Prophets in Parallel: Dynamic Cut Minimization in Series-Parallel Graphs

William Frendreiss[1] and Alejandro Toriello[2]

[1]School of Mathematics, Georgia Institute of Technology

[2]Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology

May 24, 2026

Abstract

We introduce a sequential version of the minimum s-t cut problem, defined by a directed graph with source s and sink t, and nonnegative random variables for each arc representing arc weights. We start with a working set $S = \{s\}$ and observe weight realizations for outgoing arcs from S only. We choose to either stop or to add a node to S with the objective of minimizing the expected weight of the chosen cut. When the graph is a directed s-t path, the model reduces to the classical prophet inequality problem in minimization form. We formulate the problem as a dynamic program (DP) and show that computing the expected cost of an optimal policy is #P-hard even on series-parallel graphs. We leverage a linear programming (LP) formulation of the DP to construct a separable cost-to-go function approximation (CFA), yielding a lower bound. We design a heuristic policy based on the CFA and provide a worst-case performance guarantee that depends on graph topology but is independent of arc weight distributions. Our computational experiments demonstrate the empirical quality of the heuristic policy.

Keywords: Prophet Inequalities, Optimal Stopping, Approximate Linear Programming, Dynamic Programming.

1 Introduction

In a classical optimal stopping problem of Krengel and Sucheston [45], the decision maker sequentially observes realizations of random variables in a take-it-or-leave-it setup, with the objective

of maximizing expected value. The famous *prophet inequality* [32, 45, 64] states that a threshold-based algorithm achieves at least half of the expected offline optimum. Prophet inequalities for this model and its extensions have applications in pricing, financial instruments, and revenue management, among other areas [13, 14, 15]. Extensions and special cases include i.i.d. variables [13, 33, 60], multiple-choice variants [38], and matroid constraints on choice [22, 42].

However, comparatively little work has studied the prophet setup with a minimization objective, in part because the optimal policy can perform arbitrarily poorly versus the offline optimum. Livanos and Mehta [49] studied special cases where a prophet inequality does apply, namely for i.i.d. variables with bounded hazard rate. The authors extend this work in [50] with an analysis of optimal threshold algorithms using extreme value theory. The minimization paradigm has also found applications; Qin et al. [62] connect minimization prophet inequalities to procurement auctions.

In this work, we extend the minimization setting by considering the problem of finding the minimum cut in a directed graph with uncertain arc weights. Let $D = (V, A)$ be a directed graph with source s and sink t; each arc $a \in A$ has an associated nonnegative weight distribution. We begin with $S = \{s\}$; at every iteration, we observe the realized weights of outgoing arcs from S, and must decide whether to stop or irreversibly add some $v \in V \setminus t$ to S. The objective is to minimize the expected sum of weights of outgoing arcs from S when we stop; we refer to this problem as dynamic cut minimization (DCM). From the perspective of procurement, our model can be interpreted as a generalized procurement auction depending on the graph topology, e.g. multiple items, components and sub-components, etc.

After a brief literature review, in Section 2 we formulate DCM, relate it to the famous prophet inequality and establish preliminary results, including that no algorithm can achieve a constant-factor approximation to the offline minimum cut. Instead of establishing a prophet inequality, our goal is therefore to find or approximate the optimal policy as a benchmark. We first show that finding an optimal policy is #P-hard even on *two-terminal series-parallel* graphs; we spend the remainder of the paper focusing on this class of graphs. We establish a separable lower bound based on linear programming in Section 3, and then use this to develop a heuristic policy in Section 4, for which we obtain a worst-case approximation ratio. This guarantee is independent of arc weight distributions, but depends on the instance's topology; roughly, the ratio depends on how many nodes with in-degree greater than one we encounter in any s-t path. We test our methods empirically across several graph topologies in Section 5.

1.1 Contributions

- We propose the DCM model (Section 2.2), which has not been previously studied to our knowledge. DCM generalizes both the minimization prophet model as well as the deterministic minimum cut problem, and is related to several important literature streams, as we outline below.

- Because DCM does not admit a prophet inequality against the offline optimum and is moreover #P-hard even in series-parallel graphs (Theorem 2.2), we instead propose a linear programming lower bound (5) based on a separable cost-to-go function approximation (CFA). For series-parallel graphs, the CFA lower bound decomposes into independent prophet-like problems whose optimum can be efficiently computed (Theorem 3.2). The bound requires us to intelligently "split" nodes with in-degree greater than one, and we choose this split using a super-gradient ascent scheme (Algorithm 1) that may be of independent interest.

- We also propose a heuristic policy guided by the CFA (Algorithm 2), and show that this policy has a multiplicative worst-case guarantee that depends on the graph's topology, but not on arc weight distibutions (Theorem 4.1). Roughly, the guarantee depends on the in-degrees of nodes we encounter in any s-t path.

1.2 Related Work

Ford and Fulkerson initiated the study of cuts and flows with the famous max-flow min-cut theorem, and designed an algorithm to find a maximum s-t flow [26]. The problem remains an active area of research to this day, e.g. [12, 34, 35, 75]. We refer readers to [5, 17, 65] for an in-depth discussion.

Network reliability problems extend deterministic network flows to a stochastic setting. Work in this field attempts to understand graph properties, e.g. connectivity, when parameters such as arc weights are random. Unlike the deterministic setting, Provan and Ball [61] showed many of these problems are #P-hard, leading to further work estimating connectivity probabilities, expected minimum cut weights, and so forth [2, 7, 23]. Network reliability problems have applications in power grids [56], computer networks [76], logistics [55, 58], and the internet of things [19]; for an overview, see [4, 29, 30, 59].

Another extension of the deterministic minimum cut problem related to DCM is *network interdiction*. The interdiction literature often focuses on two-player games on directed graphs, where

one player (the 'interdictor') attempts to diminish the efforts of the other player (the 'attacker') [69]. A plethora of interdiction problems focus on altering the flow that the attacker is attempting to maximize [31, 48, 54, 63]; other models focus on shortest path problems [6, 36, 37, 39, 67]. Abdolahzadeh et al. [1] studied minimum-cut interdiction explicitly, where an evader chooses an s-t cut while the interdictor can increase arc capacities under a maximum budget constraint.

More generally, there is work on many dynamic versions of combinatorial optimization problems, such as maximum independent set [53], bipartite matching [25, 57, 72, 71], knapsack [10, 8, 9, 43], and stochastic connectivity probing [28, 27, 44]. Cristi and Oren [16] study an extension of the prophet inequality in a somewhat similar setup to ours, but where the objective is to maximize the sum of arc weights on the path taken from s to t. Kleinberg and Oren [41] used a similar framework to model a problem in behavioral economics, that of time-inconsistent productivity.

2 Preliminaries & Problem Statement

2.1 Minimization Prophet Inequality

Consider n independent, nonnegative random variables $W_1, \ldots, W_n$ ordered sequentially. At time step $i \in \{1, \ldots, n\}$, we observe $w_i \sim W_i$, and irrevocably decide to either keep w_i and stop, or move to step $i+1$. Unlike the maximization setting, we *must* take w_n if we reach step n. The objective is to minimize the value of the w_i chosen. We can represent the expected cost of the optimal policy as a dynamic program (DP):

$$\text{return} \qquad \mathbb{E}[y_1^*(W_1)] : \tag{1a}$$

$$y_i^*(w_i) = \min\left\{w_i, \mathbb{E}\left[y_{i+1}^*(W_{i+1})\right]\right\} \quad \forall w_i \sim W_i \quad \forall i = 1, \ldots, n-1; \tag{1b}$$

$$y_n^*(w_n) = w_n \quad \forall w_n \sim W_n. \tag{1c}$$

The recursion in (1b) establishes that y_i^* is a piecewise linear, concave function of the realized weight w_i. We can reformulate (1) as a linear program (LP),

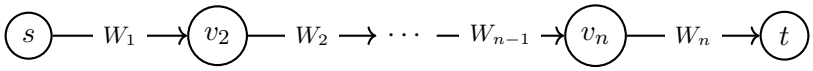
$$\max_{y} \qquad \mathbb{E}[y_1(W_1)] \tag{2a}$$

$$\text{s.t.} \qquad y_i(w_i) \le w_i \quad \forall w_i \sim W_i \quad \forall i = 1, \ldots, n \tag{2b}$$

$$y_i(w_i) \le \mathbb{E}[y_{i+1}(W_{i+1})] \quad \forall w_i \sim W_i \quad \forall i = 1, \ldots, n-1, \tag{2c}$$

where each y_i is a function of the realized weight w_i.

We can encode any such instance using a random variant of minimum s-t cut. Construct a graph using $n+1$ nodes $s = v_1, \ldots, v_{n+1} = t$, such that arc (v_i, v_{i+1}) has random weight W_i for all $i = 1 \ldots n$:



Our initial state is $S = \{s\}$, and the only arc weight we observe is (s, v_2). We can either stop and keep the weight $w_1 \sim W_1$ we observe, or choose another node to add to S. In the latter case, it is optimal to select the next node in the path, whereupon we observe the next arc's weight and the process repeats. The optimal policy corresponds exactly to that of the prophet minimization setting and is given by (1) or (2).

However, as Livanos and Mehta [49, attributed to Lucier] observe, there is no general prophet inequality when minimizing: let $n = 2$, with $W_1 = 1$ with probability 1, and $W_2 = 1/\varepsilon$ with probability ε, and 0 otherwise. The offline minimum is $\varepsilon \cdot 1 + (1-\varepsilon) \cdot 0 = \varepsilon$, while any online policy has expected cost 1 because this is the expected weight of both arcs. This leads to an arbitrarily large approximation ratio of $1/\varepsilon$.

2.2 Problem Statement

We define an instance of dynamic cut minimization (DCM) with

- a directed graph $D = (V, A)$;
- a designated *source* $s \in V$ and *sink* $t \in V$;
- an independent, nonnegative random variable W_a with support $\mathcal{W}_a \subseteq \mathbb{R}_+$ associated with each arc $a \in A$.

Starting with $S = \{s\}$, we observe the realized $w_a, a \in \delta^+(S)$ and either stop or irreversibly add a node $v \in V \setminus t$ to S. The objective is to minimize the expected value of $w(\delta^+(S)) := \sum_{a \in \delta^+(S)} w_a$ for the final S.

In a DP formulation of DCM, states are indexed by the current set S and the realized weights of arcs in $\delta^+(S)$. The cost-to-go function y^* can be written as a Bellman recursion,

$$y^*_S\big(w_a, a \in \delta^+(S)\big) = \min\Big\{ w\big(\delta^+(S)\big),$$

$$\min_{v \in V \setminus (S \cup t)} \Big\{ \mathbb{E}\big[y^*_{S \cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S) \big] \Big\} \Big\},$$

for each $S \subseteq V \setminus t$ with $S \ni s$, and the objective is $\mathbb{E}[y^*_s(W_a, a \in \delta^+(s))]$. We can then build the *cost-to-go* LP formulation [52]:

$$\begin{aligned}
\max_{y} \quad & \mathbb{E}\big[y_s\big(W_a, a \in \delta^+(s)\big)\big] \\
\text{s.t.} \quad & y_S\big(w_a, a \in \delta^+(S)\big) \le w\big(\delta^+(S)\big) \qquad \forall S \subseteq V \setminus t,\ S \ni s;\ \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^+(S) \\
& y_S\big(w_a, a \in \delta^+(S)\big) \le \mathbb{E}\big[y_{S \cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S)\big] \\
& \qquad\qquad \forall S \subseteq V \setminus t,\ S \ni s;\ \forall v \in V \setminus (S \cup t),\ \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^+(S).
\end{aligned}$$

If the arc weight support sets are infinite, this LP has infinitely many constraints and variables, where y_S is a cost-to-go function of the arc weights in $\delta^+(S)$. As with the special case defined by (1) and (2) above, y^* is optimal for this LP.

2.3 The Series-Parallel Assumption

For the remainder of the paper, we focus on the class of *series-parallel* digraphs.

Definition 2.1 (See [74, 24])**.** A *two-terminal series-parallel digraph*, or series-parallel graph for short, is a digraph D built from the following operations:

1. Start with a graph with only two nodes s, t and a single arc (s, t). Designate s as the *source*, and t as the *sink*. This graph is series-parallel.

2. (Parallel Composition). Take any series-parallel graphs D_1 and D_2, and identify the sources of D_1 and D_2 as the same node. Then, identify the two sinks as well.

3. (Series Composition). Take any series-parallel graphs D_1 and D_2, and identify the sink of D_1 with the source of D_2.

Series-parallel graphs encapsulate a wide variety of recursive topologies such as paths, parallel sums of paths, models of electrical networks, and models of reliability of many-component systems [51]. Series-parallel graphs are interesting in the field of algorithm design because many NP-hard optimization problems become much simpler when reduced to them; see [11, 40] for examples. However, DCM is #P-hard even on this simplified class of graphs.

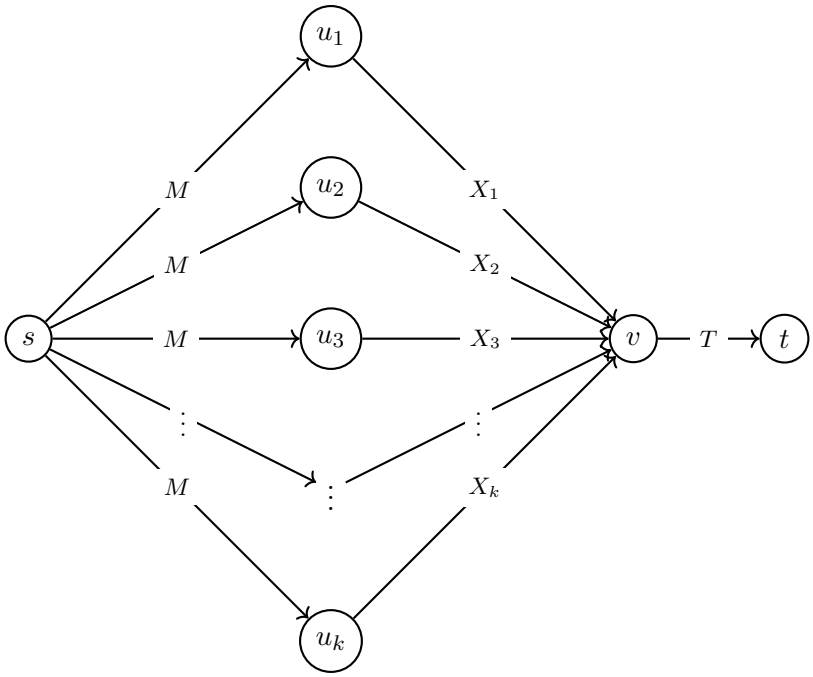


Figure 1: The Construction in Theorem 2.2

Theorem 2.2. *Computing the optimal expected cost of Dynamic Cut Minimization is #P-hard, even when each arc has a 1- or 2-point distribution and the underlying graph is a simple two-terminal series-parallel digraph with one node of in-degree more than 1. It is also #P-hard to implement the optimal policy under the same restrictions.*

Proof. We claim that #KNAPSACK [21] is reducible to computing the expected cost of the optimal policy of DCM on a suitable graph G. Consider an instance of #KNAPSACK with items $a_1, \ldots, a_k$ and capacity W. Define K as the true number of feasible solutions. Let M be a sufficiently large number, and let X_i be a weighted Bernoulli random variable realizing as 0 or a_i with equal probability $1/2$. We define $G(T)$ as a digraph with $k+3$ nodes and $2k+1$ arcs, taking a parameter $T \in \mathbb{R}$ as input. The graph is formed by k parallel paths $P_1, \ldots, P_k$ of length two. On $P_i = s \to u_i \to v$, the first arc from s has deterministic weight M and the second arc has weight distribution X_i. There is a final arc of weight T from v to t. Since $M > T$, an optimal decision maker would never allow any of the initial arcs $(s, u_1), \cdots, (s, u_k)$ to be in the final s-t cut. So the expected cost of the optimal policy is the expected minimum of $X_1 + \cdots + X_k$ and T. Let $F(T)$ be this expected cost.

