TEOM) on hard instances. A graphical vizualization of the results is presented in Figure 8 using performance profiles (Dolan and Moré 2002). In conclusion, we find that our new modeling framework outperforms existing formulations when modeling tree ensembles.

5.3. Generalized Decision Trees

Although TEO problems and multilinear optimization problems are deeply connected with one another, decision trees that predict a constant value at each node cannot model general multilinear functions. A possibly remedy is to perform regression with all interaction terms at each leaf node. However, this adds a significant number of parameters with the potential for over-fitting. In this section, inspired by the connections established earlier to multilinear optimization, we propose a generalization of decision trees where each tree can exactly represent a multilinear function, but the flexibility is limited by penalizing discontinuities at the boundaries. We show that this way of constructing decision trees provides a better fit for many nonlinear functions with/without random



(a) 200 instances.

(b) 20 hard instances.

Figure 8 Performance profiles of solution times. The horizontal axis represents the solution time and the vertical axis represents the fraction of instances solved. Figure 8a displays the performance profile of 200 instances where $C \in \{2,3\}$, $D \in \{4,5\}$, and $T \in \{1,3,5,7,9\}$. Figure 8b displays the performance profile of 20 hard instances where $C \in \{2,3\}$, $D = 6$, and $T = 20$.

noise. In this way, we expand the expressiveness of decision trees to admit multilinear functions without adversely affecting their prediction performance. With this modification, every multilinear problem can now be expressed as a generalized decision tree problem and vice-versa. We remark that the primary technique to solve TEO problems with these generalized decision trees would be to construct tight relaxations (Crama 1993, Rikun 1997, Sherali 1997, Tawarmalani and Sahinidis 2002, Luedtke et al. 2012, He and Tawarmalani 2021). The framework we have developed would, therefore, also be useful for constructing relaxations for such TEO problems.

Formally, a *generalized decision tree* (G-tree) is a piecewise-nonlinear function $f(X;T,\mathcal{G}) : \mathbb{R}^n \mapsto \mathbb{R}$, where T is a decision tree and $\mathcal{G} = \{g_\ell\}_{\ell \in \mathbf{leaves}(T)}$ is a collection of nonlinear functions defined over the hyper-rectangles corresponding to the leaves of T. Given a G-tree with T and $\mathcal{G}$, the evaluation of point $\bar{X} \in \mathbb{R}^n$ is computed as $g_{\ell(\bar{X})}(\bar{X})$ where $\ell(X)$ is the chosen leaf $\ell \in \mathbf{leaves}(T)$ corresponding to point X. A standard decision tree is a G-tree with a collection of constant functions $\mathcal{G}$. In this paper, we consider a special G-tree model, a *multilinear decision tree* (ML-tree), where $g_\ell(X)$ is multilinear in X for all $\ell \in \mathbf{leaves}(T)$.

We next introduce a two-step algorithm for training an ML-tree and then show computationally that the new model fits data generated by sampling nonlinear functions (with error) better than classical decision tree models. Given data points $\{(\bar{\boldsymbol{x}}^{\boldsymbol{i}}, \bar{y}^i)\}_{i=1}^{n_{\text{data}}}$, we first train a decision tree T using any decision tree training algorithm such as CART and, second, we find the best fitting g_ℓ for all $\ell \in \mathbf{leaves}(T)$. We next expand on the second step of the training algorithm (*i.e.*, that of finding $\{g_\ell\}_{\ell \in \mathbf{leaves}(T)}$) for an ML-tree. Let Q_ℓ be the hyper-rectangle that corresponds to leaf $\ell \in \mathbf{leaves}(T)$. The function values at vertices of a hyper-rectangle uniquely define a multilinear function (see APDX.3.3 for a formal explanation.) Therefore, to define $\mathcal{G}$ for an ML-tree, it is sufficient to specify the function value for every vertex of every hyper-rectangle Q_ℓ. Let $u_{\ell,k}$ be the function value that we will assign to the k^{th} vertex of Q_ℓ for $\ell \in \mathbf{leaves}(T)$ and $k \in [2^n]$. We denote by $g(\boldsymbol{x}; Q, \boldsymbol{s})$ the multilinear function of $\boldsymbol{x}$ over an n-dimensional hyper-rectangle Q where the function values at $\text{vert}(Q)$ are $\boldsymbol{s} \in \mathbb{R}^{2^n}$. We let $W = \bigcup_{\ell \in \mathbf{leaves}(T)} \text{vert}(Q_\ell) = \{t_1, \ldots, t_{|W|}\}$ and let $\text{idx}(\ell, k)$ be the index of the k^{th} vertex of Q_ℓ in W. We describe an optimization problem whose solution determines $\{u_{\ell,k}\}_{\ell \in \mathbf{leaves}(t), k \in [2^n]}$. Given nonnegative regularization weight parameters $\alpha, \beta \in \mathbb{R}_+$, our fitting optimization problem is

$$\min_{\boldsymbol{u},\boldsymbol{\rho},\boldsymbol{z},\boldsymbol{w}} \quad \rho_{\text{trn}} + \alpha \cdot \rho_{\text{intra}} + \beta \cdot \rho_{\text{inter}} \tag{14a}$$

$$\text{s.t.} \quad \rho_{\text{trn}} \geq \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} \left(\bar{y}^i - g(\bar{\boldsymbol{x}}^{\boldsymbol{i}}; Q_{\ell(\bar{\boldsymbol{x}}^{\boldsymbol{i}})}, \boldsymbol{u}_{\boldsymbol{\ell}(\bar{\boldsymbol{x}}^{\boldsymbol{i}})}) \right)^2, \tag{14b}$$

$$\rho_{\text{intra}} \geq \frac{1}{|\mathbf{leaves}(T)| 2^n} \sum_{\ell \in \mathbf{leaves}(T)} \sum_{k \in [2^n]} (u_{\ell,k} - z_\ell)^2, \tag{14c}$$

$$\rho_{\text{inter}} \geq \frac{1}{|\mathbf{leaves}(T)| 2^n} \sum_{\ell \in \mathbf{leaves}(T)} \sum_{k \in [2^n]} (u_{\ell,k} - w_{\text{idx}(\ell,k)})^2, \tag{14d}$$

$$\boldsymbol{u} \in \mathbb{R}^{|\mathbf{leaves}(T)| \times 2^n}, \ \rho_{\text{trn}}, \rho_{\text{intra}}, \rho_{\text{inter}} \in \mathbb{R}, \ \boldsymbol{z} \in \mathbb{R}^{|\mathbf{leaves}(T)|}, \ \boldsymbol{w} \in \mathbb{R}^{|W|}. \tag{14e}$$

Variable ρ_{trn} computes the mean squared training error. Variable ρ_{intra} sums up, over all hyper-rectangles, the squared deviations between the function values chosen for the vertices of a hyper-rectangle and their average value z_ℓ. If $\rho_{\text{intra}} = 0$, the resulting ML-tree is piecewise-constant. Variable ρ_{inter} sums up, over all vertices $t_i \in W$, the squared deviations between the function values

a vertex takes in all of the hyper-rectangles that contain it and their average value w_i. Assuming that the tree structure is given, Problem (14) is a convex quadratic programming problem.

We perform computational experiments to compare the ML-tree model with a standard decision tree model. Given a data set, we train three ML-tree models with $(\alpha, \beta) \in \{(\infty, 0), (0, \infty), (0.1, 0.1)\}$. All three models obtain the tree-structure using the CART training algorithm. Let (α, β)-ML-tree be the ML-tree model using weights α, β in solving (14). The $(\infty, 0)$-ML-tree is identical to the decision tree returned in the first step of the training algorithm where the prediction value at each leaf is the average of dependent variable values for all points contained in the leaf node.

In the experiments, we generate training, validation, and testing data-sets. We train each model by varying the maximum depth of the decision tree from 2 to 7. We choose the depth whose validation error is the smallest. Then, we compute the test error using the chosen model. The data set is composed of 15 nonlinear functions. Table 6 shows the test errors of the three models when the data is generated without/with Gaussian noise. The results show that the testing error on $(0, \infty)$ and $(0.1, 0.1)$-ML-tree is smaller than that for the standard $(\infty, 0)$-ML-tree. This demonstrates that piecewise-multilinear functions provide better fits without significant extra effort when compared to piecewise-constant functions. Incorporating such models within tree ensemble optimization problems leads to general multilinear optimization problems with continuous as well discrete variables. Our framework can be used to develop tight relaxations for these problems.

6. Conclusion

This paper investigates and exposes new connections between multilinear optimization problems and tree ensemble optimization problems. In particular, we show that multilinear optimization problems involving discrete variables are polynomially reducible to TEO problems and vice-versa. The paper then uses insights from convexification of multilinear functions to improve relaxations for tree ensemble optimization problems and uses tree ensemble representations to develop new convex relaxations for multilinear polytopes. We show that our framework generalizes various existing results in the literature on both of these problem settings, yielding better formulations. We

Table 6 Test errors of (α,β)-ML-tree models with $(\alpha,\beta)\in\{(\infty,0),(0,\infty),(0.1,0.1)\}$. Column $h(\boldsymbol{x})$ displays the nonlinear functions used in generating the data set. The 3^{rd} to 5^{th} (resp. 6^{th} to 8^{th}) columns show the test errors of (α,β)-ML-tree when using the data set without (resp. with) Gaussian noise. The second row displays the parameters (α,β) used when training an ML-tree.

No.	$h(\boldsymbol{x})$	Without Gaussian noise			With Gaussian noise		
		$(\infty,0)$	$(0,\infty)$	$(0.1,0.1)$	$(\infty,0)$	$(0,\infty)$	$(0.1,0.1)$
1	$\prod_{i=1}^{n} x_i$	0.0024	**0.0000**	0.0004	0.0046	**0.0035**	0.0036
2	$\sum_{i=1}^{n} x_i^2$	0.0116	**0.0048**	0.0092	0.0194	**0.0147**	0.0160
3	$\sum_{i=1}^{n} x_i^3$	0.0137	**0.0058**	0.0110	0.0201	**0.0155**	0.0171
4	$\sqrt{\sum_{i=1}^{n} x_i^2}$	0.0050	**0.0021**	0.0037	0.0093	**0.0072**	0.0076
5	$\sqrt[3]{\sum_{i=1}^{n} x_i^3}$	0.0042	**0.0022**	0.0032	0.0076	**0.0060**	0.0062
6	$\sum_{i=1}^{n} \sqrt{x_i}$	0.0149	**0.0029**	0.0097	0.0208	0.0201	**0.0190**
7	$\sum_{i=1}^{n} \sqrt[3]{x_i}$	0.0193	**0.0060**	0.0137	0.0238	0.0231	**0.0223**
8	$\exp(\sum_{i=1}^{n} x_i/n)$	0.0042	**0.0002**	0.0025	0.0084	**0.0060**	0.0064
9	$\exp(\sum_{i=1}^{n} x_i^2/n)$	0.0046	**0.0018**	0.0036	0.0079	**0.0062**	0.0067
10	$\exp(\prod_{i=1}^{n} x_i)$	0.0024	**0.0013**	**0.0013**	0.0070	**0.0060**	0.0061
11	$\log(1+\sum_{i=1}^{n} x_i)$	0.0037	**0.0004**	0.0023	0.0077	**0.0060**	**0.0060**
12	$\log(1+\sum_{i=1}^{n} x_i^2)$	0.0047	**0.0021**	0.0036	0.0080	**0.0060**	0.0064
13	$\log(1+\prod_{i=1}^{n} x_i)$	0.0019	**0.0001**	0.0005	0.0033	**0.0025**	0.0026
14	$\frac{x_1^2+x_2^2+x_1-x_2+1}{(x_3-1.5)(x_4-1.5)}$	0.0518	**0.0124**	0.0214	0.0672	**0.0466**	0.0485
15	$\frac{\exp(x_1x_2x_3x_4)}{x_1^2+x_2^2-x_3x_4+3}$	0.0018	0.0011	**0.0010**	0.0018	0.0011	**0.0010**

perform computational experiments to show that our formulations for TEO are tighter and have distinct computational advantages over existing formulations based on the modeling of piecewise-linear functions and/or existing tree ensemble formulations. Finally, we conclude by proposing a training model similar to decision-trees but inspired by the observation that TEO models are less expressive than multilinear problems involving continuous variables. We show that our proposed multilinear-decision-tree model fits nonlinear functions better and reduces testing error significantly. The use of such generalized decision trees in optimization models would further couple multilinear optimization and TEO.

Acknowledgments

The research presented in this paper was supported by grants 1727989 and 1917323 from the National Science Foundation Division of Civil, Mechanical and Manufacturing Innovation.

References

Anderson R, Huchette J, Ma W, Tjandraatmadja C, Vielma JP (2020) Strong mixed-integer programming formulations for trained neural networks. *Mathematical Programming* 183(1):3–39, URL http://dx.

doi.org/10.1007/s10107-020-01474-5.

Angulo G, Ahmed S, Dey SS, Kaibel V (2015) Forbidden vertices. *Mathematics of Operations Research* 40(2):350–360, URL http://dx.doi.org/10.1287/moor.2014.0673.

Balas E (1985) Disjunctive programming and a hierarchy of relaxations for discrete optimization problems. *SIAM Journal on Algebraic Discrete Methods* 6(3):466–486, URL http://dx.doi.org/https://doi.org/10.1137/0606047.

Balas E (1998) Disjunctive programming: Properties of the convex hull of feasible points. *Discrete Applied Mathematics* 89(1-3):3–44, URL http://dx.doi.org/10.1016/S0166-218X(98)00136-X.

Baltean-Lugojan R, Misener R (2018) Piecewise parametric structure in the pooling problem: from sparse strongly-polynomial solutions to NP-hardness. *Journal of Global Optimization* 71(4):655–690, URL http://dx.doi.org/10.1007/s10898-017-0577-y.

Bärmann A, Martin A, Schneider O (2020) The bipartite boolean quadric polytope with multiple-choice constraints. arXiv preprint, URL http://dx.doi.org/10.48550/arXiv.2009.11674.

Bienstock D, Munoz G (2018) LP formulations for polynomial optimization problems. *SIAM Journal on Optimization* 28(2):1121–1150, URL http://dx.doi.org/10.1137/15M1054079.

Biggs M, Hariss R, Perakis G (2017) Optimizing objective functions determined from random forests. Available at SSRN 2986630, URL http://dx.doi.org/10.2139/ssrn.2986630.

Bodlaender HL (1996) A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing* 25(6):1305–1317, URL http://dx.doi.org/10.1137/S0097539793251219.

Buchheim C, Crama Y, Rodríguez-Heck E (2019) Berge-acyclic multilinear 0–1 optimization problems. *European Journal of Operational Research* 273(1):102–107, URL http://dx.doi.org/10.1016/j.ejor.2018.07.045.

Ceria S, Soares J (1999) Convex programming for disjunctive convex optimization. *Mathematical Programming* 86(3):595–614, URL http://dx.doi.org/10.1007/s101070050106.

Chen YC, Mišić VV (2021) Assortment optimization under the decision forest model. Available at SSRN 3812654, URL http://dx.doi.org/10.2139/ssrn.3812654.

Chen YC, Mišić VV (2022) Decision forest: A nonparametric approach to modeling irrational choice. *Management Science* URL http://dx.doi.org/10.1287/mnsc.2021.4256.

Cortez P, Cerdeira A, Almeida F, Matos T, Reis J (2009) Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems* 47(4):547–553, URL http://dx.doi.org/10.1016/j.dss.2009.05.016.

