Graph Planning with Expected Finite Horizon

Krishnendu Chatterjee
IST Austria

Laurent Doyen
CNRS & LSV, ENS Paris-Saclay, France

***Abstract*—Graph planning gives rise to fundamental algorithmic questions such as shortest path, traveling salesman problem, etc. A classical problem in discrete planning is to consider a weighted graph and construct a path that maximizes the sum of weights for a given time horizon T. However, in many scenarios, the time horizon is not fixed, but the stopping time is chosen according to some distribution such that the expected stopping time is T. If the stopping time distribution is not known, then to ensure robustness, the distribution is chosen by an adversary, to represent the worst-case scenario.**

A stationary plan for every vertex always chooses the same outgoing edge. For fixed horizon or fixed stopping-time distribution, stationary plans are not sufficient for optimality. Quite surprisingly we show that when an adversary chooses the stopping-time distribution with expected stopping time T, then stationary plans are sufficient. While computing optimal stationary plans for fixed horizon is NP-complete, we show that computing optimal stationary plans under adversarial stopping-time distribution can be achieved in polynomial time. Consequently, our polynomial-time algorithm for adversarial stopping time also computes an optimal plan among all possible plans.

I. Introduction

Graph search algorithms. Reasoning about graphs is fundamental in computer science, in particular in logic (such as to describe graph properties with logic [6], [2]) and artificial intelligence [13], [9]. Graph search/planning algorithms are at the heart of such analysis, and give rise to some of the most important algorithmic problems in computer science, such as shortest path, travelling salesman problem (TSP), etc.

Finite-horizon planning. A classical problem in graph planning is the *finite-horizon* planning problem [9], where the input is a directed graph with weights assigned to every edge and a time horizon T. The weight of an edge represents the reward/cost of the edge. A *plan* is an infinite path, and for finite horizon T the utility of the plan is the sum of the weights of the first T edges. An *optimal* plan maximizes the utility. The computational problem for finite-horizon planning is to compute the optimal utility and an optimal plan. The finite-horizon planning problem has many applications: the qualitative version of the problem corresponds to finite-horizon reachability, which plays an important role in logic and verification (e.g., bounded until in RTCTL, and bounded model-checking [4], [1]); and the more general quantitative problem of optimizing the sum of rewards has applications in artificial intelligence and robotics [13, Chapter 10, Chapter 25], and in control theory and game theory [5, Chapter 2.2], [11, Chapter 6].

Solutions for finite-horizon planning. For finite-horizon planning the classical solution approach is dynamic programming (or Bellman equations), which corresponds to backward induction [8], [5]. This approach not only works for graphs, but also for other models (e.g., Markov decision processes [12]). A *stationary* plan is a path where for every vertex always the same choice of edge is made. For finite-horizon planning, stationary plans are not sufficient for optimality, and in general, optimal plans are quite involved. Represented as transducers, optimal plans require storage proportional to at least T (see later Example 1). Since in general optimal plans are involved, a related computational question is to compute effective simple plans, i.e., plans that are optimal among stationary plans.

Expected finite-horizon planning. A natural variant of the finite-horizon planning problem is to consider expected time horizon, instead of the fixed time horizon. In the finite-horizon problem the allowed stopping time of the planning problem is a Dirac distribution at time T. In expected finite-horizon problem the *expected* stopping time is T. A well-known example where the fixed finite-horizon and the expected finite-horizon problems are fundamentally different is playing Prisoner's Dilemma: if the time horizon is fixed, then defection is the only dominant strategy, whereas for expected finite-horizon problem cooperation is feasible [10, Chapter 5]. Another classical example of expected finite horizon that is well-studied is the notion of *discounting*, where at each time step the stopping probability is λ, and this corresponds

This work was partially supported by Austrian Science Fund (FWF) NFN Grant No RiSE/SHiNE S11407.