Due to the independence of the X_i variables and the fact they only take on integer values,

$$\mathbb{P}[X_1 + \cdots + X_k < W + 3/4] = \mathbb{P}[X_1 + \cdots + X_k < W + 1/4] = \frac{K}{2^k} =: p.$$

Since

$$F(W + 3/4) = p\mathbb{E}[X_1 + \cdots + X_k | \, X_1 + \cdots + X_k < W + 3/4] + (1 - p)(W + 3/4)$$
$$F(W + 1/4) = p\mathbb{E}[X_1 + \cdots + X_k | \, X_1 + \cdots + X_k < W + 1/4] + (1 - p)(W + 1/4)$$

and the conditional expected costs are the same,

$$F(W + 3/4) - F(W + 1/4) = \frac{1 - p}{2} = \frac{1}{2} - \frac{K}{2^{k+1}}.$$

So exact calculation of the optimal expected cost for general directed series-parallel graphs G can solve #KNAPSACK by running the algorithm on two graphs and subtracting the resulting value. This reduction requires computing $k + 1$ digits, which is polynomial in the input size.

For the second part of the theorem, we can modify any instance of DCM expected cost-computation on G to that of policy implementation by adding a new source s^* with one arc (s^*, s) to the old source. Give this arc deterministic value β. An optimal policy takes $s \in S$ if and only if the expected cost of the optimal policy on G is less than that of β. □

In light of Theorem 2.2, it is important to note some considerations when it comes to calculating expected values. We assume finite expected values in all cases. We also utilize the *unit-cost* arithmetic model, i.e. simple arithmetic operations do not take longer on large versus small numbers.

Our approach to DCM involves solving single-path instances; by (1), this requires calculating certain conditional expectations of the random variables W_i. For distributions with finite support, these are simply vector products. For continuous or countably-supported distributions, we have to be more careful; if W_i has probability density function f,

$$\mathbb{E}_{W_i}[\min\{W_i, C\}] = \int_0^C w_i f(w_i) \, dw_i + C \cdot \mathbb{P}[W_i > C].$$

We therefore assume an oracle for such conditional expectations. In the case of discrete variables with finite support, this implies that we assume arc weight support sizes are problem inputs or are polynomial in the graph size. This also aligns with practical considerations for continuous variables,

as single-variable integration can be efficiently calculated within a high numerical tolerance.

Despite Theorem 2.2, series-parallel graphs offer great simplification over other graphs based on their topology, as exemplified by the next fact.

Lemma 2.3. *Suppose $D = (V, A)$ is a directed series-parallel graph with source s and sink t. Suppose there is a path from s to v containing u and v such that u has a path to v. Then if D contains both*

1. *a path from u to t not containing v, and*
2. *a path from s to v not containing u,*

there is a vertex x on the path from u to v such that all paths from s to v and all paths from u to t contain x.

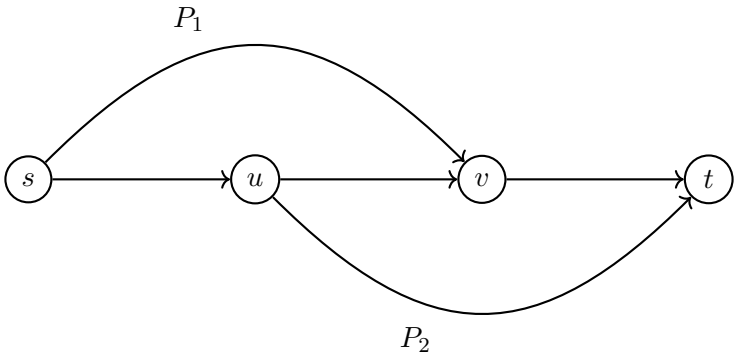


Figure 2: This graph is not series-parallel.

Proof. This is a direct consequence of the forbidden subgraph for series-parallel graphs, homeomorphic to the digraph depicted in Figure 2. See [20, 74] for more information. □

3 Separable Lower Bound

Using Lemma 2.3, we can restrict the DP recursion on series-parallel graphs by removing node options. Removing suboptimal nodes from the selection process without loss of optimality serves as the foundation for the LP approximations we employ.

Lemma 3.1. *If $D = (V, E)$ is a directed series-parallel graph, then some optimal policy for DCM on D satisfies the following: a node v is not added to the set S until every node u with a path to v is in S.*

Proof. We first claim that on *any* graph, we may restrict the options to nodes $v \in N^+(S) \setminus S$. This implies that there is a path from s to every vertex in $S \setminus s$ at every step. Let $v \notin N^+(S) \setminus S$, and suppose that

$$\mathbb{E}\big[y_{S\cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S)\big] \leq y_S\big(w_a, a \in \delta^+(S)\big).$$

We have

$$\begin{aligned}
&\mathbb{E}\big[y_{S\cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S)\big] \\
&= \mathbb{E}\left[\min\left\{w\big(\delta^+(S\cup v)\big), \min_{u\in V\setminus(S\cup v);\, u\neq s,t}\left\{\mathbb{E}\big[y_{S\cup v\cup u}\big(W_a, a\in\delta^+(S\cup v\cup u)\big)\big]\right\}\right\} \,\middle|\, w_a, a\in\delta^+(S\cup v)\right].
\end{aligned}$$

Since $v \notin N^+(S) \setminus S$, $\delta^+(S \cup v) = \delta^+(S) \cup \delta^+(v)$. So for any realization of $w(\delta^+(v))$,

$$w\big(\delta^+(S\cup v)\big) = \sum_{a\in\delta^+(v)} w_a + \sum_{a\in\delta^+(S)} w_a \geq \sum_{a\in\delta^+(S)} w_a \geq y_S\big(w_a, a\in\delta^+(S)\big).$$

Therefore, we must have that

$$\begin{aligned}
&\mathbb{E}_{w(\delta^+(v))}\big[y_{S\cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S)\big] \\
&\quad = \mathbb{E}_{w(\delta^+(v))}\mathbb{E}_{w(\delta^+(u'))}\big[y_{S\cup v\cup u'}\big(W_a, a\in\delta^+\big(S\cup v\cup u'\big)\big) \mid w_a, a\in\delta^+(S)\big]
\end{aligned}$$

for u' minimizing $\mathbb{E}[y_{S\cup v\cup u}(W_a, a \in \delta^+(S \cup v \cup u)) \mid w_a, a \in \delta^+(S)]$ over $u \in V \setminus (S \cup \{v, s, t\})$. Keep this u' and then define v' as the vertex minimizing

$$\mathbb{E}_{w(\delta^+(v))}\big[y_{S\cup v\cup u'}\big(W_a, a\in\delta^+\big(S\cup v\cup u'\big)\big) \mid w_a, a\in\delta^+(S)\big]$$

over $v \in V \setminus (S \cup \{u', s, t\})$. We then have

$$\begin{aligned}
&\mathbb{E}_{w(\delta^+(v))}\big[y_{S\cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S)\big] \\
&\quad = \mathbb{E}_{w(\delta^+(v))}\mathbb{E}_{w(\delta^+(u'))}\big[y_{S\cup v\cup u'}\big(W_a, a\in\delta^+\big(S\cup v\cup u'\big)\big) \mid w_a, a\in\delta^+(S)\big] \\
&\quad = \mathbb{E}_{w(\delta^+(u'))}\mathbb{E}_{w(\delta^+(v))}\big[y_{S\cup v\cup u'}\big(W_a, a\in\delta^+\big(S\cup v\cup u'\big)\big) \mid w_a, a\in\delta^+(S)\big] \\
&\quad \geq \mathbb{E}_{w(\delta^+(u'))}\mathbb{E}_{w(\delta^+(v'))}\big[y_{S\cup v'\cup u'}\big(W_a, a\in\delta^+\big(S\cup v'\cup u'\big)\big) \mid w_a, a\in\delta^+(S)\big] \\
&\quad \geq \mathbb{E}_{w(\delta^+(u'))}\big[y_{S\cup u'}\big(W_a, a\in\delta^+\big(S\cup u'\big)\big) \mid w_a, a\in\delta^+(S)\big].
\end{aligned}$$

Hence we can only reduce the expected DP value by taking u' instead of v, proving the claim.

Suppose our current working set is $S \subset V$, $v \in N^+(S) \setminus S$, and $u \notin S$ has a path to v. If all paths through v contain u, then $u \in S$ by our claim, a contradiction. So v has a path from s circumventing u. By Lemma 2.3, there is some x (which may be equal to v) such that all paths from u to t contain x. But since all paths from s to v contain x, the claim implies $x \in S$. Hence we can add all such u and only reduce the cut. So a policy that adds u before v cannot have a higher expected value. □

In other words, the optimal policy of DCM for series-parallel graphs can be reduced to one that only adds v to S at the current step if every arc into v has been realized. The simplified Bellman equation is then

$$\begin{aligned} y_S\big(w_a, a \in \delta^+(S)\big) = \min \Big\{ & w\big(\delta^+(S)\big), \\ & \min_{v \in N^+(S) \setminus (S \cup N^+(V \setminus S) \cup t)} \Big\{ \mathbb{E}\big[y_{S \cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S) \big] \Big\} \Big\} \qquad (3) \\ & \forall S \subseteq V \setminus t,\, S \ni s;\, \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^+(S) \end{aligned}$$

without loss of optimality. In the LP formulation as well, we may reduce the constraints:

$$\begin{aligned} \max \quad & \mathbb{E}\big[y_s\big(W_a, a \in \delta^+(s)\big) \big] && (4a) \\ \text{s.t.} \quad & y_S\big(w_a, a \in \delta^+(S)\big) \le w\big(\delta^+(S)\big) \quad \forall S \subseteq V \setminus t,\, S \ni s;\, \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^+(S) && (4b) \\ & y_S\big(w_a, a \in \delta^+(S)\big) \le \mathbb{E}\big[y_{S \cup v}\big(W_a, a \in \delta^+(S \cup v)\big) \mid w_a, a \in \delta^+(S) \big] && (4c) \\ & \qquad \forall S \subseteq V,\, S \ni s;\, \forall v \in N^+(S) \setminus \big(S \cup N^+(V \setminus S) \cup t\big),\, \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^+(S) \end{aligned}$$

3.1 Separable LP Approximation

Even if the arc weight distributions are finite, say with support size $\le M$, there are $M^{\deg^+(S)}$ constraints of the form (4b) for all S. Given this large number, we aim to build an approximation.

One common technique to build approximations of the cost-to-go LP (4) restricts the set of feasible solutions. For example, we can select state features, and define functions of these features. The literature on these *approximate LPs* dates to [18, 66, 73] and is particularly prevalent in reinforcement learning (see [46, 47], for example). Approximate LP has specifically been a useful

technique to study dynamic discrete optimization models on graphs, e.g. bipartite matching [72, 71], network revenue management [3], and the traveling salesman problem [70].

We employ a separable, arc-based *cost-to-go function approximation* (CFA) of (4) in which the cost at a given set S is given by separately evaluating the weight of each arc in $\delta^+(S)$. We define a function $y_a : \mathcal{W}_a \to \mathbb{R}$ for each arc $a \in A$, and let

$$y_S\big(w_a, a \in \delta^+(S)\big) \approx \sum_{a \in \delta^+(S)} y_a(w_a).$$

Any separable approximation feasible in (4) immediately gives us a lower bound on the optimal cost-to-go function. Indeed, we are taking the feasible set of y_S and adding the extra constraint that y_S is the sum of auxiliary variables, decreasing the objective at optimality.

Under the approximation, constraint (4c) reduces to

$$\sum_{a \in \delta^+(S)} y_a(w_a) \leq \sum_{a \in \delta^+(S) \setminus \delta^-(v)} y_a(w_a) + \sum_{a \in \delta^+(v)} \mathbb{E}[y_a(W_a)].$$

Subtracting common terms and using $\delta^-(v) \subseteq \delta^+(S)$ (from Lemma 3.1), we have

$$\sum_{a \in \delta^-(v)} y_a(w_a) \leq \sum_{a \in \delta^+(v)} \mathbb{E}[y_a(W_a)] \qquad \forall v \in V \setminus \{s,t\},\, \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^-(v).$$

We can therefore derive a simplified approximation to (4):

$$\max_{y} \sum_{a \in \delta^+(s)} \mathbb{E}[y_a(W_a)] \tag{5a}$$

$$\text{s.t.} \sum_{a \in \delta^+(S)} y_a(w_a) \leq \sum_{a \in \delta^+(S)} w_a \quad \forall S \subseteq V \setminus t,\, S \ni s;\, \forall w_a \in \mathcal{W}_a\, \forall a \in \delta^+(S) \tag{5b}$$

$$\sum_{a \in \delta^-(v)} y_a(w_a) \leq \sum_{e \in \delta^+(v)} \mathbb{E}[y_e(W_e)] \quad \forall v \in V \setminus \{s,t\},\, \forall w_a \in \mathcal{W}_a\ \forall a \in \delta^-(v). \tag{5c}$$

Despite these simplifications, constraints (5b) are still indexed by subsets of nodes, and constraints (5c) have left-hand sides that depend on the current weight of every arc into v. In other words, we still have an LP with exponentially many constraints, even in the case of two-point distributions, which is intractable to solve directly. The next result allows us to obtain a tractable reformulation with a greatly reduced constraint set.

Theorem 3.2. *For any $\lambda_a, a \in A$ satisfying*

$$\sum_{a \in \delta^-(v)} \lambda_a = 1 \quad \forall v \in V \setminus \{s,t\}; \qquad \lambda_a \geq 0 \quad \forall a \in A, \tag{6}$$

the linear program

$$\max_{y} \qquad \sum_{a \in \delta^+(s)} \mathbb{E}[y_a(W_a)] \tag{7a}$$

$$\text{s.t.} \qquad y_a(w_a) \leq w_a \qquad \forall a \in A, \forall w_a \in \mathcal{W}_a \tag{7b}$$

$$y_{uv}(w_{uv}) \leq \lambda_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}[y_a(W_a)], \quad \forall v \in V \setminus \{s,t\}, (u,v) \in \delta^-(v), w_{uv} \in \mathcal{W}_{uv} \tag{7c}$$

is a lower bound for (5)*. There exist $\lambda_a, a \in A$ satisfying* (6) *that make the two objectives equal.*

Proof. Let y^* be an optimal variable assignment for (5) with objective z^*, and let (7) have objective $\hat{z}$. For the first statement of the lemma, we claim (5b) and (5c) follow from (7b) and (7c), and hence $\hat{z} \leq z^*$. Indeed, simply add every constraint (7b) for $a \in \delta^+(S)$ to get (5b). Then, add (7c) for $a \in \delta^+(S)$ to get (5c) because $\sum_{a \in \delta^-(v)} \lambda_a = 1$.