Crama Y (1993) Concave extensions for nonlinear 0–1 maximization problems. *Mathematical Programming* 61(1):53–60, URL http://dx.doi.org/10.1007/BF01582138.

Crama Y, Rodríguez-Heck E (2017) A class of valid inequalities for multilinear 0–1 optimization problems. *Discrete Optimization* 25:28–47, URL http://dx.doi.org/10.1016/j.disopt.2017.02.001.

Dash S, Günlük O, Wei D (2018) Boolean decision rules via column generation. Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, eds., *Advances in Neural Information Processing Systems*, volume 31 (Curran Associates, Inc.), URL https://proceedings.neurips.cc/paper/2018/file/743394beff4b1282ba735e5e3723ed74-Paper.pdf.

Deepa C, Sathiya Kumari K, Pream Sudha V (2010) A tree based model for high performance concrete mix design. *International Journal of Engineering Science and Technology* 2(9):4640–4646, URL http://www.idc-online.com/technical_references/pdfs/civil_engineering/A%20TREE%20BASED.pdf.

Del Pia A, Khajavirad A (2018a) The multilinear polytope for acyclic hypergraphs. *SIAM Journal on Optimization* 28(2):1049–1076, URL http://dx.doi.org/10.1137/16M1095998.

Del Pia A, Khajavirad A (2018b) On decomposability of multilinear sets. *Mathematical Programming* 170(2):387–415, URL http://dx.doi.org/10.1007/s10107-017-1158-z.

Dolan ED, Moré JJ (2002) Benchmarking optimization software with performance profiles. *Mathematical Programming* 91(2):201–213, URL http://dx.doi.org/10.1007/s101070100263.

Dunning I, Huchette J, Lubin M (2017) JuMP: A modeling language for mathematical optimization. *SIAM Review* 59(2):295–320, URL http://dx.doi.org/10.1137/15M1020575.

Fernández-Delgado M, Cernadas E, Barro S, Amorim D (2014) Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research* 15(1):3133–3181, URL https://www.jmlr.org/papers/volume15/delgado14a/delgado14a.pdf.

Ferreira KJ, Lee BHA, Simchi-Levi D (2016) Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management* 18(1):69–88, URL http://dx.doi.org/10.1287/msom.2015.0561.

Goos M, Jain R, Watson T (2018) Extension complexity of independent set polytopes. *SIAM Journal on Computing* 47(1):241–269, URL http://dx.doi.org/10.1137/16M109884X.

Gupte A, Kalinowski T, Rigterink F, Waterer H (2020) Extended formulations for convex hulls of some bilinear functions. *Discrete Optimization* 36:100569, URL http://dx.doi.org/10.1016/j.disopt.2020.100569.

Gurobi Optimization, LLC (2021) Gurobi optimizer reference manual. URL http://www.gurobi.com.

He T, Tawarmalani M (2021) A new framework to relax composite functions in nonlinear programs. *Mathematical Programming* 190(1):427–466, URL http://dx.doi.org/10.1007/s10107-020-01541-x.

Herrera M, Torgo L, Izquierdo J, Pérez-García R (2010) Predictive models for forecasting hourly urban water demand. *Journal of Hydrology* 387(1-2):141–150, URL http://dx.doi.org/10.1016/j.jhydrol.2010.04.005.

Hoffman AJ (1976) Total unimodularity and combinatorial theorems. *Linear Algebra and its Applications* 13(1-2):103–108, URL http://dx.doi.org/10.1016/0024-3795(76)90047-1.

Huchette J, Vielma JP (2022) Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools. *Operations Research* Forthcoming, URL http://arxiv.org/abs/1708.00050.

Ibaraki T (1976) Integer programming formulation of combinatorial optimization problems. *Discrete Mathematics* 16(1):39–52, URL http://dx.doi.org/10.1016/0012-365X(76)90091-1.

Kim J, Taslimi B, Richard JPP, Tawarmalani M (2019) Polyhedral results for tree ensembles optimization. *2019 INFORMS Annual Meeting*.

Kim J, Tawarmalani M, Richard JPP (2021) Convexification of permutation-invariant sets and an application to sparse principal component analysis. *Mathematics of Operations Research* URL http://dx.doi.org/10.1287/moor.2021.1219.

Kis T, Horváth M (2021) Ideal, non-extended formulations for disjunctive constraints admitting a network representation. *Mathematical Programming* URL http://dx.doi.org/10.1007/s10107-021-01652-z.

Luedtke J, Namazifar M, Linderoth J (2012) Some results on the strength of relaxations of multilinear functions. *Mathematical Programming* 136(2):325–351, URL http://dx.doi.org/10.1007/s10107-012-0606-z.

Ma J, Sheridan RP, Liaw A, Dahl GE, Svetnik V (2015) Deep neural nets as a method for quantitative structure–activity relationships. *Journal of Chemical Information and Modeling* 55(2):263–274, URL http://dx.doi.org/10.1021/ci500747n.

Misener R, Floudas CA (2012) Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming* 136(1):155–182, URL http://dx.doi.org/10.1007/s10107-012-0555-6.

Mišić VV (2020) Optimization of tree ensembles. *Operations Research* 68(5):1605–1624, URL http://dx.doi.org/10.1287/opre.2019.1928.

Mistry M, Letsios D, Krennrich G, Lee RM, Misener R (2021) Mixed-integer convex nonlinear optimization with gradient-boosted trees embedded. *INFORMS Journal on Computing* 33(3):1103–1119, URL http://dx.doi.org/10.1287/ijoc.2020.0993.

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830, URL https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf?ref=https://githubhelp.com.

Rikun AD (1997) A convex envelope formula for multilinear functions. *Journal of Global Optimization* 10(4):425–437, URL http://dx.doi.org/10.1023/A:1008217604285.

Rockafellar RT (1970) *Convex analysis* (Princeton university press).

Schrijver A (1983) Short proofs on the matching polyhedron. *Journal of Combinatorial Theory, Series B* 34(1):104–108, URL http://dx.doi.org/10.1016/0095-8956(83)90011-4.

Sherali HD (1997) Convex envelopes of multilinear functions over a unit hypercube and over special discrete sets. *Acta mathematica vietnamica* 22(1):245–270, URL http://journals.math.ac.vn/acta/pdf/9701245.pdf.

Stubbs RA, Mehrotra S (1999) A branch-and-cut method for 0-1 mixed convex programming. *Mathematical programming* 86(3):515–532, URL http://dx.doi.org/10.1007/s101070050103.

Su G, Wei D, Varshney KR, Malioutov DM (2016) Learning sparse two-level boolean rules. *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, 1–6, URL http://dx.doi.org/10.1109/MLSP.2016.7738856.

Svetnik V, Liaw A, Tong C, Culberson JC, Sheridan RP, Feuston BP (2003) Random forest: a classification and regression tool for compound classification and qsar modeling. *Journal of chemical information and computer sciences* 43(6):1947–1958, URL http://dx.doi.org/10.1021/ci034160g.

Tawarmalani M (2010) Inclusion certificates and simultaneous convexification of functions. Available at Optimization Online, URL http://www.optimization-online.org/DB_HTML/2010/09/2722.html.

Tawarmalani M, Sahinidis NV (2002) Convex extensions and envelopes of lower semi-continuous functions. *Mathematical Programming* 93(2):247–263, URL http://dx.doi.org/10.1007/s10107-002-0308-z.

Thebelt A, Kronqvist J, Mistry M, Lee RM, Sudermann-Merx N, Misener R (2021) Entmoot: a framework for optimization over ensemble tree models. *Computers & Chemical Engineering* 151:107343, URL http://dx.doi.org/10.1016/j.compchemeng.2021.107343.

Vielma JP (2015) Mixed integer linear programming formulation techniques. *Siam Review* 57(1):3–57, URL http://dx.doi.org/10.1137/130915303.

Vielma JP, Ahmed S, Nemhauser G (2010) Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. *Operations research* 58(2):303–315, URL http://dx.doi.org/10.1287/opre.1090.0721.

Vielma JP, Nemhauser GL (2011) Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming* 128(1):49–72, URL http://dx.doi.org/10.1007/s10107-009-0295-4.

Xu Y, Adams W, Gupte A (2021) Polyhedral analysis of symmetric multilinear polynomials over box constraints. *arXiv preprint arXiv:2012.06394* URL http://dx.doi.org/10.48550/arXiv.2012.06394.

Yeh IC (1998) Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research* 28(12):1797–1808, URL http://dx.doi.org/10.1016/S0008-8846(98)00165-3.

Appendix

APDX.1. Supplements for Section 3

The following lemma explains why it is sufficient to consider a finite number of intervals for a numerical input variable.

LEMMA APDX.1. *Let $v \in [n]$ and $j \in [K_v]$. Assume that X' and X'' are such that (i) $X'_{v'} = X''_{v'}$ for $v' \neq v$ and (ii) $X'_v, X''_v \in (a_{v,j-1}, a_{v,j}] \cap \mathbb{R}$ where $a_{v,0} := -\infty$ and $a_{v,K_v} := \infty$. Then, $f(X') = f(X'')$.*

Proof. For $v' \neq v$, the answers to all queries on $X_{v'}$ over the trees for X' and X'' are identical because $X'_{v'} = X''_{v'}$. The answers to all queries on X_v over the trees for X' and X'' are identical because there are only $K_v - 1$ possible queries. For both X' and X'', the answer to query "$X_v \leq a_{v,p}$" is false when $1 \leq p \leq j-1$ and true when $j \leq p \leq K_v - 1$. Since the answers to all queries are identical for X' and X'' over all trees, each decision tree reaches the same leaf for X' and X''. Therefore, $f(X') = f(X'')$. □

Proof of Theorem 1

Consider an instance of TEO with L leaves and n_T input variables. The reduction given in Section 3.2 reduces it to an instance of MOCPS with $|\mathcal{F}|$ faces and n_M simplices. Since the reduction models each leaf as an elementary multilinear term, then $|\mathcal{F}| = L$. Since one simplex is created for each input variable, we compute that $n_M = n_T \leq L$ under Assumption 1.

Consider an instance of MOCPS with n simplices so that $P := \prod_{v \in [n]} \Delta^{K_v}$, with a set $\mathcal{F}$ of faces in P characterized by subsets $J_F^v \subseteq [K_v]$, and with cost vector $\boldsymbol{c} \in \mathbb{R}^{|\mathcal{F}|}$. The objective of MOCPS is $\sum_{F \in \mathcal{F}} c_F \mathbb{1}_F(\boldsymbol{x})$. We will construct a tree ensemble with $|\mathcal{F}|$ trees whose input variables $X = (X_1, \ldots, X_n)$ are categorical and X_v has K_v category values for all $v \in [n]$. For $v \in [n]$ and $j \in [K_v]$, we think of $x_{v,j}$ in MOCPS as indicating whether input variable X_v takes category value j in TEO. For each face $F \in \mathcal{F}$, we construct a decision tree

t_F with up to $n+1$ leaves, one of which models $c_F \mathbb{1}_F(\boldsymbol{x})$ in MOCPS; see Algorithm 1:

Algorithm 1: Construction of a decision tree associated with face F

Data: Positive integer K_v, $J_F^v \subseteq [K_v]$ for all $v \in [n]$, and $c_F \in \mathbb{R}$.

Result: A decision tree t.

```
t ← single node decision tree;
ℓ ← root of t;
for v ∈ [n] do
    if J_F^v ⊊ [K_v] then
        Create left and right children of ℓ with branching query "X_v ∈ J_F^v?";
        ℓ ← the left child of ℓ;
Set the prediction value of ℓ to c_F;
Set the prediction values of all other leaves to 0;
```

It is clear that the decision tree created by Algorithm 1 generates, for a value X of the input variables, an output value identical to $c_F \mathbb{1}_F(\boldsymbol{x})$. Figure APDX.1 illustrates the decision tree created by Algorithm 1 for the face F where $\mathbb{1}_F(\boldsymbol{x}) = (x_{11}+x_{13})x_{22}(x_{42}+x_{44})$.

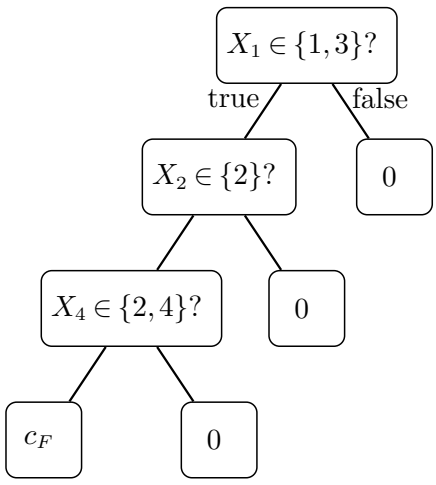


Figure APDX.1 Decision tree associated with the face F where $\mathbb{1}_F(\boldsymbol{x}) = (x_{11}+x_{13})x_{22}(x_{42}+x_{44})$ with coefficient c_F.

Let $\mathcal{T}$ be the tree ensemble composed of trees $\{t_F\}_{F\in\mathcal{F}}$ produced by Algorithm 1 for each $F\in\mathcal{F}$. Solving TEO for $\mathcal{T}$ is equivalent to solving the given instance of MOCPS because multilinear terms corresponding to any leaf which is not the leftmost one can be dropped because they have zero coefficients. After dropping these variables, the problem is identical to MOCPS. Because $\mathcal{T}$ has $|\mathcal{F}|$ trees and because each tree has at most $n+1$ leaves (as it branches at most n times according to Algorithm 1), we conclude that the total number of leaves of the trees in $\mathcal{T}$ is at most $|\mathcal{F}|(n+1)$. □

APDX.2. Supplements for Section 4

APDX.2.1. Proofs of Statements in Section 4

Proof of Theorem 2 We apply disjunctive programming (Balas 1985, 1998) to obtain a convex hull description of conv $\left(\bigvee_{F\in\mathcal{F}} P_F\right)$. First, recall that according to (6), conv(P_F) is described by the inequalities

$$\sum_{j\in J_F^v} x_{v,j} = 1, \qquad \forall v\in[n], \tag{APDX.1a}$$

$$x_{v,j} = 0, \qquad \forall v\in[n],\ \forall j\notin J_F^v, \tag{APDX.1b}$$

$$y_F = 1, \tag{APDX.1c}$$

$$y_{F'} = 0, \qquad \forall F'\in\mathcal{F}\backslash\{F\}, \tag{APDX.1d}$$

$$x_{v,j}\geq 0, \qquad \forall v\in[n],\ \forall j\in[K_v] \tag{APDX.1e}$$

for each $F\in\mathcal{F}$. Next, we introduce a multiplier λ^F for each disjunct P_F and copies of the variables $(\boldsymbol{x},\boldsymbol{y})$ for each disjunct, which we denote by $(\boldsymbol{z_F},\dot{\boldsymbol{y}}^{\boldsymbol{F}})$, so as to obtain

$$\sum_{j\in J_F^v} z_{v,j,F} = \lambda^F, \qquad \forall F\in\mathcal{F},\ \forall v\in[n], \tag{APDX.2a}$$

$$z_{v,j,F} = 0, \qquad \forall F\in\mathcal{F},\ \forall v\in[n],\ \forall j\notin J_F^v, \tag{APDX.2b}$$

$$\dot{y}_F^F = \lambda^F, \qquad \forall F\in\mathcal{F}, \tag{APDX.2c}$$

$$\dot{y}_{F'}^F = 0, \qquad \forall F\in\mathcal{F},\ \forall F'\in\mathcal{F}\backslash\{F\}, \tag{APDX.2d}$$

$$z_{v,j,F}\geq 0, \qquad \forall F\in\mathcal{F},\ \forall v\in[n],\ \forall j\in[K_v], \tag{APDX.2e}$$

$$\lambda^F \geq 0, \qquad \forall F \in \mathcal{F}, \tag{APDX.2f}$$

$$\sum_{F\in\mathcal{F}} \lambda^F = 1, \tag{APDX.2g}$$

$$x_{v,j} = \sum_{F\in\mathcal{F}} z_{v,j,F}, \qquad \forall v \in [n],\ \forall j \in [K_v], \tag{APDX.2h}$$

$$y_F = \sum_{F'\in\mathcal{F}} \dot{y}_F^{F'}, \qquad \forall F \in \mathcal{F}. \tag{APDX.2i}$$

Taken together, equalities (APDX.2c), (APDX.2d), and (APDX.2i) imply that $y_F = \dot{y}_F^F = \lambda^F$, which provides a way to eliminate variables $\dot{y}_F^{F'}$ and λ^F from the formulation. In particular, (APDX.2f) and (APDX.2g) become (7d). Similarly, (APDX.2a) becomes (7b). Using (APDX.2b) we eliminate zero variables from the formulation. In particular, (APDX.2h) becomes (7a) and (APDX.2e) implies (7e).

