$\mathcal{I}^2\mathcal{SDS}$

The Institute for Integrating Statistics in Decision Sciences

Technical Report TR-2008-21

December 4, 2008

Selecting Optimal Alternatives and Risk Reduction Strategies in Decision Trees

Hanif D. Sherali
Grado Department of Industrial and Systems Engineering
Virginia Polytechnic Institute and State University, USA

Evrim Dalkiran
Grado Department of Industrial and Systems Engineering
Virginia Polytechnic Institute and State University, USA

Theodore S. Glickman
Department of Decision Sciences
The George Washington University, USA

Selecting Optimal Alternatives and Risk Reduction Strategies in Decision Trees

Hanif D. Sherali[1], **Evrim Dalkiran**[1], **Theodore S. Glickman**[2]

[1] *Grado Department of Industrial and Systems Engineering (0118),*
Virginia Polytechnic Institute and State University,
250 Durham Hall, Blacksburg, VA 24061, U.S.A.
email: hanifs@vt.edu dalkiran@vt.edu

[2] *Department of Decision Sciences,*
The George Washington University,
415 Funger Hall, Washington, DC 20052, U.S.A.
email: glickman@gwu.edu

In this paper, we conduct a quantitative analysis for a strategic risk management problem that involves allocating certain available failure-mitigating and consequence-alleviating resources to reduce the failure risk of system safety components and subsequent losses, respectively, together with selecting optimal strategic decision alternatives, in order to minimize the risk or expected loss in the event of a hazardous occurrence. Using a novel decision tree optimization approach to represent the cascading sequences of probabilistic events as controlled by key decisions and investment alternatives, the problem is modeled as a nonconvex mixed-integer 0-1 factorable program. We develop a specialized branch-and-bound algorithm in which lower bounds are computed via tight linear relaxations of the original problem that are constructed by utilizing a polyhedral outer-approximation mechanism in concert with two alternative linearization schemes having different levels of tightness and complexity. We also suggest three alternative branching schemes, each of which is proven to guarantee convergence to a global optimum for the underlying problem. Extensive computational results are presented to demonstrate the efficacy of the proposed algorithm.

Subject classifications: Programming: integer. Programming: integer: branch-and-bound.
Decision analysis: risk.
Area of Review: Optimization.

1 Introduction

This paper addresses the strategic reduction of risk in a system that is characterized by a *decision tree*. Such a tree contains two types of nodes that represent either *event-points* or *decision-points*. At an event-point, some safety feature or measure is invoked that might lead to one of several outcomes, each represented by an arc having a specified probability of occurrence. For example, in the particular context of *Bernoulli events*, an event-point i would trigger either a failure or a success outcome state with respective probabilities p_i and $1 - p_i$ corresponding to the particular safety feature applied at this stage. On the other hand, the arcs emanating from each decision-point node are deterministic, and represent a selection among different available alternatives. These choices might involve making certain strategic decisions or system design selections at the particular stage in the process, which then govern the subsequent sequence of events and decisions. The decision tree is rooted at a *root node* or *node zero* that represents some component failure in the system under study or an external hazardous occurrence, which triggers a cascading sequence of strategic decisions and event outcomes based on applied actions or safety features that are invoked to control the damage. Hence, each branch in the tree represents a specific sequence of decision choices and event outcomes starting from the root node, and culminating in a *final outcome* or *leaf node* of the tree, where the latter entails an associated *consequence* or *loss*.

Figure 1 displays a particular example of a decision tree pertaining to the rupture of a gas-line in an offshore oil and gas platform, which has been adapted from a simpler *event tree* representation (i.e., one having only event-point nodes) as described in Andrews and Dunnett (2000), and that involves Bernoulli events. Here, each event-point, represented by the nodes indexed $1, \ldots, 9$, corresponds to applying some safety measure such as closing an isolation valve to localize the damage or opening a blow-down (BD) valve to depressurize certain critical sections of the system, and can lead to one of two immediate scenarios or outcomes depending on the success or failure of this measure. Note that a particular safety measure can be activated at different event-points in the tree to counteract the hazard at that particular stage in the process. At any decision-point in the tree (nodes $10, \ldots, 13$ in Figure 1, depending on the situation at that particular stage as governed by the sequence of events and decisions leading up to it, one of several strategic alternative options can be selected, where each such option is represented by a binary variable that takes on a value of one if this option is selected and zero otherwise. Hence, at the decision node 10 for example, one of three choices can be made with respect to subsequently activating only the closure of valve B, or the closure of valve B and the opening of the blow-down (BD) valve, or neither of these, which are respectively represented by the binary variables ϕ_1, ϕ_2, and ϕ_3 with $\phi_1 + \phi_2 + \phi_3 = 1$. The binary variables designating the alternative choices at the other decision nodes 11, 12, and 13 are specified in the legend of Figure 1 along with their constraining relationships. For example, assume that the first option is chosen at decision node 10. At the event-point represented by node 6, the valve B either fails or succeeds to operate with respective probabilities p_6 and $1 - p_6$. Following the success outcome, the leaf node 14 is reached, while the failure event leads to another decision node at which one of two emergency response choices can be made, which are represented by the binary variables ϕ_4 and ϕ_5, respectively, with $\phi_4 + \phi_5 = \phi_1$. Note that when $\phi_1 = 0$, the choice represented at decision node 11 does not arise. In this fashion, the cascading sequences of events and decisions lead to final leaf nodes 14, ..., 28, each entailing a particular loss or consequence.

Jiang et al. (2006) study maintenance selection and scheduling under tight budgetary and labor constraints to maximize the risk reduction. A composite heuristic using linear programming relaxations and dynamic programming is developed to solve the large-scale integer programming problem. Based on this, a Lagrangian relaxation approach is adopted to formulate a particular knapsack problem, and a sequence of such knapsack problems are then solved using dynamic programming in concert with a heuristic. In another study, Dillon et al. (2003) develop the Advanced Programmatic Risk Analysis and Management model as a decision tool for allocating a limited budget amongst design and residual investments that involve reinforcement (to mitigate technical failure risks) and initial budget reserves (to mitigate managerial failure risk). For all levels of residual investments, the corresponding allocations of the budget between subcomponents are listed and the alternative that minimizes the expected risk is chosen. Using sensitivity analysis, the value of additional investments is assessed, which provides insights into the amount of additional budget levels required to satisfy certain specific risk thresholds. Mehr and Tumer (2006) study resource allocation in the form of a portfolio selection problem in which risk is related to both the likelihood and consequence of an undesirable event, and each unit of resource allocation reduces the risk by a random amount. Sherali et al. (2008) also study resource allocations to minimize the expected risk in a given system. In contrast to Mehr and Tumer (2006), they differentiate between investment decisions for failure-mitigation and consequence-alleviation. A logit model is used to represent the relationships between investments and failure probabilities instead of adopting a linearity assumption as in Mehr and Tumer (2006). Furthermore, the probability of a final consequence is given via an event tree as a polynomial function of the probabilities of cascading events as opposed to using an assigned probability. The overall event tree optimization problem is modeled as a continuous nonconvex factorable program, and an equivalent transformed reformulation of the problem is solved using the commercial global optimization software BARON (Sahinidis (1996)).

Several other risk management applications where decision trees of the type described in Figure 1 arise are discussed in the literature. Beim and Hobbs (1997) utilize event trees to estimate the lock closure risks due to vessel accidents and nonstructural failures, while Acosta and Siu (1993) analyze steam generator tube rupture accidents in a power plant. Alternatively, Ulerich and Powers (1988), Andrews and Bartlett (2003), and Hayes (2002) adopt fault trees for modeling chemical processes, firewater deluge systems, and biological invasions, respectively. Dugan et al. (2000) develop a fault tree modeling and analysis tool, which decomposes the tree into static and dynamic subtrees that are solved using binary decision diagrams and Markov models, respectively. Another such computer-automated fault tree analysis tool that is applied in chemical process industries has been designed by

Khan and Abbasi (2000). For fault tree management problems, Rauzy (1993) proposes new algorithms based on binary decision diagrams that enable an efficient computation of minimal cuts along with deducing the probability of the root event. Dutuit and Rauzy (1996) describe a linear-time algorithm to detect independent subtrees in a fault tree by which computational costs can be reduced by means of a divide-and-conquer technique. Using the binary decision diagram approach of Rauzy (1993) and Dutuit and Rauzy (1996), Sinnamon and Andrews (1997a, 1997b) calculate the exact values of the top event parameters efficiently in contrast with the approximations produced by using traditional kinetic tree theory approaches. In addition, Furuta and Shiraishi (1984), Kenarangui (1991), and Huang et al. (2001) suggest using the fuzzy-set approach in order to enable the use of verbal statements for possibility measures instead of requiring the specification of event probabilities.

The remainder of this paper is organized as follows. In Section 2, we formulate a mathematical model for the DTO problem and then develop a suitable reformulation along with a tight lower-bounding representation through some transformations, polyhedral approximations, and valid inequalities in concert with two alternative linearization schemes. In Section 3, we describe a specialized branch-and-bound procedure to solve the DTO problem and establish its convergence. Computational results for a hypothetical case study and a set of simulated test cases are presented in Section 4. Finally, Section 5 concludes the paper with a summary and some recommendations for future extensions.

2 Model Formulation: Analysis and Enhancements

To model the proposed decision tree optimization problem, we begin by defining the *risk* associated with each possible final outcome or leaf node of the tree. Toward this end, consider the following notation:

- $I_e \equiv \{1, \ldots, I\}$: set of indices representing the event-point nodes.
- p_i and $(1 - p_i)$: respectively, the conditional probabilities of failure and success associ-

ated with the outcomes resulting from the application of the particular safety feature at event-point i, given that the sequence of decisions and events on the chain from the root node to node i has occurred, $i \in I_e$.

Remark 1. Note that for the sake of notational simplicity and clarity in exposition, we consider Bernoulli events in our model, although our analysis readily generalizes to multi-state events. For example, we could consider multiple levels of failure indexed by $r \in F_i$, having respective probabilities $p_{ir}, r \in F_i, \forall i \in I_e$, with the success probability given by $p_i^s \equiv (1 - \sum_{r \in F_i} p_{ir}) \in (0, 1)$. Some comments on accommodating this feature in the model are provided in our discussion. $\square$

- $\tau = \{j$: node j is a final outcome or leaf node index of the tree$\}$.
- $d = 1, \dots, D$: indices for the collective set of decisions made over the entire tree.
- $\phi_d = \begin{cases} 1 & \text{if decision } d \text{ is selected} \\ 0 & \text{otherwise.} \end{cases}$
- K: index set of decision-point nodes in the tree.
- $J_k = \{$decisions d (with corresponding binary variables ϕ_d) associated with the alternative option-based arcs that emanate from node $k\}$, $\forall k \in K$.
- $B_k = \{$ordered set of decisions d (with corresponding binary variables ϕ_d) that occur on the (unique) path from the root node 0 to node $k\}$, $\forall k \in K \cup I_e \cup \tau$.

For each $j \in \tau$:

- $C_j = \{i$: node i lies on the (unique) path from the root node 0 to node j, excluding nodes 0 and $j\}$.
- A_{ij}= arc emanating from node $i \in C_j$ that lies on the path joining the root node 0 to node $j \in \tau$.

- $S_{1j} = \{i \in C_j \cap I_e$: the (conditional) probability associated with arc A_{ij} is $p_i\}$, $\forall j \in \tau$.
- $S_{2j} = \{i \in C_j \cap I_e$: the (conditional) probability associated with arc A_{ij} is $(1 - p_i)\}$, $\forall j \in \tau$.
- $S_{3j} = \{d$: decision d is associated with arc $A_{ij}, \forall i \in C_j \cap K\}$, $\forall j \in \tau$.
- l_j= loss or consequence (in dollars, say) associated with leaf node $j \in \tau$.
- c_d= fixed cost associated with selecting decision d, $\forall d = 1, \dots, D$. (Note that for the sake of simplicity, we consider here only linear cost terms; however, the methodology described below readily extends to the case where we might have polynomial cost terms that are predicated on products of the ϕ-variables that occur along (partial) paths from node 0 to any leaf node.)

Then the *risk* associated with the leaf node $j \in \tau$ is the *expected consequence/loss* that is conditioned on the binary decisions as given by:

$$\psi_j \equiv l_j \prod_{i \in S_{1j}} p_i \prod_{i \in S_{2j}} (1 - p_i) \prod_{d \in S_{3j}} \phi_d, \ \forall j \in \tau, \tag{1}$$

and the *overall risk* is the total expected consequence given by $\sum_{j \in \tau} \psi_j$.

Thus far, the only decisions identified with respect to the decision tree analysis involve selecting binary values for the variables ϕ_d, $d = 1, \dots, D$, subject to the restrictions that

$$\sum_{d \in J_k} \phi_d = \prod_{d \in B_k} \phi_d, \ \forall k \in K, \tag{2}$$

where $\prod_{d \in B_k} \phi_d \equiv 1$ whenever $B_k = \varnothing$, $\forall k \in K$. Now, as in Sherali et al. (2008), suppose that we can additionally bring to bear certain preventive or protective *event-related resources* such as investments in improved technologies or supporting equipment for the various safety features involved so as to mitigate the conditional failure probabilities, $p_i, i \in I_e$, given the sequence of preceding actions and events. Likewise, suppose that we can invest in certain available *consequence-related resources* such as clean-up mechanisms and trained emergency response personnel in order to ameliorate the potential consequences or losses l_j associated with the leaf nodes $j \in \tau$. Accordingly, the goal would be to determine how to effectively deploy the available limited resources under some budgetary restrictions so as to appropriately manipulate the failure probabilities and the resultant consequences, as well as devise a plan for making suitable strategic decision option choices, in order to minimize the overall risk. More specifically, consider the following related notation and concepts:

- $m = 1, \ldots, M$: index set for the available event-related resources.
- s_m = total available units of the event-related resource m, $\forall m = 1, \ldots, M$.
- q_{im}= *decision variable* representing the quantity of event-related resource m that is allocated to reduce the associated (conditional) failure probability p_i of the safety feature deployed at event-point or node i, $\forall i \in I_e,\ m = 1, \ldots, M$.
- c_{im}= cost (in dollars) per unit of q_{im}.
- $n = 1, \ldots, N$: index set for the available consequence-related resources.
- t_n= total available units of the consequence-related resource n, $\forall n = 1, \ldots, N$.
- r_{jn}= *decision variable* representing the quantity of consequence-related resource n that is allocated to reduce the loss magnitude l_j associated with the leaf node j, $\forall j \in \tau,\ n = 1, \ldots, N$.
- d_{jn}= cost (in dollars) per unit of r_{jn}.
- β = total available budget (in dollars).
- *Logit model for relating* p_i *to* $\{q_{im},\ m = 1, \ldots, M\}$, $\forall i \in I_e$:

$$ln\left[\frac{p_i}{1-p_i}\right] = a_{i0} - \sum_{m=1}^{M} a_{im} q_{im},\ \forall i \in I_e, \tag{3}$$

where $(a_{i0}, ..., a_{iM}) \geq 0$.