For the second statement, for a given $a \in A$, fix a set of nodes S such that $a \in \delta^+(S)$. Then, if the weight of every other arc out of S is 0, (5b) becomes

$$y_a^*(w_a) \leq y_a^*(w_a) + \sum_{e \in \delta^+(S) \setminus a} y_e^*(0) \leq w_a + 0 = w_a.$$

Here, we used $y^* \geq 0$ and $0 \in \mathcal{W}_a$, the latter of which we may assume without loss of generality. So (7b) follows from (5b).

Next, fix $v \in V$ and choose arc realizations w such that

$$\sum_{a \in \delta^-(v)} y_a^*(w_a) = \sum_{e \in \delta^+(v)} \mathbb{E}[y_e^*(w_e)];$$

if no such assignment exists then the constraints (5c) and (7c) are redundant. Fix an $a \in \delta^-(v)$. There exists some scalar λ_a such that $y_a^*(w_a) = \lambda_a \sum_{e \in \delta^+(v)} \mathbb{E}[y_e^*(w_e)]$. By fixing the weight realizations of the other arcs into v, we have that

$$\sum_{a' \in \delta^-(v) \setminus a} y_{a'}^*(w_{a'}) = (1 - \lambda_a) \sum_{e \in \delta^+(v)} \mathbb{E}[y_e^*(w_e)].$$

So, using (5c) with varying $w \in \mathcal{W}_a$,

$$y_a^*(w) \leq \sum_{e \in \delta^+(v)} \mathbb{E}[y_e^*(w_e)] - (1-\lambda_a) \sum_{e \in \delta^+(v)} \mathbb{E}[y_e^*(w_e)] = \lambda_a \sum_{e \in \delta^+(v)} \mathbb{E}[y_e(w_e)],$$

implying (7c). Therefore y^* is feasible for (7) with these λ_a, so $\hat{z} \geq z^*$. □

Corollary 3.3. *For any λ, an optimal solution $\hat{y}$ of (7) is of the form*

$$\hat{y}_{uv}(w_{uv}) = \min\left\{ w_{uv}, \lambda_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] \right\}, \tag{8}$$

for all $(u,v) \in A$ and $w_{uv} \in \mathcal{W}_{uv}$.

In other words, optimal functions in (8) mirror the cost-to-go functions of optimal policies for single-path, prophet-type stopping problems. Assuming conditional expectations are efficiently computable (see the remarks after Theorem 2.2), (7) offers a tractable approximation to (3) for any values of λ_a, $a \in A$: we "split" vertices with in-degree greater than one, resulting in separate, single-path problems, each of which we solve analogously to (1); Figure 3 illustrates an example. Furthermore, the "splitting" of constraints only occurs for nodes $v \neq t$ with in-degree at least two. Therefore, if no such nodes exist, the separable structure comes without loss of optimality.

Theorem 3.4. *If $\max_{v \in V \setminus t} \left\{ \deg^-(v) \right\} = 1$, then the objective of (5) and (7) is the same as (4).*

The optimal policy for DCM in this case reduces to a sum of minimization prophet instances and we can leverage Corollary 3.3 to solve DCM using a threshold approach. The proof of Theorem 3.4 is included in Appendix A, and is a standard verification of our intuition of independence.

For graphs with higher max in-degree, the separable LP implicitly creates a graph satisfying the condition of Theorem 3.4, as shown in Figure 3. We duplicate v and the entire subgraph of nodes reachable from v, splitting into $\deg^-(v)$ copies. For the copy corresponding to (u,v), the cost-to-go downstream of v is weighted by λ_{uv}. When we do this over the entire graph, proceeding in a topological ordering, we create a graph for which each arc a is duplicated by the number of s-t paths containing a. On the copy of a corresponding to path P, a's weight is multiplied by the product of λ_e for arcs $e \in P$ preceding a.

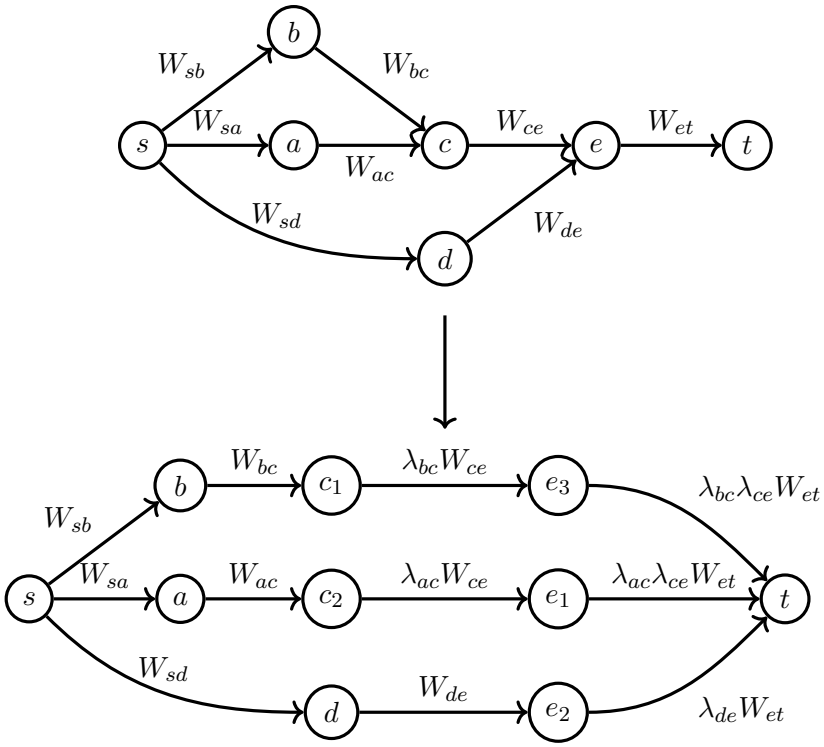


Figure 3: Intuition behind the separable approximation.

3.2 Choosing Multipliers for Theorem 3.2

If some v has in-degree greater than one, we want to choose λ_a maximizing the optimum of (7). To find these λ_a values, we first establish the structure of the objective value as a function of the multipliers.

Lemma 3.5. *For a fixed $v \in V \setminus \{s,t\}$, the optimum of (7) is concave as a function of $\{\lambda_a\}_{a\in\delta^-(v)}$. In particular, it is concave on the simplex*

$$\left\{ \{\lambda_a\}_{a\in\delta^-(v)} \middle| \sum_{a\in\delta^-(v)} \lambda_a = 1;\ \lambda_a \geq 0 \quad \forall a \in \delta^-(v) \right\}.$$

Proof. Let $\hat{y}$ be an optimal point of (7). For any arc $(u_1,u_2) \in A$, if u_2 has no path to v, then $\hat{y}_{u_1u_2}(w_{u_1u_2})$ is unchanging in λ_a for any $a \in \delta^-(v)$. If instead u_2 does, we proceed by induction. For the base case, with a fixed v and $a \in \delta^-(v)$, we have

$$\hat{y}_a(w_a) = \min \left\{ w_a, \lambda_a \sum_{e\in\delta^+(v)} \mathbb{E}[\hat{y}_e(W_e)] \right\} \tag{9}$$

for all w_a. This function is concave because it is the minimum of a constant and a linear function in λ_a. Since the expected value of a concave function is also concave, $\mathbb{E}[\hat{y}_e(W_e)]$ is concave in λ_a.

Now fix $(u_1, u_2) \in A$ and suppose every arc e in the subgraph of reachable vertices from u_2 satisfies that $\hat{y}_e(w_e)$ is concave in $\{\lambda_a; a \in \delta^-(v)\}$. For any $e \in \delta^-(u_1)$, (9) implies that $\hat{y}_{u_1,u_2}(w_{u_1u_2})$ is the minimum of $w_{u_1u_2}$ (which is constant over all λ_a) and a concave function in the set of λ_a; this is concave. Finally, $\sum_{e\in\delta^+(s)} \mathbb{E}[\hat{y}_e(W_e)]$ is concave in $\{\lambda_a\}_{a\in\delta^-(v)}$ by induction. □

Algorithm 1 Supergradient Ascent Approximation to (5)

for $v \in V \setminus \{s, t\}$ **do**
    $\lambda_a^0 \leftarrow 1/\deg^-(v) \quad \forall a \in \delta^-(v)$
Topologically sort nodes $s \preceq v_1 \preceq \cdots v_n \preceq t$.
$\hat{y}^0 \leftarrow$ optimal point of (7) with λ^0
$z^0 \leftarrow$ optimal value of (7) with λ^0
$z^1 \leftarrow \infty$
$t \leftarrow 0$
$\gamma \leftarrow$ universal learning rate
$\varepsilon \leftarrow$ universal tolerance
while $\left|\left|z^t - z^{t-1}\right|\right| > \varepsilon$ **do**
    $t \leftarrow t + 1$
    for $i = 1 \ldots n$ **do**
        $\eta_i^t \leftarrow \left(\gamma/\deg^-(v_i) \cdot \sum_{a\in\delta^+(v_i)} \mathbb{E}[\hat{y}_a(W_a)]\right)$ ($1/\deg^-(v_i)$ is a scaling factor)
        $\ell \leftarrow 0$
        while $\left|\left|\left(\lambda_a^t\right)_{a\in\delta^-(v_i)}^{\ell} - \left(\lambda_a^t\right)_{a\in\delta^-(v_i)}^{\ell-1}\right|\right| > \varepsilon$ **do**
            $\ell \leftarrow \ell + 1$
            $(\lambda_{(u,v_i)}^t)^\ell \leftarrow (\lambda_{(u,v_i)}^t)^{\ell-1} + \eta_i^t \sum_{s-u-v_i \text{ paths } P} p(u,P) \prod_{a\in P\setminus(u,v_i)} \lambda_a \quad \forall (u, v_i) \in \delta^-(v_i)$
            Project $(\lambda_a)_{a\in\delta^-(v_i)}$ onto $\Delta^{\delta^-(v_i)}$.
        $\lambda_a^t \leftarrow \left(\lambda_a^t\right)^\ell \quad \forall a \in \delta^-(v_i)$
    $\hat{y}^t \leftarrow$ optimal point of (7) with λ^t
    $z^t \leftarrow$ optimal value of (7) with λ^t
return $\left\{\lambda_a^t\right\}_{a\in\delta^-(v), v\in V\setminus\{s,t\}}$.

Lemma 3.6. *Let $v \in V \setminus \{s,t\}$, and let $P : u_1(=s), u_2, \ldots, u_\ell, u_{\ell+1}(=v)$ be an s-v path; define*

$$p(v,P) := \prod_{i=1}^{\ell} \mathbb{P}\left[W_{u_i u_{i+1}} > \lambda_{u_i u_{i+1}} \sum_{a \in \delta^+(u_{i+1})} \mathbb{E}[\hat{y}_a(W_a)] \right].$$

Denote $\hat{y}$ as an optimal point for (7). Then

$$\Big(\sum_{a \in \delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] \sum_{\substack{s\text{-}v \text{ paths } P \\ P \ni (u,v)}} p(v,P) \prod_{a \in P \setminus (u,v)} \lambda_a \Big)_{(u,v) \in \delta^-(v)} \tag{10}$$

is a supergradient for the optimum of (7) as a function of $\{\lambda_{uv}\}_{(u,v)\in\delta^-(v)}$.

The proof of the lemma is long and non-illuminating, so we have it included in Appendix A. We can relate the supergradient back to this intuition of the separable LP as splitting nodes with high in-degree. Specifically, the supergradient records the sum of probabilities of reaching any copy of the arc a, multiplied by the weighted expected cost-to-go.

As for explicit computation, despite a possibly exponential number of paths from s to v, the weighted sum in (10) can be computed in polynomial time using recursion by saving the sums for ancestor nodes. Unfortunately, Lemma 3.5 only establishes concavity when considering $\{\lambda_a\}_{a\in\delta^-(v)}$, for a fixed node $v \neq s,t$. Therefore, we implement a batched coordinate ascent: for some $v \neq s,t$, use (10) to optimize λ_a for $a \in \delta^-(v)$ with respect to the objective value of (7), while keeping the other λ_a's fixed. After each step, we perform a projection to ensure (6). Iterate this process over all such nodes until no such improvement is possible. Algorithm 1 details this procedure formally. Because the optimum increases at each step and is bounded above, we converge to a point that is optimal with respect to each $v \neq s,t$ with other λ_a fixed. Furthermore, since $\{\lambda_a\}_{a\in\delta^-(v)}$ lies in a simplex, a true gradient at convergence is a multiple of $(1,\ldots,1)$.

Corollary 3.7. *Suppose $\mathbb{P}[w_a > t]$ and $\mathbb{E}[w_a : w_a \leq t]$ are differentiable in $t > 0$ for all $a \in A$. For fixed $v \in V \setminus \{s,t\}$ and λ_a, $a \notin \delta^-(v)$, a maximizing choice of $\{\lambda_{uv}\}_{(u,v)\in\delta^-(v)}$ for (7) equalizes*

$$\sum_{\substack{s\text{-}v \text{ paths } P \\ P \ni (u,v)}} p(v,P) \prod_{a \in P \setminus (u,v)} \lambda_a$$

for all $(u,v) \in \delta^-(v)$.

For discrete distributions, there may not be a set of λ that equalize all probabilities. To fix this,

one can alter the definition of $p(v, P)$ in Lemma 3.6 to make a probabilistic choice to add v to S when $w_{uv} = \lambda_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}[y_a(W_a)]$. In this way, the probabilities can be altered to become equal without changing the lower bound.

Proof. We claim that the expected optimal value of (7) has a continuous first derivative; then the supergradient is in fact a gradient. This implies the corollary, as projected gradient ascent on a simplex converges to a point where the gradient is a multiple of $(1, \ldots, 1)^\top$. Let $f(\lambda)$ be a differentiable function of λ and let $\mathcal{D}$ be a distribution satisfying the corollary's assumptions. Then

$$\mathbb{E}_{x \sim \mathcal{D}}[\min\{x, f(\lambda)\}] = (1 - \mathbb{P}[x > f(\lambda)]) \cdot \mathbb{E}[x \mid x \leq f(\lambda)] + \mathbb{P}[x > f(\lambda)] \cdot f(\lambda)$$

is also differentiable in λ, except possibly at $\lambda = 0$. But then by recursion on (9), starting with $f(\lambda_{uv}) = \lambda_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}[y_a(W_a)]$ and taking expected values, we find that the optimal of (7) is differentiable in $\lambda_a > 0$ for any $a \in A$. □

4 CFA-based Policy

The importance of the separable approximate LP approach is not only a computable lower bound, but also an approximation of the cost-to-go function that can be used to build heuristic policies.

