Maximum Probabilistic All-or-Nothing Paths

Noam Goldberg, Michael Poss

Maximum Probabilistic All-or-Nothing Paths

Noam Goldberg* Michael Poss†

December 17, 2019

Abstract

We consider the problem of a maximum probabilistic all-or-nothing network path. Each arc is associated with a profit and a probability and the objective is to select a path with maximum value for the product of probabilities multiplied by the sum of arc profits. The problem can be motivated by applications including serial-system design or subcontracting of key project activities that may fail. When subcontracting such critical success activities, each must be completed on time, according to the specs, and in a satisfactory manner in order for the entire project to be deemed successful. We develop a dynamic programming (DP) method for this problem in the acyclic graph setting, under an independence assumption. Two different fully-polynomial approximation schemes are developed based on the DP formulations, one of which applies repeated rounding and scaling to the input data, while the other uses only rounding. In experiments we compare the DP approach with mixed-integer nonlinear programming (MINLP) using a branch-and-cut method, on synthetic randomly generated instances as well as realistic ones.

Keywords: networks, dynamic programming, integer non-linear programming, FPTAS

1 Introduction

Network optimization problems with an objective that involves a probability of the intersection of independent addtworandom events have been

*Department of Management, Bar-Ilan University, Ramat Gan 5290002, Israel noam.goldberg@biu.ac.il

†LIRMM, University of Montpellier, CNRS, France michael.poss@lirmm.fr

extensively studied. For example, in the reliability literature, some authors propose the objective of minimizing a ratio of cost to reliability (Ahuja, 1988; Katoh, 1992). The reliability of a subgraph of a given network graph is defined as the product of the edge probabilities, each of which corresponds to the probability that the edge remains functional. The closely related area of sequential system testing involves finding a least cost policy, which may correspond to a binary decision tree, of system components to test in order to determine whether the entire system is operational; see Ben-Dov (1981); Boros (1999) as well as Daldal et al. (2016) and the references therein. Similar network optimization problems also arise in the network interdiction literature (Pan and Morton, 2008; Goldberg, 2017) where the model may involve expected-value profit-probability product maximization (rather than minimization of a cost-reliability ratio). Probabilistic *all-or-nothing* expected-value maximization objectives have also been proposed in the kidney exchange game theory literature (Dickerson et al., 2019), where they are termed discounted utility. These problems share the characteristic that value is only derived from a feasible solution (a subgraph such as a network path) if all of the underlying arcs are functional. Our problem extends probabilistic all-or-nothing subsets introduced by Goldberg and Rudolf (2017) to maximize a similar objective over network paths. The all-or-nothing subset problem involves selecting a subset of (unordered) activities such that the sum of activity payoffs multiplied by the product of their success probabilities is maximized.

An important application of our model is in designing series-connected subsystems of the power grid. In particular, in renewable energy generation it is quite common to use series-connected generators, for example in a string inverter layout, many photo-voltaic (PV) panel units may be connected in series; see for example Zhang et al. (2012). Although technical considerations may impose some bounds on the number of units to be connected using a single inverter (e.g., due to voltage requirements), generally a wide range of design and configuration choices may be available. In the subsystems being considered, a failure of one of the units would imply the failure of all units connected to the same inverter. Hence, the objectives of maximizing the subsystem's reliability and maximizing the operating state's power capacity (and/or associated profit) are antagonistic. Compared with bicriteria and single-objective constrained-maximization approaches, which

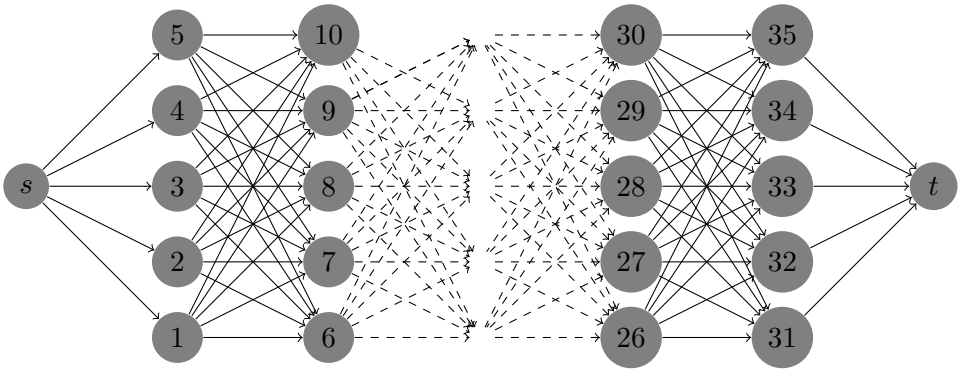

Figure 1: Series-connected power subsystem design. This example may illustrate a string array of 7 generators to be connected in series. In each position a generator is to be chosen given a choice of 5 generator types. Each generator j may correspond to a different power generation capacity c_{sj} and reliability (one minus the probability of failure) p_{sj}. Typically the generation capacity (accordingly revenue) and the failure probability for the same generator, $j = 1, \ldots, 5$, would similar irrespective of the position. So that for all $i = 1, \ldots, 5$, $c_{sj} = c_{i,j+5}$ and $p_{sj} = p_{i,j+5}$.

potentially leave a large set of candidate nondominatred solutions or require an arbitrary specification of a constraint parameter, our model maximizes the subsystem's expected profit, thereby it simultaneously accounts for both the system reliability and its generation capacity. The network corresponding to all feasible series subsystem can be represented as a directed acyclic graph (DAG). An example of a design problem of a certain series-connected power subsystem is illustrated in Figure 1. Note that in general the design network may not be a grid in cases where not all generator choices are available in a particular position and/or if the number of generators to be included is not a-priori fixed.

Our model can also be motivated by the important application of project activity networks. R&D Project activity scheduling when activities may fail has been considered by De Reyck and Leus (2008); Coolen et al. (2014). These papers propose models for scheduling project activities in the case that activities may fail and the overall project success depends on the success of each and every activity. The objective is to maximize the expected profits where each activity incurs a negative payoff (i.e., a cost) and the project completion involves a positive payoff (revenue). In De Reyck and Leus (2008), the overall net present value is considered by time-discounting the

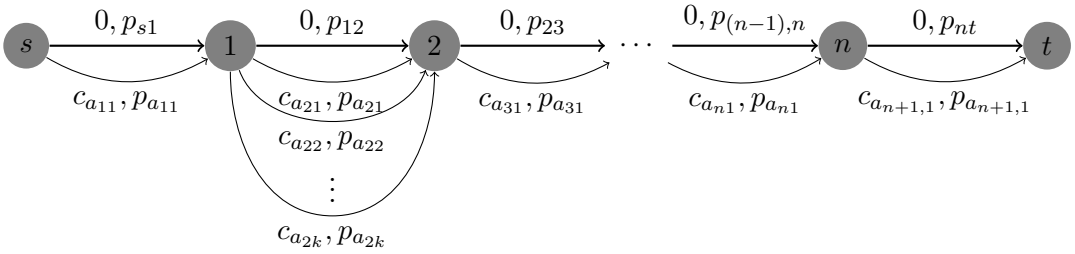

Figure 2: A multigraph that illustrates activity precedence and subcontracting: An activity arc label pair $(c_{a_{ij}}, p_{a_{ij}})$ along arc a_{ij} indicates, respectively, the cost savings relative to a baseline activity arc $(i-1, i)$ (for example, when the activity is carried out in-house) and the probability of successful on-time completion, using subcontractor j. Note that while this example is illustrated as a multigraph, it could be transformed into a directed graph (without parallel edges) using additional dummy vertices and 0-cost, probability-1 arcs).

project revenue and costs, which are associated with each of the project's underlying activities. In Coolen et al. (2014), the activities are partitioned into modules so that all modules must succeed in order for the project to be successful (while not all activities have to succeed in each module for the entire project to succeed).

In many industries such as construction, project activities are typically subcontracted. Complex engineering projects can be under significant pressures to be delivered at the lowest possible cost and consequently, costs are a major concern in procurement and subcontractor selection; see for example Kini (1999); Abbasianjahromi et al. (2014). When identifying the subcontractor to which a particular activity or task should be assigned, both cost and reliability considerations are taken into account. Reliability may be defined as the probability of completing a given task successfully and within an acceptable timeframe. Reliability considerations have been noted as especially important for critical activities (Icmeli-Tukel and Rom, 1997; Kim et al., 2012). Every candidate subcontractor for each activity may be evaluated relative to a safe baseline alternative, for example, the alternative of completing the task in-house or that of using a highly reliable "premium" subcontractor. Both of these alternatives are likely to incur higher costs but with the benefit of greater reliability. A multigraph that represents the critical path subcontracting problem is illustrated in Figure 2.