We conclude the proof by arguing that (7c) is redundant for each $v \in [n]$. First, observe that $\sum_{j\in[K_v]} x_{v,j} = 1$ in (7c), for each $v \in [n]$, is redundant as it can be obtained by summing (7a) for $j \in [K_v]$, subtracting (7b) for $F \in \mathcal{F}$, and subtracting $\sum_{F\in\mathcal{F}} y_F = 1$ in (7d). Second, $x_{v,j} \geq 0$ in (7c), for each $v \in [n]$ and $j \in [K_v]$, is also redundant as it can be obtained using (7a) and (7e). □

Proof of Proposition 1 Our proof of Proposition 1 relies on an ancillary result that we obtain first in Lemma APDX.3 as an application of Hoffman's circulation theorem.

LEMMA APDX.2 **(Hoffman (1976, Circulation Theorem))**. *Let $G = (V, E)$ be a digraph. Let $\ell, u : E \mapsto \mathbb{R} \cup \{\pm\infty\}$ denote the lower and upper bound of flow on arc e. Assume that $\ell(e) \leq u(e)$ for every $e \in E$. Then, there exists a feasible flow $\phi : E \mapsto \mathbb{R}$ that satisfies (i) $\ell(e) \leq \phi(e) \leq u(e)$ for every $e \in E$ and (ii) $\sum_{j:(i,j)\in E} \phi(i,j) = \sum_{j:(j,i)\in E} \phi(i,j)$ for all $i \in V$ if and only if*

$$\sum_{(i,j)\in E: i\in X, j\notin X} \ell(i,j) \leq \sum_{(i,j)\in E: i\notin X, j\in X} u(i,j), \qquad \forall X \subseteq V. \tag{APDX.3}$$

LEMMA APDX.3. *Consider a bipartite graph G with $V(G) = U \cup W$ and $E(G) \subseteq U \times W$, supply $s_u \in \mathbb{R}_+$ for $u \in U$ and demand $d_w \in \mathbb{R}_+$ for $w \in W$. Assume that $\sum_{u\in U} s_u = \sum_{w\in W} d_w$. The associated transportation polytope is the set of solutions $\phi : E \mapsto \mathbb{R}_+$ that satisfy the constraints*

$$\sum_{(u,w)\in E: u=u'} \phi(u,w) = s_{u'}, \qquad \forall u' \in U, \tag{APDX.4a}$$

$$\sum_{(u,w)\in E: w=w'} \phi(u,w) = d_{w'}, \qquad \forall w' \in W. \tag{APDX.4b}$$

The transportation polytope is feasible if and only if

$$\sum_{w\in W'} d_w \le \sum_{u\in \bigcup_{w\in W'} U(w)} s_u, \qquad \forall W' \subseteq W, \tag{APDX.5}$$

where $U(w) := \{u \in U : (u,w) \in E(G)\}$ *for* $w \in W$. *Further* (APDX.5) *is equivalent to*

$$\sum_{w\in W: U(w)\subseteq U'} d_w \le \sum_{u\in U'} s_u, \qquad \forall U' \subseteq U. \tag{APDX.6}$$

Proof. We convert the solutions of the given transportation polytope into flows on a network N, for which Lemma APDX.2 can be used. We define the vertex set $V(N) = U \cup W \cup \{s,t\}$ and the arc set $E(N) = E \cup E_s \cup E_t \cup \{(t,s)\}$ where $E_s = \{(s,u),\ \forall u \in U\}$ and $E_t = \{(w,t),\ \forall w \in W\}$. We set the lower and upper bounds on arcs of E to be 0 and ∞, respectively. For arcs (s,u) in E_s, these bounds are both set to s_u, while for arcs $(w,t) \in E_t$ they are both set to d_w. Finally the lower and upper bound on arc (t,s) are chosen to be $-\infty$ and ∞, respectively.

The transportation polytope and the set of feasible flows in N are equivalent because the flows on edges in $E_s \cup E_t \cup \{(t,s)\}$ are fixed. For $X \subseteq V(N)$, let $\delta^-(X) := \{(i,j) \in E(N) : i \notin X, j \in X\}$ and $\delta^+(X) := \{(i,j) \in E(N) : i \in X, j \notin X\}$. By Lemma APDX.2, there exists a feasible flow in N if and only if

$$\sum_{(i,j)\in\delta^+(X)} \ell(i,j) \le \sum_{(i,j)\in\delta^-(X)} u(i,j), \qquad \forall X \subseteq V(N). \tag{APDX.7}$$

We next compute the left- and right-hand-side of (APDX.7) for each X in terms of s_u and d_w.

Condition (APDX.7) can be disregarded when its left-hand-side is equal to $-\infty$ and its right-hand-side is not, or when its right-hand-side is equal to ∞ and its left-hand-side is not. Further, observe that the left-hand-side (resp. right-and-side) is strictly less than ∞ (resp. strictly greater than $-\infty$) since $\ell(i,j) < \infty$ (resp. $u(i,j) > -\infty$) for all $(i,j) \in E(N)$. Since the left-hand-side becomes $-\infty$ when $\delta^-(X) \cap (E \cup \{(t,s)\}) \neq \emptyset$ and since the right-hand-side becomes ∞ when $\delta^+(X) \cap \{(t,s)\} \neq \emptyset$, we may assume from now on that (i) either $\{s,t\} \subseteq X$ or $X \cap \{s,t\} = \emptyset$, and

(ii) $\bigcup_{w\in X\cap W} U(w) \subseteq X$. Under assumptions (i) and (ii), we can rewrite (APDX.7) in terms of s_u and d_w. There are two cases. If $\{s,t\} \subseteq X$, then

$$\sum_{w\in X\cap W} d_w \leq \sum_{u\in X\cap U} s_u. \tag{APDX.8}$$

If $X \cap \{s,t\} = \emptyset$, then

$$\sum_{u\in U\setminus X} s_u \leq \sum_{w\in W\setminus X} d_w. \tag{APDX.9}$$

Next, we argue that the relationships in (APDX.9) and some of the relationships in (APDX.8) are implied by a subset of conditions (APDX.8). To see this, observe first that, for any $X \subseteq V(N) \setminus \{s,t\}$ used in (APDX.9), the following holds because $\sum_{u\in U} s_u = \sum_{w\in W} d_w$, and sets U and W do not contain $\{s,t\}$:

$$\sum_{u\in U\setminus X} s_u \leq \sum_{w\in W\setminus X} d_w \quad\Leftrightarrow\quad -\sum_{w\in W\setminus X} d_w \leq -\sum_{u\in U\setminus X} s_u \quad\Leftrightarrow\quad \sum_{w\in W} d_w - \sum_{w\in W\setminus X} d_w \leq \sum_{u\in U} s_u - \sum_{u\in U\setminus X} s_u$$
$$\Leftrightarrow\quad \sum_{w\in X\cap W} d_w \leq \sum_{u\in X\cap U} s_u \quad\Leftrightarrow\quad \sum_{w\in (X\cup\{s,t\})\cap W} d_w \leq \sum_{u\in (X\cup\{s,t\})\cap U} s_u.$$

From now, we can therefore assume that (iii) $\{s,t\} \subseteq X$ holds, which is stronger than and implies (i). For $X \subseteq V(N)$, let $U_X = X \cap U$ and $W_X = X \cap W$. We argue that (APDX.8) for $X \subseteq V(N)$ under (ii) and (iii) is implied by (APDX.8) for $X' \subseteq V(N)$ defined so that $U_{X'} = \bigcup_{w\in W_X} U(w)$ and $W_{X'} = W_X$. Set X' satisfies (ii) and (iii) by definition. Further, (APDX.8) for X' implies (APDX.8) for X because $W_X = W_{X'}$ by definition and because $U_{X'} \subseteq U_X$ (otherwise X would contradict assumption (ii).) In conclusion, there exists a feasible flow in N if and only if (APDX.8) holds for $X = W' \cup \bigcup_{w\in W'} U(w)$ for all $W' \subseteq W$. When expressed using s_u and d_w, this condition is (APDX.5).

We complete the proof by showing (APDX.5) and (APDX.6) are equivalent. First, we show that any inequality of (APDX.6) is implied by (APDX.5). Consider (APDX.6) for $U' \subseteq U$. Let $W' = \{w \in W : U(w) \subseteq U'\}$. It holds that $\bigcup_{w\in W'} U(w) \subseteq U'$ by definition. Hence, (APDX.6) for U' is implied by (APDX.5) for W'. Next, we show that any inequality of (APDX.5) is implied by (APDX.6). Consider (APDX.5) for $W' \subseteq W$. Let $U' = \bigcup_{w\in W'} U(w)$. It holds that $W' \subseteq \{w \in W : U(w) \subseteq U'\}$ by definition. Hence, (APDX.5) for W' is implied by (APDX.6) for U'. Therefore, (APDX.5) and (APDX.6) are equivalent. □

Recall the discussion before the introduction of (8) that point $(\boldsymbol{x}, \boldsymbol{y}) \in \prod_{v\in[n]} \Delta^{K_v} \times \Delta^{|\mathcal{F}|}$ is in $\text{conv}(\text{ML}(D, \mathcal{F}))$ if and only if $\text{TP}(v)$ given $(\boldsymbol{x_v}, \boldsymbol{y})$ is feasible for all $v \in [n]$. By Lemma APDX.3, for $v \in [n]$, $\text{TP}(v)$ is feasible if and only if $(\boldsymbol{x_v}, \boldsymbol{y})$ satisfies (APDX.5) or (APDX.6). Constraints (9a) and (9b) for $v \in [n]$ correspond to (APDX.6) and (APDX.5), respectively. Since it is enough to satisfy one of (9a) and (9b) for $v \in [n]$, system (9) becomes a formulation for $\text{conv}(\text{ML}(D, \mathcal{F}))$ for any $N \subseteq [n]$. We exclude $J \in \{\emptyset, [K_v]\}$ and $H \in \{\emptyset, \mathcal{F}\}$ because the associated constraints reduce to either $0 \leq 0$ or $1 \leq 1$. □

Proof of Proposition 2 Consider first (9a) with indices (v, J). Assume that there exists a non-trivial partition (J_1, J_2) of J where every $F \in \mathcal{F}$ such that $J_F^v \subseteq J$ satisfies either $J_F^v \subseteq J_1$ or $J_F^v \subseteq J_2$. Then,

$$\sum_{F\in\mathcal{F}: J_F^v \subseteq J} y_F = \sum_{F\in\mathcal{F}: J_F^v \subseteq J_1} y_F + \sum_{F\in\mathcal{F}: J_F^v \subseteq J_2} y_F \leq \sum_{j\in J_1} x_{v,j} + \sum_{j\in J_2} x_{v,j} = \sum_{j\in J} x_{v,j},$$

where the first equality holds by assumption, the second holds because $\sum_{F\in\mathcal{F}: J_F^v \subseteq J_k} y_F \leq \sum_{j\in J_k} x_{v,j}$ is (9a) for indices (v, J_k) for $k = 1, 2$, and the last holds because $J_1 \cup J_2 = J$ and $J_1 \cap J_2 = \emptyset$. Therefore, (9a) for indices (v, J) is redundant since it can be obtained by summing (9a) for indices (v, J_k) for $k = 1, 2$.

Consider next (9b) with indices (v, H). Assume that there is a partition (H_1, H_2) of H where $(\cup_{F\in H_1} J_F^v) \cap (\cup_{F\in H_2} J_F^v) = \emptyset$. Then,

$$\sum_{F\in H} y_F = \sum_{F\in H_1} y_F + \sum_{F\in H_2} y_F \leq \sum_{\cup_{F\in H_1} J_F^v} x_{v,j} + \sum_{\cup_{F\in H_2} J_F^v} x_{v,j} = \sum_{\cup_{F\in H} J_F^v} x_{v,j},$$

where the first equality holds by assumption, the second holds because $\sum_{F\in H_k} y_F \leq \sum_{\cup_{F\in H_k} J_F^v} x_{v,j}$ is (9a) for (v, H_k) for $k = 1, 2$, and the last holds because $\left(\bigcup_{F\in H_1} J_F^v\right) \cap \left(\bigcup_{F\in H_2} J_F^v\right) = \emptyset$ and $H = H_1 \cup H_2$. Therefore, (9b) for indices (v, H) is redundant since it can be obtained by summing (9b) for indices (v, H_k) for $k = 1, 2$. □

Proof of Theorem 3 By choosing $N = [n]$ in Proposition 1, it is sufficient to show that (9a) is implied by (11a). Consider a constraint (9a) with indices (v, J). If $J = [a..b]$ for some a and b, then it is clearly implied because there is an identical constraint in (11a).

Assume J is not of the form $[a..b]$. We say that $[a'..b']$ is a maximal consecutive integer subset of J if $[a'..b'] \subseteq J$ and $a'-1, b'+1 \notin J$. Then, there is a unique collection of maximal consecutive integer subsets of J, $\{[a_1..b_1], \cdots, [a_k..b_k]\}$, such that $J = [a_1..b_1] \cup \cdots \cup [a_k..b_k]$. Without loss of generality, assume that $a_1 < a_2 < \cdots < a_k$.

Pick any $F \in \mathcal{F}$ such that $J_F^v \subseteq J$. By the adjacency property, $J_F^v = [c..d]$ for some $c, d \in [K_v]$. Since $J_F^v \subseteq J$, $[c..d] \subseteq [a_r..b_r]$ for some $r \in [k]$. The left-hand-side of (11a) for (v, a_r, b_r) contains y_F. Also, the left-hand-side of (11a) for (v, a_p, b_p) for $p \neq r$ does not contain y_F. Therefore, y_F for all $F \in \mathcal{F}$ such that $J_F^v \subseteq J$ appears in exactly one of (11a) for $(v, a_1, b_1), \ldots, (v, a_k, b_k)$. It follows that the aggregation of the constraints of (11a) for $(v, a_1, b_1), \ldots, (v, a_k, b_k)$ results in

$$\sum_{F \in \mathcal{F}: J_F^v \subseteq J} y_F = \sum_{p=1}^{k} \left(\sum_{F \in \mathcal{F}: J_F^v \subseteq [a_p..b_p]} y_F \right) \leq \sum_{p=1}^{k} \left(\sum_{j \in [a_p..b_p]} x_{v,j} \right) = \sum_{j \in J} x_{v,j},$$

which is (9a) for (v, J). This completes the proof.