- $[p_i^l, p_i^u]$: lower and upper bounds on p_i, where $0 < p_i^l \leq p_i \leq p_i^u < 1$, $\forall i \in I_e$, and where these bounds are either imposed or are implied by (3) and the available resources; in particular, we assume that $p_i^u = p_i^{u(3)} \equiv e^{a_{i0}}/(1+e^{a_{i0}}), \forall i \in I_e$, as implied by (3), i.e., the model seeks further possible reductions in p_i below p_i^u by using suitable event-related resource allocations as necessary.

- *Model for relating* l_j *to* $\{r_{jn}, n = 1, ..., N\}$, $\forall j \in \tau$:

$$l_j = b_{j0} - \sum_{n=1}^{N} b_{jn} r_{jn}, \ \forall j \in \tau \tag{4}$$

where $(b_{j0}, ..., b_{jN}) \geq 0$.

Remark 2. As an alternative to (4), we could consider a diminishing marginal return loss function of the type $l_j = b_{j0}\, exp\{-\sum_{n=1}^{N} b_{jn} r_{jn}\}$, $\forall j \in \tau$, having nonnegative parameter values. Our proposed algorithm can be identically applied for such loss functions upon taking logarithms. Likewise, for the general multi-failure state scenario discussed in Remark 1, the corresponding logit model would take the form $ln[p_{ir}/p_i^s] = a_{ir0} - \sum_{m=1}^{M} a_{irm} q_{im}, \forall r \in F_i, i \in I_e$, with the analysis described below following identically. □

- $[l_j^l, l_j^u]$: lower and upper bounds on l_j, where $0 < l_j^l \leq l_j \leq l_j^u < \infty$, $\forall j \in \tau$, and where these bounds are either imposed or are implied by (4) and the available resources; in particular, we assume that $l_j^u = l_j^{u(4)} \equiv b_{j0}, \forall j \in \tau$, as implied by (4), i.e., the model seeks further possible reductions in l_j below l_j^u by using suitable consequence-related resource allocations as necessary.

The *decision tree optimization* problem, **DTO**, can then be formulated as the following 0-1 mixed-integer nonlinear programming problem:

$$\textbf{DTO:} \quad \text{Minimize} \quad \sum_{d=1}^{D} c_d \phi_d + \sum_{j \in \tau} l_j \prod_{i \in S_{1j}} p_i \prod_{i \in S_{2j}} (1 - p_i) \prod_{d \in S_{3j}} \phi_d \tag{5a}$$

subject to

$$\sum_{i \in I_e} q_{im} \le s_m, \ \forall m = 1, \dots, M \tag{5b}$$

$$\sum_{j \in \tau} r_{jn} \le t_n, \ \forall n = 1, \dots, N \tag{5c}$$

$$\sum_{i \in I_e} \sum_{m=1}^{M} c_{im} q_{im} + \sum_{j \in \tau} \sum_{n=1}^{N} d_{jn} r_{jn} \le \beta, \tag{5d}$$

$$ln \left[\frac{p_i}{1 - p_i} \right] = a_{i0} - \sum_{m=1}^{M} a_{im} q_{im}, \ \forall i \in I_e \tag{5e}$$

$$l_j = b_{0j} - \sum_{n=1}^{N} b_{jn} r_{jn}, \ \forall j \in \tau \tag{5f}$$

$$\sum_{d \in J_k} \phi_d = \prod_{d \in B_k} \phi_d, \ \forall k \in K \tag{5g}$$

$$(p, l) \in \Omega \equiv \left\{ (p, l) : \begin{array}{c} 0 < p_i^l \le p_i \le p_i^u < 1, \ \forall i \in I_e \\ 0 < l_j^l \le l_j \le l_j^u < \infty, \ \forall j \in \tau \end{array} \right\}, \tag{5h}$$

$$(q, r) \in \mathscr{P} \tag{5i}$$

$$\phi_d \text{ binary}, \ \forall d = 1, \dots, D. \tag{5j}$$

In this formulation, the objective function (5a) seeks to minimize the total cost of implementing decisions plus the consequent expected loss or overall risk. (Note that the first term in the objective function is equivalent to the polynomial term $\sum_{k \in K} \sum_{d \in J_k} c_d \phi_d [\prod_{d' \in B_k} \phi_{d'}]$ because of Constraints (5g) and (5j)). Constraints (5b) and (5c) impose the resource availability restrictions; Constraint (5d) enforces the budgetary limitation; Constraints (5e), (5f), and (5g) follow from (3), (4), and (2), respectively; (5h) requires the (p, l)-variables to satisfy the specified bounding restrictions; and $\mathscr{P}$ in (5i) is a polyhedron (embedded in the nonnegative orthant) that includes any additional suitable constraints and variables in order to further restrict the (q, r)-variables, or to relate them to other influencing technological or operational decisions that are not explicitly stated in the above model. In the simplest case, $\mathscr{P} \equiv \{(q, r) : (q, r) \ge 0\}$. Observe that the nonconvexity in Problem DTO arises due to the polynomial function in (5a) and (5g), the logarithmic (factorable) term in (5e), as well as the binary decision variables in (5j).

Remark 3. There is another related model that can be formulated to study a sensitivity analysis issue in the context of decision trees, which is also of interest and can be handled by the algorithmic process discussed below. Consider a decision tree in which the event probabilities and loss values are not known with certainty, but might vary within intervals as designated in (5h) (while not being strategically controllable). Given such variabilities in the probabilities and the consequences, we might be interested in ascertaining the least expected consequence or risk value, along with the maximum possible deviation from this value that could occur due to the inherent uncertainties in the problem. Hence, we could first minimize (5a) subject to (5g), (5h), and (5j), and then fixing the resulting ϕ-variable values, we could next determine the maximum of (5a) subject to (5h) in order to resolve these two respective issues. Furthermore, as an alternative approach for making decisions that hedge against this uncertainty, we could formulate and solve a corresponding robust optimization problem in the spirit of Mulvey et al. (1995), or minimize the maximum risk by solving

$$\min_{\phi:(5\text{g}),(5\text{j})} \max_{(p,l)\in\Omega} \{(5\text{a})\}. \quad \square$$

We next define the following auxiliary variables, along with their implied bounds, in order to conveniently reformulate Problem DTO. To begin with, we transform the objective function by denoting

$$\theta_j \equiv l_j \prod_{i\in S_{1j}} p_i \prod_{i\in S_{2j}} (1-p_i), \ \forall j \in \tau. \tag{6a}$$

Thus we have,

$$\theta_j^l \leq \theta_j \leq \theta_j^u,$$

where,

$$\left\{ \begin{array}{l} \theta_j^l = l_j^l \prod_{i \in S_{1j}} p_i^l \prod_{i \in S_{2j}} (1 - p_i^u) \\ \theta_j^u = l_j^u \prod_{i \in S_{1j}} p_i^u \prod_{i \in S_{2j}} (1 - p_i^l) \end{array} \right\}, \ \forall j \in \tau. \tag{6b}$$

Furthermore, noting (1), (5a), and (6a), we denote

$$\psi_j \equiv \theta_j w_{S_{3j}}, \forall j \in \tau, \tag{6c}$$

where

$$w_{S_{3j}} \equiv \prod_{d \in S_{3j}} \phi_d, \forall j \in \tau. \tag{6d}$$

Noting that the binariness of the ϕ_d-variables implies that of the $w_{S_{3j}}$-variables, $\forall j \in \tau$, we can linearize (6c) by replacing it with the restrictions:

$$\theta_j^l w_{S_{3j}} \leq \psi_j \leq \theta_j^u w_{S_{3j}} \text{ and } \theta_j^l (1 - w_{S_{3j}}) \leq \theta_j - \psi_j \leq \theta_j^u (1 - w_{S_{3j}}), \tag{6e}$$

by which it is readily verified that (6c) holds true whenever $w_{S_{3j}}$ takes on binary values, $\forall j \in \tau$. In fact, because of the positive (unit) objective coefficients on ψ_j, this relationship will hold true even if we retain just the inequalities $\psi_j \geq \theta_j^l w_{S_{3j}}$ and $\psi_j \geq \theta_j - \theta_j^u (1 - w_{S_{3j}})$ from (6e) for each $j \in \tau$.

In the same spirit as $w_{S_{3j}}, j \in \tau$, we define a new variable w_J to represent $\prod_{d \in J} \phi_d$ for some specific sets $J \subset \{1, \ldots, D\}$ as described below, which will be used in the sequel for linearizing (5g). Here, whenever $|J| = 1$ with $J = \{d\}$, we will take the corresponding $w_J \equiv \phi_d$ itself, and whenever $J = \varnothing$, we will take $w_J \equiv 1$. Now, consider any set S_{3j} for $j \in \tau$ (in general, $w_{S_{3j}}$ might be replicated for different $j \in \tau$), and assume that the indices in S_{3j} are arranged according to the order in which the corresponding decisions d (with associated variables ϕ_d) occur along the path from the root node to node j. If $|S_{3j}| \geq 2$, this will generate w_J-variables via sets J defined by taking the first two indices from S_{3j}, the first three indices from S_{3j}, and so on, up to all the indices from S_{3j} (finally yielding $w_{S_{3j}}$ as denoted above). Letting $\mathscr{J}$ denote the resulting distinct sets J generated, we define the following identities:

$$w_J = \prod_{d \in J} \phi_d, \forall J \in \mathscr{J}. \tag{7}$$

Note that (7) subsumes (6d) since $S_{3j} \in \mathscr{J}$, $\forall j \in \tau$.

Next, to linearize (5e), we introduce the variables

$$y_{1i} = ln(p_i) \text{ and } y_{2i} = ln(1 - p_i),\ \forall i \in I_e. \tag{8a}$$

Note that,

$$y_{1i}^l \leq y_{1i} \leq y_{1i}^u \text{ and } y_{2i}^l \leq y_{2i} \leq y_{2i}^u,$$

$$\text{where } y_{1i}^l = ln(p_i^l),\ y_{1i}^u = ln(p_i^u),\ y_{2i}^l = ln(1 - p_i^u),\ \text{and } y_{2i}^u = ln(1 - p_i^l), \forall i \in I_e. \tag{8b}$$

Similarly, to linearize (6a) itself, we denote

$$z_j \equiv ln(\theta_j),\ \forall j \in \tau, \tag{9a}$$

where, based on (6b), we impose

$$z_j^l \leq z_j \leq z_j^u,\ \text{with } z_j^l = ln(\theta_j^l) \text{ and } z_j^u = ln(\theta_j^u),\ \forall j \in \tau. \tag{9b}$$

Likewise, to accommodate the term $ln(l_j)$ generated by taking logarithms in (6a), define

$$\xi_j \equiv ln(l_j), \forall j \in \tau, \tag{10a}$$

and impose the related bounds:

$$\xi_j^l \leq \xi_j \leq \xi_j^u,\ \text{where } \xi_j^l = ln(l_j^l) \text{ and } \xi_j^u = ln(l_j^u),\ \forall j \in \tau. \tag{10b}$$

Using the foregoing transformations and substitutions, we obtain the following equivalently reformulated problem DTO(Ω), which is predicated on the hyperrectangle Ω.

DTO(Ω): Minimize $\sum_{d=1}^{D} c_d \phi_d + \sum_{j \in \tau} \psi_j$ (11a)

subject to

$$\psi_j \geq \theta_j^l w_{S_{3j}} \text{ and } \psi_j \geq \theta_j - \theta_j^u (1 - w_{S_{3j}}),\ \forall j \in \tau \tag{11b}$$

$$w_J = \prod_{d \in J} \phi_d, \forall J \in \mathscr{J} \tag{11c}$$

$$\sum_{i \in I_e} q_{im} \leq s_m, \forall m = 1, \ldots, M \tag{11d}$$

$$\sum_{j \in \tau} r_{jn} \leq t_n, \forall n = 1, \ldots, N \tag{11e}$$

$$\sum_{i \in I_e} \sum_{m=1}^{M} c_{im} q_{im} + \sum_{j \in \tau} \sum_{n=1}^{N} d_{jn} r_{jn} \leq \beta, \tag{11f}$$

$$y_{1i} - y_{2i} = a_{i0} - \sum_{m=1}^{M} a_{im} q_{im}, \forall i \in I_e \tag{11g}$$

$$l_j = b_{0j} - \sum_{n=1}^{N} b_{jn} r_{jn}, \forall j \in \tau \tag{11h}$$

$$\sum_{d \in J_k} \phi_d = w_{B_k}, \forall k \in K \tag{11i}$$

$$z_j = \xi_j + \sum_{i \in S_{1j}} y_{1i} + \sum_{i \in S_{2j}} y_{2i}, \forall j \in \tau \tag{11j}$$

$$y_{1i} = ln(p_i), \forall i \in I_e \tag{11k}$$

$$y_{2i} = ln(1 - p_i), \forall i \in I_e \tag{11l}$$

$$z_j = ln(\theta_j), \forall j \in \tau \tag{11m}$$

$$\xi_j = ln(l_j), \forall j \in \tau \tag{11n}$$

$$(p, l) \in \Omega \tag{11o}$$

$$(q, r) \in \mathscr{P} \tag{11p}$$

$$\left\{ \begin{array}{l} \theta_j^l \leq \theta_j \leq \theta_j^u, z_j^l \leq z_j \leq z_j^u, \text{ and } \xi_j^l \leq \xi_j \leq \xi_j^u, \forall j \in \tau \\ y_{1i}^l \leq y_{1i} \leq y_{1i}^u \text{ and } y_{2i}^l \leq y_{2i} \leq y_{2i}^u,\ \forall i \in I_e \end{array} \right\} \tag{11q}$$

$$\phi_d \text{ binary}, \forall d = 1, \ldots, D, \text{ and } 0 \leq w_J \leq 1, \forall J \in \mathscr{J}. \tag{11r}$$

Here, Ω is as specified in (5h), and we note that the bounds in (11q) depend on Ω (even as Ω will be modified via a partitioning process in our proposed algorithmic approach below) and are given by (6b), (8b), (9b), and (10b), respectively. For the sake of convenience in discussion, we shall denote the set of variables in Problem DTO(Ω), with obvious vector notation, as:

$$x \equiv (p, l, q, r, \phi, w, \psi, \theta, y_1, y_2, z, \xi),$$

where, in particular, $y_1 \equiv (y_{1i}, \forall i \in I_e)$ and $y_2 \equiv (y_{2i}, \forall i \in I_e)$. Observe that DTO($\Omega$) is, in general, a mixed-integer 0-1 factorable program (see Sherali and Wang (2001) for continuous factorable programs). However, in our case, DTO(Ω) is linear except for the complicating identities (11c) and (11k) - (11n).