Assuming that arcs fail independent of one another, the problem that we model can be viewed as an optimization of a nonlinear utility function over Pareto efficient bicriteria optimal paths; the defined utility is the product of the sum of arc profits with the product of arc probabilities. The independence assumption is common in network reliability literature (Ahuja, 1988; Katoh, 1992; Ball et al., 1995) as well in project management literature (see for example Soroush (1994); De Reyck and Leus (2008)). Bicriteria shortest paths have been extensively studied with linear objectives, as have extensions with linear utility functions defined over the set of Pareto optimal paths. For references on approximation algorithms for multicriteria optimization problems, and in particular for shortest paths, see Hansen (1980); Warburton (1987); Henig (1994); Papadimitriou and Yannakakis (2000) and the references therein. A particular variant of a constrained shortest path has also been extensively studied (Hassin, 1992; Ergun et al., 2002; Raz et al., 2003). In particular, these studies have improved on the computational complexity of general schemes for bicriteria shortest path (Hansen, 1980; Warburton, 1987).

In Section 2 we model the probabilistic all-or-nothing path problem. We develop a dynamic program (DP) with a pseudopolynomial worst-case complexity in acyclic graphs in Section 3. Fully-polynomial time approximation schemes (FPTAS) are described in Section 5, which use scaling and/or rounding techniques together with our previously described DPs. Our problem is reformulated as a convex integer program in Section 6. In Section 7 we conduct experiments to compare the DP approaches with integer programming methods. We consider randomly generated instances and also onces that are based on real-life examples of IT and civil engineering projects.

2 Problem Definition

We now formally state the probabilistic all-or-nothing path problem. Let $G = (V, E)$ be a directed graph, $s, t \in V$ be the source and the sink, and let $\delta^+(i) = \{j \in V : (i, j) \in E\}$ and $\delta^-(i) = \{j \in V : (j, i) \in E\}$ be sets of direct successors and predecessors of node $i \in V$, respectively. For each edge $e = (i, j) \in E$, we are given a positive (integer) profit $c_e = c_{ij} \geq 0$ and a probability of success $p_e = p_{ij} \in [0, 1]$. Let $|E| = m$ and $|V| = n$. The all-or-nothing path problem is to find a path π from s to t such that the

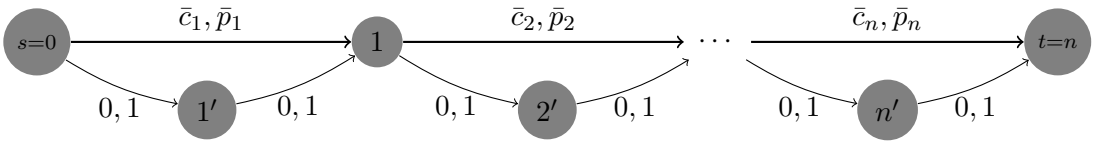

Figure 3: The instance used in the reduction of the all-or-nothing subset problem to the all-or-nothing path problem: Traversing an upper arc in the graph corresponds to selecting an item in the subset problem.

objective function

$$z(\pi) = \sum_{e \in \pi} c_e \prod_{f \in \pi} p_f \tag{1}$$

is maximized. Given any node $i \in V$, let us denote the set of all paths from s to i by $\mathcal{P}(i)$. Then, the optimization problem is to determine

$$z^* \equiv \max\{z(\pi) : \pi \in \mathcal{P}(t)\}. \tag{2}$$

When $p_e = 1$ for each $e \in E$, problem (2) amounts to finding a longest path in G. Hence, in a general graph our problem is $\mathcal{NP}$-hard in the strong sense (Garey and Johnson , 1979, pp. 199–200). We now show through a reduction from the unconstrained all-or-nothing problem that even in Directed Acyclic Graphs (DAGs), the problem is $\mathcal{NP}$-hard (although in the ordinary sense).

Proposition 1. *Problem* (2) *is $\mathcal{NP}$-hard in DAGs with a node in-degree and out-degree of at most 2.*

Proof. Consider an instance of the n-item all-or-nothing subset problem which, given item profits $\bar{c} \in \mathbb{Z}_+^n$ and item probabilities $\bar{p} \in (0,1)^n$, slightly determines a subset $S \subseteq [n]$ that maximizes $\sum_{i \in S} \bar{c}_i \prod_{j \in S} \bar{p}_j$ (over all subsets $S \subseteq [n]$). This problem is proved to be $\mathcal{NP}$-hard in Goldberg and Rudolf (2017). Then, let G be defined as the directed graph with $2n+1$ vertices, as depicted in Figure 3. Let $c_{(i-1)i} = \bar{c}_i$ and $p_{(i-1)i} = \bar{p}_i$ for $i = 1, \ldots, n$, and let $c_{(i-1)i'} = c_{i'i} = 0$ and $p_{(i-1)i'} = p_{i'i} = 1$. It follows that for every $S \subseteq [n]$, there exists a path π in G so that $i \in S$ if and only if $(i-1, i) \in \pi$. Similarly, for every path π in G there exists a subset $S \subseteq [n]$ so that every arc $(i-1, i) \in \pi$ if and only if item $i \in S$. Further, it can be verified that in both cases $\prod_{(i,j) \in \pi} p_{ij} \sum_{(k,l) \in \pi} c_{kl} = \prod_{i \in S} \bar{p}_i \sum_{j \in S} \bar{c}_j$. □

Note that the reduction in the proof of Proposition 1 uses a directed graph

that is equivalent to the path network with parallel edges in Figure 2 with exactly two edges in parallel between each node pair along the chain. Also note that compared with the problem considered in De Reyck and Leus (2008) each item/activity i is associated with a positive payoff c_i. Although, some of our solution techniques may be extended to handle either positive or negative c_i's, compared with only negative payoffs (positive activity costs) in De Reyck and Leus (2008) and a positive project payoff for the last activity. The NP-hardness result of Proposition 1 applies in the case that we consider when all of the c_i's are nonnegative. Finally, note that our objective (1) also resembles that of the minimum cost-to-reliability ratio problems for which negative computational complexity results remain open Ahuja (1988); Katoh (1992).

3 Dynamic Programming

We now develop DP algorithms to solve problem (2).

DP with State Space defined by the Profit Coefficient. Let $Z_c(C, i)$ denote the maximum objective value of a path from s to i with a realized total profit of (exactly) C for the arc profit vector c, so

$$Z_c(C, i) = \max \left\{ z(\pi) : \pi \in \mathcal{P}(i), \sum_{e \in \pi} c_e = C \right\}. \tag{3}$$

Herein, when subscript c is omitted then Z by itself will refer to the given arc coefficients c. Let $\pi(C, i)$ denote a path (a subset of arcs) that reaches the maximum in (3) (for the sake of unambiguity assume that ties are broken by selecting the lexicographically smallest path as a sequences of vertices).

Note that $Z(C, i)$ satisfies the optimality condition

$$Z(C,i) = \begin{cases} \max\limits_{j \in \delta^-(i)} z(\pi(C - c_{ji}, j) \cup \{(j,i)\}) & i \in V \setminus \delta^+(s) \text{ or } c_{si} < C \\ \max \left\{ \begin{array}{c} c_{si} p_{si}, \\ \max\limits_{\substack{j \in \delta^-(i) \setminus \{s\}: \\ c_{ji} \neq C}} z(\pi(C - c_{ji}, j) \cup \{(j,i)\}) \end{array} \right\} & i \in \delta^+(s), C = c_{si} \\ -\infty & C < 0. \end{cases} \tag{4}$$

It can be observed that for $(j,i) \in \pi(C,i)$,

$$z(\pi(C-c_{ji},j) \cup \{(j,i)\}) = C \cdot p_{ji} \prod_{f \in \pi(C-c_{ji},j)} p_f = \begin{cases} C \cdot p_{ji} \frac{Z(C-c_{ji},j)}{C-c_{ji}} & \text{if } C - c_{ji} > 0 \\ C \cdot p_{ji} & \text{otherwise,} \end{cases} \tag{5}$$

where (5) follows directly from the definition of $\pi(C-c_{ji},j)$, $Z(C-c_{ji},j)$ and z (note that the second line of (5) is needed when considering the direct successors of s). Plugging (5) into (4), for $i \in V \setminus \{s\}$, the DP equations are

$$Z(C,i) = \begin{cases} \max\limits_{j \in \delta^-(i)} \left\{ C p_{ji} \frac{Z(C-c_{ji},j)}{C-c_{ji}} \right\} & i \in V \setminus \delta^+(s) \text{ or } c_{si} < C \\ \max \left\{ c_{si} p_{si}, \max\limits_{\substack{j \in \delta^-(i) \setminus \{s\}: \\ c_{ji} \neq C}} \left\{ C p_{ji} \frac{Z(C-c_{ji},j)}{C-c_{ji}} \right\} \right\} & i \in \delta^+(s), C = c_{si} \\ -\infty & C < 0. \end{cases} \tag{6}$$

To determine z^* using the DP method given an upper bound on $\sum_{a \in \pi^*} c_a \leq \bar{C}$, (6) must be evaluated for $C = 1, \ldots, \bar{C}$. In particular

$$z^* = \max_{C=1,\ldots,\bar{C}} Z(C,t). \tag{7}$$

Let C^* denote an optimal solution of (7). A straightforward upper bound $\bar{C} \geq C^*$ in a DAG is $\sum_{a \in E} c_a$. A tighter upper bound, although its use may not affect the worst-case computational complexity of (7), is the longest path from s to t in the graph with arc profits c.