978-1-7281-3608-0/19/$31.00

	plan complexity	arbitrary	stationary
Specified distribution	memory necessary	PTIME	NP-complete
Unknown distribution (best-case)	memory necessary	PTIME	NP-complete
Unknown distribution (adversarial)	stationary sufficient	PTIME	

Table I: Plan complexity (left) and computational complexity (right).

to an expected stopping time equal to $1/\lambda$ [5].

Specified vs. unknown distribution. For the expected finite-horizon problem there are two variants: (a) *specified distribution:* the stopping-time distribution with finite support is specified; and (b) *unknown distribution:* the stopping-time distribution is unknown, and either resolved as the best-case scenario, or resolved as the worst-case scenario by an adversary. The expected finite-horizon problem with adversarial distribution represents the robust version of the planning problem, where the distribution is unknown and the adversary represents the worst-case scenario.

Motivation. We now present some motivation to study the expected stopping-time problem with adversarial distribution. As mentioned before, the well-studied discounted-sum problem is a specific example of stopping-time distribution. In comparison, our general framework is relevant in the following scenarios: First, in many scenarios the discount factor is not known precisely, and for robust analysis the factor is chosen adversarially. Second, the discounted-sum model makes an assumption on the shape of the stopping-time distribution. A weaker assumption is to consider time-varying discount factors [3]. If the discount factors are not known, then robust solutions require the adversarial choice of the factors. The above scenarios suggest that complex stopping-time distributions are required to model realistic scenarios, and if the precise parameters are unknown, then robust solutions require adversarial choices. Moreover, in all cases when the stopping-time distribution is important yet unknown, a conservative estimate (i.e., lower bound) of the optimal value is obtained using the adversarial choice. Thus the problems we consider present robust extensions of the classical finite-horizon planning that has a wide range of applications.

Results. In this work, we consider the expected finite-horizon planning problems in graphs. To the best of our knowledge this problem has not been studied in the literature.

- Our first simple result is that for the specified distribution problem, the optimal value can be computed in polynomial time (Theorem 1). However, since the specified distribution generalizes the fixed finite-horizon problem, the optimal plan description as an explicit transducer is of size T. Hence the output complexity is not polynomial in general (where the output is the optimal plan). Second, we consider the decision problem whether there is a stationary plan to ensure a given utility. We show that this problem is NP-complete (Theorem 2). We establish the same results (Theorem 6 and Theorem 7) for the best-case scenario of unknown distributions.

Our most interesting results are for the adversarial unknown distribution problem, which we describe below:

- We show that stationary plans suffice for optimality (Theorem 3).
- We show that the optimal value and an optimal stationary plan can be computed in polynomial time (Theorem 4).

We highlight the surprising aspects and novelty of the above results.

- First, the result about optimality of stationary plans for adversarial distribution is surprising and counter-intuitive. In the classical finite-horizon problem (and in the specified-distribution problem), the adversary does not have any choice, and in the best-case scenario the choice of the distribution is made favorably. In terms of the choice of plans and the choice of stopping-time distributions, in the first two cases there is only one quantification over the choice of plans, and in the last case, there are two quantifications, but no quantifier alternation. In all the above cases, stationary plans *do not* suffice for optimality. In contrast, we show that in the presence of an adversary the simpler class of stationary plans suffices for optimality. The adversarial case represents a quantifier alternation between the choice of plans and stopping-time distribution. Quite surprisingly our results establish that simpler plans suffice

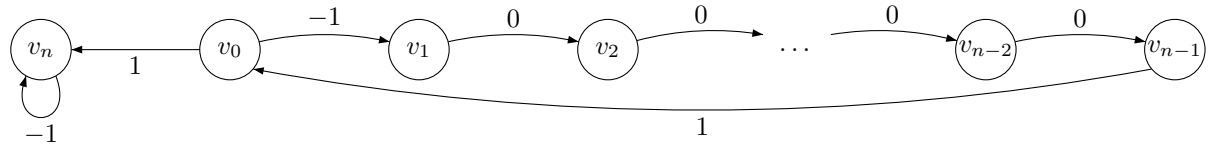


Fig. 1: A weighted graph (with $n+1$ vertices) where the optimal path (of length $T = k \cdot n + 1$) is not simple: at v_0, the optimal plan chooses k times the edge (v_0, v_1), and then the edge (v_0, v_n).

for optimality in the quantifier alternation case as compared to the cases with no quantifier alternation, or only one quantifier.