We next discuss some suitable polyhedral outer-approximation mechanisms to handle the univariate, monotone logarithmic functions in (11k) - (11n). Toward this end, generically denote any identity in (11k) - (11n) as:

$$f = ln(\gamma), \text{ where } 0 < \gamma^l \leq \gamma \leq \gamma^u < \infty. \tag{12}$$

We then replace each such constraint (12) by the following affine convex envelope:

$$f \geq ln(\gamma^l) + \frac{(\gamma - \gamma^l)}{(\gamma^u - \gamma^l)}[ln(\gamma^u) - ln(\gamma^l)], \tag{13a}$$

along with some $H \geq 2$ tangential supports:

$$f \leq ln(\gamma_h) + \frac{(\gamma - \gamma_h)}{\gamma_h}, \text{ for } h = 1, \ldots H, \text{ where } \gamma^l \equiv \gamma_1 < \gamma_2 < \ldots < \gamma_H \equiv \gamma^u. \tag{13b}$$

In order to prescribe some judicious alternatives for selecting the points of tangency $\{\gamma_2, \ldots, \gamma_{H-1}\}$ (other than the interval end-points) in (13b), consider the following result:

Proposition 1. Consider any $0 < \bar{\gamma} < \hat{\gamma}$ such that $ln(\hat{\gamma}) - ln(\bar{\gamma}) = \Delta$. Then, for $\gamma \in [\bar{\gamma}, \hat{\gamma}]$, the maximum error, E, between the function $ln(\gamma)$ and its piecewise linear approximation defined by the tangential supports at $\bar{\gamma}$ and $\hat{\gamma}$ depends on Δ alone and is given by

$$E = ln[e^{\Delta} - 1] - ln(\Delta) + \frac{\Delta}{e^{\Delta} - 1} - 1. \tag{14}$$

Proof. By the monotonicity of both the logarithmic and the affine tangential supporting functions, the stated maximum error, E, occurs at the point of intersection γ^* of the tangential supports as given by $\gamma^* = \bar{\gamma}\hat{\gamma}\Delta/(\hat{\gamma} - \bar{\gamma})$, with

$$E = \frac{\hat{\gamma} ln(\hat{\gamma}) - \bar{\gamma} ln(\bar{\gamma})}{(\hat{\gamma} - \bar{\gamma})} - 1 - ln(\gamma^*). \tag{15}$$

Substituting for γ^* in (15), and writing $ln(\hat{\gamma}) = ln(\bar{\gamma}) + \Delta$, we get

$$E = ln\left(\frac{\hat{\gamma}}{\bar{\gamma}} - 1\right) - ln(\Delta) + \frac{\Delta}{[(\hat{\gamma}/\bar{\gamma}) - 1]} - 1,$$

which, upon using $\hat{\gamma}/\bar{\gamma} = e^{\Delta}$, yields (14). $\square$

Corollary 1. Given any $\epsilon > 0$, let $\Delta = \Delta_\epsilon$ be the solution to (14) when we set $E = \epsilon$. Now, suppose that we approximate the function $f \equiv ln(\gamma)$ on $[\gamma^l, \gamma^u] \subseteq (0, \infty)$ by H tangential supports constructed at points uniformly distributed along the f-axis including the end-points, where

$$H = \left\lceil \frac{ln(\gamma^u) - ln(\gamma^l)}{\Delta_\epsilon} \right\rceil + 1. \tag{16}$$

Then, the maximum approximation error will be bounded above by ϵ.

Proof. By Proposition 1 and since $\partial E/\partial\Delta > 0$ for $\Delta > 0$ in (14), the maximum approximation error will be bounded above by ϵ provided

$$\frac{ln(\gamma^u) - ln(\gamma^l)}{H - 1} \leq \Delta_\epsilon,$$

which yields (16). $\square$

Accordingly, we define the *Bounded Error Strategy* **(BES)** for selecting points for generating the tangential supports (13b) as that prescribed by Corollary 1, given any error tolerance $\epsilon > 0$. For example, utilizing (14), the maximum approximation error will be bounded above by $\epsilon = 0.01$ (respectively, 0.001), if we select $\Delta_\epsilon = 0.28$ (respectively, 0.09). The number of tangential supports generated would then further depend on the bounding interval $[\gamma^l, \gamma^u]$ as given by (16). Alternatively, in order to control the number of tangential supports generated, we shall also apply BES with a prespecified value of H. In this case, we generate (13b) at some H uniformly distributed points along the f-axis, including the interval end-points. This yields

$$\gamma_h = exp\left\{ln(\gamma^l) + \left(\frac{h-1}{H-1}\right) ln\left[\frac{\gamma^u}{\gamma^l}\right]\right\}, \text{ for } h = 1, \ldots, H.$$

We shall refer to this strategy as **BES(H)**. We recommend the value H=4 based on our computational results reported in Section 4, where we investigated using H=4,..., 20.

Next, in order to generate a linear programming relaxation LP(Ω) for computing lower bounds in a branch-and-bound framework, we additionally adopt the following alternative linearization strategies for (11c) as identified in Section 2.1 below:

2.1 Linearization of (11c)

In this section, we shall discuss two alternative schemes for linearizing (11c) that differ in their size or complexity and the relative tightness of the resulting LP relaxation. These are respectively denoted as linearization methods LM1 and LM2, and, together with the polyhedral outer-approximation (13) applied to (11k)-(11n), produce corresponding LP relaxations **LP1(Ω)** and **LP2(Ω)**, respectively. We shall also use the terminology **LP(Ω)** to refer generically to either of the foregoing relaxations LP1(Ω) and LP2(Ω).

2.1.1 Linearization Method LM1:

This is a standard linearization technique that utilizes the following constraints for each $J \in \mathscr{J}$ such that $|J| \geq 2$ (noting (11r)):

$$w_J \leq \phi_d, \forall d \in J, \text{ and } w_J \geq \sum_{d \in J} \phi_d - |J| + 1. \tag{17}$$

Note that as portended by (11c), and using (11r), when $\phi_d = 0$ for any $d \in J$, then (17) implies that $w_J = 0$, whereas when $\phi_d = 1$, $\forall d \in J$, then (17) implies that $w_J = 1$ as well.

2.1.2 Linearization Method LM2:

In this method, consider any decision-point node $k \in K$. Recall that B_k denotes the ordered set of decision indices that occur on the path from node 0 to node k. If $B_k = \varnothing$, then no additional constraints are generated for this node k. Else, suppose that $|B_k| \geq 1$. Then, we generate the restrictions:

$$w_{B_k+d} = \phi_d, \forall d \in J_k \text{ (for each } k \ \in K : B_k \neq \varnothing), \tag{18}$$

where we denote $w_{B_k+d} \equiv w_{B_k \cup \{d\}}$, with the index d appearing last in the resulting set $B_k \cup \{d\}, \forall d \in J_k$.

Proposition 2. The constraints (18) are valid and together with (11i) and (11r), imply that (11c) holds true.

Proof. First of all, note that (18) is valid since if $\phi_d = 0$ for any $d \in J_k$, then we must have $w_{B_k+d} = 0$ by its interpretation, and if $\phi_d = 1$ for any $d \in J_k$, then by (11i) and (11r), we must have $w_{B_k} = 1$, so that again the interpretation of w_{B_k+d} implies that we must have $w_{B_k+d} = 1$.

Next, let us establish that (11c) holds true by induction on $|J|$. Consider any $k \in K$ such that $B_k = \{d'\}$ and $d \in J_k$ so that $|J| = 2$ with $J \equiv \{d', d\}$ (the case $|J| = 1$ is trivial by definition). Now, if $\phi_{d'} = 0$, then (11i) implies that $\phi_d = 0$, so that (18) yields $w_{d'd} = 0$. On the other hand, if $\phi_{d'} = 1$, then (11i) implies that $\phi_d = 0$ or 1, which respectively yields $w_{d'd} = 0$ or 1 via (18).

Inductively, to complete the proof, consider $J = \{d_1, \ldots, d_h\}$, where $h \geq 3$, and assume that (11c) holds true for any strict subset of this set. Hence, there exists $k \in K$ such that $B_k = \{d_1, \ldots, d_{h-1}\}$, with $d_h \in J_k$. Let us show that (11c) is satisfied for any such J. If $\phi_d = 0$ for any $d \in \{d_1, \ldots, d_{h-1}\}$, then by the induction hypothesis, we have that $w_{B_k} = 0$, which implies by (11i) that $\phi_d = 0$, $\forall d \in J_k$. Therefore, $\phi_{d_h} = 0$, which implies by (18) that $w_J = 0$. Else, if $\phi_{d_1} = \ldots = \phi_{d_{h-1}} = 1$, then $w_{B_k} = 1$ by the induction hypothesis, and so by (11i), $\phi_{d_h} = 0$ or 1, which respectively yields $w_J = 0$ or 1 via (18), and so (11c) is again satisfied in either case. □

The next result demonstrates that LM2 yields a tighter representation than LM1 in the continuous (LP) sense.

Proposition 3. The constraints represented in (18), (11i), and $\phi \geq 0$ of LM2 imply those in (17) of LM1.

Proof. Consider any $J' \in \mathscr{J}$ and assume without loss of generality that $J' = \{1, \ldots, h\}$, with $\phi_1, \ldots, \phi_h$ occurring in this order along the path from node 0 to some node j. Hence, by (17), LM1 generates the constraints

$$w_{J'} \leq \phi_d, \ \forall d = 1, \ldots, h, \tag{19a}$$

$$w_{J'} \geq \sum_{d=1}^{h} \phi_d - (h-1). \tag{19b}$$

Now, by (18) applied to the node from which the arc having ϕ_h emanates, LM2 directly produces $w_{J'} = \phi_h$ and (11i) yields $\phi_h \leq w_{J'-\{h\}}$, which, by (18) applied at the node from which the arc having the variable ϕ_{h-1} emanates, produces $w_{J'-\{h\}} = \phi_{h-1}$. Hence, $w_{J'} \leq \phi_{h-1}$ as well. Continuing in this fashion along the chain from node j to node 0 yields that (19a) is implied.

To complete the proof, we next show that (19b) is implied as well. Note that (18) yields for any $j_1 \in J_k,\ k \in K$, that

$$w_{B_k+j_1} = \phi_{j_1} \geq w_{B_k} + \phi_{j_1} - 1, \tag{20}$$

where the last inequality follows from the fact that $w_{B_k} \leq 1$ in (11r). Hence, applying (20) at the node from which the arc having the variable ϕ_h emanates with $B_k + j_1 \equiv J'$ and $j_1 \equiv h$ we get,

$$w_{J'} \geq w_{J'-\{h\}} + \phi_h - 1. \tag{21}$$

Repeating this at the node from which the arc having the variable ϕ_{h-1} emanates yields $w_{J'-\{h\}} \geq w_{J'-\{h,h-1\}} + \phi_{h-1} - 1$, which, together with (21), implies that

$$w_{J'} \geq w_{J'-\{h,h-1\}} + \phi_h + \phi_{h-1} - 2. \tag{22}$$

Continuing (22) along the chain from node j to node 0, we get

$$w_{J'} \geq w_{J'-\{h,h-1,\ldots,2\}} + \phi_h + \phi_{h-1} + \ldots + \phi_2 - (h-1) = \sum_{d=1}^{h} \phi_d - (h-1),$$

where the last equality follows by noting that $w_{J'-\{h,h-1,\ldots,2\}} = w_1 \equiv \phi_1$. Hence, (19b) is also implied. $\square$

Now, for any $j \in K \cup I_e \cup \tau$, let $d(j)$ be the first decision index that is encountered in the (reverse) path from node j to the root node (whenever $d(j)$ does not exist, we define $\phi_{d(j)} \equiv 1$). Then, we can simplify LM2 by eliminating the w_J-variables upon using (18), which effectively equates $w_{S_{3j}} = \phi_{d(j)}, \forall j \in \tau$ in (11b) and $w_{B_k} = \phi_{d(k)}, \forall k \in K$ in (11i). Hence, by Proposition 2, *LM2 can be equivalently written by using these substitutions in* (11b) *and* (11i)*, and eliminating* (11c) *and the w_J-inequalities in* (11r). The following result reveals a partial convex hull representation inherent within LM2.

Proposition 4. Consider the following polyhedral set defined by the constraints of LM2.

$$\Lambda \equiv \{\phi \geq 0 : \sum_{d \in J_k} \phi_d = \phi_{d(k)}, \forall k \in K\}. \tag{23}$$

Then, Λ is nonempty and compact with binary-valued extreme points.