DP with State Space defined by the Probability Coefficient. In this case we first revise our DP formulation such that the product of probabilities is the state variable. Also, we must now assume that the probabilities are rational numbers, so for all $e \in E$, $p_e \in [0,1] \cap \mathbb{Q}$. Let $\widehat{Z}_p(P,i)$ denote the expected total profit of a path from s to i with a probability of (exactly) P given probability vector $p \in [0,1]^{|E|}$. Thus for each $i \in V$

$$\widehat{Z}_p(P,i) = \begin{cases} \max\limits_{j\in\delta^-(i)} \{p_{ji}\widehat{Z}_p(P/p_{ji},j) + P\cdot c_{ji}\} & i \neq s, P \leq 1 \\ 0 & i = s, P = 1 \\ -\infty & i = s, P \neq 1. \end{cases} \tag{8}$$

Since the probabilities are rational, let us further suppose that they are given as $p_e = q_e/d_e$ for $e \in E$, where q_e,$d_e \in \mathbb{Z}$. Then, the DP (8) can be solved in finitely many steps with a maximum expected total profit from node s to node t given by

$$\max_{k=1,\ldots,\prod_{e\in E} q_e} \widehat{Z}_p\left(\frac{k}{\prod_{e\in E} d_e}, t\right). \tag{9}$$

Now suppose that there exists some $b > 0$ such that for each $e \in E$, $p_e = b^{k_e}$ for some $k_e \in \mathbb{Z}$. Then, the next proposition establishes that the running time of DP (8) is polynomial in m, n and $\max_{e\in E} k_e$ under this assumption.

Proposition 2. *If there exists $b > 0$ such that for all $e \in E$, $p_e = b^{k_e}$ for some $k_e \in \mathbb{Z}$, then DP* (8) *can be applied to solve problem* (2) *with a running time complexity of $O(mn \max_{e\in E} k_e)$.*

Proof. The correctness of (8) follows by induction: For all $\pi \in \mathcal{P}(j)$ and $(j,i) \in \delta^+(j)$, $z(\pi \cup \{(j,i)\}) = (z(\pi) + c_{ji})p_{ji}$. The maximizing path to $i \in V$ is found by determining the maximum of $z(\pi \cup \{(j,i)\})$ over the predecessors $j \in \delta^-(i)$. The base case has $\pi = \{(s,i)\}$, $z(\pi) = p_{si}c_{si}$ and trivially $z(\emptyset) = 0$. Then, an optimal solution of (2) is determined by solving

$$\max_{k=0,1,\ldots,n\max_{e\in E} k_e} \widehat{Z}_p\left(b^k, t\right),$$

which has a running time complexity bound of $O(mn \max_{e\in E} k_e)$. □

We consider straightforward bounds on C^* and z^* as well as further tightening of these bounds in the following section.

4 Bounding z^* and C^*

The running time of the DP algorithm (7) directly depends on $\bar{C}$ as an upper bound on C^*. Bounds on the optimal objective value may also be necessary to derive an approximation result. Some of the analysis that follows will be

used to support the approximation results that appear in Section 5, while some will be used for the exact DP method experiments in Section 7.

For convenience, for each $a \in E$ define

$$R(a) \equiv \max_{\pi \in \mathcal{P}(t)} \prod_{e \in \pi} r_e(a), \qquad \text{where} \qquad r_e(a) = \begin{cases} c_e p_e & e = a, \\ p_e & \text{otherwise.} \end{cases}$$

Note that as each π is an elementary path, then arc $a \in A$ appears at most once in a path that is a maximizer in the definition of $R(a)$. Denote such a maximizer by $\bar{\pi}$. It follows that $R(a) = \begin{cases} c_a \prod_{e \in \bar{\pi}} p_e & a \in \bar{\pi}, \\ \prod_{e \in \bar{\pi}} p_e & \textbf{otherwise.} \end{cases}$ The following proposition establishes a lower bound on the optimal objective value.

Proposition 3. *A lower bound on the optimal objective value is given by*

$$z_{LB} \equiv \max_{\pi \in \mathcal{P}(t)} \left\{ \prod_{e \in \pi} p_e \max_{f \in \pi} \{c_f\} \right\} \leq z^*. \tag{10}$$

Further, this bound satisfies $z_{LB} = \max_{a \in E} R(a)$.

It follows that z_{LB} can be computed by solving at most m maximum-reliability path problems, where each provides the value $R(a)$ for a given $a \in E$. In a DAG the maximum-reliability path problem (similar to the shortest path problem) can be solved in $O(m)$ running time using a topological sort algorithm (Cormen, 2009).

Let $H(v)$ denote the maximum number of hops, or the longest elementary path in terms of number of arcs, from s to v. The next proposition establishes an upper bound on the optimal solution value.

Proposition 4. $z^* \leq H(t) \cdot z_{LB}$.

Proof. Let π^* denote a solution that is optimal for (2) and let $\hat{\pi}$ denote a solution that is optimal for (10). By the optimality of $\hat{\pi}$,

$$z_{\text{LB}} = \prod_{e \in \hat{\pi}} p_e \max_{f \in \hat{\pi}} c_f \geq \prod_{e \in \pi^*} p_e \max_{f \in \pi^*} c_f.$$

Since π^* is an elementary path with $|\pi^*| \leq H(t)$,

$$H(t)z_{\text{LB}} \geq \prod_{e\in\pi^*} p_e H(t) \max_{f\in\pi^*} c_f \geq \prod_{e\in\pi^*} p_e \sum_{f\in\pi^*} c_f = z^*.$$

□

For convenience in the following, for a vector $v \in \mathbb{R}^\beta$, let $v_{\max} = \max_{i=1}^{\beta} v_i$ and $v_{\min} = \min_{i=1}^{\beta} v_i$. The bounds on the optimal objective value do not necessarily imply bounds on $C^* \in \text{argmax}_C\{Z(C,t)\}$. However, given a lower bound on the optimal objective value, it may be possible in some cases to tighten the otherwise loose bounds on C^*. The following proposition summarizes the bounds on C^*.

Proposition 5. *Suppose that* $C^* \in \text{argmax}_C\{Z(C,t)\}$. *Then,*

(i) $C^* \in [\min_{\pi\in\mathcal{P}(t)} \sum_{e:e\subseteq\pi} c_e, \max_{\pi\in\mathcal{P}(t)} \sum_{e:e\subseteq\pi} c_e] \subseteq [c_{\min}, H(t)c_{\max}]$.

(ii) $C^* \in \bigcup_{e\in E}[c_e, H(t)c_e]$.

(iii) Let $a^* \in \text{argmax}_a R(a)$. *If for all* $a \neq a^*$ *satisfying* $H(t)R(a) > z_{LB}$, $c_{a^*} \geq c_a$, *then* $[\min_{a:(n-1)R(a)>z_{LB}} c_a, H(t)c_{a^*}] \cap \text{argmax}_C\, Z(C,t) \neq \emptyset$.

Proof.
Proofs of (i) and (ii). Straightforward.
Proof of (iii). Suppose that $c_{a^*} \geq c_a$ for all $a \neq a^*$ such that $H(t)R(a) > z_{\text{LB}}$ and suppose also that for all π^* that are optimal to (2), $C^* = \sum_{a\in\pi^*} c_a > H(t)c_{a^*}$. Let $f \in \text{argmax}_{e\in\pi^*}\{c_e\}$. Then, $(n-1)R(f) \geq (n-1)c_f \prod_{e\in\pi^*} p_e \geq z(\pi^*) > z(\pi) \geq z_{\text{LB}}$. Since $H(t)R(f) > z_{\text{LB}}$, it follows that $c_{a^*} \geq c_f$. But then $H(t)c_{a^*} \geq H(t)c_f \geq C^*$ thereby establishing a contradiction. □

Note that the tightening of the bounds in Proposition 5-(i), in particular the upper bound, relies on the computation of the longest path in a graph and so it can be determined in polynomial time in DAGs. This is while the straightforward upper bound $H(t)c_{\max} \leq (n-1)c_{\max}$ applies also in the case of general graphs. Also, note that although the bound of Proposition 5-(ii) corresponds to m intervals, the advantage of using this bounding set is that each interval has an upper bound that is only a (polynomial) factor of $(n-1)$ times the lower bound.

We now describe a family of (deterministic) instances to illustrate the utility of tightening the bound on C^* based on Proposition 5-(iii).

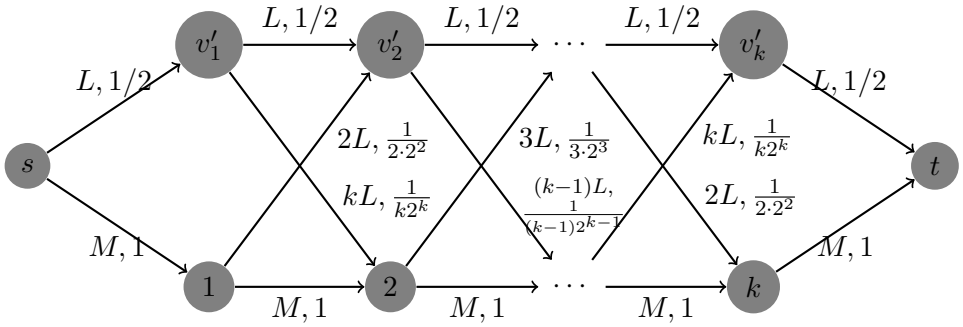

Figure 4: Deterministic instances that illustrate the utility of tightening the bound on C^* using the result of 5-(iii).