Proof of Proposition 3 Since $\mathcal{F}'$ is a refinement of $\mathcal{F}$, there exists $A \in \mathbb{R}^{|\mathcal{F}| \times |\mathcal{F}'|}$ such that $\mathrm{ML}(D, \mathcal{F}) = \{(\boldsymbol{x}, A\boldsymbol{y'}) \mid (\boldsymbol{x}, \boldsymbol{y'}) \in \mathrm{ML}(D, \mathcal{F}')\}$. Since affine transformations and convex hull operators commute, it holds that $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F})) = \{(\boldsymbol{x}, \boldsymbol{y}) \mid (\boldsymbol{x}, \boldsymbol{y'}) \in \mathrm{conv}(\mathrm{ML}(D, \mathcal{F}')), \boldsymbol{y} = A\boldsymbol{y'}\}$. Therefore, $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F}))$ can be formulated as an LP by adding $|F|$ variables $(\boldsymbol{y})$ and $|F|$ constraints $(\boldsymbol{y} = A\boldsymbol{y'})$ to the LP formulation for $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F}'))$. □

Proof of Theorem 4 Theorem 1.1 in Goos et al. (2018) proves that there exists a collection of simple graph $G_n = (V_n, E_n)$ for which every extended formulation of the independent set polytope of G_n has size (the total number of constraints) at least $\Omega(2^{n/\log n})$. Let $P_n = \prod_{v \in [n]} \Delta^2$ and let $D_n = \mathrm{vert}(P_n)$. Let $\mathcal{F}_n = \{F_{i,j}\}_{(i,j) \in E_n}$ where $F_{i,j}$ is a face of P_n whose associated elementary multilinear term is $\mathbb{1}_{F_{i,j}}(\boldsymbol{x}) = x_{i,1} x_{j,1}$ for $(i,j) \in E_n$. The independent set polytope of G_n can be written as

$$I(G_n) = \mathrm{proj}_{\boldsymbol{x}} \, \mathrm{conv} \left\{ (\boldsymbol{x}, \boldsymbol{y}) \in \mathrm{ML}(D_n, \mathcal{F}_n) \,\middle|\, \sum_{(i,j) \in E_n} y_{i,j} = 0 \right\},$$

where $y_{i,j}$ is the indicator variable for $F_{i,j}$ for all $(i,j) \in E_n$. Since affine transformations commute with convexification, it holds that

$$I(G_n) = \text{proj}_{\boldsymbol{x}} \left\{ (\boldsymbol{x}, \boldsymbol{y}) \in \text{conv}(\text{ML}(D_n, \mathcal{F}_n)) \middle| \sum_{(i,j)\in E_n} y_{i,j} = 0 \right\}.$$

Assume now by contradiction that there exists a P_n-completion $\mathcal{F}'_n$ of $\mathcal{F}_n$ of size bounded above by a polynomial of n. By Proposition 3, there exists a formulation for $\text{conv}(\text{ML}(D_n, \mathcal{F}_n))$ whose size is polynomial in n, $|\mathcal{F}_n|$, and $|\mathcal{F}'_n|$. Since $|\mathcal{F}_n| \leq \binom{n}{2}$ and $|\mathcal{F}'_n|$ is polynomial in n, the size of this formulation is polynomial in n. This contradicts the fact that the minimum size of extended formulations for the independent set polytope of G_n belongs to $\Omega(2^{n/\log n})$. □

Proof of Proposition 4 The proof follows from Remark 1 by constructing a decision tree in which there is a leaf for each F in $\mathcal{F}$. We use induction to show that when $\mathcal{F}$ is a collection of vertices $v_1, \dots v_k$ of P, there exists a tree T with no more than $nk+1$ leaves such each vertex v_j can be mapped to a leaf of T whose associated region is exactly the vertex. For the basis of induction, observe that, when $|\mathcal{F}| = 1$, Algorithm 1 can be used to produce a tree with the desired property that has no more than $n+1$ leaves. For the inductive step, assume the result holds for collections of k vertices and consider a collection $\mathcal{F}$ such that $|\mathcal{F}| = k+1$. For any vertex v in the collection $\mathcal{F}$, consider the collection $\tilde{\mathcal{F}}$ obtained by removing v from $\mathcal{F}$. By induction, there exists a tree $\tilde{T}$ with no more than $nk+1$ leaves such each vertex v_j can be mapped to a leaf of $\tilde{T}$ whose associated region is exactly the vertex. Next, identify the leaf ℓ' in $\tilde{T}$ whose associated region contains v. Applying lines 3-8 of Algorithm 1 by setting $\ell = \ell'$, we obtain a new tree T. This procedure does not add more than n leaves to $\tilde{T}$ showing that the number of leaves of T is bounded above by $n(k+1)+1$. Further, the analysis of Algorithm 1 performed in Theorem 1 also shows that one of these leaves has an associated region that is exactly vertex v. □

Proof of Proposition 5 We first introduce the following ancillary result.

LEMMA APDX.4. *Let $\mathcal{F}_1, \dots, \mathcal{F}_T$ be facial decompositions of P. Then, there is a common P-completion $\mathcal{F}$ of $\mathcal{F}_1, \dots, \mathcal{F}_T$ such that $|\mathcal{F}| \leq \prod_{t=1}^{T} |\mathcal{F}_t|$.*

Proof. Define $\mathcal{F} := \bigcup_{(F_{1,j_1},\dots,F_{T,j_T})\in\mathcal{F}_1\times\cdots\times\mathcal{F}_T} \{\bigcap_{t\in[T]} F_{t,j_t}\}\setminus\{\emptyset\}$. We claim that $\mathcal{F}$ is a common P-completion of $\mathcal{F}_1,\dots,\mathcal{F}_T$. Collection $\mathcal{F}$ is a facial decomposition of P because (i) every element in $\mathcal{F}$ is a face in P as, in a polytope, the intersection of multiple faces is also a face, and (ii) every vertex $v\in\text{vert}(P)$ is contained in a unique face $F_i = F_{1,j_1}\cap\cdots\cap F_{T,j_T}$ in $\mathcal{F}$ where F_{t,j_t} is the unique face containing v for each $t\in[T]$ as $\mathcal{F}_t$ is a facial decomposition of P for all $t\in[T]$.

We next argue that $\mathcal{F}$ is a refinement of $\mathcal{F}_t$ for $t\in[T]$. Pick a face $F_{t^*,j^*}\in\mathcal{F}_{t^*}$. Let $\tilde{\mathcal{F}} = \bigcup_{v\in\text{vert}(F_{t^*,j^*})}\{F\in\mathcal{F}\mid v\in F\}$. It holds that $F_{t^*,j^*}\subseteq\text{conv}\left(\bigcup_{F\in\tilde{\mathcal{F}}} F\right)$ because there exists a face in $\mathcal{F}$ containing v for every vertex $v\in\text{vert}(P)$ by (ii). Also, $\text{conv}\left(\bigcup_{F\in\tilde{\mathcal{F}}} F\right)\subseteq F_{t^*,j^*}$ holds because every $F\in\tilde{\mathcal{F}}$ is obtained as $F_{1,j_1}\cap\cdots\cap F_{t^*,j^*}\cap\cdots\cap F_{T,j_T}$ for suitable faces $F_{t,j_t}\in\mathcal{F}_t$ for $t=1,\dots,t^*-1,t^*+1,\dots,T$. Therefore, for any $t\in[n]$, any face in $\mathcal{F}_t$ can be expressed as the convex hull of a union of a subset of $\mathcal{F}$, *i.e.*, $\mathcal{F}$ is a refinement of $\mathcal{F}_t$ for all $t\in[T]$. It follows that $\mathcal{F}$ is a common P-completion of $\mathcal{F}_1,\dots,\mathcal{F}_T$ with $|\mathcal{F}|\le\prod_{t\in[T]}|\mathcal{F}_t|$, where the inequality holds by construction of $\mathcal{F}$. □

Let $\mathcal{F}=\{F_1,\dots,F_r\}$ be the given collection of a constant number of faces. Let n be the number of simplices. By Proposition 4, there exists a P-completion $\mathcal{F}_i$ of $\{F_i\}$ with $|\mathcal{F}_i|\le n+1$ for all $i\in[r]$. By Lemma APDX.4, there is a common P-completion $\mathcal{F}'$ of $\mathcal{F}_1,\dots,\mathcal{F}_r$ with $|\mathcal{F}'|\le\prod_{i\in[r]}|\mathcal{F}_i|\le(n+1)^r$. Then, $\mathcal{F}'$ is also a P-completion of $\mathcal{F}$ because $\mathcal{F}\subseteq\bigcup_{i\in[r]}\mathcal{F}_i$. Moreover, $|\mathcal{F}'|$ is polynomial in n since r is constant. Therefore, $\mathcal{F}$ is P-poly-completable. □

Proof of Lemma 1 Since each node is a subset of P, for each node $F\in\mathcal{F}$ we can find a sequence of sets ordered so that each set is covered by the previous set and the sequence begins at P and ends at F. This shows that P is connected to all the faces in $\mathcal{F}$. Now, we show that the path to each face F is unique. Instead, consider two paths $(R_1,\dots,R_t)$ and $(S_1,\dots,S_k)$ that are not identical. Assume further that $R_1=S_1=P$ and $R_t=S_k=F$ and without loss of generality that $t\ge k$. If the two paths are not identical, there is an $R'\in\{R_1,\dots R_t\}\setminus\{S_1,\dots,S_k\}$. Let R' be the highest indexed set that belongs to the first path but not the second. Then, it follows that the next set in the sequence, say R'' is contained in a set $S'\cap R'$, where we choose S' to be the highest indexed set in $\{S_1,\dots,S_k\}$ that strictly contains R''. Since R'' is next to R' in one of the paths,

$S' \not\subseteq R'$. Similarly, S' is next to R'' in $\{S_1, \ldots, S_k\}$, S' and, so, $R' \not\subseteq S'$. Moreover, since R' is not in the second path $R' \neq S'$. However, since $R'' \subseteq R' \cap S'$ and R'' is not empty, R' and S' violate the arborescence property. Therefore, the path from P to F must be unique. □

Proof of Theorem 5 Let G be the graph of $\mathcal{F}$. We construct an arborescence G' by adding nodes that correspond to faces of P so that, for all $v \in V(G')$, we have $\bigcup_{w \in \mathcal{N}^+_{G'}(v)} w = v$. Then, we show that the leaves of G', *i.e.*, $\mathcal{F}' = \{v \in V(G') \mid \mathcal{N}^+_{G'}(v) = \emptyset\}$, form a P-completion of $\mathcal{F}$ and that $|\mathcal{F}'|$ is polynomial in n, $K^{\max}$, and $|\mathcal{F}|$.

Let $G^1 = G$. We begin with $k = 1$. We let $I^1 = \{u \mid \mathcal{N}^+_{G^1}(u) \neq \emptyset\}$ be the set of internal nodes of G^1. Throughout the procedure, we ensure that (i) G^k is the graph of $V(G^k)$, (ii) the neighbors of any $u \in I^k$ remain unaltered so that it remains u-poly-completable, and (iii) if $u \in I^1 \setminus I^k$ then $\bigcup_{w \in \mathcal{N}^+_{G^k}(u)} w = u$. At each step, we pick $v \in I^k$. Since $\mathcal{N}^+_{G^k}(v)$ is v-poly-completable by our assumption, there exists a v-completion S_v of $\mathcal{N}^+_{G^k}(v)$ such that $\mathcal{N}^+_{G^k}(v) \subseteq S_v$ and $|S_v|$ is polynomial in n, $K^{\max}$, and $|\mathcal{N}^+_{G^k}(v)|$. We create G^{k+1} so that $V(G^{k+1}) = V(G^k) \cup S_v$ and $E(G^{k+1}) = E(G^k) \cup \{(v, v') \mid v' \in S_v\}$. We define $I^{k+1} = I^k \setminus \{v\}$.

We argue that (i) holds for G^{k+1}. To see this, we add nodes in S_v one by one. Let v' be a node newly added to the graph. We show that there cannot be a cover relation (w, u) so that, now $u \subseteq v' \subseteq w$. There are four cases to consider. If $w \cap v = \emptyset$, we cannot have $\emptyset \neq v' \subseteq v \cap w = \emptyset$. If $w \supsetneq v$, then $v \subseteq u$ which violates that $v' \subsetneq v$. If $w \subsetneq v$, there is a path in G^k from v to w. Let v'' be the node next to v on this path. Since $v' \neq v''$, we have $w \cap v' \subseteq v'' \cap v' = \emptyset$. Finally, we consider $v = w$. Then, $\emptyset \neq u = u \cap v' = \emptyset$. Therefore, v' does not have any outneighbors. Now, we show that v must have v' as its outneighbor in G^{k+1}. Assume instead that v' is the outneighbor of $w \neq v$. Since $v' \subseteq w \cap v$, $w \cap v$ cannot be empty. If $v \subsetneq w$, we would not connect w to v'. Therefore, $v' \subseteq w \subsetneq v$. However, then there is a node adjacent to v on the path from v to w in G^k, say v''. But, $v'' \cap v' = \emptyset$ since v' and v'' belong to a facial decomposition of v. Therefore, $w = v$.

We next argue that (ii) and (iii) hold for G^{k+1}. It follows from the above construction that for any $v' \in S_v \setminus \mathcal{N}^+_{G^k}(v)$, $\mathcal{N}^+_{G^{k+1}}(v') = \emptyset$. Therefore, the set of internal nodes of G^{k+1} is the same as that

of G^k. For any $u \in I^{k+1}$, its neighbors in G^{k+1} are the same as those of u in G^k since $v \notin I^{k+1}$. Therefore, each node in $u \in I^{k+1}$ is still u-poly-completable, *i.e.*, (ii) holds. Moreover, for all $u \in I^1 \setminus I^{k+1}$, we have $\bigcup_{w \in \mathcal{N}^+_{G^{k+1}}(u)} w = u$ since the property remains true for nodes in $I^1 \setminus I^k$ as their neighbors were not altered and, for v the property was guaranteed by construction, *i.e.*, (iii) holds.

Now, we iterate with $k \leftarrow k+1$. Since during each iteration the size of I^k reduces, the procedure converges in $t = |I^1|$ steps with a P-completion. This is because for any node $u \in G^t$ that is not a leaf, we have $\bigcup_{w \in \mathcal{N}^+_{G^t}(u)} w = u$. To verify that the leaves give a P-completion, observe that there is no path between the leaves of the arborescence. Therefore, the leaves are disjoint. Moreover, for any node in u that is not a leaf, there is an outneighbor of u that contains it. Therefore, recursively applying the idea, shows that there is a leaf that contains the node.

Now, we count the number of leaves in the arborescence at the end of the procedure. At each step, we add no more than $|S_v| - 1$ nodes, which is bounded above by a polynomial in n, $K^{\max}$, and $|\mathcal{F}|$, say $p(n, K^{\max}, |\mathcal{F}|)$. The maximum number of steps in the procedure is the number of internal nodes in G, which is bounded above by $|V(G)| = |\mathcal{F}|$. Therefore, $|\mathcal{F}'| \le |\mathcal{F}| \times p(n, K^{\max}, |\mathcal{F}|)$ which is a polynomial in n, $K^{\max}$, and $|\mathcal{F}|$. □

Proof of Theorem 6 The main idea of the proof is stated in Lemma APDX.5. A similar decomposition principle has been used in several papers; see Schrijver (1983), for instance.