Proof. The compactness of Λ follows readily by noting that the constraints in Λ imply that $0 \leq \phi_d \leq 1, \forall d = 1, \ldots, D$, recalling that $\phi_{d(k)} \equiv 1$ whenever $d(k)$ is null. Now, to complete the proof, let us show that for any arbitrary objective vector $(C_d, d = 1, \ldots, D)$, the linear program

$$\text{Minimize } \left\{ \sum_{d=1}^{D} C_d \phi_d : \phi \in \Lambda \right\} \tag{24}$$

has an optimal solution for which ϕ_d is binary-valued.

We solve (24) sequentially as follows. Consider any decision node $k \in \arg\max\{|B_k|\}$, and notice that the variables ϕ_d for $d \in J_k$ appear only in the single corresponding constraint for k in (23). Hence, there exists an optimal solution in which $\phi_{\hat{d}} = \phi_{d(k)}$, where $\hat{d} \in \underset{d \in J_k}{\arg\min}\{C_d\}$, and $\phi_d = 0,\ \forall d \in J_k - \{\hat{d}\}$. This eliminates the variables ϕ_d for $d \in J_k$ from (24) along with the associated constraint for this k in (23), where, whenever $B_k \neq \varnothing$ (so that $d(k)$ exists), we also update the objective coefficient $C_{d(k)}$ for the upstream (toward the root node) variable $\phi_{d(k)}$ according to $C_{d(k)} \leftarrow C_{d(k)} + C_{\hat{d}}$. Repeating this step, we will finally solve separable problems for decision nodes $k \in K$ having $B_k = \varnothing$, for which the reduced linear program is given as follows for some transformed objective coefficients $(\hat{C}_d, d \in J_k)$:

$$\text{Minimize} \left\{ \sum_{d \in J_k} \hat{C}_d \phi_d : \sum_{d \in J_k} \phi_d = 1, \phi_d \geq 0, \forall d \in J_k \right\},$$

and for which there exists an optimal binary solution for $\phi_d,\ d \in J_k$. By back-substituting these binary values for the recorded solutions for the downstream decision nodes, we will get a binary optimal solution for $\phi_d,\ d \in J_k, \forall k \in K$. $\square$

2.1.3 Illustrative Example

Consider the illustration of the decision tree in Figure 1 that contains the restrictions (5g):

$$\phi_1 + \phi_2 + \phi_3 = 1, \tag{25a}$$

$$\phi_4 + \phi_5 = \phi_1, \tag{25b}$$

$$\phi_6 + \phi_7 = \phi_3, \tag{25c}$$

$$\phi_8 + \phi_9 = \phi_3\phi_7, \tag{25d}$$

along with product terms of the type (7) for $J \in \mathscr{J}$ as given by:

$$\phi_1\phi_4,\ \phi_1\phi_5,\ \phi_3\phi_6,\ \phi_3\phi_7,\ \phi_3\phi_7\phi_8,\ \text{and } \phi_3\phi_7\phi_9. \tag{26}$$

For this situation, the methods LM1 and LM2 will produce the following additional constraints and variables to replace (11c) in Problem DTO:

LM1 (Equation (17)):

For $d = 4$: $\{w_{14} \leq \phi_1,\ w_{14} \leq \phi_4,\ w_{14} \geq \phi_1 + \phi_4 - 1\}$

$d = 5$: $\{w_{15} \leq \phi_1,\ w_{15} \leq \phi_5,\ w_{15} \geq \phi_1 + \phi_5 - 1\}$

$d = 6$: $\{w_{36} \leq \phi_3,\ w_{36} \leq \phi_6,\ w_{36} \geq \phi_3 + \phi_6 - 1\}$

$d = 7$: $\{w_{37} \leq \phi_3,\ w_{37} \leq \phi_7,\ w_{37} \geq \phi_3 + \phi_7 - 1\}$

$d = 8$: $\{w_{378} \leq \phi_3,\ w_{378} \leq \phi_7,\ w_{378} \leq \phi_8, w_{378} \geq \phi_3 + \phi_7 + \phi_8 - 2\}$

$d = 9$: $\{w_{379} \leq \phi_3,\ w_{379} \leq \phi_7,\ w_{379} \leq \phi_9,\ w_{379} \geq \phi_3 + \phi_7 + \phi_9 - 2\}$.

LM2 (Equation (18), where these identities are substituted into (11b) and (11i)**):**

For $k = 11$: $\{w_{14} = \phi_4,\ w_{15} = \phi_5\}$

$k = 12$: $\{w_{36} = \phi_6,\ w_{37} = \phi_7\}$

$k = 13$: $\{w_{378} = \phi_8,\ w_{379} = \phi_9\}$.

Remark 4. Observe that the size of the decision tree can be reduced whenever there exist consecutive decision-point nodes. If a decision-point node k_1, having the variable ϕ_{d_1} associated with an emanating arc is immediately followed by another decision-point node k_2 having $J_{k_2} \equiv \{d_2, d_3, \ldots, d_n\}$, we can collapse node k_2 into k_1 and define new variables $\phi_{d_1d_2}, \phi_{d_1d_3}, \ldots, \phi_{d_1d_n}$ associated with corresponding arcs emanating from k_1 to replace the variables $\phi_{d_1}, \phi_{d_2}, \ldots, \phi_{d_n}$. Compared to the original decision tree, we reduce the number of decision-point nodes and the number of decision alternatives by one for each such step. To illustrate this for the decision tree of Figure 1, note that we can collapse both nodes 12 and 13 (sequentially adopting the foregoing step) into node 10, and generate path-based arcs connecting node 10 to nodes 21, 22, and 23, having respective associated variables ϕ_{36}, ϕ_{378}, and ϕ_{379}. The constraints (5g) in this case would then be written as follows:

$$\phi_1 + \phi_2 + \phi_{36} + \phi_{378} + \phi_{379} = 1, \tag{27a}$$

$$\phi_4 + \phi_5 = \phi_1, \tag{27b}$$

and the product terms of the type (7) for $J \in \mathscr{J}$ would be given by

$$\phi_1\phi_4 \text{ and } \phi_1\phi_5. \ \square \tag{28}$$

2.2 Structure of optimal solutions

Let $\bar{x} = (\bar{p}, \bar{l}, \bar{q}, \bar{r}, \bar{\phi}, \bar{w}, \bar{\psi}, \bar{\theta}, \bar{y_1}, \bar{y_2}, \bar{z}, \bar{\xi})$ represent an optimal solution to LP(Ω). To reduce the size of this relaxation, we *a priori* identify constraints that would be inactive at optimality by analyzing the structural behavior of optimal solutions. In this spirit, Proposition 5 below establishes that the affine convex envelope for the constructed outer-approximation of the functional form $z_j = ln(\theta_j), j \in \tau$ can be omitted without affecting optimality.

Proposition 5. In Problem LP(Ω), the affine convex envelope of $ln(\theta_j)$, for any $j \in \tau$, will not be active at optimality unless if the optimal θ_j-value is at its lower or upper bound.

Moreover, at least one of the tangential supports will be active.

Proof. Given a feasible solution x', let the variables $(p, l, q, r, \phi, w, y_1, y_2, \xi, z)$ be fixed in LP(Ω) according to x'. The resulting linear program in ψ and θ effectively bounds θ_j as $\underline{\theta_j} \leq \theta_j \leq \overline{\theta_j}, \forall j \in \tau$, where $\underline{\theta_j}$ and $\overline{\theta_j}$ are respectively determined by some tangential support (13b) and the affine convex envelope (13a), corresponding to the left-hand side in (13a, 13b) fixed at z'_j. By the nature of the objective function (11a) and the constraints (11b), we would therefore have $\theta_j = \underline{\theta_j}$, $\forall j \in \tau$, at an optimal solution. Hence, for each $j \in \tau$, some tangential support is active at optimality and the affine convex envelope is inactive at optimality unless z'_j, and therefore θ_j, equals its original lower or upper bound. $\square$

Remark 5. By the nature of the objective function (11a) and the constraint relationships (11b), (11g), (11h), and (11j) in LP(Ω), the y_1-, y_2-, and ξ-variables tend to be at their lower/upper bounds at optimality. Although removing the tangential supports associated with the corresponding functional forms (11k), (11l), and (11n) other than those at the interval end-points might worsen the lower bound obtained via the relaxed problem, the total computational time may improve as a result of the decrease in the size of LP(Ω) that is solved at each node of the branch-and-bound tree. Hence, in our computations, we shall experiment with this reduced modeling strategy. $\square$

2.3 Further Properties of LP(Ω)

The next set of results lay the groundwork for composing our proposed global optimization strategy for solving Problem DTO. Henceforth, for any Problem P, we shall denote its optimal value as $v[P]$.

Proposition 6. v[LP(Ω)] gives a lower bound for v[DTO(Ω)]. Moreover, if $\bar{x}$ solves LP(Ω) and satisfies (11k) - (11n) and (11r), then $\bar{x}$ also solves DTO(Ω) with the same objective value.

Proof. Follows from the construction of LP(Ω), noting that under the hypothesis of the proposition and the validity of LMr, r=1,2, we also then have that (11c) holds true. □

Proposition 7. Let $\bar{x}$ solve LP(Ω). Let $(\hat{l}, \hat{q}, \hat{r}) \equiv (\bar{l}, \bar{q}, \bar{r})$ and let $\hat{p}$ be computed by (5e), i.e., $\hat{p}_i = \bar{g}_i/(1+\bar{g}_i)$ where $\bar{g}_i = exp\{a_{i0} - \sum_{m=1}^{M} a_{im}\bar{q}_{im}\}$, $\forall i \in I_e$. Furthermore, set $\hat{\theta}_j \equiv \hat{l}_j \prod_{i\in S_{1j}} \hat{p}_i \prod_{i \in S_{2j}} (1-\hat{p}_i), \forall j \in \tau$, and let $\hat{\phi}$ be a binary optimal solution to the linear program composed in Proposition 4 as given by:

$$\hat{LP}: \text{ Minimize } \left\{ \sum_{d=1}^{D} c_d \phi_d + \sum_{j \in \tau} \hat{\theta}_j \phi_{d(j)} : \phi \in \Lambda \right\}. \tag{29}$$

Then, $(\hat{p}, \hat{l}, \hat{q}, \hat{r}, \hat{\phi})$ is a feasible solution to Problem DTO with objective value $v[\hat{LP}]$.

Proof. From (11q) (as in (8b)), we have $ln(p_i^l) \le \bar{y}_{1i} \le ln(p_i^u)$ and $ln(1-p_i^u) \le \bar{y}_{2i} \le ln(1-p_i^l)$. Thus, from (11g), $ln(p_i^l) - ln(1-p_i^l) \le \bar{y}_{1i} - \bar{y}_{2i} = a_{i0} - \sum\limits_{m=1}^{M} a_{im}\bar{q}_{im} \le ln(p_i^u) - ln(1-p_i^u)$. Hence, since $\hat{p}$ has been computed by using the constraint (5e), we have

$$ln\Big(\frac{p_i^l}{1-p_i^l}\Big) \le ln\Big(\frac{\hat{p}_i}{1-\hat{p}_i}\Big) = a_{i0} - \sum_{m=1}^{M} a_{im}\bar{q}_{im} \le ln\Big(\frac{p_i^u}{1-p_i^u}\Big),$$

which results in $p_i^l \le \hat{p}_i \le p_i^u$. Thus, from the constraints of LP(Ω), we have that $(\hat{p}, \hat{l}, \hat{q}, \hat{r})$ is feasible to (5b)-(5f), (5h), and (5i). Furthermore, by the structure of LM2 and Propositions 2 and 4, $\hat{LP}$ defined by (29) then yields a binary optimal solution $\hat{\phi}$ that represents the best optimal completion $(\hat{p}, \hat{l}, \hat{q}, \hat{r}, \hat{\phi})$ to the foregoing partial solution. □

Proposition 8. Let $\bar{x}$ solve LP(Ω) with objective value v[LP(Ω)]. If each of the variable values $\bar{p}_i$, $\bar{l}_j$, and $\bar{\theta}_j$ equals either its corresponding lower or upper bound and $\bar{\phi}$ is binary-valued, then $\bar{x}$ solves DTO(Ω) with objective value v[DTO(Ω)] $= v$[LP(Ω)].

Proof. From the construction of LP(Ω), it is sufficient to show that $\bar{x}$ satisfies (11c) and (11k) - (11n). For the generic case (12), if γ equals either of its bounds, then Constraints

(13a) and (13b) imply that $f = ln(\gamma)$. Likewise, if each of the variable values $\bar{p}_i$, $\bar{l}_j$, and $\bar{\theta}_i$ equals either of its corresponding bounds, then (11k) - (11n) are satisfied. Moreover, as established for LMr, r=1,2, in Section 2.1.2 and Proposition 2, if $\bar{\phi}$ is binary-valued, then, (11c) is satisfied. Thus, $\bar{x}$ is feasible to DTO(Ω). □

Proposition 9. Let Ω be such that $p_i^l = p_i^u, \forall i \in I_e$, and $l_j^l = l_j^u, \forall j \in \tau$. If $\bar{x}$ solves LP(Ω) with $\bar{\phi}$ being binary-valued, then $\bar{x}$ also solves DTO(Ω).

Proof. If $p_i^l = p_i^u$ and $l_j^l = l_j^u$, we have that $\theta_j^l = \theta_j^u$ by (6b). The proof now follows from Proposition 8. □

2.4 Optimality-Induced Valid Inequalities

In order to further tighten the model representation, we introduce in this section certain valid inequalities (denoted **VIs**) that are implied by optimality (rather than feasibility) considerations.