Example 1. *In the example of Figure 4 let* $M > 1$ *and* $L = \left\lfloor \frac{(M-1)2^{k+1}}{k+1} \right\rfloor$. *Evidently, path* $\pi^* = \{(s,1),(1,2),\ldots,(k-1,k),(k,t)\}$, $z^* = M(k+1)$ *and* $z_{LB} = M$. *Observe that for each* $a \notin \pi^*$, $R(a) = L/2^{k+1} \leq (M-1)/(k+1) < z_{LB}/(k+1)$. *Then, Proposition 5-(iii) implies that* $C^* \leq (k+1)M << 2L+(k-1)M$ *(where* $2L+(k-1)M$ *is the next longest path). In particular,* $C^* \in O(kM)$ *and for all (exponentially many in* n*) paths other than* π^* *the path length is in* $O(M2^k)$.

Example 1 illustrates an application of Proposition 5-(iii). In practice the tightened bound can be used to make the DP more efficient or, perhaps, to preprocess a given graph more effectively by deleting arcs that cannot be a part of an optimal path. In particular, in Example 1 the graph can be reduced to the single path $\pi^* = \{(s,1),(1,2),\ldots,(k,t)\}$.

5 Approximation Schemes

We now consider two different approximation schemes. First, a scaling and rounding scheme is applied to the c coefficients.

Profit coefficient scaling and rounding scheme. Consider Algorithm 1 as an approximation scheme given a relative error bound $0 < \epsilon < 1$ and an interval $[\ell, u]$ so that there exists a $C^* \in [\ell, u]$ where $C^* \in \operatorname{argmax}_C Z(C,t)$. Following the discussion of the previous section, a straightforward ratio of bounds is given by $\frac{u}{\ell} \in O(nc_{\max}/c_{\min})$. The computational complexity of our algorithm will depend on the magnitude of this ratio. Thus, we will also

consider the repeated application of the algorithm when given a union of intervals of polynomially-bounded upper to lower bound ratios.

Algorithm 1 Profit Rounding and Scaling Approximation Scheme

Input: $G = (V, E), p, c, 0 < \epsilon < 1, \ell, u$ {Bounds ℓ, u satisfy $\ell \leq C^* \leq u$.}

1: $a^* \leftarrow \text{argmax}_{a \in E} R(a)$
2: **while** $q = 0, \ldots, \lceil \log u - \log \ell \rceil - 1$ **do**
3: $K \leftarrow \frac{\epsilon 2^q \ell}{n-1}$
4: $\bar{c}_e \leftarrow \begin{cases} -\infty & c_e > 2^{q+1}\ell \\ \lfloor c_e / K \rfloor & \text{otherwise} \end{cases}$
5: $\bar{z}_q \leftarrow K \cdot \max_{C=0,\ldots,\lfloor \frac{2(n-1)}{\epsilon} \rfloor} Z_{\bar{c}}(C, t)$; let $\pi_q = \pi(C, t)$ be the corresponding path.
6: **end while**
7: Select $q^* \in \text{argmax}\{z_q\}$

Output: $z(\pi_q)$

Following Algorithm 1, we define the scaled objective as

$$z_{\bar{c}}(\pi) = \sum_{e \in \pi} \bar{c}_e \prod_{f \in \pi} p_f,$$

and the scaled problem is $\max\{z_{\bar{c}}(\pi) : \pi \in \mathcal{P}(t)\}$. The next lemma establishes an approximation relation between solutions of the scaled problem when $\ell 2^q \leq \frac{z^*}{\prod_{e \in \pi^*} p_e} \leq \ell 2^{q+1}$ in the main loop of Algorithm 1 and solutions that are optimal to (2).

Lemma 1. *If $K \leq \frac{\epsilon L}{(n-1)}$ where $L \leq \frac{z^*}{\prod_{e \in \pi^*} p_e}$, then $\bar{\pi} \equiv \pi_q$ in the main loop of Algorithm 1 satisfies*

$$\sum_{e \in \bar{\pi}} c_e \prod_{f \in \bar{\pi}} p_f \geq (1 - \epsilon) z^*.$$

Proof. The result follows from the optimality of $\bar{\pi}$ for the rounded profits $\bar{\mathbf{c}}$

and the assumption on K, specifically

$$\begin{aligned}
\sum_{e\in\bar{\pi}} c_e \prod_{f\in\bar{\pi}} p_f &\geq K z_{\overline{c}}(\bar{\pi}) && \text{(Rounding of } \overline{\mathbf{c}}\text{)} \\
&\geq \sum_{e\in\pi^*} \lfloor c_e/K \rfloor K \prod_{f\in\pi^*} p_f && \text{(Optimality of } \bar{\pi} \text{ for } \overline{\mathbf{c}}\text{)} \\
&\geq (\sum_{e\in\pi^*} c_e - (n-1)K) \prod_{f\in\pi^*} p_f && \text{(Rounding of } \overline{\mathbf{c}}\text{)} \\
&\geq (1-\epsilon) z^* && \text{(Assumed upper bound on } K\text{)}
\end{aligned}$$

□

Intuitively, Lemma 1 shows that the desired approximation guarantee is attained as long as the scaling factor K is sufficiently small. It remains to show that such a "sufficiently small" scaling factor also allows for a polynomial running time complexity bound. The correctness and complexity bound of Algorithm 1 are established in the following proposition.

Proposition 6. *The output z of Algorithm 1 satisfies $z \geq (1-\epsilon)z^*$. The complexity of Algorithm 1 is $O(\frac{1}{\epsilon} mn \log(u/\ell))$.*

Proof. The first part of the claim is evident from Lemma 1 and from the fact that the scaled DP is run over an interval that contains a maximizer C^*. In particular, the main loop goes through $q = 0, \ldots, \lceil \log u - \log l \rceil - 1$ and the intervals $[2^q\ell, 2^{q+1}\ell]$ partition the interval $[\ell, u]$. Each maximization using the DP with scaled profits in line 5 has a complexity bound of $O(mn/\epsilon)$, while the number of iterations of the main loop is bounded by $O(\log(u/\ell))$. Together, these bounds imply an overall complexity bound of $O(\frac{1}{\epsilon} mn \log(u/\ell))$. □

Corollary 1. *Algorithm 1 invoked with $\ell = c_{\min}, u = nc_{\max}$ outputs a solution with objective $z \geq (1-\epsilon)z^*$. The complexity of Algorithm 1 is $O(\frac{1}{\epsilon} mn \log(nc_{\max}/c_{\min}))$.*

Observe that Proposition 6 establishes a polynomial complexity bound for Algorithm 1, although it is not strongly polynomial. The following proposition establishes a strongly polynomial time complexity at the cost of repeatedly invoking Algorithm 1 (m times).

Proposition 7. *Algorithm 1 can be invoked repeatedly to output a solution with objective $z \geq (1-\epsilon)z^*$ with a complexity bound of $O(\frac{1}{\epsilon}m^2 n \log n)$.*

Proof. By invoking Algorithm 1 once for each arc $a \in E$, with $\ell = c_a$ and $u = (n-1)c_a$, and outputting the best solution value. This approach uses the straightforward observation that $C^* \in \bigcup_{e\in E}[c_e, (n-1)c_e]$ (see also Proposition 5-(ii). So, the analysis of the complexity bound is similar to that in the proof of Proposition 6 except that the algorithm is invoked m times and each invocation now involves $O(\log(u/\ell)) = O(\log n)$ iterations. □

Note that for every π^* that is optimal to (2), the pair $(C^*, P^*) \equiv (\sum_{e\in\pi^*} c_e, \prod_{e\in\pi^*} p_e)$ must be Pareto efficient: for all $\pi \in \mathcal{P}(t)$, if $PC \neq P^*C^*$ then either $C = \sum_{e\in\pi} c_e < C^*$ or $P = \prod_{e\in\pi} p_e < P^*$. It follows that approximation schemes for multicriteria shortest path problems could be adapted to determine an optimal solution for our problem. Hansen (1980) and Warburton (1987) developed a fully-polynomial time approximation scheme (FPTAS) for bicriteria and constant-number multicriteria path problems, respectively.

More efficient approximation schemes were developed by Hassin (1992) for the related problem of the (linear) constrained shortest path. In contrast to the situation in the current problem, in the constrained shortest path case, it is possible to binary search a bounded interval of the logarithm of objective values. Subsequent improvements to Hassin's scaling-based approximation scheme for constrained shortest paths were developed by Lorenz and Raz (2001) and Ergun et al. (2002).