LEMMA APDX.5. *For $i = 1, 2$, let Q_i be an integral polytope in variables $(\boldsymbol{x_i}, \boldsymbol{y}) \in \mathbb{R}^{p_i} \times \mathbb{R}^{p_0}$. Assume that the projections of Q_1 and Q_2 in the space of $\boldsymbol{y}$ form a common simplex Δ. Then, polytope $S = \{(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y}) \mid (\boldsymbol{x_1}, \boldsymbol{y}) \in Q_1, (\boldsymbol{x_2}, \boldsymbol{y}) \in Q_2\}$ is integral.*

Proof. Given a point $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y}) \in S$, we express it as a convex combination of points $(\boldsymbol{x_1^i}, \boldsymbol{x_2^i}, \boldsymbol{y^i})$ where $(\boldsymbol{x_1^i}, \boldsymbol{y^i})$ is an extreme point of Q_1 and $(\boldsymbol{x_2^i}, \boldsymbol{y^i})$ is an extreme point of Q_2. To do so, we consider the extreme points $\boldsymbol{y^k}$ of Δ. Then, we write $(\boldsymbol{x_1}, \boldsymbol{y}) = \sum_k \sum_{i \in I_k} \lambda_{i,k} (\boldsymbol{x_1^i}, \boldsymbol{y^k})$ and $(\boldsymbol{x_2}, \boldsymbol{y}) = \sum_k \sum_{j \in J_k} \gamma_{j,k} (\boldsymbol{x_2^j}, \boldsymbol{y^k})$. We can write $(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{y}) = \sum_k \sum_{i \in I_k} \sum_{j \in J_k} \frac{\lambda_{i,k} \gamma_{j,k}}{\sum_{i \in I_k} \lambda_{i,k}} (\boldsymbol{x_1^i}, \boldsymbol{x_2^j}, \boldsymbol{y^k})$. To see that this works, observe that $\sum_{i \in I_k} \lambda_{i,k} = \sum_{j \in J_k} \gamma_{j,k}$, for each k, because $\{\boldsymbol{y^k}\}_k$ are the extreme points of a simplex. □

Let $\bar{M} := \mathrm{ML}(D_I, \mathcal{F}_I^0 \cup \mathcal{F}_I')$. Both $\mathrm{conv}(\bar{M})$ and $\mathrm{conv}(M)$ are integral polytopes. The projection in the space of $\boldsymbol{z}$ variables of $\mathrm{conv}(M)$ forms a simplex $\Delta^{\boldsymbol{z}} := \{\boldsymbol{z} \in \mathbb{R}_+^{|\mathcal{F}_I^1|} \mid \mathbf{1}^\intercal \boldsymbol{z} \leq 1\}$. If $\bigcup_{F \in \mathcal{F}_I^1} F \subsetneq P_I$, then the projection in the space of $\boldsymbol{z}$ variables of $\mathrm{conv}(\bar{M})$ is also $\Delta^{\boldsymbol{z}}$. Then, Lemma APDX.5 proves the result. Since $\bigcup_{F \in \mathcal{F}_I^1} F \subseteq P_I$ holds, it is sufficient to assume next that $\bigcup_{F \in \mathcal{F}_I^1} F = P_I$. The projection in the space of $\boldsymbol{z}$ variables of $\mathrm{conv}(\bar{M})$ is $\Delta^{\boldsymbol{z}} \cap \{\boldsymbol{z} \mid \mathbf{1}^\intercal \boldsymbol{z} = 1\}$. Let E be the hyperplane in the space of variables $(\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2})$ defined by the constraint $\mathbf{1}^\intercal \boldsymbol{z} = 1$ Then, it holds that $\mathrm{ML}(D, \mathcal{F}) = \{(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \mid (\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in \bar{M}, (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in M\} == \{(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \mid (\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in \bar{M}, (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in M \cap E\}$ since every point in the set satisfies $\mathbf{1}^\intercal \boldsymbol{z} = 1$. We complete the proof by arguing that

$$\begin{aligned}
\mathrm{conv}(\mathrm{ML}(D, \mathcal{F})) &= \mathrm{proj}_{\boldsymbol{x},\boldsymbol{y}}\big\{(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \mid (\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in \mathrm{conv}(\bar{M}), (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in \mathrm{conv}(M \cap E)\big\} \\
&= \mathrm{proj}_{\boldsymbol{x},\boldsymbol{y}}\big\{(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \mid (\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in \mathrm{conv}(\bar{M}), (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in \mathrm{conv}(M) \cap E\big\} \\
&= \mathrm{proj}_{\boldsymbol{x},\boldsymbol{y}}\big\{(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y}) \mid (\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in \mathrm{conv}(\bar{M}), (\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in \mathrm{conv}(M)\big\}.
\end{aligned}$$

The first equality holds because of Lemma APDX.5 and the fact that convex hull and projection operators commute. The second equality holds because E is a supporting hyperplane of M and the convex hull of the intersection of a set and its supporting hyperplane equals to the intersection of the supporting hyperplane and the convex hull of a set. The third equality holds because $(\boldsymbol{x_{\bar{I}}}, \boldsymbol{z}, \boldsymbol{y^1}, \boldsymbol{y^2}) \in E$ is implied by $(\boldsymbol{x_I}, \boldsymbol{z}, \boldsymbol{y^0}) \in \mathrm{conv}(\bar{M})$. □

APDX.2.2. Polynomial Separation Algorithm

PROPOSITION APDX.1. *Let* $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}) \in \prod_{v \in [n]} \Delta^{K_v} \times \Delta^{|\mathcal{F}|}$. *Verifying whether* $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ *belongs to* $\mathrm{conv}(ML(D, \mathcal{F}))$ *can be done through the solution of the following separation problems*

$$z_v^* = \max \quad \sum_{F \in \mathcal{F}} \bar{y}_F \mu_F - \sum_{j \in [K_v]} \bar{x}_{v,j} \lambda_{v,j} \tag{APDX.10a}$$

$$s.t. \quad \mu_F \leq \lambda_{v,j}, \qquad \forall F \in \mathcal{F},\ \forall j \in J_F^v, \tag{APDX.10b}$$

$$\lambda_{v,j} \in [0,1], \qquad \forall j \in [K_v], \tag{APDX.10c}$$

$$\mu_F \in [0,1], \qquad \forall F \in \mathcal{F}, \tag{APDX.10d}$$

for $v \in [n]$. *In particular, if* $z_v^* = 0$ *for all* $v \in [n]$, *then* $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}) \in \text{conv}(ML(D, \mathcal{F}))$. *Otherwise,* $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}) \notin \text{conv}(ML(D, \mathcal{F}))$ *and for any* $v \in [n]$ *such that* $z_v^* > 0$, *inequality*

$$\sum_{F \in \mathcal{F}} \mu_F^* y_F \leq \sum_{j \in [K_v]} \lambda_{v,j}^* x_{v,j}, \tag{APDX.11}$$

where $(\boldsymbol{\lambda_v^*}, \boldsymbol{\mu^*})$ *is an optimal solution to* (APDX.10) *for* $v \in [n]$, *separates* $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ *from* $\text{conv}(ML(D, \mathcal{F}))$.

Choose $N = [n]$. According to Proposition 1, if $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ does not belong to $\text{conv}(\text{ML}(D, \mathcal{F}))$, then there exist $v \in [n]$ and $J \subseteq [K_v]$ such that

$$\sum_{F \in \mathcal{F}: J_F^v \subseteq J} \bar{y}_F > \sum_{j \in J} \bar{x}_{v,j}. \tag{APDX.12}$$

Since inequalities (9a) decompose by $v \in [n]$, it is sufficient to consider a separation problem for each $v \in [n]$:

$$\begin{aligned}
\max \quad & \sum_{F \in \mathcal{F}} \bar{y}_F \mu_F - \sum_{j \in [K_v]} \bar{x}_{v,j} \lambda_{v,j} & & \\
\text{s.t.} \quad & \mu_F \leq \lambda_{v,j}, & \forall F \in \mathcal{F},\ \forall j \in J_F^v, & \quad \text{(APDX.13a)} \\
& \lambda_{v,j} \in \{0,1\}, & \forall j \in [K_v], & \quad \text{(APDX.13b)} \\
& \mu_F \in \{0,1\}, & \forall F \in \mathcal{F}. & \quad \text{(APDX.13c)}
\end{aligned}$$

Constraint (APDX.13a) imposes that if y_F appears on the left-hand-side of (9a), then $x_{v,j}$ for all $j \in J_F^v$ appears on the right-hand-side. Constraints (APDX.13b) and (APDX.13c) impose that the coefficients of $x_{v,j}$ and y_F are either 0 or 1. The objective function represents the amount of the violation at $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$. Therefore, given $v \in [n]$, problem (APDX.13) finds an inequality of the form (APDX.11) that $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ violates the most. Finally, binary conditions in (APDX.13) can be relaxed as the constraint matrix is totally unimodular.

APDX.2.3. Connecting our Framework to the Literature

In this section, we show in APDX.2.3.1, APDX.2.3.2, and APDX.2.3.3 that Theorem 6 can be applied to obtain existing results about decomposability or polynomially-sized convex hull formulations for certain multilinear sets over $0-1$ variables given in the literature. We also present in APDX.2.3.4 an example that shows that our results provide a strict generalization.

Let $S = \{(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{x} \in \{0,1\}^n, y_j = \prod_{i \in e_j} x_i, \forall j \in [m]\}$ where $e_j \subseteq [n]$ for $j \in [m]$. Set S is commonly represented using a hypergraph H where $V(H) = [n]$ and $E(H) = \bigcup_{j \in [m]} \{e_j\}$. For this reason, we refer to S as the multilinear set associated with hypergraph H. We denote by $H[V']$ the section hypergraph of H induced by a subset of nodes $V' \subseteq V(H)$.

APDX.2.3.1. The multilinear set of a laminar hypergraph Theorem 10 in Del Pia and Khajavirad (2018a) provides a polynomially-sized formulation for the convex hull of the multilinear set S associated with a laminar hypergraph H. A hypergraph H is laminar if for every $e_i, e_j \in E(H)$, it holds that either $e_i \subseteq e_j$, $e_j \subseteq e_i$, or $e_i \cap e_j = \emptyset$. This result is a special case of Theorem 6. In fact, if a hypergraph H is laminar, $E(H)$ has the arborescence property. Set $V = \{\{v\}\}_{v \in V(H)} \cup E(H)$ has the arborescence property since H is laminar. Therefore, we can construct an arborescence G using Lemma 1 such that $V(G) = V$ and for every $u \in V(G)$, it holds that $z_u = \prod_{w \in V(G):(u,w) \in E(G)} z_w$ where, for each $u \in \{\{v\}\}_{v \in V(H)}$, z_u is x_u if $u \in V(H)$ and z_u is y_u if $u \in E(H)$. This arborescence-structured dependency allows S to be decomposed based on any arc $(u, w) \in E(G)$ into S_1 and S_2 where S_1 is the multilinear set containing z_w together with the multilinear terms associated with the descendants of w, and where S_2 is the set containing the remaining multilinear terms. Theorem 6 can then be applied without further refinement as S_2 depends on the single multilinear term z_w in S_1. Since both S_1 and S_2 can be expressed as the multilinear sets of laminar hypergraphs, the procedure can be applied recursively. At termination, we have expressed the convex hull of S using a polynomially-sized collection of convex hulls of multilinear sets with a single multilinear term, which can be easily convexified using the standard linearization.

Next, we restate Theorem 10 in Del Pia and Khajavirad (2018a) in Corollary APDX.1. We then provide a formal constructive proof that is a direct application of Theorem 6.

COROLLARY APDX.1 **(Theorem 10 in Del Pia and Khajavirad (2018a))**. *Let H be a laminar hypergraph. There exists a polynomially-sized formulation for the convex hull of the multilinear set S associated with H.*

Proof. Consider the multilinear set S of a hypergraph H and assume that H is laminar. Without loss of generality, we assume that H is connected as S can be convexified by treating each component separately otherwise. We claim that $[n] \in E(H)$ if H is laminar and connected. Assume by contradiction $[n] \notin E(H)$. Pick $e_k \in E(H)$ such that $|e_k| = \max_{j\in[m]} |e_j|$. By the assumption, $e_k \subsetneq [n]$. Pick $i \in [n] \setminus e_k$. Since H is laminar and there is no e_j such that $e_k \subsetneq e_j$, there is no $e_\ell \in E(H)$ that contains i and any node in e_k. Moreover, this holds for all $j \in [n] \setminus e_k$. Therefore, node i and the nodes in e_k are disconnected, which is a contradiction to the assumption that H is connected. This establishes that $[n] \in E(H)$. We can now construct, using Lemma 1, an arborescence G from $E(H)$ where the root is $[n]$. Select a leaf e_k from G and decompose S into S_1 and S_2 using Theorem 6 with $I = e_k$. Consider $\mathrm{ML}(D, \mathcal{F})$ corresponding to S. Since e_k is a leaf, for all $e_j \in E(H)$, it holds either $e_k \subseteq e_j$ or $e_k \cap e_j = \emptyset$. Let $\bar{F}$ be the face corresponding to e_k and $\mathcal{F}^0, \mathcal{F}^1, \mathcal{F}^2 = \texttt{partition}(\mathcal{F}, I)$. Then, $\mathcal{F}^0 = \{\bar{F}\}$, $\mathcal{F}^1 \cup \mathcal{F}^2 = \mathcal{F} \setminus \{\bar{F}\}$, and $\mathcal{F}^1_I = \{\bar{F}\}$. Since $\mathcal{F}^1_I$ is a disjoint set, we can apply Theorem 6 to decompose S into S_1 and S_2 where S_1 is the multilinear set of a hypergraph H_1 with $V(H_1) = e_k$ and $E(H_1) = \{e_k\}$ and S_2 is the multilinear set of a hypergraph H_2 with $V(H_2) = V(H) \setminus e_k \cup \{e_k\}$ and $E(H_2) = \{e \setminus e_k \cup \{e_k\}, \forall e \in E(H) : e_k \subsetneq e_j\} \cup \{e_j \in E(H) : e_j \cap e_k = \emptyset\}$. The multilinear set of H_1 is equivalent to $Q = \{(a_1, \ldots, a_{|e_k|}, b) \in \{0,1\}^{|e_k|+1} \mid b = a_1 \cdots a_{|e_k|}\}$, which can be easily convexified by $|e_k|+1$ linear inequalities and $|e_k|$ nonnegativity constraints using the standard linearization (Rikun 1997). The multilinear set of H_2 can be obtained from the multilinear set of H by substituting $\prod_{i\in e_k} x_i$ with y_j in all expressions, removing $\{x_i\}_{i\in e_k}$, and setting y_j to be a binary variable. Observe that, in this case, we did not need to introduce additional z variables. Therefore, this decomposition yields a set that can be easily convexified and another that corresponds to a strictly smaller hypergraph with $n - |e_k| + 1$ nodes. Moreover, H_2 is laminar because we removed e_k from $E(H)$ and replaced the nodes $v, \forall v \in e_k$ with e_k. Therefore, applying Theorem 6 on H_2 results in a decomposition of S into $\{S_t\}_{t=1}^{m+n}$, where the arborescence of S_t consists of a root e_t with children e_j for $j \in J_t$ and x_i for $i \in I_t$. Then, $S_t = \{(\boldsymbol{x}_{\boldsymbol{I_t}}, \boldsymbol{y}_{\boldsymbol{J_t}}, y_t) \in \{0,1\}^{|I_t|+|J_t|+1} \mid y_t = \prod_{i\in I_t} x_i \prod_{j\in J_t} y_j\}$. By convexifying each S_t using standard linearization techniques, we obtain the desired formulation for $\mathrm{conv}(S)$. □

APDX.2.3.2. Decomposability of a multilinear set. Theorem 1 in Del Pia and Khajavirad (2018b) gives the following sufficient conditions for decomposability of the multilinear set S associated with a hypergraph H: if there exist $V_1, V_2 \subseteq V(H)$ such that $V_1 \cup V_2 = V(H)$, $H[V_1 \cap V_2]$ is complete, and either $e \subseteq V_1$ or $e \subseteq V_2$ holds for every $e \in E(H)$, then S is decomposable into the multilinear sets associated with $H[V_1]$ and $H[V_2]$. The conditions are equivalent to the special case of Theorem 6 with $I = V_1$ or V_2, and $\mathcal{F}'_I = \text{vert}(P_I)$. This is because the complete refinement of all multilinear terms in $V_1 \cap V_2$ is polynomial in the number of multilinear terms relating variables in $V_1 \cap V_2$.