Proposition 10. There exists an optimal solution to Problem DTO satisfying the following inequalities:

$$q_{im} \leq s_m \phi_{d(i)}, \ \forall i \in I_e, \ m = 1, \ldots, M, \tag{30a}$$

$$r_{jn} \leq t_n \phi_{d(j)}, \ \forall j \in \tau, \ n = 1, \ldots, N. \tag{30b}$$

Proof. First of all, note that Problem DTO has an optimal solution since it is bounded and feasible (the solution $(q, r) = (0, 0)$, with $p_i = p_i^u \equiv p_i^{u(3)}$, $\forall i \in I_e$, $l_j = l_j^u \equiv l_j^{u(4)}$, $\forall j \in \tau$, and any ϕ satisfying (5g) and (5j) gives a feasible solution). Now, for any $i \in I_e$, if $\phi_{d(i)} = 0$, then event i is inconsequential to the problem, and so, we need not allocate any event-related resources to reduce p_i below p_i^u in (5e). Hence, we can set $q_{im} = 0$, $\forall m = 1, \ldots, M$, i.e., (30a) is valid. On the other hand, if $\phi_{d(i)} = 1$, then (30a) is again valid because it is implied by (5b). Likewise, (30b) is satisfied at an optimal solution because for any $j \in \tau$, if $\phi_{d(j)}$=0, then consequence $j \in \tau$ does not arise (or does not impact the objective function), and we can therefore set $r_{jn} = 0,\ \forall n = 1, \ldots, N$, while if $\phi_{d(j)} = 1$, then (30b) is implied by (5c). □

Henceforth, we shall assume that the inequalities (30a) and (30b) are incorporated within Problem DTO, and hence, within DTO(Ω) and LP(Ω), $\forall\, \Omega$. In addition, we shall perform the following variable fixings in Problem DTO(Ω) and LP(Ω) as prompted by Proposition 11 below. This strategy will become relevant in the sequel when we revise the bounds on the p- and l-variables in a branch-and-bound framework. Naturally, if any such fixings render a particular node subproblem infeasible, then we can fathom this node.

Proposition 11. Consider any (node subproblem) DTO(Ω) predicated on a set of imposed bounds Ω (and incorporating the VIs (30a) and (30b)). Then,

$$p_i^u < p_i^{u(3)} \Rightarrow \phi_d = 1,\ \forall d \in B_i,\ \text{for each } i \in I_e, \tag{31a}$$

$$l_j^u < l_j^{u(4)} \Rightarrow \phi_d = 1,\ \forall d \in B_j,\ \text{for each } j \in \tau. \tag{31b}$$

Proof. Consider any $i \in I_e$, and suppose that the currently imposed upper bound p_i^u on p_i satisfies $p_i^u < p_i^{u(3)}$. Then, by (11g), (11k), and (11l), we have that $q_{im} > 0$ for some $m \in \{1, \ldots, M\}$, and so, (30a) and (11r) imply that $\phi_{d(i)} = 1$. But since the constraints (11c, 11i, 11r) imply (18), we hence have $\phi_d = 1, \forall d \in B_i$. Similarly, for any $j \in \tau$, if the currently imposed upper bound l_j^u on l_j satisfies $l_j^u < l_j^{u(4)}$, then (11h) implies that $r_{jn} > 0$ for some $n \in \{1, \ldots, N\}$, and then (30b) and (18) yield $\phi_d = 1,\ \forall d \in B_j$. □

2.5 Upper Bounding Scheme: Procedure UB

In this subsection, we use Proposition 7 to develop a method for computing upper bounds on Problem DTO based on the solution of any lower bounding relaxation LP(Ω). (If any optimization problem is detected to be infeasible in this process, we abort the procedure.)

Procedure UB:

Step 1: Solve LP(Ω) (including Constraints (30)) and obtain an optimal solution $\bar{x}$. Apply Proposition 7 to obtain the solution $(\hat{p}, \hat{l}, \hat{q}, \hat{r}, \hat{\phi})$.

Step 2: Fix $\phi = \hat{\phi}$, resolve LP(Ω) to obtain an optimum $\bar{x}$, and apply Proposition 7 to derive a possibly revised solution $(\hat{p}, \hat{l}, \hat{q}, \hat{r}, \hat{\phi})$. Repeat this step until no further improvement is obtained in the objective function value for Problem DTO.

Step 3: Fix $\phi = \hat{\phi}$, and use a nonlinear programming (local search) solver to optimize DTO(Ω), starting with the solution obtained at Step 2 as an initial solution. (We used SNOPT Version 7 (Gill et al. (2005)) for this purpose.) Output the resulting solution value as an upper bound on Problem DTO, and update the incumbent solution x^* and its objective value v^*, if necessary.

2.6 Range Reduction

The imposed range for each ϕ-, p-, l-, and θ-variable can be tightened by sequentially solving a pair of linear programs that minimize and maximize each variable in turn over the feasible region of LP(Ω), while additionally restricting the original objective function to take on values lesser than or equal to the best known upper bound v^* obtained by Procedure UB. Starting with the ϕ-variables, note that if the minimum value of ϕ_d is positive for any $d \in \{1, \ldots, D\}$, then we can fix $\phi_d \equiv 1$, and likewise, if the maximum value is less than one, we can fix $\phi_d \equiv 0$. Next, considering the p-variables, the foregoing pair of associated linear programs is used to update the lower and upper bounds on each p_i-variable in turn, where we also update the bounds on the affected θ_j-variables using Equation (33), for each j such that $i \in S_{1j} \cup S_{2j}$. Moreover, noting that the bounds on the p- and θ-variables define the bounds on the y_1-, y_2-, and z-variables according to (8b) and (9b), if any of the bounds on the former variables are revised, then the bounds on the corresponding latter variables are also updated. The same procedure is followed for l_j, $j \in \tau$, during which the bounds on θ_j along with the bounds on the corresponding z- and ξ-variables are also updated using (33), (9b), and (10b). Finally, a pair of linear programs is solved to directly tighten the bounds on $\theta_j, \forall j \in \tau$. Following this range reduction process, the polyhedral outer approximations for the logarithmic relationships are constructed based on the revised bounds on the variables.

Next, LP(Ω) is (re-)solved and the lower and upper bounds on Problem DTO are updated as possible. Note that at each node of the branch-and-bound tree, we invoke range reduction only once.

3 Global Optimization Branch-and-Bound Algorithms

We now design three alternative approaches to solve Problem DTO via DTO(Ω) and its relaxation LP(Ω). In **Algorithm A**, we utilize a specialized branch-and-bound process based on partitioning the hyperrectangle Ω, where the bounds on the variables ($\psi,\ \theta,\ y_1,\ y_2,\ z,\ \xi$) are accordingly computed by (6e), (6b), (8b), (8b), (9b), and (10b), respectively. For any node subproblem DTO(Ω) that is associated with a particular Ω, we construct the relaxation LP(Ω) and solve it to compute a lower bound. Let $\bar{x}$ solve LP(Ω). If the conditions of Proposition 6 hold, we will have also solved subproblem DTO(Ω). Otherwise, we apply Procedure UB to find a feasible solution to Problem DTO and update the incumbent solution and the associated upper bound if possible, perform range reductions to update Ω and (11q), and as necessary, we branch at this node by partitioning Ω as follows:

Branching Rule A: While selecting the branching variable, priority is given to the ϕ_d-variables. Thus, we first check if $\bar{\phi}$ is binary-valued. If not, then we define the set $K' = \{k \in K : (\bar{\phi}_d,\ d \in J_k) \text{ is not binary-valued}\}$ and find $\hat{k} \in \arg\operatorname*{lexmin}_{k\in K'}\{|B_k|, \min_{d\in J_k} |\bar{\phi}_d - 0.5|\}$. We then branch on $\phi_{\hat{d}}$, where $\hat{d} \in \arg\min_{d\in J_{\hat{k}}}\{|\bar{\phi}_d - 0.5|\}$, by using the dichotomy that $\phi_{\hat{d}} = 1 \vee \phi_{\hat{d}} = 0$. Note that Constraints (11b) and (11i) together with Equations (17) and (18) are invoked to fix additional (ϕ, w)-variables as possible whenever we branch on a ϕ-variable. On the other hand, if $\bar{\phi}$ is binary-valued, then we find a variable p_i or l_j having the largest bounding interval according to:

$$\max\{(p_i^u - p_i^l), (l_j^u - l_j^l), \forall i, j\}, \tag{32a}$$

where ties are broken by favoring the variable that gives the largest discrepancy in

$$\max\{|\bar{y}_{1i} - ln(\bar{p}_i)|, |\bar{y}_{2i} - ln(1 - \bar{p}_i)|, |\bar{\xi}_j - ln(\bar{l}_j)|, \forall i, j\}, \tag{32b}$$

where the first two terms in (32b) relate to p_i and the third term to l_j. If the identified term in (32) relates to p_i, then we branch on this variable by partitioning its interval according to the dichotomy that $p_i \in [p_i^l, (p_i^l + p_i^u)/2] \;\vee\; p_i \in [(p_i^l + p_i^u)/2, p_i^u]$. If the identified term corresponds to l_j, then we partition its interval according to the dichotomy that $l_j \in [l_j^l, (l_j^l + l_j^u)/2] \;\vee\; l_j \in [(l_j^l + l_j^u)/2, l_j^u]$. $\square$

Algorithm B is the same as Algorithm A, except that we also include θ_j in the partitioning process, as motivated by Proposition 8. In this case, given any imposed bounds $[\theta_j^l, \theta_j^u]$ on the variable θ_j, $\forall j \in \tau$, we update these bounds based on the implied bounds derived via (6b) according to:

$$\begin{bmatrix} \theta_j^l \leftarrow max\{\theta_j^l,\ l_j^l \prod_{i \in S_{1j}} p_i^l \prod_{i \in S_{2j}} (1 - p_i^u)\} \\ \theta_j^u \leftarrow min\{\theta_j^u,\ l_j^u \prod_{i \in S_{1j}} p_i^u \prod_{i \in S_{2j}} (1 - p_i^l)\} \end{bmatrix}, \forall j \in \tau. \tag{33}$$

Now, define Ω' as

$$\Omega' \equiv \{(p, l, \theta) : p_i^l \leq p_i \leq p_i^u, \forall i,\ l_j^l \leq l_j \leq l_j^u, \forall j,\ \theta_j^l \leq \theta_j \leq \theta_j^u, \forall j\}, \tag{34}$$

and let DTO(Ω') be identical to DTO(Ω), except that we primarily impose the bounds $(p, l, \theta) \in \Omega'$, and then compute the bounds on the variables (ψ, y_1, y_2, z, ξ) using (6e), (8b), (8b), (9b), and (10b), respectively. Whenever range reduction is performed, all these bounds are updated accordingly.

At each node of the branch-and-bound tree for Algorithm B, we proceed exactly as in Algorithm A except that we now solve the relaxation problem LP(Ω') for computing lower bounds, and also, for the partitioning scheme, the Branching Rule A is substituted by the following, where $\bar{x}$ solves LP(Ω'):

Branching Rule B: We apply the same scheme as in Branching Rule A in case $\bar{\phi}$ is not binary-valued. Else, we find a variable that yields the maximum discrepancy according to:

$$\max\{|\bar{y}_{1i} - ln(\bar{p}_i)|, |\bar{y}_{2i} - ln(1 - \bar{p}_i)|, |\bar{\xi}_j - ln(\bar{l}_j)|, |\bar{z}_j - ln(\bar{\theta}_j)|, \forall i, j\}, \tag{35}$$

where ties are broken by favoring the variable having the largest bounding interval. If the maximum is attained by one of the first two terms, then we branch on the corresponding variable p_i by partitioning its interval according to $[p_i^l, \bar{p}_i] \vee [\bar{p}_i, p_i^u]$. Similarly, if the maximum is attained by the third or fourth term, we branch with respect to l_j or θ_j, partitioning their intervals as $[l_j^l, \bar{l}_j] \ \vee \ [\bar{l}_j, l_j^u]$, or $[\theta_j^l, \bar{\theta}_j] \ \vee \ [\bar{\theta}_j, \theta_j^u]$, respectively. $\square$

The third alternative, called **Algorithm C**, can be viewed as a combination of Algorithms A and B. Motivated by Propositions 8 and 9, we adopt the following partitioning scheme:

Branching Rule C: If $\bar{\phi}$ is not binary-valued, then we apply Branching Rule A as before. Else, we select a variable having the largest bounding interval among the set of variables having at least a δ-deviation from their associated logarithmic functions, where $0 \leq \delta \leq 10^{-1}$. The value of δ changes through the optimization process depending on the number of variables that violate the exact corresponding logarithmic relations. Specifically, we select a variable p_i or l_j having the largest bounding interval according to:

$$\max\{(p_i^u - p_i^l), (l_j^u - l_j^l), \forall i \in I', \forall j \in \tau'\}, \tag{36}$$

where the index sets I' and τ' are defined by

$$I' \equiv \{i : |\bar{y}_{1i} - ln(\bar{p}_i)| > \delta\} \cup \{i : |\bar{y}_{2i} - ln(1 - \bar{p}_i)| > \delta\} \text{ and } \tau' \equiv \{j : |\bar{\xi}_j - ln(\bar{l}_j)| > \delta\}. \tag{37}$$

If both I' and τ' are empty, we set $\delta \leftarrow \delta/10^v$ for the smallest integer $v \geq 1$ such that at least one of these index sets becomes nonempty, and then apply (36). For the selected branching variable, we split its interval at the geometric mean, i.e., if the identified term in (36) corresponds to some p_i-variable, then we partition its interval according to $[p_i^l, \sqrt{p_i^l p_i^u}] \vee [\sqrt{p_i^l p_i^u}, p_i^u]$, and if it corresponds to some l_j-variable, then we partition its interval according to $[l_j^l, \sqrt{l_j^l l_j^u}] \vee [\sqrt{l_j^l l_j^u}, l_j^u]$. $\square$

Remark 6. Note that, for a function $f = ln(\gamma), 0 < \gamma^l < \gamma < \gamma^u < \infty$, splitting the γ-interval at its geometric mean corresponds to bisecting the implied bounds on the f-variables at its arithmetic mean, i.e., at $[ln(\gamma^l) + ln(\gamma^u)]/2$. Figure 2 displays this feature, and exhibits the evident potential for generating improved bounds via the child-nodes by using the geometric mean splitting technique. For the sake of comparison, we shall also implement the interval partitioning technique predicated by Branching Rules A and B within Branching Rule C. $\square$

Now, we formally describe the branch-and-bound procedure Algorithm A for solving Problem DTO. At each stage s of this algorithm, $s = 0, 1, \ldots$, we define A_s as the set of non-fathomed, or *active nodes.* Each active node $a \in A_s$ is associated with a hyperrectangle Ω^a. A lower bound LB_a on an active node is obtained by solving the linear programming relaxation LP(Ω^a) to yield $LB_a = v[\text{LP}(\Omega^a)]$. At each stage s, we define the global lower bound on Problem DTO by

$$LB(s) \equiv \min\{LB_a : a \in A_s\}.$$

Whenever LP(Ω^a) is solved for a node $a \in A_s$, we apply Procedure UB to possibly find a feasible solution for updating the upper bound for Problem DTO, and perform range reductions as necessary. These bounds are used in concert with a least lower bound node selection rule and the aforementioned partitioning schemes as detailed below.