Probability-rounding scheme. We now consider a rounding scheme (without scaling) that is applied to the probability coefficients. Proposition 2 shows that under the assumption that the probabilities are powers of a common base, then the exact solution can be determined using DP (8) in polynomial time. In the absence of this assumption, we develop an approximation scheme that further exploits the result of Proposition 2. This rounding based scheme is given by Algorithm 2.

For convenience let $z_p(\pi)$ denote the objective value of path π (similar to $\widehat{Z}_p$) with probability vector p. The following proposition establishes that Algorithm 2 is an FPTAS.

Algorithm 2 Probability Rounding Approximation Scheme

Input: $G = (V, E), \mathbf{p}, \mathbf{c}, \epsilon$

1: For each $e \in E$, let $p'_e = (1-\epsilon)^{\lceil \log_{(1-\epsilon)^{1/n}} p_e \rceil / n}$.

2: $z' = \max_{k=1,\ldots,n\lceil \log_{(1-\epsilon)^{1/n}}(p_{\min})\rceil} \widehat{Z}_{\mathbf{p}'}\left((1-\epsilon)^{k/n}, t\right)$. Let π be the corresponding path.

Output: $z(\pi)$

Proposition 8. *Algorithm 2 computes a solution for* (2) *with an objective at least* $(1-\epsilon)z^*$ *in time* $O(-\frac{1}{\epsilon}mn^2 \log(p_{\min}))$.

Proof. Consider π, which is the output of Algorithm 2, and π^*, which is optimal to (2). By (i) the definition of p', (ii) the maximum number of arcs that form an elementary path and (iii) the optimality of π and π^* for their respective problems, it follows that

$$z_p(\pi) \geq z'(\pi) \geq z'(\pi^*) \geq \prod_{e \in \pi^*} p_e (1-\epsilon)^{1/n} \sum_{f \in \pi^*} c_f \geq (1-\epsilon) z(\pi^*).$$

The complexity of the algorithm (recalling that for positive x, $\log x \leq x - 1$) is

$$O(-m \sum_{f \in E} \log_{(1-\epsilon)^{1/n}} p_f) \subseteq O\left(-\frac{1}{\epsilon} mn^2 \log(p_{\min})\right). \quad \square$$

Discussion. To summarize the approximation results of this section, our strongly polynomial profit scaling and rounding based FPTAS has a complexity of $O(\frac{1}{\epsilon}m^2 n \log n)$. Our alternative, simple probability rounding based FPTAS has a runtime complexity of $O(-\frac{1}{\epsilon}mn^2 \ln p_{\min})$. The latter may be preferable if the probabilities are sufficiently large with respect to n, in particular, if $p_{\min} \in \Omega(e^{-n \log n})$. Also, note that using the directed graph reduction in the proof of Proposition 1, the all-or-nothing subset problem can be solved using our methods with $m \in O(n)$. This implies a strongly polynomial runtime complexity of $O(\frac{1}{\epsilon}n^3 \log n)$ to solve the subset problem, which outperforms the $O(n^4/\epsilon)$ runtime complexity of the corresponding algorithm presented by Goldberg and Rudolf (2017).

6 A Convex Mixed-Integer Nonlinear Program (MINLP) Formulation

In this section we develop a MINLP model and a specialized MINLP method. The latter will serve as a benchmark for empirically evaluating the efficiency of the DP as an exact solution technique in the following section. Introducing decision variables $\mathbf{x} \in \{0,1\}^m$ to indicate which arcs are selected to form a path, and accordingly the set of feasible paths

$$\Pi \equiv \left\{ \mathbf{x} \in \mathbb{R}^m \;\middle|\; \begin{array}{ll} \sum_{(j,i)\in E} x_{ji} = \sum_{(i,j)\in E} x_{ij}, & i \in V \setminus \{s,t\} \\ \sum_{(s,i)\in E} x_{si} = 1, & \sum_{(j,t)\in E} x_{jt} = 1 \end{array} \right\},$$

the problem can be formulated as a MINLP. Following Goldberg and Rudolf (2017) we rewrite $z(\pi)$ as

$$z(\mathbf{x}) = \sum_{e\in E} c_e x_e \prod_{f\in E} p_f^{x_f}. \tag{11}$$

Further, maximizing $z(\mathbf{x})$ is equivalent to maximizing its logarithm, so (11) can be replaced by the concave objective function

$$\ln(z(\mathbf{x})) = \ln\left(\sum_{e\in E} c_e x_e\right) + \sum_{f\in E} \ln(p_f) x_f. \tag{12}$$

Then the derived MINLP formulation is

$$\max\left\{ \ln(z(\mathbf{x})) \;\middle|\; \mathbf{x} \in \Pi \cap \{0,1\}^m \right\}. \tag{13}$$

It can be observed that as a concave maximization problem with linear constraints, this formulation is a convex MINLP. In Section 7 we conduct experiments using this formulation and treat it as a benchmark for our proposed computational techniques.

To solve this model using state-of-the art linear integer programming solvers, we implement our own MINLP linearization and objective-based cutting plane techniques. While our convex MINLP method implementation is fairly standard, it specializes the general techniques for our problem. In particular, following the concavity of $\ln(z(\mathbf{x}))$, subgradient objective cuts can be derived for problem formulation (13). For any $\bar{\mathbf{x}}$ that is feasible for

formulation (13), the following inequality

$$\begin{aligned}\ln(z(\mathbf{x})) &\leq \ln(z(\bar{\mathbf{x}})) + \nabla \ln z(\bar{\mathbf{x}})^T(\mathbf{x}-\bar{\mathbf{x}}) \\ &= \ln(z(\bar{\mathbf{x}})) + \left[\frac{1}{\sum_{e\in E} c_e\bar{x}_e}c + \ln(p)\right]^T(\mathbf{x}-\bar{\mathbf{x}}) \Leftrightarrow \\ \ln\left(\sum_{e\in E} c_e x_e\right) &\leq \ln\left(\sum_{e\in E} c_e\bar{x}_e\right) + \frac{\sum_{e\in E} c_e x_e}{\sum_{e\in E} c_e\bar{x}_e} - 1\end{aligned}$$

is valid for the set of optimal solutions of (11). For convenience define, for $\mathbf{x}, \bar{\mathbf{x}} \in \Pi$ and $0 \leq t \leq \ln\left(\sum_{e\in E} c_e\right)$,

$$g(\mathbf{x}, \bar{\mathbf{x}}, u) \equiv u - \frac{\sum_{e\in E} c_e x_e}{\sum_{e\in E} c_e\bar{x}_e} - \ln\left(\sum_{e\in E} c_e\bar{x}_e\right) + 1 \leq 0. \tag{14}$$

Now consider a linearized subproblem given a set of inequalities (14) corresponding to a (finite) set of solutions $\Pi^0 \subset \Pi \cup \{\mathbb{1}\}$, also known as the relaxed master problem in the literature on decomposition algorithms.

$$\begin{aligned}
\max_{\mathbf{x},u} \quad & u + \sum_{e\in E} \ln(p_e)x_e && && (15a)\\
\text{subject to} \quad & \mathbf{x} \in \Pi && && (15b)\\
& g(\mathbf{x}, \bar{\mathbf{x}}, u) \leq 0 && \bar{\mathbf{x}} \in \Pi^0 && (15c)\\
& \mathbf{x} \in \{0,1\}^m. && && (15d)
\end{aligned}$$

For a solution (x, u) that is feasible to (15), Algorithm 3 dynamically generates the cuts (14) required to solve (13) as a sequence of mixed-integer linear programs (MILPs) (15) to within a given relative-optimality gap $\gamma \in [0, 1)$.

Note that although Algorithm 3 solves a sequence of MILPs, where each MILP may be solved to optimality, it can instead be implemented such that step 3 outputs any upper bound. Indeed, an approach that is usually more efficient in practice is to generate the inequalities (15c) dynamically and to "hot-start" the previous iteration's branch-and-bound tree in step 3 of the algorithm, or equivalently, to generate the inequalities (15c) as a part of a branch-and-cut algorithm. We conduct experiments using a branch-and-cut

Algorithm 3

Input: $G = (V, E)$, $\mathbf{c}$, $\mathbf{p}$, $0 \leq \gamma < 1$

$\Pi^0 \leftarrow \{\mathbb{1}\}$
while True **do**
  Solve (15) to determine a feasible solution $\bar{\mathbf{x}}, u$.
  if $\ln z(\bar{\mathbf{x}}) \geq (1-\gamma)\left(u + \sum_{e \in E} \ln(p_e) x_e\right)$ **then**
    break.
  else
    $\Pi^0 \leftarrow \Pi^0 \cup \{\bar{\mathbf{x}}\}$
  end if
end while

Output: $z(\bar{\mathbf{x}})$

approach in Section 7.

7 Computational Results

In this experimental analysis, we compare our MINLP method (Algorithm 3) and DP (7) with a standard MINLP solver for solving problem (13), Bonmin. Algorithm 3 was implemented using Gurobi and was based on its cut generation routine. The next subsection contains further implementation details, while the following subsections detail our experiments using three sets of instances: random graph instances, specific types of graphs, and realistic instances based on project management.