We restate Theorem 1 in Del Pia and Khajavirad (2018b) in Corollary APDX.2. We then provide a proof that uses Theorem 6 to show that S corresponds to a multilinear set over the Cartesian product of simplices that is decomposable.

COROLLARY APDX.2 **(Theorem 1 in Del Pia and Khajavirad (2018b))**. *Let S be the multilinear set of a hypergraph H. Assume $V_1, V_2 \subsetneq V(H)$ are such that $V(H) = V_1 \cup V_2$, $E(H) = E(H[V_1]) \cup E(H[V_2])$, and $H[V_1 \cap V_2]$ is a complete hypergraph. Then, the convex hull of S is obtained by convexifying S_1 and S_2 separately where S_1 (resp. S_2) is the multilinear set of $H[V_1]$ (resp. $H[V_2]$).*

Proof. Let $E_1 = E(H[V_1])$ and $E_2 = E(H[V_2])$. We define the multilinear set over the Cartesian product of simplices corresponding to S. Let $P_v = [0,1]$, $\forall v \in V(H)$, $P = \prod_{v \in V(H)} P_v$, and $D = \text{vert}(P)$. We denote by $F(e)$ the face corresponding to hyperedge $e \in E(H)$ (we convert x_v to $x_{v,1}$) and denote by $\mathcal{F}(E)$ the set of faces corresponding to hyperedges in E, *i.e.*, $\mathcal{F}(E) = \bigcup_{e \in E} \{F(e)\}$. Let $\mathcal{F} = \mathcal{F}(E(H)) \cup \mathcal{F}'$ where $\mathcal{F}' = \bigcup_{F \in \text{vert}(P_{V_1 \cap V_2})} F \times P_{V(H) \setminus (V_1 \cap V_2)}$. The number of faces in $\mathcal{F}$ is a polynomial in $|V(H)|$ and $|E(H)|$ because $|\mathcal{F}| \leq |E(H)| + 2^{|V_1 \cap V_2|} \leq 2|E(H)| + |V(H)| + 1$ where the inequality holds because $|E(H)| \geq 2^{|V_1 \cap V_2|} - |V(H)| - 1$ as $H[V_1 \cap V_2]$ is complete. Now, we apply Theorem 6 to $\text{ML}(D, \mathcal{F})$. Let $I = V_1$ and let $\bar{I} = V(H) \setminus V_1$. Let $\mathcal{F}^0, \mathcal{F}^1, \mathcal{F}^2 = \texttt{partition}(\mathcal{F}, I)$. Then, $\mathcal{F}^0 = \mathcal{F}(E_1) \cup \mathcal{F}'$ and $\mathcal{F}^1 \cup \mathcal{F}^2 = \mathcal{F}(E_2 \setminus E_1)$. Let $\mathcal{F}^1_I = \bigcup_{F \in \mathcal{F}^1} \{F_I\}$ and let $\mathcal{F}'_I = \bigcup_{F \in \mathcal{F}'} \{F_I\}$. Set $\mathcal{F}'_I$ is a refinement of $\mathcal{F}^1_I$ because every face in $\mathcal{F}^1_I$ is in the form of $P_{V_1 \setminus V_2} \times F$ for some face F in $P_{V_1 \cap V_2}$ and $\mathcal{F}'_I = \bigcup_{F \in \text{vert}(P_{V_1 \cap V_2})} P_{V_1 \setminus V_2} \times F$. For each $F \in \mathcal{F}^1_I$, let $C(F)$ be such that

$C(F) \subseteq \mathcal{F}'_I$ and $\text{conv}\big(\bigcup_{\bar{F}\in C(F)} \bar{F}\big) = F$, *i.e.*, $C(F)$ is the set of faces in $\mathcal{F}'$ that refine F. Therefore, $\text{conv}(\text{ML}(D,\mathcal{F}))$ can be obtained by convexifying $\text{ML}(D_I,\mathcal{F}^0)$ and $\text{ML}(D_{\bar{I}} \times \Delta^{2^{|V_1\cap V_2|}}, \hat{\mathcal{F}}^1 \cup \hat{\mathcal{F}}^2)$ where $\hat{\mathcal{F}}^2 = \bigcup_{F\in\mathcal{F}^2}\{F_{V_2\setminus V_1} \times \Delta^{2^{|V_1\cap V_2|}}\}$ and $\hat{\mathcal{F}}^1 = \bigcup_{F\in\mathcal{F}^1}\{F_{V_2\setminus V_1} \times \text{conv}(\bigcup_{\bar{F}\in C(F)} e_{\text{idx}(\bar{F})})\}$ where $e_i \in \{0,1\}^{2^{|V_1\cap V_2|}}$ is the i^{th} principal vector. This is an equivalent decomposition because $\text{ML}(D_I,\mathcal{F}^0)$ is the multilinear set over the variables corresponding to V_1 and $\text{ML}(D_{V_2\setminus V_1} \times \Delta^{2^{|V_1\cap V_2|}}, \hat{\mathcal{F}}^1 \cup \hat{\mathcal{F}}^2\}$ is the multilinear set over the variables corresponding to V_2 where the variables corresponding to $V_1 \cap V_2$ is lifted to a space of $2^{|I|}$ variables. □

APDX.2.3.3. Connection to Bienstock and Munoz (2018). Bienstock and Munoz (2018) study formulations for the convex hull of S where the structural sparsity of S is described by a tree-decomposition. They show their formulations to be polynomially-sized when the tree-width is constant. Theorem 6 allows us to also derive formulations that are polynomial under these assumptions. For our set, the notion of tree-decomposition becomes

Definition APDX.1. Consider a set of proper faces $\mathcal{F}$ of $P := \prod_{v\in[n]} P_v$ where $P_v := \Delta^{K_v}$ for all $v \in [n]$. Let $D = \text{vert}(P)$. A *tree-decomposition* of $\text{ML}(D,\mathcal{F})$ is a pair (T,Q), where T is a tree and $Q = \{Q_t\}_{t\in V(T)}$ is a collection of subsets of $[n]$ (the indices of simplices) such that

(i) For all $v \in [n]$, the set $\{t \in V(T) : v \in Q_t\}$ forms a subtree T_v of T,

(ii) For each $F \in \mathcal{F}$, there is a $t \in V(T)$ such that $e(F) \subseteq Q_t$ where $e(F) := \{v \in [n] : J^v_F \subsetneq [K_v]\}\}$,

(iii) $\bigcup_{t\in V(T)} Q_t = [n]$.

The *width* of a tree-decomposition is defined as $\max_{t\in V(T)} |Q_t| - 1$. The *treewidth* of $\text{ML}(D,\mathcal{F})$ is the minimum width over all tree-decompositions of $\text{ML}(D,\mathcal{F})$.

It is known that, if the treewidth of $\text{ML}(D,\mathcal{F})$ is a constant, then a tree-decomposition (T,Q) can be found in time linear in n and $|\mathcal{F}|$; see Bodlaender (1996).

Consider a set $\text{ML}(D,\mathcal{F})$ for which we have obtained a tree-decomposition (T,Q). We can decompose $\text{ML}(D,\mathcal{F})$ using Theorem 6 into $|V(T)|$ multilinear sets such that the convex hull of each set is described by a constant number of variables and constraints. This construction leads to

COROLLARY APDX.3. *Consider a set of proper faces $\mathcal{F}$ of $P := \prod_{v\in[n]} P_v$ where $P_v := \Delta^{K_v}$ for all $v \in [n]$. Let $D = \text{vert}(P)$. Assume that the treewidth of $ML(D,\mathcal{F})$ and that K_v for all $v \in [n]$ are all constants. Then, there exists an extended formulation for $\text{conv}(ML(D,\mathcal{F}))$ whose size is polynomial in n and $|\mathcal{F}|$.*

Proof. Let r be the treewidth of $\text{ML}(D,\mathcal{F})$. Let (T,Q) be a tree-decomposition of $\text{ML}(D,\mathcal{F})$ obtained in linear time using the algorithm in (Bodlaender 1996). It follows that the number of vertices in T is bounded above by a linear function in n and $|\mathcal{F}|$. Consider any edge $\{u,w\} \in E(T)$ such that $Q_u \cap Q_w = \emptyset$. Let $I_u = \bigcup_{t\in V_u} Q_t$ (resp. $I_w = \bigcup_{t\in V_w} Q_t$) where V_u (resp. V_w) is the set of vertices in $V(T)$ such that every vertex in V_u (resp. V_w) is reachable from u (resp. w) without using $\{u,w\}$. Sets I_u and I_w form a partition of $[n]$ by the definition of a tree-decomposition. Moreover, every face F in $\mathcal{F}$ can be expressed as either $F_{I_u} \times P_{I_w}$ or $P_{I_u} \times F_{I_w}$. Therefore, we can convexify $\text{ML}(D,\mathcal{F})$ by convexifying $\text{ML}(D_{I_u},\mathcal{F}_{I_u})$ and $\text{ML}(D_{I_w},\mathcal{F}_{I_w})$ separately where $\mathcal{F}_{I_u} = \{F_{I_u} \mid F \in \mathcal{F}\}$ and $\mathcal{F}_{I_w} = \{F_{I_w} \mid F \in \mathcal{F}\}$. We may therefore assume that for every edge $\{u,w\} \in E(T)$, $Q_u \cap Q_w \neq \emptyset$. We next argue that we may assume that for every edge $\{u,w\} \in E(T)$, both $Q_u \subsetneq Q_w$ and $Q_w \subsetneq Q_u$ hold. Otherwise, we could shrink $\{u,w\}$ (*i.e.*, remove edge $\{u,w\}$, combine vertices u and w into one vertex v, and define $Q_v := Q_u \cup Q_w$) so as to obtain another tree-decomposition of $\text{ML}(D,\mathcal{F})$ with fewer vertices. Recursively shrinking such edges would then lead to a tree-decomposition that satisfies the assumption.

Using Proposition 3, we show that there is a polynomially-sized formulation for $\text{conv}(\text{ML}(D,\mathcal{F}))$ by constructing a refinement $\mathcal{F}'$ of $\mathcal{F}$ where $|\mathcal{F}'|$ is a linear function in n and $|\mathcal{F}|$. Define $\mathcal{F}' := \bigcup_{\{t\}\in V(T)} \mathcal{F}_t$ where $\mathcal{F}_t := \{F \times P_{[n]\setminus Q_t}\}_{F\in\text{vert}(P_{Q_t})}$ for all $t \in V(T)$. We first argue that $\mathcal{F}'$ is a refinement of $\mathcal{F}$. For every $F \in \mathcal{F}$, there is $t \in V(T)$ such that $F = F_{Q_t} \times P_{[n]\setminus Q_t}$ and F_{Q_t} is a face of P_{Q_t} by (ii) in the definition of tree-decomposition. Since for every vertex v of P_{Q_t}, $v \times P_{[n]\setminus Q_t}$ is in $\mathcal{F}'$, it follows that $\mathcal{F}'$ refines F, therefore, $\mathcal{F}'$ refines $\mathcal{F}$. We next argue that $|\mathcal{F}'|$ is bounded above by a linear function of n and $|\mathcal{F}|$. By the definition of $\mathcal{F}'$, we have $|\mathcal{F}'| \leq \sum_{t\in V(T)} \prod_{v\in Q_t} K_v$. The right-hand-side of this expression is a linear function of n and $|\mathcal{F}|$ because (i) $\{K_v\}_{v\in[n]}$ are

constants by assumption, (ii) the maximum cardinality of Q_t for $t \in V(T)$ is bounded above by the treewidth of $\mathrm{ML}(D, \mathcal{F})$ plus one, which is constant, and (iii) $|V(T)|$ is a linear function of n and $|\mathcal{F}|$. Therefore, $\mathcal{F}'$ is a refinement of $\mathcal{F}$ where $|\mathcal{F}'|$ is a linear function of n and $|\mathcal{F}|$.

Next, we obtain a formulation for $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F}'))$. We apply Theorem 6 to decompose $\mathrm{ML}(D, \mathcal{F}')$. Pick an edge $\{u, w\}$ in $E(T)$ where w is a leaf. Let $I = \bigcup_{t \in V(T): t \neq w} Q_t$. It follows from the assumption that $Q_w \subsetneq Q_u$ and from (i) in the definition of a tree-decomposition that $I \subsetneq [n]$. Let $\mathcal{F}^0, \mathcal{F}^1, \mathcal{F}^2 = \texttt{partition}(\mathcal{F}', I)$. We argue that $\mathcal{F}^0 = \mathcal{F}' \setminus \mathcal{F}_w$, $\mathcal{F}^1 = \mathcal{F}_w$, and $\mathcal{F}^2 = \emptyset$. Recall that $\mathcal{F}' = \bigcup_{t \in V(T)} \mathcal{F}_t$. For $t \in V(T)$ that is distinct from w, $\mathcal{F}_t \subseteq \mathcal{F}^0$ since $Q_t \subseteq I$. For $F \in \mathcal{F}_w$, $F \in \mathcal{F}^1$ since F is a vertex of P_{Q_w}, $Q_w \cap I \neq \emptyset$, and $Q_w \cap \bar{I} \neq \emptyset$. Therefore, $\mathcal{F}^0 = \mathcal{F}' \setminus \mathcal{F}_w$, $\mathcal{F}^1 = \mathcal{F}_w$, and $\mathcal{F}^2 = \emptyset$. Let $\mathcal{F}^i_I = \{F_I \mid F \in \mathcal{F}^i\}$ for $i \in \{0, 1\}$. We argue that $\mathcal{F}^0_I$ is a refinement of $\mathcal{F}^1_I$. Since $\mathcal{F}^1 = \mathcal{F}_w$, then $\mathcal{F}^1_I = \{F_{I \cap Q_w} \times P_{I \setminus Q_w}\}_{F \in \mathrm{vert}(P_{Q_w})} = \{F \times P_{I \setminus Q_w}\}_{F \in \mathrm{vert}(P_{Q_w \cap I})}$. Also, since $\mathcal{F}_u \subseteq \mathcal{F}^0$, then $\mathcal{F}^0_I \supseteq \{F \times P_{I \setminus Q_u}\}_{F \in \mathrm{vert}(P_{Q_u})}$. Since $Q_w \cap I = Q_w \cap Q_u \subseteq Q_u$, then $\mathcal{F}^0_I$ is a refinement of $\mathcal{F}^1_I$. Therefore, we can apply Theorem 6 to decompose $\mathrm{ML}(D, \mathcal{F}')$ without adding any additional faces. Let $\mathrm{idx}(F)$ be the index of F in $\mathcal{F}_u$ for every $F \in \mathcal{F}_u$. Let $e_i \in \{0, 1\}^{|\mathcal{F}_u|}$ be the i^{th} principal vector. Let $C(F)$ be such that $C(F) \subseteq \mathcal{F}_u$ and $\mathrm{conv}\left(\bigcup_{\bar{F} \in C(F)} \bar{F}_I\right) = F$ for $F \in \mathcal{F}^1_I$. By Theorem 6, we can obtain a formulation for $\mathrm{conv}(\mathrm{ML}(D, \mathcal{F}'))$ by separately convexifying $\mathrm{ML}(D_I, \mathcal{F}^0_I)$ and $\mathrm{ML}(D_{\bar{I}} \times \Delta_{0,1}^{|\mathcal{F}_u|}, \hat{\mathcal{F}}^1)$ where $\hat{\mathcal{F}}^1 = \left\{F_{\bar{I}} \times \mathrm{conv}\left(\bigcup_{\bar{F} \in C(F)} \{e_{\mathrm{idx}(\bar{F})}\}\right)\right\}_{F \in \mathcal{F}^1}$. The latter set is a multilinear set over the Cartesian product of a constant number of simplices with a constant number of faces. The former set either is of the same form as the latter (when it corresponds to a tree with a single node) or is a multilinear set represented by a constant-treewidth tree-decomposition with $V(T) - 1$ vertices, *i.e.*, the tree-decomposition obtained by removing edge $\{u, w\}$ from T. Therefore, by recursively applying Theorem 6, we can convexify $\mathrm{ML}(D, \mathcal{F}')$ by separately convexifying $|V(T)|$ multilinear sets, for which constant-size convex hulls descriptions can be explicitly constructed since every multilinear set is associated with a constant number of simplices and a constant number of faces. This construction yields a polynomially-sized formulation for $\mathrm{ML}(D, \mathcal{F}')$. In turn, this formulation can be used to obtain a polynomially-sized formulation for $\mathrm{ML}(D, \mathcal{F})$. $\square$