3.1 Branch-and-Bound Algorithm A for Problem DTO

Step 0: Initialization. Set $s = 0$, $A_s = \{0\}$, $a(s) = 0, a = 0$, and $\Omega^0 = \Omega$. Solve LP(Ω^0) and let x^0 be the optimal solution obtained. Set $LB_0 = v[\text{LP}(\Omega^0)]$. Apply Procedure UB of Section 2.5 with $\bar{x} \equiv x^0$ to derive an incumbent solution x^* with objective value v^*. Invoke the range reduction scheme of Section 2.6 to revise LB_0 and (x^*, v^*) as possible. If $LB_0 \geq v^*(1-\epsilon)$ for some optimality tolerance $\epsilon \geq 0$, then stop with the incumbent solution as (ϵ)-optimal to Problem DTO. Otherwise, proceed to Step 1.

Step 1: Partitioning Step. Invoke the Branching Rule A to partition the selected node $a(s)$ into two subnodes $a+1$ and $a+2$ with associated hyperrectangles Ω^{a+1} and Ω^{a+2}, respectively. Replace $A_s \leftarrow A_s \cup \{a+1, a+2\} - \{a(s)\}$.

Step 2: Bounding Step. Solve the relaxed problems LP(Ω^{a+1}) and LP(Ω^{a+2}) after fixing ϕ-variables as possible using Proposition 11. Apply Procedure UB to the solutions found at each node and update the incumbent solution if possible, and perform range reductions as necessary.

Step 3: Fathoming Step. Fathom any non-improving node and update $A_{s+1} \leftarrow A_s - \{a \in A_s : LB_a \geq v^*(1-\epsilon)\}$. Increment s by 1.

Step 4: Termination Check and Node Selection Step. If $A_s = \varnothing$, stop with the incumbent solution as an $\epsilon-$optimum. Otherwise, select a node $a(s) \in \arg\min\{LB_a : a \in A_s\}$ and return to Step 1.

Proposition 12. (Main Convergence Result)
Algorithm A (with $\epsilon = 0$) either terminates finitely with the incumbent solution as an optimum to DTO, or else, an infinite sequence of stages is generated such that along any infinite branch of the tree, any accumulation point of the (p, l, q, r, ϕ)-variable part of the sequence of linear programming relaxation solutions solves Problem DTO.

Proof. The case of finite termination is clear. Hence, suppose that an infinite sequence of stages is generated. Consider any infinite branch of the tree having a nested hyperrectangle sequence $\Omega^{a(s)}$ for s belonging to some index set S. For each stage s, we have $LB(s) = LB_{a(s)} = v[\text{LP}(\Omega^{a(s)})], \forall s \in S$. For each node $a(s)$, let $x^{a(s)}$ solve LP($\Omega^{a(s)}$). By the compactness of the feasible region, there exists a convergent subsequence for $\{x^{a(s)}, \Omega^{a(s)}\}_{s \in S}$. Without loss of generality, assume that $\{x^{a(s)}, \Omega^{a(s)}\}_{s \in S} \to (x^*, \Omega^*)$. We must show that the $(p^*, l^*, q^*, r^*, \phi^*)$-variable part of x^* solves Problem DTO.

Note that $LB_{a(s)} = \min\{LB_a : a \in A_s\} \leq v[\text{DTO}], \forall s \in S$, which is also preserved in the limit:

$$v^* \equiv \lim_{s \to \infty, s \in S} LB_{a(s)} \leq v[\text{DTO}]. \tag{38}$$

Since we can only branch on the ϕ-variables finitely often, we have that $\phi^{a(s)}$ is binary-valued for all $s \in S$ sufficiently large. Hence, along the infinite branch, at least one of the p_i- or l_j-variables is chosen as the branching variable infinitely often. Since we bisect the corresponding interval at each step, the length of the interval for a such variable converges to 0. Therefore, according to Branching Rule A, we have in the limit as $s \to \infty, s \in S$ that $p_i^{*l} = p_i^{*u}, \forall i$, and $l_j^{*l} = l_j^{*u}, \forall j$. By Proposition 9, we thus have that x^* is feasible to DTO with objective function value $v^* \equiv \lim_{s \to \infty, s \in S} LB_{a(s)}$, and so, $v^* \geq v[\text{DTO}]$. Together with (38), we have that x^* solves DTO with objective function value v^*. $\square$

3.2 Branch-and-Bound Algorithm B for Problem DTO

This alternative branch-and-bound procedure is the same as Algorithm A, with the exception of the branching variable selection at each stage s along with the partitioning of the hyperrectangle Ω' associated with each active node $a \in A_s$, where Ω' is defined by (34). The following result establishes the global convergence of Algorithm B.

Proposition 13. Similar to the main convergence result of Algorithm A stated in Proposition 12, Algorithm B (with $\epsilon = 0$) solves Problem DTO.

Proof. The case of convergence in a finite number of steps is clear. For the case of infinite stages, taking any infinite branch of the tree where the stages s are indexed by a set S, we have as in the proof of Proposition 12 that $\{x^{a(s)}, \Omega'^{a(s)}\}_{s\in S} \to \{x^*, \Omega'^*\}$, and that $LB(s) = LB_{a(s)} = v[\text{LP}(\Omega'^{a(s)})] \leq v[\text{DTO}], \forall s \in S$, which also holds true in the limit:

$$v^* = \lim_{s\to\infty, s\in S} LB_{a(s)} \leq v[\text{DTO}]. \tag{39}$$

We now show that x^* is feasible to Problem DTO. Along the convergent subsequence, for s large enough, $\phi^{a(s)} = \phi^*$ is binary-valued, and so some p_i-, l_j-, or θ_j-variable is partitioned infinitely often at stages $s \in S_1$, say, where $S_1 \subseteq S$. From Branching Rule B, this variable equals one of its bounds in the limit, and therefore, the discrepancy related to it in (35) approaches zero. Since this variable has the maximum discrepancy in (35), $\forall s \in S_1$, the discrepancies related to the other p-, l-, and θ-variables also approach zero as $s \to \infty, s \in S_1$. Consequently, by Proposition 8, x^* satisfies Constraints (11c), and (11k)-(11n) with $\phi^*_d \in \{0,1\}, \forall d \in D$. Hence, x^* is feasible to DTO with objective value v^*, so that $v^* \geq v[\text{DTO}]$. Together with (39), we have that x^* solves DTO with objective value v^*. □

3.3 Branch-and-Bound Algorithm C for Problem DTO

As described earlier in the section, this procedure follows the same scheme as that of Algorithms A and B, while adopting a combination of these two methods for selecting a branching variable and partitioning its interval, and including the alternative option of splitting the interval of the selected variable at its geometric mean. As such, its global convergence proof follows identically to that in Propositions 12 and 13 above, and is omitted for the sake of brevity.

4 Computational Results

In this section, we study the effectiveness of the proposed branch-and-bound procedures for solving Problem DTO. We begin by considering the gas-line rupture application illustrated in Figure 1, and solve it using the proposed algorithms as well as the commercial global optimization software BARON, Version 8.1.5 (Sahinidis (1996)). We further analyze the sensitivity of the solution with respect to the budget and resource availability restrictions.

Next, using a suitable experimental design, we explore the best combination of algorithmic features including the various branching variable selection schemes and partitioning strategies. We also investigate the performance of the different proposed linearization methods and techniques for generating tangential supports, and assess the effect of incorporating range reductions and VIs on CPU time. In addition, we explore the direct implementation of the software BARON to solve the original problem formulation DTO (given by (5)) as well as its transformed version DTO(Ω) (given by (11)), in comparison with our best proposed procedure. Finally, the special case of the event tree optimization problem (**ETO**) introduced by Sherali et al. (2008) is solved using the best proposed algorithmic strategy, and the results are compared with the solutions generated by BARON (which was used to solve ETO by Sherali et al. (2008)). Runs with BARON have been made on a remote 1.6 GHz Intel Pentium M processor running Linux (courtesy of N. V. Sahinidis), whereas the proposed procedures have been implemented on a local 2.33 Ghz Intel Pentium M processor running Windows.

4.1 Gas-Line Rupture: Illustrative Hypothetical Case Study

Consider the scenario in which, in the aftermath of a gas leak (represented by node 0), cascading sequences of probabilistic events and decisions result in consequences as depicted in Figure 1. For this specific gas-line rupture application, we assume that five event-related and five consequence-related resources are available to prevent failures and to alleviate consequences.

The model coefficients a_{im} and b_{jn} were generated uniformly in the respective intervals [0.5, 1] and [1000, 5000]. Additionally, we set $a_{i0} = ln(0.01/(1-0.01)), \forall i$, and generated $b_{j0}, \forall j$, randomly on [100000, 200000]. The total available event-related and consequence-related resources were set to $s_m = 10, \forall m$ and $t_n = 50, \forall n$, respectively, where the corresponding per-unit costs of allocating resources, c_{im} and d_{jn}, were generated uniformly on [20, 40]. The total available budget for resource allocation was initially taken as $\beta = 3000$.

The lower bound on $l_j, \forall j \in \tau$, was determined depending on the scenario produced by the unique path connecting the root node to the leaf node j. More specifically, a contribution to l_j^l of $1000(k)^{(2.5)}$ was set to be incurred for each failure event on the path from node 0 to the leaf node, where k is the nodal-distance between the failed event and the leaf node. The lower bound on l_j for any leaf node that has no failure event along the path to it from node 0 was set to 10. The upper bound was determined as $l_j^{u(4)}, \forall j \in \tau$, whereas the lower and upper bounds on p_i were set to 0.0001 and $p_i^{u(3)}$, respectively, $\forall i \in I_e$. Moreover, the direct cost for each selected decision alternative $d = 1, \ldots, D = 9$ was generated randomly on [2, 4]. (The online supplement provides the complete data set.)

We solved the gas-line rupture problem with 10 randomly generated decision cost vectors (on the interval [2, 4]) using ϵ=0.001. Table 1 displays the results obtained. The proposed algorithm solved all 10 problem instances to optimality exploring 9-23 nodes within 1.1-2.6 CPU seconds. On the other hand, using default settings, BARON (Version 8.1.5) optimized four of these 10 problems within 0.9-1.9 CPU seconds and yielded the same objective value as given by the proposed algorithm, while for four other problem instances, it terminated with a slightly worse (3-8 %) objective value as well as with different binary decisions. (Table 1 provides the ratio z^*_{BARON}/z^* of the best solution values produced by BARON and the proposed algorithm.) Moreover, when we re-ran BARON after fixing the binary decisions as given by the proposed algorithm, it found exactly the same optimal solution value. For the remaining two problems, BARON terminated with the maximum limiting CPU time (1000 seconds) without satisfying the specified optimality gap restriction.

Using five levels of the budget, $\beta \in \{1000, 2000, \ldots, 5000\}$, and four levels of event-related resources, $s \in \{1, 2, 3, 4\}$, we computed the optimal objective function value as plotted in Figure 3. As expected, the objective function value improves significantly with initial additional event-related resources, but further resource increases result in diminishing marginal returns. On the other hand, the relation between the number of nodes explored and the budget level is not readily evident. Analyzing the optimal solution for s=3, it was observed that for β =1000, 2000, and 3000, we obtained the same optimal (p, q)-values, and the additional budget increments were used to acquire and allocate consequence-related resources. Concomitant with the budget increase, the state space of the q-variables increased, which resulted in a higher number of nodes explored. However, for $\beta = 4000$ and 5000, the failure probabilities of critical events and loss amounts at critical outcomes were readily minimized to their lower bounding values, thus resulting in fewer enumerated nodes.

4.2 Random Test Cases

We next generated random decision tree problem instances to assess the effectiveness of the proposed solution techniques as well as that of the global optimization software BARON. The two inputs provided to the tree generation process are the desired *decision node density*, as defined by the ratio of the number of decision nodes to the total number of nodes, and the size of the tree as measured by its depth. Each tree was accordingly generated by initially constructing nodes 0 and 1 as in Figure 1. Then, while sequentially generating additional nodes (including the case of node 1), we first checked whether the node was at the maximum depth, in which case it was labeled as a leaf node. Else, we determined the type (event or decision) of the node randomly, depending on the desired and current ratios of the number of event and decision nodes. If the node turned out to be a decision node, we randomly determined the number of decision alternatives emanating from it. Finally, the leaf node representing the direct consequence of a failure at node 1 was created.

To complete the instance specification, we randomly generated the logit coefficients a_{im} and b_{jn}, the cost coefficients c_{im} and d_{jn}, and the decision costs c_d as explained in Section 4.1. We set b_{j0} equal to the sum of the lower bound on l_j and a uniformly generated random variable over the interval [100000, 150000]. The budget and the decision node density were respectively 3000 and 0.2-0.4 (specified desired value = 0.3) for the first five instances, and

1500 and 0-6-0.7 (specified desired value = 0.7) for the last five instances.

In the following runs, we define the **base case approach** as the one that incorporates the linearization scheme LM2 (Section 2.1.2), the valid inequalities VIs (Section 2.4), and invokes range reduction (Section 2.6) while generating four tangential supports for each functional form $f = ln(\gamma)$ using the BES ($H = 4$) strategy, and utilizes an optimality gap tolerance of $\epsilon = 0.001$.