7.1 Implementation details

All experiments were carried out on a CPU Intel i7-5500 3.0 GHz processor under the Ubuntu 16.04 operating system. Our algorithms were coded in Julia 5.1 using the package JuMP (Dunning et al., 2017) to interact with the MIP solvers. The MINLP (13) was solved in two different ways: using Bonmin 1.8.4 (with MUMPS as the linear solver) and using a branch-and-cut algorithm that generates the cuts (14) at the integer nodes found along the tree and interacts with Gurobi 7.02 through callbacks. It is essentially similar to Algorithm 3 (setting $\gamma = 10^{-4}$) but using Gurobi's callback facility to incorporate the generation of cuts within its branch-and-cut implementation.

7.2 General Random Graph Instances

We consider below two sets of random instances generated uniformly as topologically ordered DAGs, which means that for an ordering $1, ..., n \in V$, all edges $(i, j) \in V \times V$ with $i < j$ have the same probability of appearing in each graph. Furthermore, we consider $s = 1$ and $t = n$ and ensure that there exists at least one path between s and t (otherwise the current graph is discarded and another one is generated). Similar to the random generation of knapsack instances (see for example Pisinger (1997)), as well as other constrained combinatorial optimization problems, we find that the degree of correlation between item profits and constraint coefficients may heavily affect the observed difficulty in solving the generated instances. Thus, we consider two different schemes for generating the probability coefficients (analogous to budget constraint coefficients in constrained settings):

Probabilities generated uniformly at random (independent of profits). All graphs are generated given the following parameters: the number of vertices, the graph density, and the maximum edge profit. The number of vertices n ranges in $\{50, 100, 150, 200, 250, 300, 350, 400\}$, edge densities in $\{0.1, 0.2, 0.3, 0.4, 0.5\}$, maximum profit $c_{\max} \in \{50, 100, 150, 200\}$, and minimum probability $p_{\min} \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. The cost c_e for each edge e is an integer uniformly generated between 1 and $c_{\max}$ and the probability p_e is uniformly generated in $[p_{\min}, 1]$.

Probabilities inversely related with profits. For the second set of instances, we focus on smaller graphs ($n = 50$) with $c_{\max}$ ranging in $\{50, 200, 400, 600, 800, 1000\}$. The cost c_e is again uniformly generated between 1 and $c_{\max}$ while the probability is now given by the relation $p_e = e^{-\alpha \frac{c_e}{c_{\max}}}$ where α ranges in $\{1, 10, 20, 30\}$.

We now examine the results of our experiments. Figure 5 reports the average solution times of the five implementations on acyclic graphs: MINLP using Gurobi, MINLP using Bonmin, and DP using the following three bounds on C^* used. The first bound computes the longest path from s to each node using lengths $\bar{c}$ (local UB), the second bound only computes the longest path from s to t (global UB), and the third bound applies Proposition 5-(iii). We limited our experiments with Bonmin to instances with $n = 100$ nodes, as larger instances took an excessive length of time. Clearly,

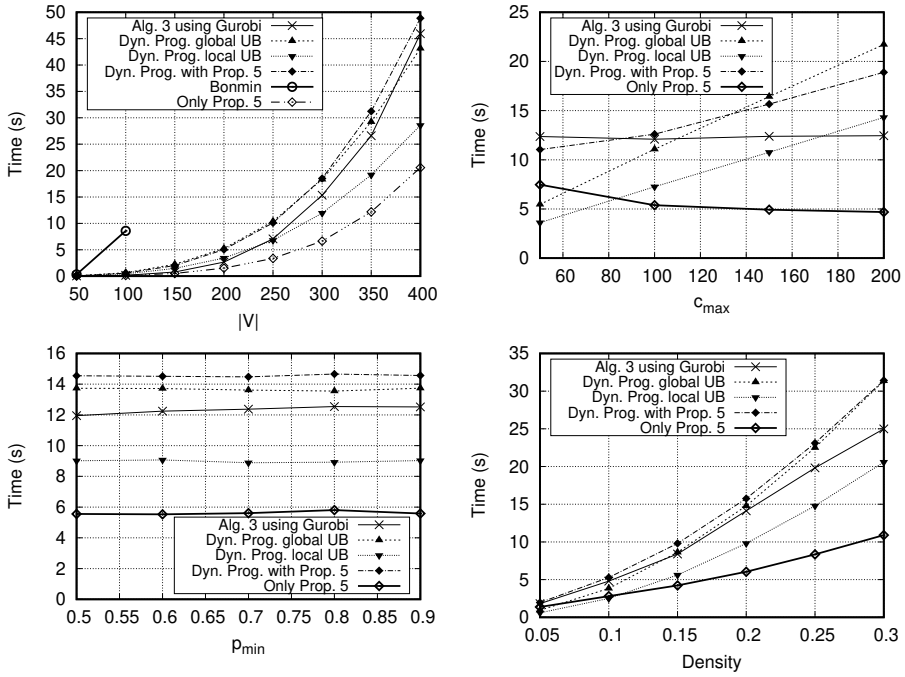

Figure 5: Non-correlated probabilities: A comparison of solution times for the case where probabilities are independent of profits.

as $|V|$ increases, Bonmin is outperformed by the other algorithms by orders of magnitude. We also see that, for small values of $c_{\max}$, the dynamic programming algorithms are faster than MINLP (Algorithm 3 using Gurobi). When $c_{\max}$ increases, the DP algorithms slow down while the MINLP is barely affected. In fact, the MINLP is fastest for $c_{\max} \geq 150$. The most efficient variant of the DP algorithm is the one that uses a local upper bound based on the longest path value at each graph vertex. It is evident that, for these instances, the running time of computing the bound of Proposition 5-(iii) on C^* is not justified by the actual bound reduction. The inefficiency of Proposition 5-(iii) for these instances is further illustrated on In fact, when examining these experiments it appeared the upper bound computed using Proposition 5-(iii) provided no reduction at all on these instances.

Figure 6 reports the average solution times for the same algorithms, with the exception of Bonmin, on the second group of instances. These show that Proposition 5 now provides a worthwhile time reduction, especially when $\alpha \geq 20$. This is in-line with the bounds reported in Figure 7, where we report the averages ratios C_i/C^* for $i \in \{1, 2\}$ where C_1 is the upper

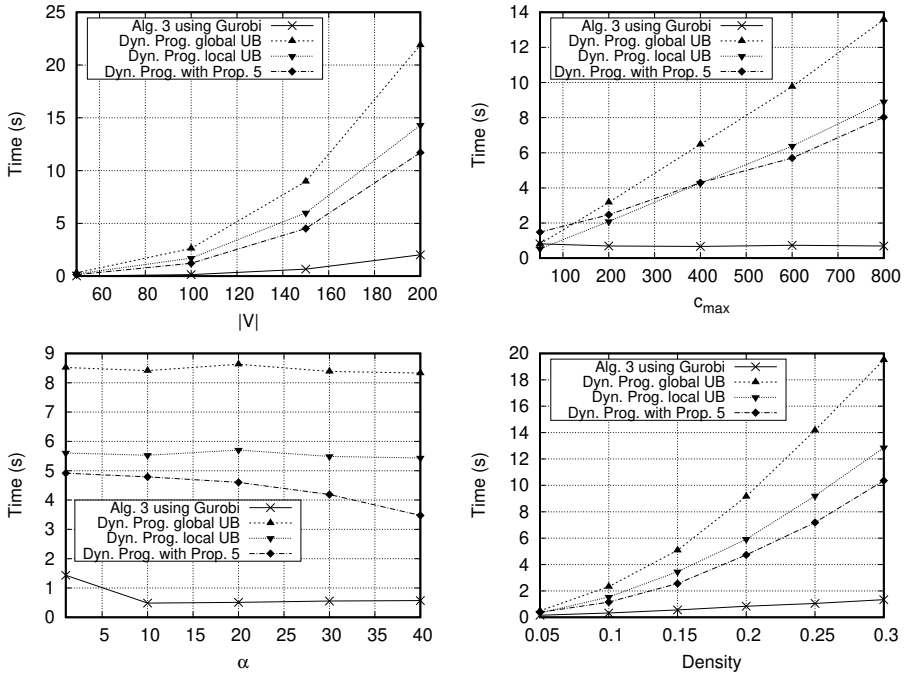

Figure 6: Correlated probabilities: A comparison of solution times for the case where probabilities are set to be inversely related to the profits.

bound provided by the aforementioned longest path given arc length vector c (referred to as global UB in Figure 6), and $C_2 = H(t)c_{a^*}$ with a^* defined in Proposition 5-(iii) (referred to Prop 5. in Figure 6). It also appears the greatest improvement of the bound computed based on Proposition 5-(iii) is obtained with "small" correlated probability values corresponding to a large value of α. At the same time, it should be noted that for these instances Gurobi is much faster than the dynamic programming algorithms, which is explained by the larger bounds on C^* used, which is consistent with the large ratios indicated by Figure 7 (in the experiments of Figure 5 with non-correlated instances the corresponding ratios appeared to be significantly less than 400%).