We build a heuristic policy from the CFA using an optimal solution $\hat{y}$ of (7), calculated with (8) and a set of multipliers $\{\lambda_a\}_{a \in A}$; as we explain later, it is important to use multipliers satisfying the conditions of Corollary 3.7. The heuristic algorithm proceeds as follows. At a state defined by cut S and arc weights $w_a, a \in \delta^+(S)$, the decision maker can either halt, or choose

$$S \leftarrow S \cup \arg \min_{v \in N^+(S), (u,v) \in A} \left\{ \lambda_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] - w_{uv} \right\} \tag{11}$$

when the minimum is negative. The policy relies on the following intuition. Each arc $a \in \delta^-(v)$ has a chosen multiplier λ_a which explains the "share" of weight expected to be contributed by $\hat{y}_a(W_a)$ to $\sum_{e \in \delta^-(v)} \hat{y}_e(W_e)$. If any arc weight is realized at more than this share, we irrevocably add the node v into S. This attempts to emulate the optimal policy in the separable lower bound given by (8) where we have replicated node v. The difference is that if we add v to S, we must include all "copies" of v – our actions on one path have effects in the others.

Algorithm 2 CFA-Based Policy

1: Find multipliers $\lambda_a, a \in A$ satisfying Corollary 3.7.
2: $\hat{y} \leftarrow$ optimal point of (7) with λ from step 1.
3: $S \leftarrow \{s\}$.
4: **while** `STOP` not chosen **do**
5: $\quad S \leftarrow S \cup \arg\min_{(u,v)\in\delta^+(S)} \left\{\lambda_{uv} \sum_{a\in\delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] - w_{uv}\right\}$ or `STOP`.
6: $\quad S \leftarrow S \cup \{\text{every node with a path to } v\}$ if not stopped.

Theorem 4.1. *The expected cost* Alg *of Algorithm 2 satisfies*

$$\frac{\mathbb{E}[\text{Alg}]}{\hat{z}} \leq \max_{s\text{-}t \text{ path } P} \prod_{v\in P\setminus\{s,t\}} \deg^-(v), \tag{12}$$

where $\hat{z}$ is the objective of (7) *for $\lambda_a, a \in A$ satisfying Corollary 3.7. This ratio is tight; that is, it is achieved by a family of series-parallel DCM instances.*

The proof of the bound is included in Appendix A. As for tightness, we include the example below.

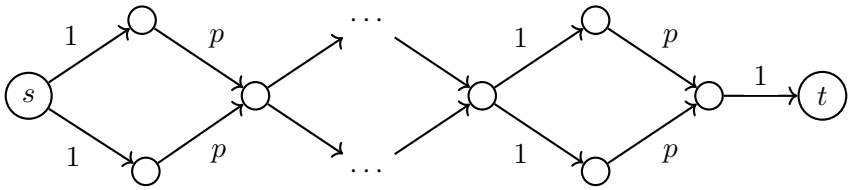


Figure 4: The graph in Example 4.2 with $k = 2$

Example 4.2 (Tightness of (12))**.** Consider a graph formed by first taking the parallel composition of k paths of length two, and then taking the series composition of the resulting graph n times. In each two-path, the first arc has weight one with probability one, while the second arc's weight is given by a Bernoulli distribution with probability p to be determined later. We add n copies of this parallel path graph in series. Finally, we add another arc with weight one at the end as another series composition; see Figure 4 for an example.

An optimal policy only stops if it encounters a cut of zero weight; otherwise, it takes the final arc as its cut. This policy in fact achieves the offline minimum cut. The expected value of the policy is

$$\mathbb{E}[\text{OPT}] = \mathbb{P}[\exists\ s\text{-}t \text{ path with weight-one arcs}] = \left(1 - (1-p)^k\right)^n.$$

As for the lower bound from LP (7), we can obtain a recursive formula. Let $\mathrm{LB}(n)$ be the lower bound for the graph with n copies of the k parallel paths, where we set $\mathrm{LB}(0) = 1$. Suppose the multipliers for the first node with in-degree k are $\lambda_1, \ldots, \lambda_k$; we have

$$\begin{aligned}\mathrm{LB}(n) &= \sum_{i=1}^{k} \left(p \cdot \min\left\{1, \lambda_i \cdot \mathrm{LB}(n-1)\right\} + (1-p) \cdot 0\right) \\ &\leq \sum_{i=1}^{k} p \cdot \lambda_i \cdot \mathrm{LB}(n-1) = p \cdot \mathrm{LB}(n-1).\end{aligned}$$

Applying this n times, we have $\mathrm{LB}(n) \leq p^n$. So the approximation ratio satisfies

$$\frac{\mathbb{E}[\mathrm{OPT}]}{\mathrm{LB}(n)} \geq \frac{\left(1-(1-p)^k\right)^n}{p^n} = \left(\frac{1-(1-p)^k}{p}\right)^n.$$

But as $p \to 0$, $\frac{1-(1-p)^k}{p} \to k$. So, with an appropriate choice of p, our approximation ratio is lower bounded by k^n, which matches (12) for this instance.

5 Computational Experiments

We performed a test suite of computations on graphs with up to 150+ nodes in order to test the practical effectiveness of Algorithm 2 versus the LP lower bound (7).

5.1 Instance Design

We conducted experiments on a variety of topologies. For each graph topology, we tested 100 different instances defined by different random weight distributions. For each distribution, we gave an arc a support of $\Omega = \{0.0, 0.1, 0.2, \ldots, 9.9, 10.0\}$ and an independent, uniformly sampled value $f : \Omega \to [0, 1]^{101}$ for the probability of each weight realizing. To make this a valid distribution, we normalized by dividing by the sum.

We first used a varied set of 10 series-parallel graph topologies on small graphs with 5-15 nodes each. These are numbered $G_0, \ldots, G_9$, and their topologies are shown in Appendix B. For larger-scale tests, we first tested on large examples of parallel-path graphs with a deterministic final arc with weight B. For each of the examples with $k \in \{2, 3, 4\}$ paths, we tested path lengths of $10, 25, 50$ arcs on each path. Then, we fixed the path lengths at two and tested on $10, 25, 50$ paths. For a topology with k paths, we tested values of B at $2.5k, 5k, 7.5k$ due to the distributional midpoint

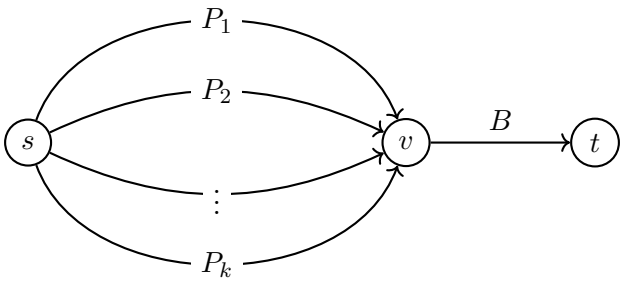


Figure 5: Joined Path Topologies

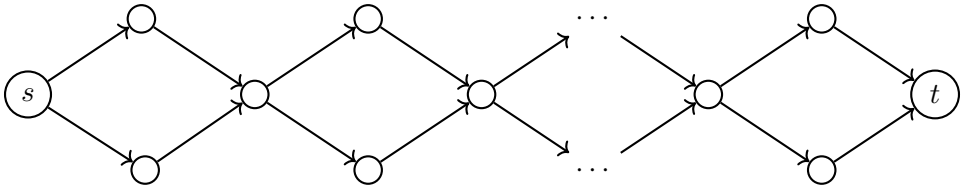


Figure 6: A Chain-link Graph

being roughly 5 for each arc.

We also tested topologies similar to the one in Example 4.2 as a form of stress testing. This group is 'chain-link' graphs, formed by consecutive serial graph sums of 4-node diamonds. Each diamond forms a 'link;' we tested with $5, 10, 15, 20$ links.

5.2 Experiment Details

We implemented two control benchmarks to test the effectiveness of the upper and lower bound. As a lower bound benchmark, we computed the minimum cut of each simulated realization using `NetworkX` and computed the sample mean. As an upper bound benchmark, we used a simple greedy heuristic (Algorithm 3), where we add $S \leftarrow S \cup v$ for $v \in N^+(S)$ if we are expected to reduce the cut by adding v. We slightly modified this greedy policy by using Lemma 3.1; after we add a node v to S, we add every node u with a path to v.

For each topology/distribution split pairing, we first computed the lower bound LP value in (7). For this, we used Algorithm 1 with a learning rate of $\gamma = 0.05/\sqrt{\ell}$, a maximum iteration count of $T = 1000$, and a tolerance of $\varepsilon = 10^{-5}$. We used the `pyproximal` library for projection back onto the simplex. Convergence of this nonsummable diminishing step size supergradient ascent is guaranteed by [68].

Our CFA policy implementation (Algorithm 4) slightly differs from Algorithm 2. We order the arcs $(u, v) \in \delta^+(S)$ in increasing order of $\lambda_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] - w_{uv}$, for arcs where this

Algorithm 3 Greedy Policy

$S \leftarrow \{s\}$.
while `not FLAG` **do**
  for $(u, v) \in \delta^+(S)$ **do**
    $S' \leftarrow S \cup v$
    $S' \leftarrow S' \cup \{\text{every node with a path to } v\}$
    $X_v \leftarrow \mathbb{E}[w(\delta^+(S'))|\,\text{realized weights}]$
  $X \leftarrow \min_{v \in N^+(S)} X_v$
  if $X < \sum_{a \in \delta^+(S)} w_a$ **then**
    $S \leftarrow S'$
  else
    `FLAG`

difference is negative. Let $R(v) := \{u : \exists u \to v \text{ path}\}$. Proceeding through the ordered list of nodes, we perform a *secondary* check,

$$\sum_{a \in \delta^+(S)} w_a > \sum_{a \in \delta^+(S \cup v \cup R(v)) \setminus \delta^+(S)} \mathbb{E}[\hat{y}_a(W_a)] + \sum_{a \in \delta^+(S) \cap \delta^+(S \cup v \cup R(v))} w_a;$$

if the secondary check is not satisfied, we move on to the next element in the list. We either add the first node satisfying the secondary check, or we halt. Intuitively, Algorithm 2 treats arcs separately when deciding to add a node; the secondary check exploits the fact that if we add node v, we also add its predecessors that aren't already in the cut (cf. Lemma 3.1). The secondary check assesses how this series of actions would impact the approximate cost-to-go. Empirically, this modification yields a small improvement versus Algorithm 2.

We then estimated the offline expected min-cut and the expected values of Algorithms 3 and 4 using the sample mean of 5,000 simulations per instance. Our choice of 5,000 simulations was due in part to the high variance we observed in sample mean distributions for smaller simulation numbers. Then, for each instance, we calculated three ratios:

1. Algorithm 4 average / maximum of LP (7) and simulated min-cut average;
2. Algorithm 3 average / Algorithm 4 average;
3. LP (7) average / simulated min-cut average.

Together, these ratios characterize the performance of both our lower bound and heuristic policy. We calculated their geometric means and standard deviations across each topology example, and report these in Section 5.3.

Algorithm 4 Improved Cost-to-go Function Approximation-based Policy

Using Algorithm 1, find multipliers $\lambda_a, a \in A$.
$\hat{y} \leftarrow$ optimal point of (7) with λ from step 1.
$S \leftarrow \{s\}$.
$Q \leftarrow \varnothing$ is a priority queue in ascending order
while `not FLAG` **do**
    $Q \leftarrow \left\{(u,v) \in \delta^+(S) : \lambda_{(u,v)} \sum_{a \in \delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] - w_{(u,v)} < 0\right\}$
    if $Q = \varnothing$ **then**
        `FLAG`
    while $Q \neq \varnothing$ **do**
        $(u,v) \leftarrow$ `pop`(Q)
        $S' \leftarrow S \cup v$
        $S' \leftarrow S' \cup \{$every node with a path to $v\}$
        $X \leftarrow \mathbb{E}[\hat{y}(\delta^+(S'))|$ realized weights$]$
        if $X < \sum_{a \in \delta^+(S)} w_a$ **then**
            $S \leftarrow S'$
            `break` inner loop
        else
            `continue`

5.2.1 Technical Specifications

Our experiments were performed on a 2019 MacBook Air with a 1.6GHz Dual-Core Intel i5 processor. The code for calculation of lower bounds and for algorithm implementation was written in Python 3.12.3. We made liberal use of the `NetworkX` and `NumPy` libraries. Our random seed was 30332, the ZIP Code for Georgia Tech.

The solving time was upwards of several hours total for each topology grouping. The test group for $G_0, \ldots, G_9$ had the shortest runtime of approximately three hours total. The longest runtime was for the long chains test suite, which took about 36 hours to run all tests for the varied lengths. The most serious runtime bottleneck was the number of simulations; simplifying the support of our distributions did not have an appreciable effect in decreasing this runtime.

5.3 Results Summary

From our experiments, we conclude that Algorithm 4 not only significantly outperforms the greedy benchmark policy (Algorithm 3), but also greatly outperforms its own worst-case approximation ratio in practice. The greedy policy Algorithm 3 performs much worse than Algorithm 4 in most cases, and the performance of (7) versus the expected min-cut is mixed.

For the benchmark ratios, we recorded the geometric mean and standard deviations. We observe

an interesting trend in the performance ratios: when the geometric mean of a ratio is less than one, most topologies have 100% of instances below one. When it is at least one, this count was also near 100%. Hence the ratio reported, even when close to one, accurately reflects the performance difference in question.

In a few cases, the ratio (Algorithm 4/Best lower bound) is reported as less than one, a theoretical impossibility. We believe this is a simulation artifact, because retesting select instances with $50,000$ simulations (rather than 5,000) yielded ratios of at least one.

5.3.1 Experiments on Varied Series-Parallel Topologies

We first ran our test suite on a collection of small series-parallel graph examples; our results are collected in Table 1. In every case, the geometric mean ratio of Alg. 4 over LB (7) is less than 1.5, peaking for G_7 with a 41.9% gap. This is in stark contrast to the worst-case ratio we proved in Theorem 4.1.

Graph	$\prod \deg^-(v)$	**Alg. 4 / Best LB**	**Alg. 3 / Alg. 4**	**LP (7) / Offline**
G_0	3	1.097 ± 0.013	1.340 ± 0.039	1.051 ± 0.016
G_1	3	1.315 ± 0.015	1.047 ± 0.013	1.108 ± 0.010
G_2	3	1.130 ± 0.010	1.315 ± 0.042	0.945 ± 0.010
G_3	6	1.034 ± 0.003	0.997 ± 0.002	0.970 ± 0.007
G_4	4	1.168 ± 0.013	1.719 ± 0.063	0.957 ± 0.013
G_5	2	1.074 ± 0.008	1.274 ± 0.021	1.105 ± 0.010
G_6	2	1.070 ± 0.010	1.106 ± 0.011	1.182 ± 0.011
G_7	8	1.419 ± 0.015	1.219 ± 0.025	0.970 ± 0.011
G_8	6	1.149 ± 0.015	2.248 ± 0.07	1.023 ± 0.017
G_9	3	1.073 ± 0.005	1.230 ± 0.022	0.981 ± 0.007

Table 1: Geometric Means and Standard Deviations for Ratios on Varied Small Graph Topologies

Algorithm 4 also outperformed Algorithm 3. In each case except G_3, the geometric mean of the ratio of Algorithm 3 over LP (7) is higher, peaking for G_8 at a 124.8% gap. For G_3, the performance is about equal, with Algorithm 3 actually slightly outperforming. The simulated offline min-cut average had varied performance versus the optimum of LP (7).