The proof of Corollary APDX.3 recursively applies the idea that for each leaf-node, if we track all possible multilinear terms that relate to variables in the neighboring node, then the convex hull can be developed independent of the rest of the tree. This decomposition follows from Theorem 6 since the convex hull of all multilinear terms for a Cartesian product of simplices is a simplex. It follows that the main burden in the explicit construction of this convex hull resides in the determination of the tree-decomposition, which can be found in time linear in n and $|\mathcal{F}|$.

Theorem 3.5 in Bienstock and Munoz (2018) can then be viewed as a special case of Corollary APDX.3.

APDX.2.3.4. An Example Showing Strict Generalization. Example APDX.1 shows that the framework for convexification presented in this paper is in fact strictly more general than the results of the literature reviewed above. It does so by describing a set whose convex hull cannot be directly obtained using these results but can be derived from Theorem 6, together with other theorems in this paper.

EXAMPLE APDX.1. Consider

$$S = \left\{ (\boldsymbol{x}, \boldsymbol{y}) \in \{0,1\}^{2n} \times \{0,1\}^{3n} \;\middle|\; \begin{array}{ll} y_j = (1-x_j)\prod_{i=1}^{j-1} x_i, & \forall j \in [n] \\ y_{n+j} = y_j x_{n+j}, & \forall j \in [n] \\ y_{2n+j} = y_j x_{n+j} x_{n+j+1}, & \forall j \in [n-1] \\ y_{3n} = y_n x_{2n} x_{n+1} & \end{array} \right\}.$$

Theorem 6 can be used to decompose S into S_1 and S_2 where S_1 is the multilinear set over $x_1, \ldots, x_n$ with multilinear terms $y_1, \ldots, y_n$ and S_2 is the multilinear set over $x_{n+1}, \ldots, x_{2n}, y_1, \ldots, y_n$ with multilinear terms $y_{n+1}, \ldots, y_{3n}$ using $I = [n]$. Specifically,

$$S_1 = \left\{ (\boldsymbol{x_I}, \boldsymbol{y_I}) \in \{0,1\}^n \times \{0,1\}^n \;\middle|\; y_j = (1-x_j)\prod_{i=1}^{j-1} x_i, \;\; \forall j \in [n] \right\}$$

$$S_2 = \left\{ (\boldsymbol{x_{\bar{I}}}, \boldsymbol{y_I}, \boldsymbol{y_{\bar{I}}}) \in \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^{2n} \;\middle|\; \begin{array}{ll} y_{n+j} = y_j x_{n+j}, & \forall j \in [n] \\ y_{2n+j} = y_j x_{n+j} x_{n+j+1}, & \forall j \in [n-1] \\ y_{3n} = y_n x_{2n} x_{n+1} & \\ \mathbf{1}^\intercal \boldsymbol{y_I} \leq 1 & \end{array} \right\}.$$

We remark that if we replace y_j for $j \in [n]$ with $\bar{y}_j = \prod_{i=1}^{j} x_i$, the resulting set can still be convexified using the techniques in this paper. This is because, as mentioned before, S_1 is an affine transform of the set obtained after replacing y_j. We can obtain $\text{conv}(S)$ by convexifying S_1 and S_2 separately. A polynomially-sized formulation for $\text{conv}(S_1)$ can be obtained using Theorem 3 because $\boldsymbol{y_I}$ corresponds to a facial decomposition of the Cartesian product of simplices corresponding to $\boldsymbol{x_I}$ and because it has the adjacency property. We can obtain a polynomially-sized formulation for $\text{conv}(S_2)$ because the set of faces corresponding to $\{y_j\}_{j=n+1}^{3n}$ is poly-completable according to Theorem 5.

Existing results in the literature do not apply to S. The treewidth of S is at least $n+2$ because the degree of y_{3n} in terms of $\boldsymbol{x}$ variables is $n+2$. Hence, Theorem 3.5 in Bienstock and Munoz (2018) does not yield a polynomially-sized formulation. Consider next the hypergraph H associated with S obtained by creating hyperedges for all monomials that appear in the expression: $V(H) = [2n]$ and $E(H) = E_1 \cup E_2 \cup E_3$ where

$$E_1 = \{\{1,2\},\{1,2,3\},\ldots,\{1,2,\ldots,n\}\}, \quad E_2 = \{\{1,n+1\},\{\{1,2,n+2\},\ldots,\{1,2,\ldots,n,2n\}\},$$

$$E_3 = \{\{1,n+1,n+2\},\{1,2,n+2,n+3\},\ldots,\{1,\ldots,n-1,2n-1,2n\},\{1,\ldots,n+1,2n\}\}.$$

Sets E_1, E_2, and E_3 are associated with multilinear terms $\{y_j\}_{j\in[n]}$, $\{y_{n+j}\}_{j\in[n]}$, and $\{y_{2n+j}\}_{j\in[n]}$, respectively. The hypergraph H is neither laminar nor γ-acyclic. In fact, H contains many γ-cycles in H, for example, $v_1 - e_1 - v_2 - e_2 - v_3 - e_3 - v_1$ where $v_1 = 1$, $v_2 = k$, $v_3 = n+k$, $e_1 = \{1,\ldots,k\}$, $e_2 = \{1,\ldots,k,n+k\}$, $e_3 = \{1,\ldots,k-1,n+k-1,n+k\}$ for all $k = 2,\ldots,n$ and $v_1 - e_1 - v_2 - e_n - \ldots - v_n - e_n - v_1$ where $v_j = n+j$ for all $j \in [n]$, $e_j = \{1,\ldots,j,n+j,n+j+1\}$ for all $j \in [n-1]$, and $e_n = \{1,\ldots,n,n+1,2n\}$. Therefore, we cannot apply either Theorem 10 or Corollary 18 from Del Pia and Khajavirad (2018a) to convexify S. Further, decomposition Theorem 1 in Del Pia and Khajavirad (2018b) cannot be used to convexify S. In fact, $H[V']$ is complete for $V' \subseteq V(H)$ only if $V' = \{1,2\}$ or $|V'| = 1$. However, there is no $V_1, V_2 \subseteq V(H)$ such that $V_1 \cap V_2 = \{1,2\}$ or $|V_1 \cap V_2| = 1$, and either $e \subseteq V_1$ or $e \subseteq V_2$ holds for every $e \in E(H)$. ■

APDX.3. Supplements for Section 5

APDX.3.1 explains the procedure to generate transportation problem instances used in Section 5.2. APDX.3.2 provides the explicit formulations used in the computational experiment in Section 5.2. APDX.3.3 provides the proof of the uniqueness of a multilinear function over a hyper-rectangle given all function values on its vertices.

APDX.3.1. Generation of Multi-Commodity Flow Problem Instances

In this section, we introduce the procedure we use to generate instances of the multi-commodity flow problem used in the computational experiments. First, we randomly generate supply and demand with the idea that there is 20% more supply than demand. We set the total demand (resp. total supply) of the i^{th} commodity to be $D_i = 100 \cdot 3^{i-1}$ (resp. $S_i = 1.2D_i$) for $i \in [C]$. For $i \in [C]$, $s_{u,i}$ for $u \in U$ represents the i^{th} commodity's supply and $d_{v,i}$ for $v \in V$ represents the i^{th} commodity's demand. We randomly distribute the total demand (resp. supply) to the demand (resp. supply) nodes using the following procedure: (i) pick $p_{v,i} = \text{Uniform}(1,2)$ for all $v \in V$ and $i \in [C]$ (resp. pick $p_{u,i} = \text{Uniform}(1,2)$ for all $u \in U$ and $i \in [C]$), (ii) compute $d_{v,i} = \lfloor \frac{p_{v,i}}{\sum_{v \in V} p_{v,i}} D_i \rfloor$ for all $v \in V$ and $i \in [C]$ (resp. $s_{u,i} = \lceil \frac{p_{u,i}}{\sum_{u \in U} p_{u,i}} S_i \rceil$ for all $u \in U$ and $i \in [C]$). Second, we deduce some suitable values for capacities, that is, for $i \in [C]$ and $e \in E$, the arc capacity $c_{e,i}$ for the i^{th} commodity on arc e is computed as $\lceil 0.25\,(s_{u,i} + d_{v,i}) \rceil$. Third, for each arc $e \in E$, we define a hidden cost function $g_e(\boldsymbol{z_e}) = (1 + p_1(\boldsymbol{z_e}) + p_2(\boldsymbol{z_e}))\,||\boldsymbol{z_e}||_2$ where $c_e = \sum_{i \in [C]} c_{e,i}$, $p_1(\boldsymbol{z_e}) = \frac{\max\{0.25 c_e - \sum_{i \in [C]} z_{e,i}, 0\}}{0.25 c_e} \in [0,1]$, and $p_2(\boldsymbol{z_e}) = \frac{5 \max\{\sum_{i \in [C]} z_{e,i} - 0.75 c_e, 0\}}{0.25 c_e} \in [0,5]$ for all $e \in E$. The underlying idea is that $g_e(\boldsymbol{z_e})$ is proportional to the Euclidean distance between the flow $\boldsymbol{z_e}$ and the zero flow, and that there are penalties when total flow on arc e is less than 25% or more than 75% of its total capacity c_e. We then generate the data points for the cost incurred on arc e as $\{(\bar{\boldsymbol{z}}, g_e(\bar{\boldsymbol{z}}))\}_{\bar{\boldsymbol{z}} \in \mathbb{Z} \cap \prod_{i \in [C]} [0, c_{e,i}]}$, that is, we compute the value of g at all the integer points in the hyper-rectangular domain bounded by individual capacities. We train a random forest model using `RandomForestRegressor` in `scikit-learn` package in Python (Pedregosa et al. 2011) with parameters (D, T) to describe $\{(\bar{\boldsymbol{z}}_{\boldsymbol{e}}^{\boldsymbol{j}}, g_e(\bar{\boldsymbol{z}}_{\boldsymbol{e}}^{\boldsymbol{j}}))\}_{j \in [n_{\text{data}}]}$ for each arc $e \in E$ and use those trees in the multi-commodity transportation problem instance.

APDX.3.2. Formulations Used in the Experiments

In this section, we provide explicit formulations for (TEO), (TEOM), (MC), (DCC), and (DLog). For $e \in E$ and $i \in [C]$, we let $d_{e,i,1} \leq \ldots \leq d_{e,i,K_{e,i}}$ be the unique split values of the flow variables $z_{e,i}$ in the decision trees associated with the cost on arc e. Clearly, $K_{e,i}$ represents the number of unique split values on $z_{e,i}$. We introduce interval indicator variables (in incremental form) for $z_{e,i}$ that we call $x_{e,i,j}$, so that $x_{e,i,j} = \mathbb{1}[z_{e,i} \leq d_{e,i,j}]$. Let $\mathbf{leaves}(e,t)$ (resp. $\mathbf{nleaves}(e,t)$) be the set of leaves (resp. non-leaves) in the t^{th} tree describing the cost on arc $e \in E$. For $e \in E$, $t \in [T_e]$, $\ell \in \mathbf{leaves}(e,t)$, we denote by $\delta_{e,t,\ell,i}$ the interval corresponding to the projection in the space of $z_{e,i}$ of the hyper-rectangle of leaf ℓ in the t^{th} tree associated with arc $e \in E$. To streamline presentation, we use $x_{e,i,0} := 0$, $x_{e,i,K_{e,i}+1} := 1$, $d_{e,i,0} := -\infty$, and $d_{e,i,K_{e,i}+1} := \infty$ for all $i \in [C]$ and $e \in E$. Also, $\ell \in \mathbf{leaves}(e,t)$ for $e \in E$ and $t \in [T_e]$ indicates the index of the leaf. The explicit formulation of (TEO) is as follows:

$$\min \quad \sum_{e\in E} \sum_{t\in[T_e]} \sum_{\ell\in\mathbf{leaves}(e,t)} p_{e,t,\ell} y_{e,t,\ell} \tag{APDX.14a}$$

$$\text{s.t.} \quad \sum_{e=(u',v')\in E: u'=u} z_{e,i} \leq s_{u,i}, \qquad \forall u \in U, \forall i \in [C], \tag{APDX.14b}$$

$$\sum_{e=(u',v')\in E: v'=v} z_{e,i} \geq d_{v,i}, \qquad \forall v \in V, i \in [C], \tag{APDX.14c}$$

$$z_{e,i} \geq \sum_{j\in[K_{e,i}]} (d_{e,i,j-1} - d_{e,i,j}) x_{e,i,j}, \qquad \forall e \in E, \forall i \in [C], \tag{APDX.14d}$$

$$z_{e,i} \leq c_{e,i} + \sum_{j\in[K_{e,i}]} (d_{e,i,j} - d_{e,i,j+1}) x_{e,i,j}, \qquad \forall e \in E, \forall i \in [C], \tag{APDX.14e}$$

$$\sum_{\ell\in\mathbf{leaves}(e,t):\delta_{e,t,\ell,i}\subseteq[d_{e,i,j_1},d_{e,i,j_2}]} y_{e,t,\ell} \leq x_{i,j_2} - x_{i,j_1}, \quad \forall e \in E, \forall t \in [T_e], \forall i \in [C], \forall j_1, j_2 \in [0..(K_{e,i}+1)] : j_1 < j_2, \tag{APDX.14f}$$

$$x_{e,i,j} \leq x_{e,i,j+1}, \qquad \forall e \in E, \forall i \in [C], \forall j \in [K_{e,i}-1], \tag{APDX.14g}$$

$$x_{e,i,j} \in \{0,1\}, \qquad \forall e \in E, \forall i \in [C], \forall j \in [K_{e,i}], \tag{APDX.14h}$$

$$\boldsymbol{y_{e,t}} \in \Delta^{|\mathbf{leaves}(e,t)|}, \qquad \forall e \in E, \forall t \in [T_e], \tag{APDX.14i}$$

$$z_{e,i} \in \mathbb{R}, \qquad \forall e \in E, \forall i \in [C]. \tag{APDX.14j}$$