7.3 DAGs to Further Illustrate the Effect of the Tightening of Bounds on C^*

The results of Section 7.2 for random-probability graphs appear to indicate that the bound on C^* based on Proposition 5-(iii) reduces the overall effi-

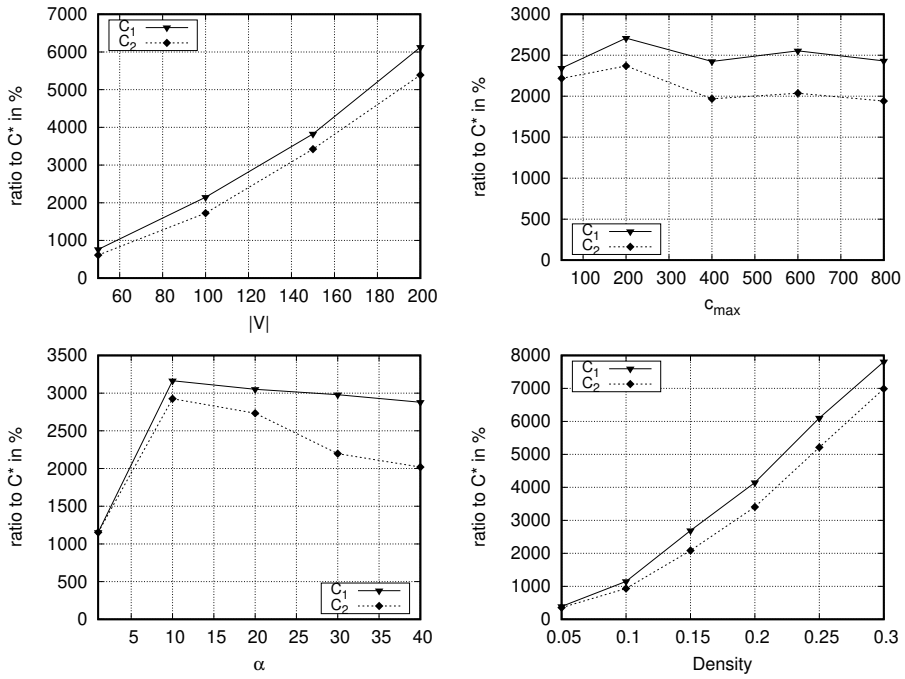

Figure 7: Correlated probabilities: A comparison of bounds on C^* for the case where probabilities are set to be inversely related to the profits.

ciency of the DP algorithm (it has a marginally positive effect in the case of the inversely related probabilities). This is partly due to the running time of computing the bound, which appears to increase in the number of vertices and graph density. Figure 8 shows an example graph with edge probabilities that are inversely related to the edge profits. One optimal path has $z^* = C^* = 2$ and all other paths π are optimal with $\sum_{e\in\pi} c_e = 2M$, which can be orders of magnitude larger than C^*. In particular, experiments with $2000 \leq M \leq 8000$ are shown in Figure 9. Here the DP algorithm (7) with $\bar{C}$ based on Proposition 5-(iii) yields the lowest running time for all values of $|V|$ and M considered. The DP with the local upper bounds on $\sum_{e\in\pi} c_e$, with $\pi \in \mathcal{P}(v)$ for each $v \in V$, performs second best among the DP variants. It is outperformed by the branch-and-cut algorithm (Algorithm 3 using Gurobi) for small values of $|V|$ but it performs much better than the branch-and-cut method as $|V|$ grows larger. The bounds on C^*, illustrated on Figure 10, are in line with the solution times observed in Figure 9.

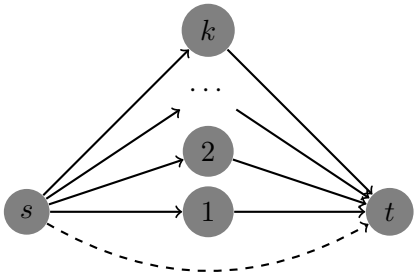

Figure 8: Deterministic instances that illustrate the utility of tightening the bound on C^* using the result of 5-(iii). The dashed arc has a profit of 2 and a probability of 1 while the solid arcs each have a profit of M and probability of $1/\sqrt{M}$.

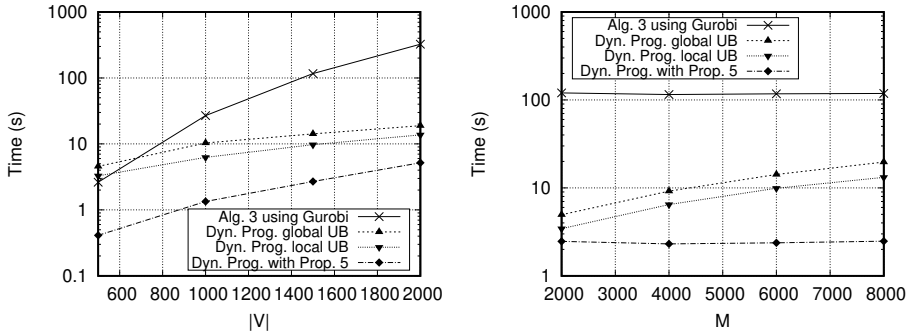

Figure 9: A comparison of the running times using the deterministic graphs from Figure 8.

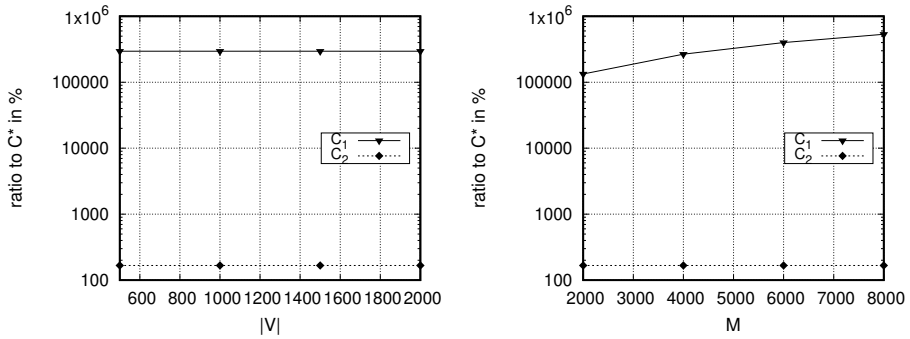

Figure 10: Non-correlated probabilities: A comparison of bounds on C^* using the deterministic graphs from Figure 8.

7.4 Real Project Critical Path Experiments

In this subsection we perform experiments using the critical paths of real projects appearing in Batselier and Vanhoucke (2015). In particular, for each project, an activity-on-arc network is created in order to determine the critical path given the planned activity durations. Then, for each arc a on the critical path, an arc a_0 is created in the subcontracting chain graph with $c_{a_0} = 0$ and $p_{a_0} = p_{\max} = 0.99$. In addition, $s \in \{5, 10, 20, 50, 100\}$ "parallel arcs" are created, with $c_{a_1}, \ldots, c_{a_s}$, each joined with a dummy vertex and additional arc with cost 0 and probability 1, which correspond to the alternatives to subcontracting activity a (similar to the multigraph in Figure 2 but converted into an equivalent graph). For $i = 1, \ldots, s$, c_{a_i} is assigned $r/2$ of the total budgeted cost of activity a_i, where r is generated uniformly at random from $[0, 1]$. Then, for a given $\alpha > 0$, the success probability of subcontract arc a_i is set to $p_{a_i} = p_{\max} e^{-\alpha r}$.

Based on the results with random probabilities in Section 7.2, it is evident that the MINLP algorithm outperforms the DP method for all but negligible values of the cost parameter. In the current experiments, we ran MINLP algorithm 3. The results are shown in Tables 1 and 2. For each experiment with different values of the parameters s (accordingly $|V|$) and α, the results of the running time, the optimal solution probability and the objective (expected) value are indicated. The number of subcontracted activities is also reported. The results confirm the running time efficiency of the MINLP algorithm in practice for actual project network instances. It can be observed overall that the number of activities that are subcontracted, as well as the probability P^* and the objective value of the optimal solution, increase as α decreases. Further, as would be expected, for a given value of α, as s (the number of subcontracting alternatives) increases, the optimal objective value also tends to increase. Perhaps less intuitive is the finding that P^* tends to decrease rather than increase under the same circumstances. Therefore, it appears that a greater variety of subcontracting alternatives results in somewhat riskier optimal solutions with higher returns to justify the increased risk.