Using the base case approach, we first studied the efficiency of the three branching variable selection rules (A, B, and C) together with the three splitting rules (arithmetic mean, geometric mean, and the current optimal value), for a total of eight combinations as displayed in Table 2. (Note that Rule A combined with the current optimal value partitioning strategy is not included because the current optimal value could be at the lower or upper bound of the selected branching variable, which would consequently not produce valid child-nodes.) Observe also that Algorithm A is defined by the branching variable selection rule A and the arithmetic mean splitting rule (Combination 1), whereas Algorithm B is defined by the branching variable selection rule B and the current optimal value splitting rule (Combination 5). Algorithm C is (principally) defined by the branching variable selection rule C and the geometric mean splitting rule (Combination 7).

The branching variable selection rule C and the arithmetic mean splitting rule outperformed the other combinations and was used for further evaluating various algorithmic variants as discussed next. (Algorithm C defined by Combination 7 comes a close second, and is also further evaluated in subsequent runs. In fact, with BES(H=5) used in lieu of BES(H=4), the mean CPU times for Combination 6 and 7 were 26.4 and 26.0 CPU seconds, respectively.) In each of the ensuing experiments, one of the features in the base case was varied and its best setting was determined with respect to CPU time using the randomly generated problem instances. Tables 3-7 provide results for the following experimental studies:

- Table 3: Comparison of linearization methods LM1 versus LM2.

- Table 4: Effect of the number of tangential supports: BES with $\epsilon = 0.01, 0.05, 0.1$, and 0.2 versus BES($H$) with H=4, 5, 10, and 20.
- Table 5: As in Remark 5 in Section 2.2, the y_1-, y_2-, and z-variables tend to be at their lower/upper bounds at optimality. Hence, we tested generating only H=2 tangential supports at the two interval end-points for these variables, but used BES(H=4) for the remaining variables.
- Table 6: Effect of implementing range reductions.
- Table 7: Effect of incorporating the valid inequalities VIs (30) of Proposition 10 and conducting the related tests (31) of Proposition 11.

The results displayed in Tables 3-7 indicate that utilizing the linearization method LM2, generating H=4 tangential supports via BES(H=4), and invoking range reduction and valid inequalities outperformed other alternatives. Note that the relative efficiency of the linearization methods depends on the decision node density as indicated in Table 3. Whereas both linearization schemes were comparable for the first five instances (low decision node density - LM2 consumed 2.5% lesser effort), we observed a 13% improvement in the CPU time for the last five instances (high decision node density) with LM2 over LM1. This is to be expected because higher decision node density instances involve more complex binary substructures in the model, which therefore benefit more substantially by using the improved representation LM2 over LM1. We use these settings in further runs with our proposed algorithm, hereinafter referred to as **Algorithm Best**. We also explore the efficiency of Combination 7 (Algorithm C) along with Algorithm Best since the difference in performance between these two algorithms is not significant as displayed in Table 2.

The decision tree optimization problem can alternatively be directly solved by using off-the-shelf commercial software on the original model (5) or on the better-structured, equivalently transformed formulation (11). The global optimization software BARON was therefore used to solve problem instances modeled alternatively as in (5) or (11) using LM2, as well as via (11) while using both LM2 and the proposed VIs (30), in order to assess their direct solvability. Table 8 displays the results obtained, and also includes the performance of Algorithm Best for comparison purposes. Using the original DTO formulation (5), BARON terminated prematurely for four of the 10 problems (with optimality gaps of 98.8%, 1%, 1.2% and 0.3%, respectively), whereas the proposed algorithm optimized all the problems within the maximum limiting CPU time. On average, it took 79 CPU seconds for BARON to solve Problem DTO (*excluding* the early termination cases), while the same set of problems were optimized in 15.5 CPU seconds with the proposed algorithm. We also mention here that Algorithm C solved these instances in 23.6 CPU seconds on average, where this value decreases to 15 CPU seconds when the same two premature termination cases are excluded (detailed results are not displayed for the sake of brevity). Moreover, we observed premature termination for all problem instances using BARON on the transformed DTO formulation (11) with LM2, which may be due to the increase in the number of variables as well as the number of nonlinear constraints. However, when we further invoked the valid inequalities VIs, BARON managed to solve eight of the 10 problem instances using this transformed formulation.

Finally, we solved some 15 random instances of Problem ETO (having event nodes only) as a special case of Problem DTO using the proposed algorithm (Algorithm Best), and compared the results against applying BARON to solve Model DTO(Ω) as given by (11), which is tantamount to the strategy utilized by Sherali et al. (2008). The experimental design and results are presented in Table 9. We set the budget equal to 3000, 4000, and 5000 for the instances having 17, 33, and 65 leaf nodes, respectively. Algorithm Best successfully solved 13 out of 15 random instances of Problem ETO, whereas BARON solved only one of them within the maximum allowable time limit. Although the final upper bounds derived by BARON were very close to those obtained by Algorithm Best, the quality and the convergence rate of the lower bounds resulted in its relatively poor performance. As the problem size increased, the optimality gap produced at termination using BARON steadily increased. For the two instances that remained unsolved using the proposed algorithm within the 1000 CPU seconds time limit, the optimality gaps at termination were respectively 0.7 % and 0.8%, versus gaps of 2.9% and 3.4% for these same two problems when implementing BARON. Furthermore, we explored the efficiency of Algorithm C for these test cases. The performance was comparable to Algorithm Best, where the CPU times consumed for the three sets of instances were 20, 187, and 749 CPU seconds, as compared with 23, 188, and 736 CPU seconds, respectively for Algorithm Best.

5 Summary and Conclusions

This paper addresses the strategic reduction of risk by allocating certain resources to reduce failure probabilities of system safety components and subsequent losses related to final outcomes, as well as by selecting optimal strategic decisions or system design alternatives in the context of a hazardous event. Using a novel decision tree approach, the problem was modeled as a nonconvex mixed-integer 0-1 factorable program. We developed a specialized branch-and-bound algorithm in concert with polyhedral approximations, valid inequalities, alternative linearization schemes, and range reduction strategies, and established its theoretical convergence to global optimality.

Among the various algorithmic variants tested, we advocate the strategy that implements branching variable selection rule C along with the arithmetic (or geometric) mean splitting rule, while utilizing the linearization method LM2 (6% CPU time reduction over LM1 on average, but 13% reduction on average for relatively higher decision node density problems); generating tangential supports via the BES(H=4) scheme; invoking the range reduction mechanism (all instances are solved to ϵ-optimality with range reduction compared to 100% early termination without range reduction), and incorporating the derived valid inequalities (13% CPU time reduction on average). In our computational experience, this proposed algorithmic approach outperformed the commercial software BARON (Version

8.1.5). We also solved random instances of Problem ETO, introduced by Sherali et al. (2008), for which the performance of the proposed algorithm greatly surpassed BARON (which was the solver adopted in Sherali et al. (2008)).

This research can be extended in several directions. For instance, instead of minimizing the overall risk, we could minimize the maximum risk in the system. Additionally, as explained in Remark 2, we can use a diminishing marginal return loss function in lieu of the linear correspondent given by (4), as well as replace Bernoulli events with more general multistate events. Furthermore, as an algorithmic extension, we could explore a GUB-based partitioning whenever we branch on a $\phi-$variable. Finally, it is of interest to investigate specific applications of the proposed generic DTO problem framework in event-decision contexts that arise in various areas such as homeland security and health-care.

Notes:

[1]:Algorithm terminated with the maximum limiting number of explored nodes without satisfying the specified optimality gap restriction.

[2]:The optimality gap is computed as $\frac{(UB-LB)}{UB}$ where UB and LB are the upper and lower bounds on the optimal objective function value at termination.

Acknowledgement: This research has been supported by the *National Science Foundation* under Grant No. CMMI-0552676.

References

Acosta, C., N. Siu. 1993. Dynamic event trees in accident sequence analysis: Application to steam generator tube rupture. *Reliability Engineering and System Safety* **41** 135–154.

Andrews, J. D., L. M. Bartlett. 2003. Genetic algorithm optimization of a firewater deluge system. *Quality and Reliability Engineering International* **19** 39–52.

Andrews, J. D., S. J. Dunnett. 2000. Event-tree analysis using binary decision diagrams. *IEEE Transactions on Reliability* **49**(2) 230–238.

Beim, G. K., B. F. Hobbs. 1997. Event tree analysis of lock closure risks. *Journal of Water Resources Planning and Management* **123**(3) 169–178.

Dillon, R. B., M. E. Pate-Cornell, S.D. Guikema. 2003. Programmatic risk analysis for critical engineering systems under tight resource constraints. *Operations Research* **51**(3) 354–370.

Dugan, J. B., K. J. Sullivan, D. Coppit. 2000. Developing a low-cost high-quality software tool for dynamic fault-tree analysis. *IEEE Transactions on Reliability* **49** 49–59.

Dutuit, Y., A. Rauzy. 1996. A linear-time algorithm to find modules of fault trees. *IEEE Transactions on Reliability* **45**(3) 422–425.

Furuta, H., N. Shiraishi. 1984. Fuzzy importance in fault tree analysis. *Fuzzy Sets and Systems* **12** 205–213.

Gill, P. E., W. Murray, M. A. Saunders. 2005. SNOPT: An SQP algorithm for large-scale constrained optimization. *Siam Review* **47**(1) 99–131.

Hayes, K. R. 2002. Identifying hazards in complex ecological systems - Part 1: Fault tree analysis for biological invasions. *Biological Invasions* **4** 235–249.

Huang, D., T. Chen, M. J. Wang. 2001. Fuzzy set approach for event tree analysis. *Fuzzy Sets and Systems* **118**(1) 153–165.

Jiang, Y., J. D. McClley, T. Van Voorhis. 2006. Risk-based resource optimization for transmission system maintenance. *IEEE Transactions on Power Systems* **21**(3) 1191–2000.

Kenarangui, R. 1991. Event-tree analysis by fuzzy probability. *IEEE Transactions on Reliability* **40**(1) 120–124.

Khan, F. I., S. A. Abbasi. 2000. Analytical simulation and Profat II: A new methodology and a computer automated tool for fault tree analysis in chemical process industries. *Journal of Hazardous Materials* **75**(1) 1–27.

Mehr, A. F., I. Y. Tumer. 2006. Risk-based decision-making for managing resources during the design of complex space exploration systems. *Journal of Mechanical Design* **128** 1014–1022.

Mulvey, J. M., R. J. Vanderbei, S. A. Zenios. 1995. Robust optimization of large-scale systems. *Operations Research* **32**(2) 264–281.

Rauzy, A. 1993. New algorithms for fault tree analysis. *Reliability Engineering and System Safety* **40** 203–211.

Sahinidis, N. V. 1996. BARON : A general purpose global optimization software package. *Journal of Global Optimization* **8**(2) 201–205.

Sherali, H. D., J. Desai, T. S. Glickman. 2008. Optimal allocation of risk reduction resources in event trees. *Management Science* **54**(7) 1313–1321.

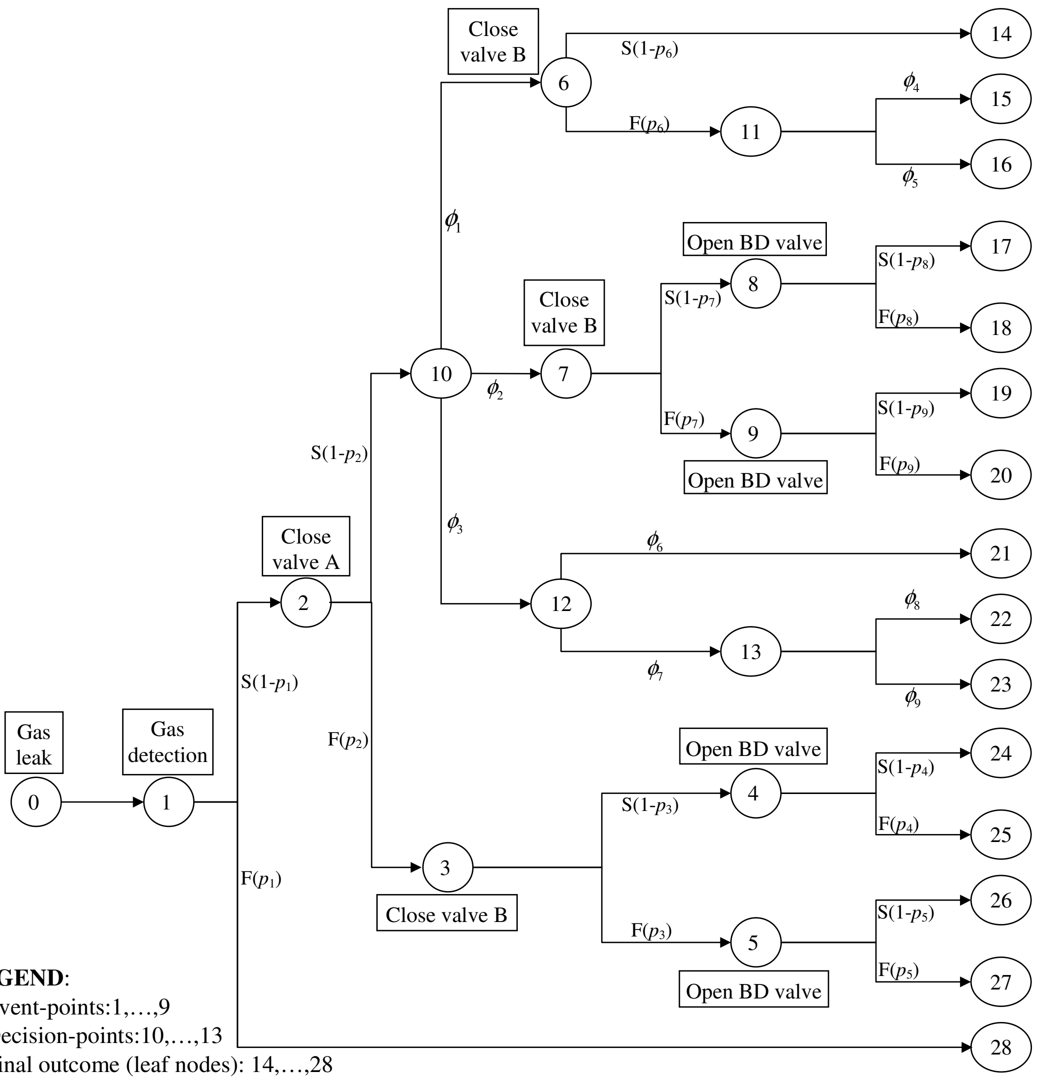
Sherali, H. D., H. Wang. 2001. Global optimization of nonconvex factorable programming problems. *Mathematical Programming* **89** 459–478.