5.3.2 Experiments on Joined Path Topologies

In our variable-size graphs, we can see some broader trends appear. Our long-path results are in Table 2. The performance of Algorithm 4 generally increases with the path length and as B increases. We attribute this trend to a lesser likelihood of picking the final arc. In that case, our graph topology becomes more similar to disjoint parallel paths, a case where the lower bound should match Algorithm 4. As for Algorithm 3, the performance becomes substantially worse for longer paths, with a near 700% gap in the longest path cases. The optimal policy's expected value becomes much lower for these long paths, whereas Algorithm 3 only factors in locally optimal moves.

Another interesting trend is the performance of the separable LP versus the offline minimum cut. As the path lengths and value of B increase, the LP lower bound seems to steadily grow versus the offline benchmark. When the length of each path increases, the optimal algorithm is less and less likely to take the arc with weight B, because it will find a low-weight group of arcs with higher probability. The LP performance should therefore approach that of the optimal algorithm and the LP expected value is going to tend higher versus the offline minimum-cut. Similarly, as B increases, the optimal algorithm is less and less likely to take the final arc. So the heuristic begins to more closely resemble the optimal algorithm, which is generally much higher than the offline minimum cut.

We also ran our test suite on cases where the path length is fixed at two, but we greatly increased our number of paths to $10, 25, 50$. We summarize our results in Table 3. Broadly, the cost-to-go approximation performs best when the final arc with weight B is either rarely or almost always taken. The algorithms perform much more poorly in the middle range.

Unlike most other cases we tested, the greedy algorithm benchmark performed about the same or better versus our heuristic; we observed up to 9% gaps in these cases. When B is about $5k$, we expect Algorithm 4 to be more eager to move forward to the second arc on the path versus Algorithm 3, because of the minimizing effect of λB after each second arc. As B increases further, this affects Algorithm 4's decision-making less, so the greedy and CFA-approximation performances converge. We don't see this in the long-path case, because in longer paths there is a greater chance for a decrease in arc weights versus the current cut.

The trend of the LP bound's performance versus the offline minimum cut mostly mirrors the long-path case. For similar reasons, the LP value approaches the optimal expected value as B

$\prod \deg^-(v)$	**Path Length**	B	**Alg. 4 / Best LB**	**Alg. 3 / Alg. 4**	**LP (7) / Offline**
2	10	5.0	1.212 ± 0.015	2.003 ± 0.055	1.186 ± 0.014
		10.0	1.107 ± 0.014	1.778 ± 0.048	1.446 ± 0.019
		15.0	1.036 ± 0.012	1.793 ± 0.047	1.531 ± 0.025
	25	5.0	1.155 ± 0.016	3.986 ± 0.128	1.576 ± 0.024
		10.0	1.061 ± 0.015	3.903 ± 0.121	1.756 ± 0.032
		15.0	1.021 ± 0.016	3.961 ± 0.123	1.804 ± 0.028
	50	5.0	1.105 ± 0.021	7.813 ± 0.333	1.891 ± 0.035
		10.0	1.038 ± 0.021	7.786 ± 0.323	2.007 ± 0.044
		15.0	1.011 ± 0.017	7.861 ± 0.297	2.046 ± 0.046
3	10	7.5	1.380 ± 0.013	1.759 ± 0.038	1.180 ± 0.014
		15.0	1.191 ± 0.017	1.656 ± 0.037	1.444 ± 0.013
		22.5	1.033 ± 0.010	1.803 ± 0.040	1.538 ± 0.019
	25	7.5	1.295 ± 0.021	3.568 ± 0.112	1.575 ± 0.021
		15.0	1.114 ± 0.015	3.720 ± 0.106	1.758 ± 0.025
		22.5	1.021 ± 0.012	3.951 ± 0.119	1.804 ± 0.025
	50	7.5	1.200 ± 0.024	7.166 ± 0.224	1.883 ± 0.034
		15.0	1.070 ± 0.025	7.583 ± 0.290	2.015 ± 0.034
		22.5	1.013 ± 0.013	7.862 ± 0.237	2.038 ± 0.036
4	10	10.0	1.519 ± 0.017	1.592 ± 0.030	1.177 ± 0.010
		20.0	1.216 ± 0.015	1.627 ± 0.034	1.445 ± 0.016
		30.0	1.035 ± 0.010	1.802 ± 0.034	1.536 ± 0.016
	25	10.0	1.420 ± 0.023	3.266 ± 0.094	1.573 ± 0.018
		20.0	1.121 ± 0.018	3.719 ± 0.095	1.756 ± 0.022
		30.0	1.017 ± 0.010	3.963 ± 0.087	1.805 ± 0.024
	50	10.0	1.293 ± 0.026	6.693 ± 0.221	1.888 ± 0.026
		20.0	1.075 ± 0.022	7.543 ± 0.228	2.014 ± 0.029
		30.0	1.012 ± 0.012	7.838 ± 0.193	2.043 ± 0.028

Table 2: Geometric Means and Standard Deviations for Ratios on Joined Paths

$\prod \deg^-(v)$	B	**Alg. 4 / Best LB**	**Alg. 3 / Alg. 4**	**LP (7) / Offline**
10	25.0	1.018 ± 0.002	0.999 ± 0.000	0.789 ± 0.004
	50	1.244 ± 0.006	0.932 ± 0.003	0.915 ± 0.005
	75.0	1.050 ± 0.004	0.996 ± 0.001	1.077 ± 0.005
25	62.5	1.003 ± 0.000	1.000 ± 0.000	0.776 ± 0.003
	125	1.306 ± 0.006	0.918 ± 0.002	0.913 ± 0.003
	187.5	1.047 ± 0.002	0.999 ± 0.000	1.077 ± 0.003
50	125.0	1.000 ± 0.000	1.000 ± 0.000	0.771 ± 0.003
	250	1.341 ± 0.005	0.915 ± 0.002	0.914 ± 0.002
	375.0	1.048 ± 0.002	0.999 ± 0.000	1.077 ± 0.002

Table 3: Geometric Means and Standard Deviations for Ratios on Joined Paths

increases. This is in contrast to the trends of our other recorded ratios, which peak with the values of B requiring the most nuanced decision-making.

Interestingly, the number of paths has little effect on the performance of either benchmark in both the longer-path and many-path regimes. Algorithm 3 performs slightly better as the number of paths increases, but the LP stays about the same. Algorithm 4 performs slightly worse as the number of paths increases, which is expected based on the ratio in Theorem 4.1.

5.3.3 Experiments on Long Chains

Links	$\prod \deg^-(v)$	**Alg. 4 / Best LB**	**Alg. 3 / Alg. 4**	**LP (7) / Offline**
5	16	1.484 ± 0.019	1.303 ± 0.033	0.925 ± 0.014
10	512	1.752 ± 0.027	1.598 ± 0.053	0.752 ± 0.011
15	16384	1.991 ± 0.034	1.767 ± 0.060	0.643 ± 0.011
20	524288	2.205 ± 0.040	1.865 ± 0.069	0.569 ± 0.012

Table 4: Geometric Means and Standard Deviations for Ratios on Long Chains

We ran our test suite on chain-link graphs (see Figure 6) with $5, 10, 15, 20$ links. Table 4 shows that the performance of Algorithm 4 progressively worsens as the number of links increases. The gaps for this class of instances is the worst of any in our experiments, increasing from 48.4% with five links up to 120.5% when there are 20 links. This aligns with the intuition from Example 4.2. Furthermore, the greedy policy's performance drops off much faster than Algorithm 4.

As the number of links increases, the LP also performs very poorly versus the expected offline minimum cut. With 20 links, the LP is just above half the offline benchmark. Again, this matches our example, where the LP has a multiplicative performance drop-off for Bernoulli distributions.

6 Conclusion

We introduced the Dynamic Cut Minimization (DCM) problem on random graphs, a generalization of both the deterministic minimum cut problem and the minimization version of the optimal stopping problem from the prophet inequality literature. We showed that in contrast to deterministic minimum cut, finding an optimal policy for DCM is #P-hard even on very simple graph topologies and arc weight distributions. Then, we approximated the optimal policy using an approximate linear programming approach and built a corresponding heuristic policy.

There are a plethora of unanswered questions about DCM problems on specific classes of graphs. We showed in Section 2 that, even when restricted to one- or two-point distributions, the expected value of an optimal policy for DCM may be arbitrarily larger than the offline optimal. However, it is possible that we may see bounded ratios when restricting the class of arc weight distributions. Livanos and Mehta [49] investigate asymptotic prophet inequality bounds for specific independent and identically distributed (i.i.d.) instances of the optimal stopping minimization setup. The i.i.d. assumption is a natural next step. To further the connection to network reliability literature, another class of distributions to consider is unweighted Bernoulli arcs.

Aside from other random variable distributions, an additional direction of work is non- series-parallel graph topologies. For example, Cristi and Oren [16] develop a prophet-inequality framework on graphs where all vertices lie on a directed path. Finally, our separable LP approximation can perform quite poorly on graphs with large in-degrees. While the optimal policy is theoretically intractable, there may be better approximations for graphs with larger in-degrees. These lower and upper bounds would likely require a different approach with a different approximation architecture.

7 Acknowledgments

The first author's work was partially supported by the National Science Foundation via a Graduate Research Fellowship, DGE-2039655. The authors' work was also partially supported by the Air Force Office of Scientific Research, grant FA9550-25-1-0341.

References

[1] A. Abdolahzadeh, M. Aman, and J. Tayyebi. Minimum *st*-cut interdiction problem. *Computers & Industrial Engineering*, 148:106708, 08 2020. doi: 10.1016/j.cie.2020.106708.

[2] J. A. Abraham. An improved algorithm for network reliability. *IEEE Transactions on Reliability*, 28 (1):58–61, 1979.

[3] D. Adelman. Dynamic bid prices in revenue management. *Operations Research*, 55:647–661, 2007.

[4] A. Agrawal and R. E. Barlow. A survey of network reliability and domination theory. *Operations Research*, 32(3):478–492, 1984.

[5] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, et al. *Network flows: theory, algorithms, and applications*, volume 1. Prentice hall Englewood Cliffs, NJ, 1993.

[6] H. Bayrak and M. D. Bailey. Shortest path network interdiction with asymmetric information. *Networks: An International Journal*, 52(3):133–140, 2008.

[7] M. G. Bell. Measuring network reliability: a game theoretic approach. *Journal of advanced transportation*, 33(2):135–146, 1999.

[8] D. Blado and A. Toriello. Relaxation analysis for the dynamic knapsack problem with stochastic item sizes. *SIAM Journal on Optimization*, 29(1):1–30, 2019.

[9] D. Blado and A. Toriello. A column and constraint generation algorithm for the dynamic knapsack problem with stochastic item sizes. *Mathematical Programming Computation*, 13:185–223, 2021.

[10] D. Blado, W. Hu, and A. Toriello. Semi-infinite relaxations for the dynamic knapsack problem with stochastic item sizes. *SIAM Journal on Optimization*, 26(3):1625–1648, 2016.

[11] B. Chaourar. A linear time algorithm for a variant of the max cut problem in series parallel graphs. *Advances in Operations Research*, 2017(1):1267108, 2017.

[12] L. Chen, R. Kyng, Y. Liu, R. Peng, M. Probst Gutenberg, and S. Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. *Journal of the ACM*, 2022.

[13] J. Correa, P. Foncea, R. Hoeksma, T. Oosterwijk, and T. Vredeveld. Posted price mechanisms for a random stream of customers. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 169–186, 2017.

[14] J. Correa, P. Foncea, R. Hoeksma, T. Oosterwijk, and T. Vredeveld. Recent developments in prophet inequalities. *ACM SIGecom Exchanges*, 17(1):61–70, 2019.

[15] J. Correa, P. Foncea, D. Pizarro, and V. Verdugo. From pricing to prophets, and back! *Operations Research Letters*, 47(1):25–29, 2019.

[16] A. Cristi and S. Oren. Planning against a prophet: a graph-theoretic framework for making sequential decisions. In *Proceedings of the 25th ACM Conference on Economics and Computation*, EC '24, page 806, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400707049. doi: 10.1145/3670865.3673625. URL `https://doi.org/10.1145/3670865.3673625`.

[17] O. Cruz-Mejía and A. N. Letchford. A survey on exact algorithms for the maximum flow and minimum-cost flow problems. *Networks*, 82(2):167–176, 2023.

[18] D. P. De Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations research*, 51(6):850–865, 2003.

[19] D. Deif and Y. Gadallah. A comprehensive wireless sensor network reliability metric for critical internet of things applications. *EURASIP Journal on Wireless Communications and Networking*, 2017:1–18, 2017.

[20] R. J. Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10(2):303–318, 1965.

[21] M. Dyer, A. Frieze, R. Kannan, A. Kapoor, L. Perkovic, and U. Vazirani. A mildly exponential time algorithm for approximating the number of solutions to a multidimensional knapsack problem. *Combinatorics, Probability and Computing*, 2(3):271–284, 1993.

[22] S. Ehsani, M. T. Hajiaghayi, T. Kesselheim, and S. Singla. Prophet secretary for combinatorial auctions and matroids. In *Proceedings of the twenty-ninth annual acm-siam symposium on discrete algorithms*, pages 700–714. SIAM, 2018.

[23] T. Elperin, I. Gertsbakh, and M. Lomonosov. Estimation of network reliability using graph evolution models. *IEEE Transactions on Reliability*, 40(5):572–581, 1991.

[24] D. Eppstein. Parallel recognition of series-parallel graphs. *Information and Computation*, 98(1):41–55, 1992.

[25] T. Ezra, M. Feldman, N. Gravin, and Z. G. Tang. Prophet matching with general arrivals. *Mathematics of Operations Research*, 47(2):878–898, 2022.

[26] L. R. Ford Jr and D. R. Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8:399–404, 1956.

[27] L. Fu, X. Wang, and P. Kumar. Optimal determination of source-destination connectivity in random graphs. In *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*, pages 205–214, 2014.

[28] L. Fu, X. Fu, Z. Xu, Q. Peng, X. Wang, and S. Lu. Determining source–destination connectivity in uncertain networks: Modeling and solutions. *IEEE/ACM Transactions on Networking*, 25(6):3237–3252, 2017.

[29] V. Gaur, O. P. Yadav, G. Soni, and A. P. S. Rathore. A literature review on network reliability analysis and its engineering applications. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 235(2):167–181, 2021.

[30] I. B. Gertsbakh and Y. Shpungin. *Models of network reliability: analysis, combinatorics, and Monte Carlo*. CRC press, 2016.

[31] P. M. Ghare, D. C. Montgomery, and W. C. Turner. Optimal interdiction policy for a flow network. *Naval Research Logistics Quarterly*, 18(1):37–45, 1971.

[32] T. Hill and R. Kertz. Ratio comparisons of supremum and stop rule expectations. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 56:283–285, 1981.

[33] T. P. Hill and R. P. Kertz. Comparisons of stop rule and supremum expectations of i.i.d. random variables. *The Annals of Probability*, pages 336–345, 1982.