Table 1: Project subcontracting instances based on the IT-project critical-path data of Batselier and Vanhoucke (2015). The network was generated based on that project's original 22-arc critical path. Running times in seconds are indicated for solving the instances using the MINLP Algorithm 3.

s	$\|V\|$	α	Time	P^*	z^*	Sub #
5	133	0.6	0.0	0.49	1205.31	3
10	243	0.6	0.0	0.45	1324.04	2
20	463	0.6	2.0	0.45	1320.96	2
50	1123	0.6	18.0	0.45	1323.76	2
5	133	0.4	0.0	0.49	1454.29	3
10	243	0.4	0.0	0.46	1638.98	3
20	463	0.4	2.0	0.46	1591.61	3
50	1123	0.4	27.0	0.45	1657.44	3
s	$\|V\|$	α	Time	P^*	z^*	Sub #
5	133	0.2	0.0	0.53	2134.87	5
10	243	0.2	0.0	0.51	2285.07	5
20	463	0.2	2.0	0.51	2279.60	5
50	1123	0.2	32.0	0.50	2306.15	5
5	133	0.1	2.0	0.62	2875.47	6
10	243	0.1	1.0	0.61	2938.63	6
20	463	0.1	2.0	0.61	2943.30	6
50	1123	0.1	31.0	0.60	2995.01	6
5	133	0.05	0.0	0.68	3161.81	8
10	243	0.05	1.0	0.67	3361.73	8
20	463	0.05	2.0	0.67	3320.29	8
50	1123	0.05	22.0	0.67	3525.86	8

Table 2: Project subcontracting instances based on the Tappan-Zee Bridge critical-path data of Batselier and Vanhoucke (2015). The network was generated based on the 9-arc critical path of that project. Running times in seconds are indicated for solving the instances using the MINLP Algorithm 3.

s	$\|V\|$	α	Time	P^*	z^*	Sub #
5	55	0.6	0.0	0.57	3.40E8	2
10	100	0.6	0.0	0.52	3.63E8	2
20	190	0.6	0.0	0.53	3.59E8	2
50	460	0.6	2.0	0.51	3.70E8	2
100	910	0.6	9.0	0.51	3.71E8	2
5	55	0.4	0.0	0.66	4.24E8	2
10	100	0.4	0.0	0.63	4.49E8	2
20	190	0.4	0.0	0.63	4.33E8	2
50	460	0.4	2.0	0.62	4.51E8	2
100	910	0.4	9.0	0.62	4.51E8	2
s	$\|V\|$	α	Time	P^*	z^*	Sub #
5	55	0.2	0.0	0.72	5.14E8	3
10	100	0.2	0.0	0.70	5.51E8	3
20	190	0.2	0.0	0.70	5.69E8	3
50	460	0.2	2.0	0.69	5.60E8	3
100	910	0.2	12.0	0.68	5.73E8	3
5	55	0.1	0.0	0.72	6.56E8	6
10	100	0.1	0.0	0.71	6.62E8	6
20	190	0.1	0.0	0.69	6.81E8	6
50	460	0.1	2.0	0.69	6.81E8	6
100	910	0.1	10.0	0.69	6.89E8	6
5	55	0.05	0.0	0.82	7.12E8	6
10	100	0.05	0.0	0.81	7.07E8	6
20	190	0.05	0.0	0.80	7.71E8	6
50	460	0.05	2.0	0.80	7.88E8	6
100	910	0.05	10.0	0.80	8.00E8	6

8 Conclusion

In this paper we introduced a new probabilistic all-or-nothing path problem motivated by critical-path analysis and procurement applications. We developed the analysis and a suite of solution techniques to address this problem: approximation schemes with theoretical guarantees with respect to the quality of the solutions that are determined, as well as practical computational approaches. In our computational experiments we found that our specialized MINLP algorithm was most effective in general, in particular for large values of $c_{\max}$, although the DP method, including its variants that compute tightened state space parameter bounds, outperformed the MINLP algorithm in some cases. We showed that the MINLP method applied to networks constructed based on critical paths of real projects runs within less than a minute of CPU time. The observed optimal solutions show that increasing the variety of task subcontracting alternatives may result in somewhat riskier optimal solutions but with sufficiently higher profits to justify this risk.

In ongoing work we consider solution methods for a partial success model similar to the one considered in Dickerson et al. (2019), in which functional subpaths of a network path solution also yield a partial value. This extension may be useful for additional applications such as for the problem of determining optimal transplant-donation chains in kidney exchange compatibility graphs.

References

Hamidreza Abbasianjahromi, Hossein Rajaie, Eghbal Shakeri, and Farzad Chokan. A new decision making model for subcontractor selection and its order allocation. *Project Management Journal*, 45:55–66, 2014. ISSN 8756-9728. doi: 10.1002/pmj.21394.

Ravindra K Ahuja. Minimum cost-reliability ratio path problem. *Computers & operations research*, 15(1):83–89, 1988.

Michael O. Ball, Charles J. Colbourn, and J. Scott Provan. *Network reliability*, chapter 11, pages 673–762. Elsevier Science B.V., 1995. doi: 10.1016/s0927-0507(05)80128-8.

Jordy Batselier and Mario Vanhoucke. Construction and evaluation framework for a real-life project database. *International Journal of Project Management*, 33:697–710, 2015. ISSN 0263-7863. doi: 10.1016/j.ijproman.2014.09.004.

Yosi Ben-Dov. Optimal testing procedures for special structures of coherent systems. *Management Science*, 27(12):1410–1420, 1981.

Endre Boros and Tonguç Ünlüyurt. Diagnosing double regular systems. *Annals of Mathematics and Artificial Intelligence*, 26:171–191, 1999.

Kris Coolen, Wenchao Wei, Fabrice Talla Nobibon, and Roel Leus. Scheudling modular projects on a bottleneck resource. *Journal of Scheduling*, 17:67–85, 2014.

Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.

Rebi Daldal, Iftah Gamzu, Danny Segev, and Tonguç Ünlüyurt. Approximation algorithms for sequential batch-testing of series systems. *Naval Research Logistics (NRL)*, 63(4):275–286, 2016.

Bert De Reyck, Roel Leus. R&D project scheduling when activities may fail. *IIE Transactions*, 40:367–384, 2008.

John P. Dickerson, Ariel D. Procaccia, and Tuomas Sandholm. Failure-aware kidney exchange. *Management Science*, 65(4):1768–1791, 2019. doi: 10.1287/mnsc.2018.3026. URL `https://doi.org/10.1287/mnsc.2018.3026`.

Iain Dunning, Joey Huchette, and Miles Lubin. Jump: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

Funda Ergun, Rakesh Sinha, and Lisa Zhang. An improved FPTAS for restricted shortest path. *Information Processing Letters*, 83(5):287–291, 2002.

Michael R. Garey and David S. Johnson Computers and intractability: A guide to the theory of NP-completeness. W.H. Freeman San Francisco, 1979.

Noam Goldberg. Non-zero-sum nonlinear network path interdiction with an application to inspection in terror networks. *Naval Research Logistics (NRL)*, 64(2):139–153, 2017.

Noam Goldberg and Gabor Rudolf. On the complexity and approximation of the maximum expected value all-or-nothing subset. *Discrete Applied Mathematics*, in press, 2019. doi: 10.1016/j.dam.2019.08.012. URL `https://doi.org/10.1016/j.dam.2019.08.012`.

Pierre Hansen. Bicriterion path problems. In *Multiple criteria decision making theory and application*, pages 109–127. Springer, 1980.

Refael Hassin. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations research*, 17(1):36–42, 1992.

Mordechai I. Henig. Efficient interactive methods for a class of multiattribute shortest path problems. *Management Science*, 40(7):891–897, 1994.

Oya Icmeli-Tukel and Walter O. Rom. Ensuring quality in resource constrained project scheduling. *European Journal of Operational Research*, 103:483–496, 1997. ISSN 0377-2217. doi: 10.1016/s0377-2217(96)00305-0.

Naoki Katoh. A fully polynomial time approximation scheme for minimum cost-reliability ratio problems. *Discrete applied mathematics*, 35(2):143–155, 1992.

JongYul Kim, ChangWook Kang, and InKeuk Hwang. A practical approach to project scheduling: considering the potential quality loss cost in the time–cost tradeoff problem. *International Journal of Project Management*, 30:264–272, 2012. ISSN 0263-7863. doi: 10.1016/j.ijproman.2011.05.004.

Damodara U Kini. Materials management: The key to successful project management. *Journal of management in engineering*, 15(1):30–34, 1999.

Dean H Lorenz and Danny Raz. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters*, 28 (5):213–219, 2001.

Feng Pan and David P Morton. Minimizing a stochastic maximum-reliability path. *Networks*, 52(3):111–119, 2008.

Christos H Papadimitriou and Mihalis Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 86–92. IEEE, 2000.

David Pisinger. A minimal algorithm for the 0-1 knapsack problem. *Operations Research*, 45(5):758–767, 1997.

Tzvi Raz, Robert Barnes, and Dov Dvir. A critical look at critical chain. *Project Management Journal*, 2003.

Siqian Shen, J Cole Smith, and Shabbir Ahmed. Expectation and chance-constrained models and algorithms for insuring critical paths. *Management Science*, 56(10):1794–1814, 2010.

Hossein M. Soroush. The most critical path in a PERT network: A heuristic approach. *European Journal of Operational Research*, 78:93–105, 1994. ISSN 0377-2217. doi: 10.1016/0377-2217(94)90124-4.

Arthur Warburton. Approximation of pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35(1):70–79, 1987.

Peng Zhang, Yang Wang, Weidong Xiao, and Wenyuan Li. Reliability evaluation of grid-connected photovoltaic power systems. *IEEE transactions on sustainable energy*, 3(3):379–389, 2012.