Sinnamon, R. M., J. D. Andrews. 1997a. Improved accuracy in quantitative fault tree analysis. *Quality and Reliability Engineering International* **13** 285–292.

Sinnamon, R. M., J. D. Andrews. 1997b. Improved efficiency in qualitative fault tree analysis. *Quality and Reliability Engineering International* **13** 293–298.

Ulerich, N. H., G. J. Powers. 1988. On-line hazard aversion and fault diagnosis in chemical processes: The digraph + fault-tree method. *IEEE Transactions on Reliability* **37**(2) 171–177.

Figure 1: A decision tree representing a gas-line rupture.



LEGEND:

- Event-points:1,…,9
- Decision-points:10,…,13
- Final outcome (leaf nodes): 14,…,28
- Binary variables:

ϕ_1 : Activate only valve B
ϕ_2 : Activate valves B and BD
ϕ_3 : Activate neither of valves B and BD
$\left.\right\}$ $\phi_1 + \phi_2 + \phi_3 = 1$

$\left.\begin{array}{l}\phi_4 : \\ \phi_5 :\end{array}\right\}$ Emergency response choices, with $\phi_4 + \phi_5 = \phi_1$

$\left.\begin{array}{l}\phi_6 : \\ \phi_7 :\end{array}\right\}$ Primary cleaning options, with $\phi_6 + \phi_7 = \phi_3$

$\left.\begin{array}{l}\phi_8 : \\ \phi_9 :\end{array}\right\}$ Secondary cleaning options, with $\phi_8 + \phi_9 = \phi_3 \phi_7$

Figure 2: Partitioning at the arithmetic mean versus the geometric mean.

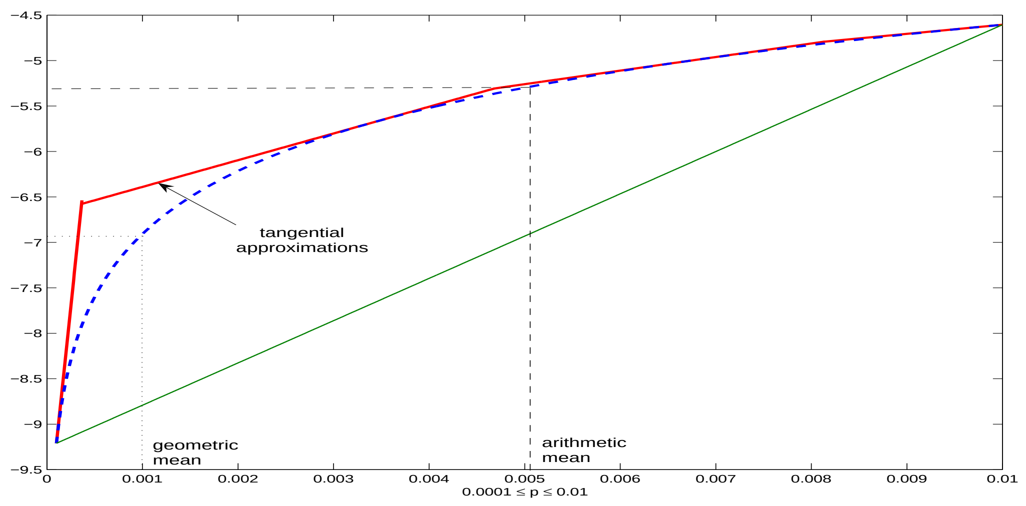
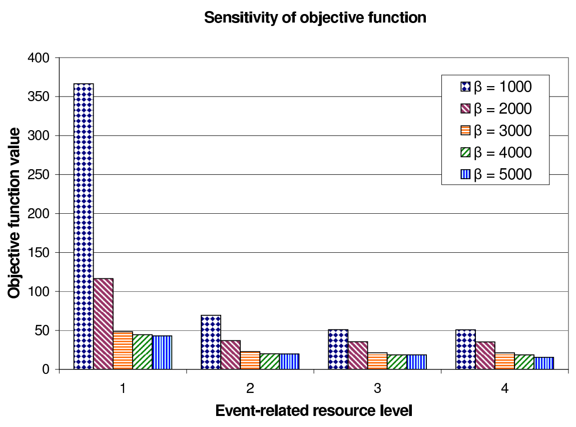


Figure 3: Sensitivity of the objective function value and the number of nodes explored with respect to budget and resources.



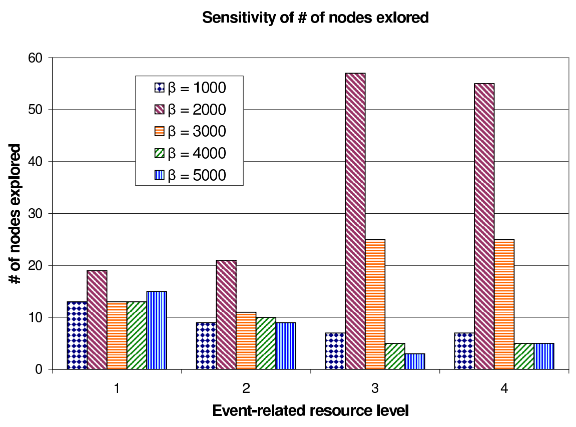


Table 1: Results for BARON and the proposed algorithm for the gas-line rupture problem.

	Proposed Algorithm	BARON (Version 8.1.5)	
Problem Instance	CPU Time	CPU Time	z^*_{BARON}/z^*
1	2.59	1.62	1.08
2	2.63	1.68	1.03
3	1.14	0.89	1
4	2.56	1.38	1
5	2.31	1.63	1
6	2.61	1000	1
7	2.47	1.88	1
8	1.7	1.36	1.03
9	1.64	1.42	1.06
10	1.28	1000	1

Table 2: Performance of the eight combinations.

	Branching Variable		Problem Instance										
	Selection	Splitting	1	2	3	4	5	6	7	8	9	10	**Mean**
1 (Alg. A)	Rule A	Arith.	10.8	150	8.6	31.4	5.2	16.3	11.9	8.8	22.9	15.3	**28.1**
2	Rule A	Geom.	10.4	139.8	8.2	27.4	5.1	16.4	10.7	9	22.6	14.8	**26.4**
3	Rule B	Arith.	11.6	132.8	9.8	71.3	11.3	16.6	11.8	8.8	22.9	14.8	**31.2**
4	Rule B	Geom.	10.9	187.7	12.1	142.8	9.8	16.3	11.4	8.9	22.8	15	**43.8**
5 (Alg. B)	Rule B	Current	14.4	215.2	12.3	124.2	10.6	15.9	11.7	9.1	22.6	14.9	**45.1**
6	Rule C	Arith.	10.1	115.1	8.4	26.4	5.1	16.3	11.3	8.8	23	14.8	**23.9**
7 (Alg. C)	Rule C	Geom.	10.4	125	8	25.7	5	16.3	10.7	8.9	22.8	14.6	**24.7**
8	Rule C	Current	10.5	157.5	8.2	27.5	4.9	16.6	11.4	8.8	22.6	15	**28.3**

Table 3: Performance of the two proposed linearization methods LM1 and LM2.

	LM1		LM2	
Problem Instance	CPU Time	# of Nodes Explored	CPU Time	# of Nodes Explored
1	9.6	57	10.1	57
2	119.5	587	115.1	587
3	8.3	63	8.4	65
4	26.8	201	26.4	201
5	5.3	45	5.1	41
6	15.6	145	16.3	151
7	10.4	107	11.3	105
8	17.5	113	8.8	57
9	27.4	175	23	155
10	14.7	97	14.8	97
Mean	**25.5**	**159**	**23.9**	**152**
Mean (1-5)	**33.9**	**191**	**33**	**190**
Mean (6-10)	**17.1**	**127**	**14.8**	**113**

Table 4: Effect of the number and placement of tangential supports on CPU time.

Problem	BES							
Instance	(H=4)	(H=5)	(H=10)	(H=20)	(ϵ=0.01)	(ϵ=0.05)	(ϵ=0.1)	(ϵ=0.2)
1	10.1	9.7	11.8	18.5	21.9	13.6	12.2	11.0
2	115.1	143.1	125.8	192.5	161.2	145.7	155.5	289.6
3	8.4	7.5	9.5	14.6	15.9	10.8	9.4	9.6
4	26.4	28.5	34.3	49.2	44.8	35.0	35.2	39.4
5	5.1	5.3	6.2	9.1	8.9	6.0	6.6	7.1
6	16.3	11.0	20.2	30.5	27.5	16.9	16.7	15.9
7	11.3	11.1	13.9	21.1	20.7	14.1	13.2	12.4
8	8.8	6.5	9.8	11.6	15.7	9.3	9.9	10.7
9	23.0	25.0	24.5	40.9	35.2	26.6	22.8	25.6
10	14.8	15.8	21.8	31.7	33.5	18.7	17.5	18.6
Mean	**23.9**	**26.4**	**27.8**	**42**	**38.5**	**29.7**	**29.9**	**44**

Table 5: Two versus four tangential supports for the y_1-, y_2-, and ξ-variables.

	Two tangential supports		BES (H=4)	
Problem instance	CPU time	# of Nodes explored	CPU time	# of Nodes explored
1	9.6	57	10.1	57
2	108.1	587	115.1	587
3	8.3	65	8.4	65
4	26.0	201	26.4	201
5	4.8	41	5.1	41
6	15.6	145	16.3	151
7	10.8	103	11.3	105
8	9.7	61	8.8	57
9	21.4	155	23.0	155
10	15.2	97	14.8	97
Mean	**23**	**151**	**23.9**	**152**

Table 6: Effect of the range reduction strategy.

	With range reduction		Without [1] range reduction		
Problem instance	CPU time	# of Nodes explored	CPU time	# of Nodes explored	Optimality [2] Gap (%)
1	9.6	57	226.4	2000	3.2
2	108.1	587	209.2	2000	10
3	8.3	65	215.8	2000	6.1
4	26.0	201	194.8	2000	4.1
5	4.8	41	212	2000	6.3
6	15.6	145	169.9	2000	1.1
7	10.8	103	169.5	2000	2.7
8	9.7	61	223	2000	0.6
9	21.4	155	202.9	2000	11.8
10	15.2	97	262.5	2000	0.6
Mean	**23**	**151**	**208.6**	**2000**	**4.7**

Table 7: Effect of implementing the proposed valid inequalities VIs.

	With VIs		Without VIs	
Problem instance	CPU time	# of Nodes explored	CPU time	# of Nodes explored
1	9.6	57	7.9	43
2	108.1	587	153.2	587
3	8.3	65	5.7	47
4	26.0	201	14.6	89
5	4.8	41	2.8	21
6	15.6	145	7.5	45
7	10.8	103	11.3	81
8	9.7	61	10.7	53
9	21.4	155	29.3	155
10	15.2	97	21.5	77
Mean	**23**	**151**	**26.4**	**120**

Table 8: Results for BARON using the original and the transformed DTO formulations, versus Algorithm Best.

	Original DTO (5)		Transformed DTO (11) with LM2		Transformed DTO (11) with LM2 and VIs (30)		Algorithm Best	
Problem instance	CPU time	Obj. value (**Opt. Gap (%)**)	CPU time	Obj. value (**Opt. Gap (%)**)	CPU time	Obj. value (**Opt. Gap (%)**)	CPU time	Obj. value
1	11.8	29.00	**1000**	**44.01 (50.2)**	99.5	29.00	9.6	29.00
2	**1000**	**2968.61 (98.8)**	**1000**	**65.49 (60.0)**	**1000**	**67.64 (61.2)**	108.1	64.59
3	**1000**	**38.1 (1.0)**	**1000**	**44.16 (33.8)**	481.7	38.10	8.3	38.10
4	35.0	34.78	**1000**	**48.1 (59.1)**	512.2	34.82	26.0	34.78
5	**1000**	**31.8 (1.2)**	**1000**	**37.69 (25.2)**	56.2	31.80	4.8	31.80
6	**1000**	**39.76 (0.3)**	**1000**	**39.66 (29.5)**	238.7	39.27	15.6	39.24
7	10.5	40.62	**1000**	**42.36 (39.6)**	166.6	40.62	10.8	40.62
8	162.3	33.79	**1000**	**34.0 (27.9)**	125.7	33.84	9.7	33.79
9	212.3	69.83	**1000**	**72.91 (52.2)**	**1000**	**71.22 (49.8)**	21.4	69.82
10	42.4	32.57	**1000**	**35.20 (38.6)**	172.4	32.59	15.2	32.57
Mean	**447**		**1000**		**385.3**		**23**	

Table 9: Results for the BARON and the proposed algorithm for Problem ETO.

	BARON		Algorithm Best	
# of leaf nodes	CPU time	Objective value (**Opt. Gap**(%))	CPU time	Objective value (**Opt. Gap**(%))
2^4+1	3	43.26	24	43.26
	1000	**43.54 (1.4)**	16	43.54
	1000	**61.29 (1)**	9	61.29
	1000	**54.46 (1)**	18	54.46
	1000	**51.6 (0.9)**	47	51.58
Mean	**801**		**23**	
2^5+1	**1000**	**72.92 (2.1)**	32	72.92
	1000	**67.23 (1.8)**	58	67.23
	1000	**58.07 (2)**	50	58.07
	1000	**58.12 (2.4)**	552	58.12
	1000	**54.25 (2.5)**	249	54.25
Mean	**1000**		**188**	
2^6+1	**1000**	**78.26 (2.9)**	**1000**	**78.26 (0.7)**
	1000	**78.72 (3.3)**	347	78.72
	1000	**99.23 (2.5)**	564	99.23
	1000	**79.82 (2.8)**	765	79.82
	1000	**74.10 (3.4)**	**1000**	**74.10 (0.9)**
Mean	**1000**		**736**	