[34] D. S. Hochbaum. The pseudoflow algorithm: A new algorithm for the maximum-flow problem. *Operations research*, 56(4):992–1009, 2008.

[35] D. S. Hochbaum and J. B. Orlin. Simplifications and speedups of the pseudoflow algorithm. *Networks*, 61(1):40–57, 2013.

[36] T. Holzmann and J. C. Smith. The shortest path interdiction problem with randomized interdiction strategies: Complexity and algorithms. *Operations Research*, 69(1):82–99, 2021.

[37] E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks: An International Journal*, 40(2):97–111, 2002.

[38] D. P. Kennedy. Prophet-type inequalities for multi-choice optimal stopping. *Stochastic Processes and their applications*, 24(1):77–88, 1987.

[39] L. Khachiyan, E. Boros, K. Borys, K. Elbassioni, V. Gurvich, G. Rudolf, and J. Zhao. On short paths interdiction problems: Total and node-wise limited interdiction. *Theory of Computing Systems*, 43(2): 204–233, 2008.

[40] T. Kikuno, N. Yoshida, and Y. Kakuda. A linear algorithm for the domination number of a series-parallel graph. *Discrete Applied Mathematics*, 5(3):299–311, 1983.

[41] J. Kleinberg and S. Oren. Time-inconsistent planning: a computational problem in behavioral economics. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 547–564, 2014.

[42] R. Kleinberg and S. M. Weinberg. Matroid prophet inequalities. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 123–136, 2012.

[43] A. J. Kleywegt and J. D. Papastavrou. The dynamic and stochastic knapsack problem. *Operations research*, 46(1):17–35, 1998.

[44] H. J. Kowshik. *Information aggregation in sensor networks*. University of Illinois at Urbana-Champaign, 2011.

[45] U. Krengel and L. Sucheston. On semiamarts, amarts, and processes with finite value. In *Probability on Banach spaces*, volume 4 of *Adv. Probab. Related Topics*, pages 197–266. Dekker, New York, 1978.

[46] F. L. Lewis and D. Liu. *Reinforcement learning and approximate dynamic programming for feedback control*. John Wiley & Sons, 2013.

[47] F. L. Lewis and D. Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE circuits and systems magazine*, 9(3):32–50, 2009.

[48] C. Lim and J. C. Smith. Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26, 2007.

[49] V. Livanos and R. Mehta. Minimization is harder in the prophet world. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 424–461. SIAM, Philadelphia, PA, 2024. doi: 10.1137/1.9781611977912.17. URL `https://doi.org/10.1137/1.9781611977912.17`.

[50] V. Livanos and R. Mehta. Minimization iid prophet inequality via extreme value theory: A unified approach. In *Proceedings of the 26th ACM Conference on Economics and Computation*, pages 1157–1179, 2025.

[51] Z. Lomnicki. Two-terminal series-parallel networks. *Advances in Applied Probability*, 4(1):109–150, 1972.

[52] A. Manne. Linear Programming and Sequential Decisions. *Management Science*, 6:259–267, 1960.

[53] C. Muir and A. Toriello. Dynamic node packing. *Mathematical Programming*, pages 1–32, 2022.

[54] A. T. Murray, T. C. Matisziw, and T. H. Grubesic. Critical network infrastructure analysis: interdiction and system flow. *Journal of Geographical Systems*, 9(2):103–117, 2007.

[55] Y.-F. Niu, W. H. Lam, and Z. Gao. An efficient algorithm for evaluating logistics network reliability subject to distribution cost. *Transportation Research Part E: Logistics and Transportation Review*, 67: 175–189, 2014.

[56] G. A. Pagani and M. Aiello. The power grid as a complex network: a survey. *Physica A: Statistical Mechanics and its Applications*, 392(11):2688–2700, 2013.

[57] C. Papadimitriou, T. Pollner, A. Saberi, and D. Wajc. Online stochastic max-weight bipartite matching: Beyond prophet inequalities. In *Proceedings of the 22nd ACM Conference on Economics and Computation*, pages 763–764, 2021.

[58] P. Peng, L. V. Snyder, A. Lim, and Z. Liu. Reliable logistics networks design with facility disruptions. *Transportation Research Part B: Methodological*, 45(8):1190–1211, 2011.

[59] H. Pérez-Rosés. Sixty years of network reliability. *Mathematics in Computer Science*, 12:275–293, 2018.

[60] S. Perez-Salazar, M. Singh, and A. Toriello. The i.i.d. prophet inequality with limited flexibility. *Mathematics of Operations Research*, 2025.

[61] J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.*, 12(4):777–788, 1983. ISSN 0097-5397. doi: 10.1137/0212053. URL `https://doi.org/10.1137/0212053`.

[62] J. Qin, S. Vardi, and A. Wierman. Minimization fractional prophet inequalities for sequential procurement. *Mathematics of Operations Research*, 49(2):928–947, 2024.

[63] M. A. Rad and H. T. Kakhki. Maximum dynamic network flow interdiction problem: New formulation and solution procedures. *Computers & Industrial Engineering*, 65(4):531–536, 2013.

[64] E. Samuel-Cahn. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *Ann. Probab.*, 12(4):1213–1216, 1984. ISSN 0091-1798. URL http://links.jstor.org/sici?sici=0091-1798(198411)12:4<1213:COTSRA>2.0.CO;2-N&origin=MSN.

[65] A. Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

[66] P. J. Schweitzer and A. Seidmann. Generalized polynomial approximations in markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110(2):568–582, 1985. ISSN 0022-247X. doi: https://doi.org/10.1016/0022-247X(85)90317-8. URL https://www.sciencedirect.com/science/article/pii/0022247X85903178.

[67] J. A. Sefair and J. C. Smith. Dynamic shortest-path interdiction. *Networks*, 68(4):315–330, 2016.

[68] N. Z. Shor. *Minimization methods for non-differentiable functions*, volume 3. Springer Science & Business Media, 2012.

[69] J. C. Smith and Y. Song. A survey of network interdiction models and algorithms. *European Journal of Operational Research*, 283(3):797–811, 2020.

[70] A. Toriello, W. Haskell, and M. Poremba. A Dynamic Traveling Salesman Problem with Stochastic Arc Costs. *Operations Research*, 62:1107–1125, 2014.

[71] A. Torrico and A. Toriello. Dynamic Relaxations for Online Bipartite Matching. *INFORMS Journal on Computing*, 34:1871–1884, 2022.

[72] A. Torrico, S. Ahmed, and A. Toriello. A polyhedral approach to online bipartite matching. *Mathematical Programming*, 172:443–465, 2018.

[73] M. A. Trick and S. E. Zin. Spline approximations to value functions: linear programming approach. *Macroeconomic Dynamics*, 1(1):255–277, 1997.

[74] J. Valdes, R. E. Tarjan, and E. L. Lawler. The recognition of series parallel digraphs. In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 1–12, 1979.

[75] J. Van Den Brand, L. Chen, R. Kyng, Y. P. Liu, R. Peng, M. P. Gutenberg, S. Sachdeva, and A. Sidford. A deterministic almost-linear time algorithm for minimum-cost flow. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 503–514. IEEE, 2023.

[76] R. Wilkov. Analysis and design of reliable computer networks. *IEEE Transactions on Communications*, 20(3):660–678, 1972.

A Proofs

A.1 Proof of Theorem 3.4

Proof. We may assume without loss of generality that $0 \in \mathcal{W}_a$ for all $a \in A$. Indeed, we can always add 0 to the support of an arc's weight and give it probability zero of appearing. This will not change the optimum of (4).

Let z^* be the optimal value of (4) and let $\hat{z}$ be the optimal value of (5)/(7). Since every feasible point of (5) is feasible for (4), $z^* \geq \hat{z}$. Let y^* be an optimal point for (4). Since every vertex aside from t has in-degree at most 1, the number of paths k from s to t is exactly $\deg^-(t)$. Number these paths $P_1, \ldots, P_k$ arbitrarily. Define the *index* of each arc $(u, v) \in A$ as

$$i(u, v) := \min_{j=1\ldots k} \{P_j \text{ passes through } (u, v)\}.$$

Without loss of optimality, Lemma 3.1 implies we need only consider the constraints (in either LP) for vertex sets S forming arborescence subgraphs of D with root s. Define

$$S(u, v) := \arg\min_{S \ni s} \big\{|S| : \delta^+(S) \ni (u, v), S \text{ is reachable from } s\big\}.$$

With these definitions, we set

$$\begin{aligned}\hat{y}_{uv}(w_{uv}) := \mathbb{E}\Big[y^*_{S(u,v)}\Big(0, \ldots, 0, \overbrace{w_{uv}}^{i(u,v)}, W_{a_j}, \ldots, W_{a_{\deg^+(S(u,v))}}\Big)\Big] \\ - \mathbb{E}\Big[y^*_{S(u,v)}\Big(0, \ldots, 0, \overbrace{0}^{i(u,v)}, W_{a_j}, \ldots, W_{a_{\deg^+(S(u,v))}}\Big)\Big],\end{aligned}$$

where the weight tuple in y^* is in increasing order of the indices of the arcs in $\delta^+(S(u, v))$. We now verify that $\hat{y}$ is feasible for (7). Then, by Theorem 3.2, $\hat{y}$ is feasible for (5).

For all (u, v) and w_{uv}, we claim (7b) holds. Indeed, one can create a policy which 'pretends' $w_{uv} = 0$ and otherwise mimics the optimal policy. For any realization of weights, the working sets S are the same as the optimal where $w_{uv} = 0$. The only difference is the arc (u, v) contributing w_{uv} to the final $w(\delta^+(S))$. This policy is suboptimal, so the optimal policy adds at most w_{uv} to the final weight versus if $w_{uv} = 0$.

As for (7c), by the structure of the graph, we notice that $\lambda_a = 1$ for all $a \in A$. Moreover, if an edge (u, v) is currently in $\delta^+(S)$, then no other edge in $\delta^+(S)$ has head v. By collapsing a telescoping sum,

$$\sum_{e \in \delta^+(v)} \mathbb{E}[\hat{y}_e(W_e)] = \sum_{e \in \delta^+(v)} \left(\mathbb{E}\Big[y^*_{S(e)}\Big(0, \ldots, 0, \overbrace{0, \ldots, W_e, \ldots}^{\in \delta^+(v)}, W_{a_j}, \ldots, W_{a_{\deg^+(S(u,v) \cup v)}}\Big)\Big]\right.$$

$$
\begin{aligned}
&\qquad - \mathbb{E}\Big[y^*_{S(e)}\Big(0,\dots,0,\overbrace{0,\dots,0,\dots}^{\in\delta^+(v)},W_{a_j},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big]\Bigg)\\
&= \mathbb{E}\Big[y^*_{S(u,v)\cup v}\Big(0,\dots,0,W_{e_1},\dots,W_{\deg^+(v)},W_{a_1},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big]\\
&\qquad - \mathbb{E}\Big[y^*_{S(u,v)\cup v}\Big(0,\dots,0,0,\dots,0,W_{a_1},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big].
\end{aligned}
$$

We used here that $S(e) = S(u,v)\cup v$ for all $e\in\delta^+(v)$. We also have

$$
\begin{aligned}
&\mathbb{E}\Big[y^*_{S(u,v)\cup v}\Big(0,\dots,0,\overbrace{0,\dots,0}^{\delta^+(v)},W_{a_1},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big]\\
&\qquad = \mathbb{E}\Big[y^*_{S(u,v)}\Big(0,\dots,0,\overbrace{0}^{(u,v)},W_{a_1},\dots,W_{a_{\deg^+(S(u,v))}}\Big)\Big]. \qquad (13)
\end{aligned}
$$

Indeed, when $w_{uv}=0$, an optimal decision-maker will never add v to S, and the subgraph of reachable vertices from this arc will contribute weight 0 to the final cut. The same applies to when all the weight out of v is 0 as well. This leads to

$$
\begin{aligned}
\hat{y}_{uv}(w_{uv}) &= \mathbb{E}\Big[y^*_{S(u,v)}\Big(0,\dots,0,w_{uv},W_{a_j},\dots,W_{a_{\deg^+(S(u,v))}}\Big)\Big]\\
&\qquad - \mathbb{E}\Big[y^*_{S(u,v)}\Big(0,\dots,0,0,W_{a_j},\dots,W_{a_{\deg^+(S(u,v))}}\Big)\Big]\\
\text{via (4c)}\quad &\le \mathbb{E}\Big[y^*_{S(u,v)\cup v}\Big(0,\dots,0,W_{e_1},\dots,W_{\deg^+(v)},W_{a_1},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big]\\
&\qquad - \mathbb{E}\Big[y^*_{S(u,v)}\Big(0,\dots,0,0,W_{a_1},\dots,W_{a_{\deg^+(S(u,v))}}\Big)\Big]\\
\text{via (13)}\quad &= \mathbb{E}\Big[y^*_{S(u,v)\cup v}\Big(0,\dots,0,W_{e_1},\dots,W_{\deg^+(v)},W_{a_1},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big]\\
&\qquad - \mathbb{E}\Big[y^*_{S(u,v)\cup v}\Big(0,\dots,0,W_{a_1},\dots,W_{a_{\deg^+(S(u,v)\cup v)}}\Big)\Big]\\
&= \sum_{e\in\delta^+(v)}\mathbb{E}[\hat{y}_e(W_e)].
\end{aligned}
$$

The last line follows by expanding the difference into a telescoping sum. Hence $\hat{y}$ satisfies (7c), rounding off the proof that $\hat{y}$ is feasible for (7).

Every arc $a\in\delta^+(s)$ has $S(a)=\{s\}$ by minimality. So

$$
\sum_{a\in\delta^+(s)}\mathbb{E}[\hat{y}_a(W_a)] = \sum_{j=1}^{\deg^+(s)}\Bigg(\mathbb{E}\Big[y^*_{\{s\}}\Big(0,\dots,0,W_{a_j},W_{a_{j+1}},\dots,W_{a_{\deg^+(s)}}\Big)\Big]
$$

$$- \mathbb{E}\Big[y^*_{\{s\}}\Big(0,\dots,0,0,W_{a_{j+1}},\dots,W_{a_{\deg^+(s)}}\Big)\Big]\Bigg)$$

$$\text{telescoping sum} \qquad = \mathbb{E}\Big[y^*_{\{s\}}\Big(W_{a_1},\dots,W_{a_{\deg^+(s)}}\Big)\Big] - y^*_{\{s\}}(0,\dots,0) = z^* - 0 = z^*,$$

meaning $\hat{y}$ has objective value z^* when plugged into (5a). Since $z^* \geq \hat{z}$ but a feasible point of (7) has objective z^*, $\hat{z} = z^*$, as desired. □

A.2 Proof of Lemma 3.6

Proof. Let y^1 be an optimal point of (7) when $\lambda = \lambda^1$ and y^2 the optimal when $\lambda = \lambda^2$, where λ^1 and λ^2 differ *only* at $\{(u,v) \in \delta^+(v)\}$. We treat λ^1, λ^2 as vectors with dimension $|A|$.