We next obtain (TEOM) from (TEO). Let $\mathbf{lchild}(s)$ (resp. $\mathbf{rchild}(s)$) be the set of leaves under the left (resp. right) child of s. For $e \in E$, $t \in [T_e]$, $s \in \mathbf{nleaves}(e,t)$, let $i(s)$ (resp. $j(s)$) be the index of the independent variable (resp. the split value) associated with the query on n, that is, "$z_{e,i(s)} \le d_{e,i(s),j(s)}$?" We obtain (TEOM) from (TEO) by replacing (APDX.14f) with (APDX.15):

$$\sum_{\ell \in \mathbf{lchild}(s)} y_{e,t,\ell} \le x_{e,i(s),j(s)}, \qquad \forall e \in E, \forall t \in [T_e], \forall s \in \mathbf{nleaves}(e,t), \tag{APDX.15a}$$

$$\sum_{\ell \in \mathbf{rchild}(s)} y_{e,t,\ell} \le 1 - x_{e,i(s),j(s)}, \qquad \forall e \in E, \forall t \in [T_e], \forall s \in \mathbf{nleaves}(e,t), \tag{APDX.15b}$$

that is,

$$\begin{aligned} \min \quad & \sum_{e \in E} \sum_{t \in [T_e]} \sum_{\ell \in \mathbf{leaves}(e,t)} p_{e,t,\ell} y_{e,t,\ell} \\ \text{s.t.} \quad & \text{(APDX.14b)–(APDX.14e), (APDX.14g)–(APDX.14j), (APDX.15).} \end{aligned}$$

Formulations (MC), (DCC), and (DLog) are developed based on general disjunctive programming techniques (Vielma et al. 2010). The formulations use $\boldsymbol{y}$ as binary variable to indicate active hyper-rectangles. For $e \in E$, $t \in [T_e]$, $i \in [C]$, $\ell \in \mathbf{leaves}(e,t)$, let $\mathrm{lb}(e,t,i,\ell)$ (resp. $\mathrm{ub}(e,t,i,\ell)$) be the lower- (resp. upper-) limit on $z_{e,i}$ of the hyper-rectangle corresponding to leaf ℓ. The explicit formulation of (MC) is as follows:

$$\min \quad \sum_{e \in E} \sum_{t \in [T_e]} \sum_{\ell \in \mathbf{leaves}(e,t)} p_{e,t,\ell} y_{e,t,\ell} \tag{APDX.16a}$$

$$\text{s.t.} \quad \sum_{e=(u',v') \in E : u' = u} z_{e,i} \le s_{u,i}, \qquad \forall u \in U, \forall i \in [C], \tag{APDX.16b}$$

$$\sum_{e=(u',v') \in E : v' = v} z_{e,i} \ge d_{v,i}, \qquad \forall v \in V, i \in [C], \tag{APDX.16c}$$

$$z_{e,i} = \sum_{\ell \in \mathbf{leaves}(e,t)} z_{e,i,t,\ell}, \qquad \forall e \in E, t \in [T_e], \tag{APDX.16d}$$

$$z_{e,i,t,\ell} \geq \max(0, \mathrm{lb}(e,t,i,\ell)) y_{e,t,\ell}, \qquad \forall e \in E, \forall i \in [C], t \in [T_e], \ell \in \mathbf{leaves}(e,t), \quad \text{(APDX.16e)}$$

$$z_{e,i,t,\ell} \leq \min(c_{e,i}, \mathrm{ub}(e,t,i,\ell)) y_{e,t,\ell}, \qquad \forall e \in E, \forall i \in [C], t \in [T_e], \ell \in \mathbf{leaves}(e,t), \quad \text{(APDX.16f)}$$

$$\boldsymbol{y_{e,t}} \in \Delta_{0,1}^{|\mathbf{leaves}(e,t)|}, \qquad \forall e \in E, \forall t \in [T_e], \quad \text{(APDX.16g)}$$

$$z_{e,i} \in \mathbb{R}, \qquad \forall e \in E, \forall i \in [C], \quad \text{(APDX.16h)}$$

$$z_{e,i,t,\ell} \in \mathbb{R}, \qquad \forall e \in E, \forall i \in [C], t \in [T_e], \ell \in \mathbf{leaves}(e,t). \quad \text{(APDX.16i)}$$

For $e \in E$, $t \in [T_e]$, $\ell \in \mathbf{leaves}(e,t)$, let $Q_{e,t,\ell} \subseteq \prod_{i \in [C]} [0, c_{e,i}]$ be the hyper-rectangle corresponding to the leaf ℓ in tree t. Let $\bar{\boldsymbol{z}}_{e,t,\ell,k} \in \mathbb{R}^C$ be the k^{th} extreme point (among 2^C of them) of $Q_{e,t,\ell}$. The explicit formulation of (DCC) is as follows:

$$\min \quad \sum_{e \in E} \sum_{t \in [T_e]} \sum_{\ell \in \mathbf{leaves}(e,t)} p_{e,t,\ell} y_{e,t,\ell} \quad \text{(APDX.17a)}$$

$$\text{s.t.} \quad \sum_{e=(u',v') \in E : u' = u} z_{e,i} \leq s_{u,i}, \qquad \forall u \in U, \forall i \in [C], \quad \text{(APDX.17b)}$$

$$\sum_{e=(u',v') \in E : v' = v} z_{e,i} \geq d_{v,i}, \qquad \forall v \in V, i \in [C], \quad \text{(APDX.17c)}$$

$$\boldsymbol{z_e} = \sum_{\ell \in \mathbf{leaves}(e,t)} \sum_{k \in [2^C]} \lambda_{e,t,\ell,k} \bar{\boldsymbol{z}}_{e,t,\ell,k}, \qquad \forall e \in E, t \in [T_e], \quad \text{(APDX.17d)}$$

$$\sum_{\ell \in \mathbf{leaves}(e,t)} \sum_{k \in [2^C]} \lambda_{e,t,\ell,k} = 1, \qquad \forall e \in E, t \in [T_e], \quad \text{(APDX.17e)}$$

$$\sum_{k \in [2^C]} \lambda_{e,t,\ell,k} \leq y_{e,t,\ell}, \qquad \forall e \in E, t \in [T_e], \ell \in \mathbf{leaves}(e,t), \quad \text{(APDX.17f)}$$

$$\boldsymbol{y_{e,t}} \in \Delta_{0,1}^{|\mathbf{leaves}(e,t)|}, \qquad \forall e \in E, \forall t \in [T_e], \quad \text{(APDX.17g)}$$

$$z_{e,i} \in \mathbb{R}, \qquad \forall e \in E, \forall i \in [C], \quad \text{(APDX.17h)}$$

$$\lambda_{e,t,\ell,k} \in \mathbb{R}_+, \qquad \forall e \in E, t \in [T_e], \ell \in \mathbf{leaves}(e,t), k \in [2^C]. \quad \text{(APDX.17i)}$$

The explicit formulation of (DLog) is as follows:

$$\min \quad \sum_{e \in E} \sum_{t \in [T_e]} \sum_{\ell \in \mathbf{leaves}(e,t)} p_{e,t,\ell} y_{e,t,\ell} \quad \text{(APDX.18a)}$$

$$\text{s.t.} \quad \sum_{e=(u',v') \in E : u' = u} z_{e,i} \leq s_{u,i}, \qquad \forall u \in U, \forall i \in [C], \quad \text{(APDX.18b)}$$

$$\sum_{e=(u',v')\in E:v'=v} z_{e,i} \geq d_{v,i}, \qquad \forall v \in V, i \in [C], \tag{APDX.18c}$$

$$\boldsymbol{z_e} = \sum_{\ell\in\mathbf{leaves}(e,t)} \sum_{k\in[2^C]} \lambda_{e,t,\ell,k}\bar{\boldsymbol{z}}_{e,t,\ell,k}, \qquad \forall e \in E, t \in [T_e], \tag{APDX.18d}$$

$$\sum_{\ell\in\mathbf{leaves}(e,t)} \sum_{k\in[2^C]} \lambda_{e,t,\ell,k} = 1, \qquad \forall e \in E, t \in [T_e], \tag{APDX.18e}$$

$$\sum_{\ell\in\mathbf{leaves}(e,t):\ell\%2^k<2^{k-1}} \sum_{k\in[2^C]} \lambda_{e,t,\ell,k} \leq y_{e,t,\ell}, \qquad \forall e \in E, t \in [T_e], k \in [\lceil \log_2 |\mathbf{leaves}(e,t)| \rceil], \tag{APDX.18f}$$

$$\sum_{\ell\in\mathbf{leaves}(e,t):\ell\%2^k\geq 2^{k-1}} \sum_{k\in[2^C]} \lambda_{e,t,\ell,k} \leq 1 - y_{e,t,\ell}, \qquad \forall e \in E, t \in [T_e], k \in [\lceil \log_2 |\mathbf{leaves}(e,t)| \rceil], \tag{APDX.18g}$$

$$\boldsymbol{y_{e,t}} \in \{0,1\}^{\lceil \log_2 |\mathbf{leaves}(e,t)| \rceil}, \qquad \forall e \in E, \forall t \in [T_e], \tag{APDX.18h}$$

$$z_{e,i} \in \mathbb{R}, \qquad \forall e \in E, \forall i \in [C], \tag{APDX.18i}$$

$$\lambda_{e,t,\ell,k} \in \mathbb{R}_+, \qquad \forall e \in E, t \in [T_e], \ell \in \mathbf{leaves}(e,t), k \in [2^C], \tag{APDX.18j}$$

where, for positive integers a and b, we use the notation $a\%b$ to denote the remainder of the division of a by b.

APDX.3.3. Unique Multilinear Function Over a Hyper-Rectangle

LEMMA APDX.6. *Let $\{\bar{\boldsymbol{x}}^{\mathbf{1}}, \ldots, \bar{\boldsymbol{x}}^{\boldsymbol{m}}\}$ be the set of vertices of a full-dimensional (axis-parallel) hyper-rectangle $\mathcal{H} \subset \mathbb{R}^n$, i.e., $m = 2^n$. Given $\bar{\boldsymbol{y}} \in \mathbb{R}^m$, there exists a unique vector $\boldsymbol{c} \in \mathbb{R}^m$ such that the function $g(x_1, \ldots, x_n) = \sum_{S\subseteq[n]} c_S \prod_{i\in S} x_i$ satisfies $g(\bar{\boldsymbol{x}}^i) = \bar{y}_i$ for all $i \in [m]$.*

Proof. We prove the statement by induction on n. In the proof, we only consider unit hypercubes $[0,1]^n$ in $\mathbb{R}^n$ because the result easily generalizes to other (axis-parallel) full-dimensional hyper-rectangles. When $n = 1$, the result is clear. In this case, $\mathcal{H} = [0,1]$ with $\bar{x}^1 = 0$ and $\bar{x}^2 = 1$. Given $\bar{y}_1, \bar{y}_2 \in \mathbb{R}$, there exists a unique linear (multilinear reduces to linear when $n = 1$) function g that is such that $g(\bar{x}^1) = \bar{y}_1$ and $g(\bar{x}^2) = \bar{y}_2$, *i.e.*, $c_\emptyset = \bar{y}_1$ and $c_{\{1\}} = \bar{y}_2 - \bar{y}_1$.

Assume now that the statement holds for all hypercubes $\mathcal{H} \subseteq \mathbb{R}^n$ and for all vectors $\boldsymbol{y} \in \mathbb{R}^{2^n}$ when $n \leq k$ for some positive integer k. We will show that the statement holds for $n = k+1$. Let $\{\bar{\boldsymbol{x}}^1, \ldots, \bar{\boldsymbol{x}}^m\}$ be the set of vertices of $\mathcal{H} \subset \mathbb{R}^{k+1}$ where $m = 2^{k+1}$ and let $\bar{\boldsymbol{y}} \in \mathbb{R}^m$. We partition $[m]$ into $I_j = \{i \in [m] \mid \bar{x}^i_{k+1} = j\}$ for $j \in \{0,1\}$. For $j \in \{0,1\}$, there exists, by the inductive hypothesis, a unique vector $\boldsymbol{c}^j$ such that $g_j := \sum_{S \subseteq [k]} c^j_S \prod_{i \in S} x_i$ takes values $\{\bar{y}_i\}_{i \in I_j}$ at the extreme points $\{(\bar{x}^i_1, \ldots, \bar{x}^i_k)\}_{i \in I_j}$ of the hyper-rectangle these points define in $\mathbb{R}^k$. It is then simple to verify that the function $g(x_1, \ldots, x_{k+1}) = (1 - x_{k+1}) g_0(x_1, \ldots, x_k) + x_{k+1} g_1(x_1, \ldots, x_k)$ is of the form $\sum_{S \subseteq [k+1]} \tilde{c}_S \prod_{i \in S} x_i$ for some suitable values of $\tilde{c}_S$ and takes the desired values at the extreme points of $\mathcal{H}$. This shows that at least one vector $\boldsymbol{c}$ with the desired property exists.

We next show that the vector $\boldsymbol{c}$ is unique. Take any function $g(x_1, \ldots, x_{k+1}) = \sum_{S \subseteq [k+1]} c_S \prod_{i \in S} x_i$ satisfying $g(\bar{\boldsymbol{x}}^i) = \bar{y}_i$ for all $i \in [m]$. By factoring out variable x_{k+1}, it can be rewritten as

$$g(x_1, \ldots, x_{k+1}) = \sum_{S \subseteq [k]} c_S \prod_{i \in S} x_i + \sum_{S \subseteq [k]} c_{S \cup \{k+1\}} x_{k+1} \prod_{i \in S} x_i.$$

Plugging the values of 0 or 1 for x_{k+1} in this expression yields

$$g(x_1, \ldots, x_k, 0) = \sum_{S \subseteq [k]} c_S \prod_{i \in S} x_i,$$
$$g(x_1, \ldots, x_k, 1) = \sum_{S \subseteq [k]} (c_S + c_{S \cup \{k+1\}}) \prod_{i \in S} x_i.$$

For $j \in \{0,1\}$, $g(x_1, \ldots, x_k, j)$ is of the form $\sum_{S \subseteq [k]} \tilde{c}^j_S \prod_{i \in S} x_i$ for suitable $\tilde{c}^j_S$ and takes the values $\{\bar{y}_i\}_{i \in I_j}$ at the vertices $\{(\bar{x}^i_1, \ldots, \bar{x}^i_k)\}_{i \in I_j}$ of a full-dimensional hypercube in $\mathbb{R}^k$. By induction, the coefficient $\{c_S\}_{S \subseteq [k]}$ and $\{c_S + c_{S \cup \{k+1\}}\}_{S \subseteq [k]}$ are uniquely determined. This implies that coefficients $\{c_S\}_{S \subseteq [k+1]}$ are also uniquely determined. This completes the proof of the inductive step. $\square$