We first claim that

$$\mathbb{E}\big[y^2_{uv}(W_{uv}) - y^1_{uv}(W_{uv})\big] \leq \big(\lambda^2_{uv} - \lambda^1_{uv}\big)\mathbb{P}\left[W_{uv} > \lambda^1_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]\right]\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big] \quad (14)$$

for all $(u,v) \in \delta^-(v)$.

Assume first that $\lambda^2_{uv} > \lambda^1_{uv}$. If $a \in A$ is such that there is no path from s to v including a, $y^1_a(w) = y^2_a(w)$ for all $w \in \mathcal{W}_a$. So the change in λ_{uv} does not affect any y_a for a 'downstream,' i.e. reachable from v. So for fixed $w \in \mathcal{W}_{uv}$,

$$\min\Big\{w, \lambda^2_{uv}\textstyle\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^2_a(W_a)\big]\Big\} = \min\Big\{w, \lambda^2_{uv}\textstyle\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]\Big\}.$$

This implies

$$\frac{y^2_{uv}(w) - y^1_{uv}(w)}{\lambda^2_{uv} - \lambda^1_{uv}} = \frac{\min\Big\{w, \lambda^2_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]\Big\} - \min\Big\{w, \lambda^1_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]\Big\}}{\lambda^2_{uv} - \lambda^1_{uv}}.$$

We can expand this out as

$$\begin{aligned}\frac{y^2_{uv}(w) - y^1_{uv}(w)}{\lambda^2_{uv} - \lambda^1_{uv}} = \mathbf{1}&\left[w > \lambda^2_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]\right]\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big] \\ &+ \mathbf{1}\left[\lambda^2_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big] \geq w > \lambda^1_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]\right] \\ &\cdot \frac{w - \lambda^1_{uv}\sum_{a\in\delta^+(v)}\mathbb{E}\big[y^1_a(W_a)\big]}{\lambda^2_{uv} - \lambda^1_{uv}}\end{aligned}$$

$$+ \frac{w - w}{\lambda^2_{uv} - \lambda^1_{uv}}.$$

Using $w \le \lambda^2_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]$ in the middle summand, we can combine the the two indicator variables and see

$$\frac{y^2_{uv}(w) - y^1_{uv}(w)}{\lambda^2_{uv} - \lambda^1_{uv}} \le \mathbf{1}\left[w > \lambda^1_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]$$

Hence, taking the expected value over w,

$$\mathbb{E}\big[y^2_{uv}(W_{uv}) - y^1_{uv}(W_{uv})\big] \le \big(\lambda^2_{uv} - \lambda^1_{uv}\big)\mathbb{P}\left[W_{uv} > \lambda^1_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big].$$

If instead $\lambda^2_{uv} < \lambda^1_{uv}$, we have

$$\begin{aligned}\frac{y^2_{uv}(w) - y^1_{uv}(w)}{\lambda^2_{uv} - \lambda^1_{uv}} = \mathbf{1}&\left[w > \lambda^1_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big] \\ &+ \mathbf{1}\left[\lambda^1_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big] \ge w > \lambda^2_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \\ &\quad \cdot \frac{\lambda^2_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big] - w}{\lambda^2_{uv} - \lambda^1_{uv}} \\ &+ \frac{w - w}{\lambda^2_{uv} - \lambda^1_{uv}}.\end{aligned}$$

Using $w \le y^1_a(W_a)\lambda^1_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]$ in the middle summand, we can combine the indicators to yield

$$\frac{y^2_{uv}(w) - y^1_{uv}(w)}{\lambda^2_{uv} - \lambda^1_{uv}} \ge \mathbf{1}\left[w > \lambda^2_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big].$$

We take the expectation over w to give us

$$\frac{\mathbb{E}\big[y^2_{uv}(W_{uv}) - y^1_{uv}(W_{uv})\big]}{\lambda^2_{uv} - \lambda^1_{uv}} \ge \mathbb{P}\left[W_{uv} > \lambda^2_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big].$$

But since $\lambda^2_{uv} < \lambda^1_{uv}$, the probability that W_{uv} is greater than $\lambda^2_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]$ is *greater* than the probability that W_{uv} is greater than $\lambda^1_{uv} \sum_{a \in \delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]$. And since all terms on the

right side are positive,

$$\frac{\mathbb{E}\big[y^2_{uv}(W_{uv}) - y^1_{uv}(W_{uv})\big]}{\lambda^2_{uv} - \lambda^1_{uv}} \geq \mathbb{P}\left[W_{uv} > \lambda^1_{uv} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big].$$

Multiplying by the negative number $\big(\lambda^2_{uv} - \lambda^1_{uv}\big)$ on both sides yields (14). Finally, if $\lambda^2_{uv} = \lambda^1_{uv}$, (14) is $0 \leq 0$, which is of course true. Hence the equation is valid in all cases.

We now claim that for any $(r, q) \in A \setminus \delta^-(v)$,

$$\mathbb{E}\big[y^2_{rq}(W_{rq}) - y^1_{rq}(W_{rq})\big] \leq \lambda^1_{rq}\mathbb{P}\Big[W_{rq} > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big]\Big] \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a) - y^1_a(W_a)\big]. \quad (15)$$

Note that $\lambda^1_{rq} = \lambda^2_{rq}$, since λ^2 only differs from λ^1 at the arcs into v. We again split into cases. First suppose $\sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big] < \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]$. We have

$$\begin{aligned}
y^2_{rq}(w) - y^1_{rq}(w) &= \min\Big\{w, \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]\Big\} - \min\Big\{w, \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big]\Big\} \\
&= \mathbf{1}\Big[w > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]\Big]\lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^2_a(W_a) - y^1_a(W_a)\big] \\
&\quad + \mathbf{1}\Big[\lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big] \geq w > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big]\Big] \\
&\quad \cdot \Big(w - \lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\Big).
\end{aligned}$$

Using $w \leq \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]$, we combine the indicator terms and obtain

$$y^2_{rq}(w) - y^1_{rq}(w) \leq \mathbf{1}\Big[w > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big]\Big]\lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^2_a(W_a) - y^1_a(W_a)\big]$$

Taking the expected value over w verifies (15).

Now suppose that $\sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big] > \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]$. Then

$$\begin{aligned}
y^2_{rq}(w) - y^1_{rq}(w) &= \mathbf{1}\Big[w > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big]\Big]\lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^2_a(W_a) - y^1_a(W_a)\big] \\
&\quad + \mathbf{1}\Big[\sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big] \geq w > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]\Big]
\end{aligned}$$

$$\cdot \Big(\overbrace{\lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^2_a(W_a)\big] - w}^{\leq 0} \Big)$$
$$\leq \mathbf{1}\Big[w > \lambda^1_{rq} \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big]\Big] \lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^2_a(W_a) - y^1_a(W_a)\big].$$

Again, taking the expectation verifies (15). Finally, if $\sum_{a\in\delta^+(q)} \mathbb{E}\big[y^1_a(W_a)\big] = \sum_{a\in\delta^+(q)} \mathbb{E}\big[y^2_a(W_a)\big]$, then the thresholds in (8) are the same and so (15) reduces to $0 \leq 0$.

We now prove the main statement of the theorem. Let r be a vertex with a path to v. We claim

$$\sum_{a\in\delta^+(r)} \Big(\mathbb{E}\big[y^2(W_a)\big] - \mathbb{E}\big[y^1(W_a)\big]\Big)$$
$$\leq \sum_{u\in N^-(v)} \big(\lambda^2_{uv} - \lambda^1_{uv}\big) \sum_{\substack{r\text{-}v \text{ path } P \\ P\ni(u,v)}} p(v,P) \prod_{a\in P\setminus(u,v)} \lambda^1_a \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]. \quad (16)$$

Note that in the case that $r = s$, (16) reduces to the statement of the theorem. Indeed, we can rewrite (16) as

$$\sum_{a\in\delta^+(s)} \mathbb{E}\big[y^2(W_a)\big] - \sum_{a\in\delta^+(s)} \mathbb{E}\big[y^1(W_a)\big]$$
$$\leq \Big\langle \big(\lambda^2_{uv} - \lambda^1_{uv}\big)_{u\in N^-(v)}, \Big(\sum_{\substack{s\text{-}v \text{ path } P \\ P\ni(u,v)}} p(v,P) \prod_{a\in P\setminus(u,v)} \lambda^1_a \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big] \Big)_{u\in N^-(v)} \Big\rangle,$$

which is the definition of a supergradient.

Let $d(u,v)$ be the maximum number of arcs in a path from u to v. We prove (16) using induction on the $d(r,v)$ from r to v. If $d(r,v) = 1$, then $r \in N^-(v)$ and (14) is exactly (16). So suppose (16) holds for all vertices q with $d(q,v) < d(r,v)$. Since $d(q,v) < d(r,v)$ for all $q \in N^+(r)$,

$$\sum_{(r,q)\in\delta^+(r)} \Big(\mathbb{E}\big[y^2(W_a)\big] - \mathbb{E}\big[y^1(W_a)\big]\Big)$$
$$\text{(via (15))} \leq \sum_{(r,q)\in\delta^+(r)} \lambda^1_{rq} \cdot \mathbb{P}\left[W_{rq} > \lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right] \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^2_a(W_a) - y^1_a(W_a)\big]$$
$$\text{(Induction)} \leq \sum_{(r,q)\in\delta^+(r)} \lambda^1_{rq} \cdot \mathbb{P}\left[W_{rq} > \lambda^1_{rq} \sum_{a\in\delta^+(v)} \mathbb{E}\big[y^1_a(W_a)\big]\right]$$

$$\cdot \sum_{u \in N^-(v)} \left(\lambda^2_{uv} - \lambda^1_{uv}\right) \sum_{\substack{q\text{-}v \text{ path } P \\ P \ni (u,v)}} p(v,P) \prod_{a \in P \setminus (u,v)} \lambda^1_a \sum_{a \in \delta^+(v)} \mathbb{E}\left[y^1_a(W_a)\right]$$

$$= \sum_{u \in N^-(v)} \left(\lambda^2_{uv} - \lambda^1_{uv}\right) \sum_{\substack{r\text{-}v \text{ path } P \\ P \ni (u,v)}} p(v,P) \prod_{a \in P \setminus (u,v)} \lambda^1_a \sum_{a \in \delta^+(v)} \mathbb{E}\left[y^1_a(W_a)\right],$$

as desired. □

A.3 Proof of Theorem 4.1

Proof. We prove assuming finite or countable supports. The proof is identical for continuous distributions, albeit with integral notation. Consider any arc (u, v). To have (u, v) in the final cut from Algorithm 2, we must have all of the following:

A : Reach the arc (u, v) by making the choice $S \leftarrow S \cup u$ in Algorithm 2.

B : Conditioned on reaching the arc (u, v), choose to stop at the arc (u, v) by **not** making the choice $S \leftarrow S \cup v$.

C : Retaining the arc (u, v) by not taking v when we add x, where v has a path to x.

For brevity, we denote $\mathbb{P}\left[B_{(r,u)}\right] := \sum_{w \in \mathcal{W}_{(r,u)}} \mathbb{P}[W_{ru} = w]\mathbb{P}\left[B_{(r,u),w}\right]$, which is the probability of the event B at arc (u, v). In a deterministic optimal policy, $\mathbb{P}\left[B_{(u,v),w}\right]$ is always either 0 or 1. By Lemma 2.3, event C does not occur for (u, v) where $u \notin S$. So in order to reach an arc, we must do so using only additions from events $A \cap \neg B$. These are made independently by each arc in $\delta^-(u)$ by design. By the union bound,

$$\mathbb{P}\left[A_{(u,v)}\right] = \mathbb{P}\left[\bigcup_{r \in N^-(u)} \left(A_{(r,u)} \cap \neg B_{(r,u)}\right)\right] \leq \sum_{r \in N^-(u)} (1 - \mathbb{P}\left[B_{(r,u)}\right])\mathbb{P}\left[A_{(r,u)}\right]. \tag{17}$$

We claim

$$\mathbb{P}\left[A_{(u,v)}\right] \leq \max_{s\text{-}u \text{ path } P} \prod_{x \in P} \deg^-(x) \cdot \sum_{s\text{-}u \text{ path } P} p(u,P) \prod_{a \in e(P)} \lambda_a. \tag{18}$$

To prove this, we proceed by induction on the maximum number of arcs in a path from s to u, denoted $d(s, u)$. If $d(s, u) = 1$, then there is only one path $P = (s, u)$ from s to u because we assume our graphs are simple. Furthermore, $\deg^-(u) = 1$, and so the product of the λ_a's is 1. Assume now

that (18) holds for all vertices r with $d(s,r) < d(s,u)$. Using (17) with the induction step, we have

$$\begin{aligned}
\mathbb{P}\big[A_{(u,v)}\big] &\le \sum_{r\in N^-(u)} (1-\mathbb{P}\big[B_{(r,u)}\big]) \\
&\qquad \cdot \max_{s\text{-}r \text{ path } P} \prod_{x\in P} \deg^-(x) \cdot \sum_{s\text{-}r \text{ path } P} p(r,P) \prod_{a\in e(P)} \lambda_a \\
&= \sum_{r\in N^-(u)} \max_{s\text{-}r \text{ path } P} \prod_{x\in P} \deg^-(x) \\
&\qquad \cdot \sum_{s\text{-}r \text{ path } P} \big(1-\mathbb{P}\big[B_{(r,u)}\big]\big) p(r,P) \prod_{a\in e(P)} \lambda_a \\
&= \sum_{r\in N^-(u)} \max_{s\text{-}r \text{ path } P} \prod_{x\in P} \deg^-(x) \\
&\qquad \cdot \sum_{(s\text{-}r \text{ path } P)\cup(r,u)} p(u, P\cup(r,u)) \prod_{a\in e(P)} \lambda_a.
\end{aligned}$$

We subsumed $(1-\mathbb{P}\big[B_{(r,u)}\big])$ into $p(r,P)$ using the definition of $p(\cdot,\cdot)$. By Corollary 3.7, all of the summands in the final line are the same. Hence any linear combination is the same as each individual. So using $\sum_{(r,u)\in\delta^-(u)} \lambda_{ru} = 1$,

$$\begin{aligned}
\mathbb{P}\big[A_{(u,v)}\big] &\le \sum_{r\in N^-(u)} \max_{s\text{-}r \text{ path } P} \prod_{x\in P} \deg^-(x) \\
&\qquad \cdot \sum_{r\in N^-(u)} \lambda_{ru} \sum_{(s\text{-}r \text{ path } P)\cup(r,u)} p(u, P\cup(r,u)) \prod_{a\in e(P)} \lambda_a \\
&\le \deg^-(u) \cdot \max_{r\in N^-(u)} \left\{ \max_{s\text{-}r \text{ path } P} \prod_{x\in P} \deg^-(x) \right\} \\
&\qquad \cdot \sum_{r\in N^-(u)} \lambda_{ru} \sum_{(s\text{-}r \text{ path } P)\cup(r,u)} p(u, P\cup(r,u)) \prod_{a\in e(P)} \lambda_a \\
&\le \max_{s\text{-}u \text{ path } P} \left\{ \prod_{x\in P} \deg^-(x) \right\} \cdot \sum_{s\text{-}u \text{ path } P} p(u,P) \prod_{a\in e(P)} \lambda_a,
\end{aligned}$$

as desired. We can now write

$$\begin{aligned}
\mathbb{E}[\mathrm{Alg}] = \sum_{(u,v)\in A} \sum_{w\in\mathcal{W}_{uv}} & w\cdot \mathbb{P}[W_{uv}=w] \\
& \cdot \mathbb{P}\big[B_{(u,v),w}\big] \mathbb{P}[A_{uv}] \mathbb{P}\big[C_{(u,v),w} \,\big|\, A_{uv}\cap B_{(u,v),w}\big].
\end{aligned}$$

Applying (18) and using $\mathbb{P}\big[C_{(u,v),w}|\, A_{uv} \cap B_{(u,v),w}\big] \le 1$,

$$\begin{aligned}
\mathbb{E}[\mathrm{Alg}] &\le \sum_{(u,v)\in A} \sum_{w\in\mathcal{W}_{uv}} w \cdot \mathbb{P}[W_{uv} = w] \cdot \mathbb{P}\big[B_{(u,v),w}\big] \\
&\qquad \cdot \max_{s\text{-}u \text{ path } P} \prod_{x\in P\setminus t} \deg^-(x) \cdot \sum_{s\text{-}u \text{ path } P} p(u,P) \prod_{a\in e(P)} \lambda_a \\
&\le \max_{s\text{-}t \text{ path } P} \prod_{x\in P} \deg^-(x) \\
&\qquad \cdot \sum_{(u,v)\in A} \sum_{w\in\mathcal{W}_{uv}} w \cdot \mathbb{P}[W_{uv} = w] \cdot \mathbb{P}\big[B_{(u,v),w}\big] \cdot \sum_{s\text{-}u \text{ path } P} p(u,P) \prod_{a\in e(P)} \lambda_a.
\end{aligned}$$

We claim

$$\hat{z} = \sum_{(u,v)\in A} \sum_{w\in\mathcal{W}_{uv}} w \cdot \mathbb{P}[W_{uv} = w] \cdot \mathbb{P}\big[B_{(u,v),w}\big] \sum_{s\text{-}u \text{ path } P} p(u,P) \prod_{a\in e(P)} \lambda_a,$$

which would complete the proof. We will actually show that

$$\sum_{a\in\delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] = \sum_{\substack{(u,x)\in A \\ \exists v\text{-}u \text{ path}}} \sum_{w\in\mathcal{W}_{ux}} w \cdot \mathbb{P}[W_{ux} = w] \cdot \mathbb{P}\big[B_{(u,x),w}\big] \sum_{v\text{-}u \text{ path } P} p(u,P) \prod_{a\in e(P)} \lambda_a. \tag{19}$$

for all $v \in V \setminus t$, which implies the claim. We proceed by induction on $d(v,t)$, which we defined above. Suppose $d(v,t) = 1$. Then $v \in \delta^-(t)$ and so $\mathbb{P}\big[B_{(v,t),w}\big] = 0$. Moreover, there is only one arc in $\delta^+(v)$ and $\lambda_{vt} = 1$. So

$$\sum_{a\in\delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] = \sum_{w\in\mathcal{W}_{vt}} w \cdot \mathbb{P}[W_{vt} = w],$$

and (19) is verified. So suppose (19) holds for all vertices with $d(x,t) < d(v,t)$. Then

$$\begin{aligned}
&\sum_{a\in\delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)] \\
&= \sum_{(v,x)\in\delta^+(v)} \mathbb{E}\left[\min\left\{W_{vx}, \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]\right\}\right] \\
&= \sum_{(v,x)\in\delta^+(v)} \mathbb{P}\Big[W_{vx} \le \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]\Big] \cdot \mathbb{E}\Big[W_{vx} \mid W_{vx} \le \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]\Big] \\
&\qquad + \sum_{(v,x)\in\delta^+(v)} \mathbb{P}\Big[W_{vx} > \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]\Big] \cdot \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]
\end{aligned}$$

$$= \sum_{(v,x)\in\delta^+(v)} \sum_{w\in\mathcal{W}_{vx}} w \cdot \mathbb{P}[W_{vx} = w] \cdot \mathbb{P}\big[B_{(v,x),w}\big]$$
$$+ \sum_{(v,x)\in\delta^+(v)} \mathbb{P}\Big[W_{vx} > \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]\Big] \cdot \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)].$$

Applying the induction step,

$$\sum_{a\in\delta^+(v)} \mathbb{E}[\hat{y}_a(W_a)]$$
$$= \sum_{(v,x)\in\delta^+(v)} \sum_{w\in\mathcal{W}_{vx}} w \cdot \mathbb{P}[W_{vx} = w] \cdot \mathbb{P}\big[B_{(v,x),w}\big]$$
$$+ \sum_{(v,x)\in\delta^+(v)} \mathbb{P}\Big[W_{vx} > \lambda_{vx} \sum_{a\in\delta^+(x)} \mathbb{E}[\hat{y}_a(W_a)]\Big]$$
$$\cdot \lambda_{vx} \sum_{\substack{(r,q)\in A \\ \exists x\text{-}r \text{ path}}} \sum_{w\in\mathcal{W}_{rq}} w \cdot \mathbb{P}[W_{rq} = w] \cdot \mathbb{P}\big[B_{(r,q),w}\big] \sum_{x\text{-}r \text{ path } P} p(r,P) \prod_{a\in e(P)} \lambda_a$$
$$= \sum_{(v,x)\in\delta^+(v)} \sum_{w\in\mathcal{W}_{vx}} w \cdot \mathbb{P}[W_{vx} = w] \cdot \mathbb{P}\big[B_{(v,x),w}\big]$$
$$+ \sum_{(v,x)\in\delta^+(v)} \sum_{\substack{(r,q)\in A \\ \exists x\text{-}r \text{ path}}} \sum_{w\in\mathcal{W}_{rq}} w \cdot \mathbb{P}[W_{rq} = w] \cdot \mathbb{P}\big[B_{(r,q),w}\big] \sum_{\substack{v\text{-}r \text{ path } P \\ (v,x)\in P}} p(r,P) \prod_{a\in e(P)} \lambda_a$$
$$= \sum_{(v,x)\in\delta^+(v)} \sum_{w\in\mathcal{W}_{vx}} w \cdot \mathbb{P}[W_{vx} = w] \cdot \mathbb{P}\big[B_{(v,x),w}\big]$$
$$+ \sum_{\substack{(r,q)\in A \\ \exists v\text{-}r \text{ path} \\ r\notin N^+(v)}} \sum_{w\in\mathcal{W}_{rq}} w \cdot \mathbb{P}[W_{rq} = w] \cdot \mathbb{P}\big[B_{(r,q),w}\big] \sum_{\substack{v\text{-}r \text{ path } P \\ (v,x)\in P}} p(r,P) \prod_{a\in e(P)} \lambda_a$$
$$= \sum_{\substack{(u,x)\in A \\ \exists v\text{-}u \text{ path}}} \sum_{w\in\mathcal{W}_{ux}} w \cdot \mathbb{P}[W_{ux} = w] \cdot \mathbb{P}\big[B_{(u,x),w}\big] \sum_{v\text{-}u \text{ path } P} p(u,P) \prod_{a\in e(P)} \lambda_a,$$

verifying (19) as desired. □

B Test Graphs for Section 5

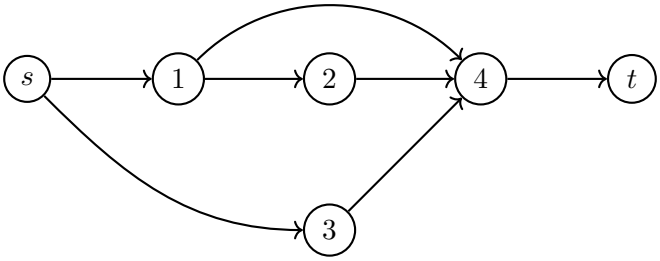


Figure 7: G_0

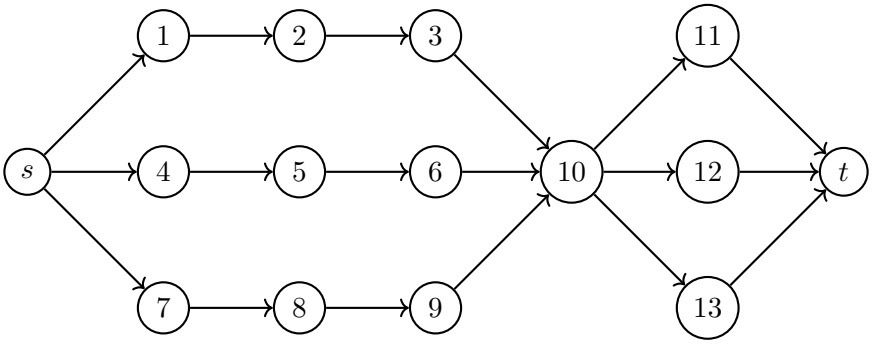


Figure 8: G_1

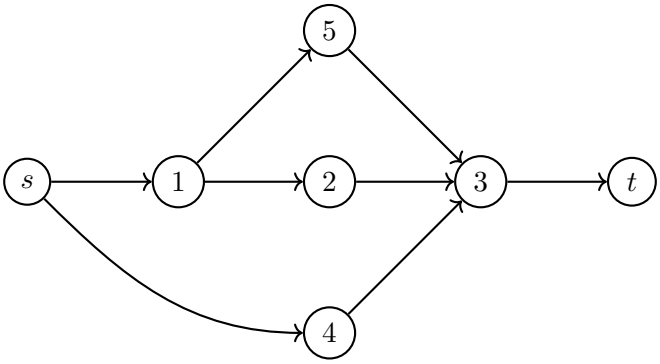


Figure 9: G_2

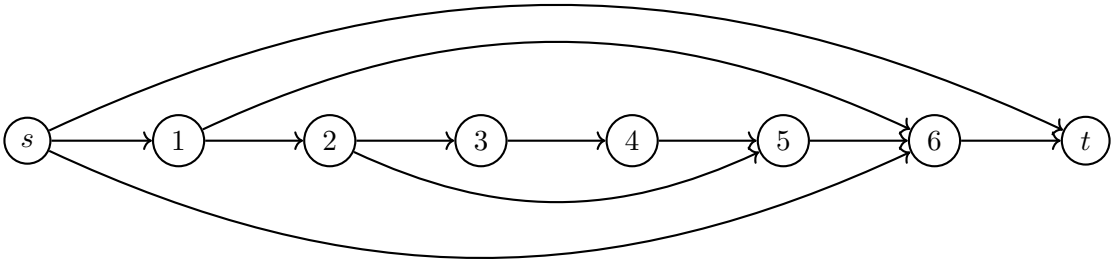


Figure 10: G_3

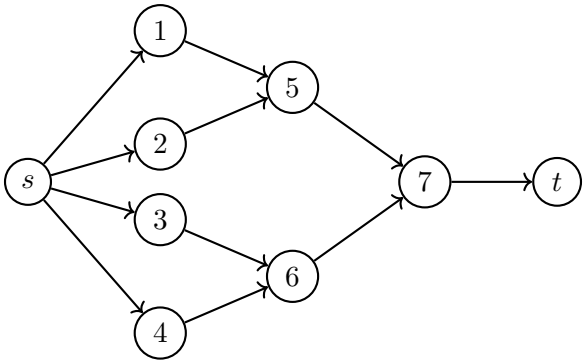


Figure 11: G_4

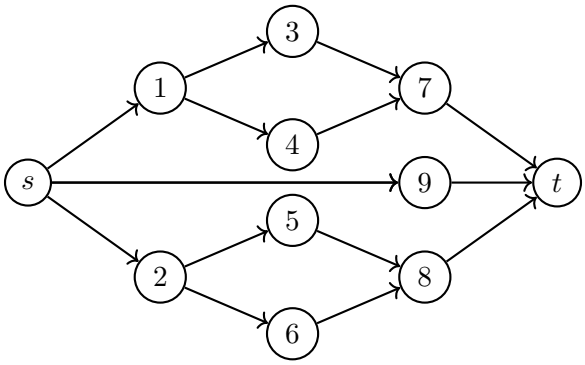


Figure 12: G_5

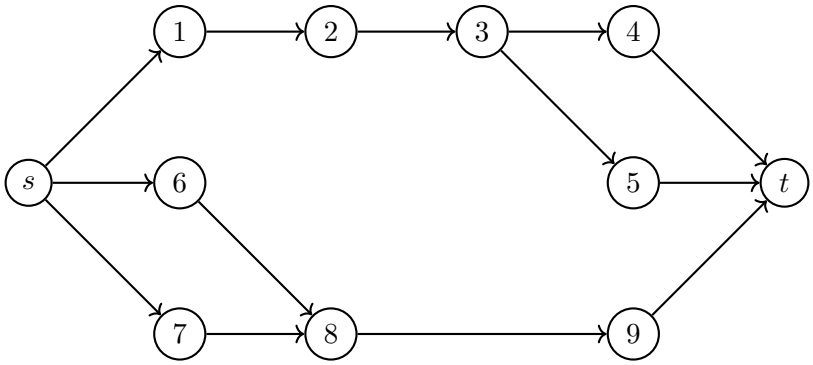


Figure 13: G_6

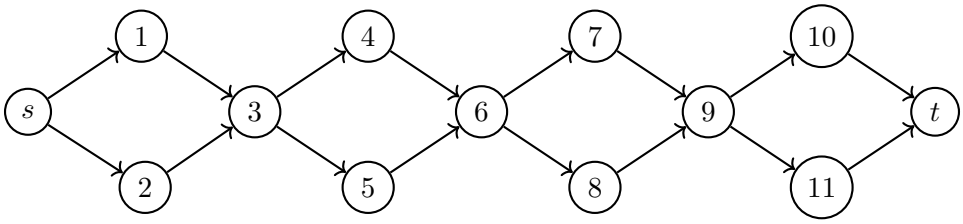


Figure 14: G_7

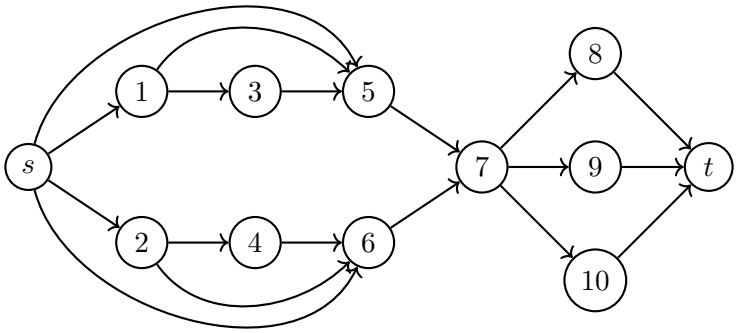


Figure 15: G_8

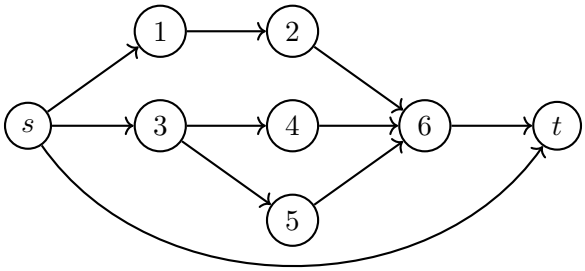


Figure 16: G_9