- For the expected finite-horizon problem with adversarial distribution, the backward induction approach does not work, as there is no a-priori bound on the stopping time. We develop new algorithmic ideas to establish polynomial-time complexity. Note that our algorithm also computes stationary optimal plans (which are as well optimal among all plans) in polynomial time, whereas computing stationary optimal plans for fixed finite horizon, or specified distribution, is NP-complete. Thus again our algorithm establishes a surprising result: a problem with quantifier alternation can be solved in polynomial-time, whereas the same problem without quantifier alternation is NP-complete.

Our results are summarized in Table I and are relevant for synthesis of robust plans for expected finite-horizon planning.

II. Preliminaries

Weighted graphs. A *weighted graph* $G = \langle V, E, w \rangle$ consists of a finite set V of vertices, a set $E \subseteq V \times V$ of edges, and a function $w \colon E \to \mathbb{Z}$ that assigns a weight to each edge of the graph.

Plans and utilities. A *plan* is an infinite *path* in G from a vertex v_0, that is a sequence $\rho = e_0 e_1 \dots$ of edges $e_i = (v_i, v_i') \in E$ such that $v_i' = v_{i+1}$ for all $i \geq 0$. A path induces the sequence of utilities $u_0, u_1, \dots$ where $u_i = \sum_{0 \leq k \leq i} w(e_k)$ for all $i \geq 0$. We denote by U_G the set of all sequences of utilities induced by the paths of G. For finite paths $\rho = e_0 e_1 \dots e_k$ (i.e., finite prefixes of paths), we denote by $\mathsf{start}(\rho) = v_0$ and $\mathsf{end}(\rho) = v_k'$ the initial and last vertex of ρ, and by $|\rho| = k + 1$ the length of ρ.

Plans as transducers. A plan uses finite memory if it can be described by a *transducer* (Mealy machine or Moore machine [7]) that given a prefix of the path (i.e., a finite sequence of edges) chooses the next edge. A *stationary* plan is a path where for every vertex the same choice of edge is made always. A stationary plan as a Mealy machine has one state, and as a Moore machine has at most $|V|$ states. Given a graph G we denote by S_G the set of all sequences of utilities induced by stationary plans in G.

Distributions and stopping times. A *sub-distribution* is a function $\delta \colon \mathbb{N} \to [0, 1]$ such that $p_\delta = \sum_{t \in \mathbb{N}} \delta(t) \in (0, 1]$. The value p_δ is the probability mass of δ. Note that $p_\delta \neq 0$. The support of δ is $\mathsf{Supp}(\delta) = \{t \in \mathbb{N} \mid \delta(t) \neq 0\}$, and we say that δ is the sum of two sub-distributions δ_1 and δ_2, written $\delta = \delta_1 + \delta_2$, if $\delta(t) = \delta_1(t) + \delta_2(t)$ for all $t \in \mathbb{N}$. A *stopping-time distribution* (or simply, a distribution) is a sub-distribution with probability mass equal to 1. We denote by Δ the set of all stopping-time distributions, and by $\Delta^{\uparrow\uparrow}$ the set of all distributions δ with $|\mathsf{Supp}(\delta)| \leq 2$, called the bi-Dirac distributions.

Expected utility and expected time. The *expected utility* of a sequence $u = u_0, u_1, \dots$ of utilities under a sub-distribution δ is $\mathbb{E}_\delta(u) = \frac{1}{p_\delta} \cdot \sum_{t \in \mathbb{N}} u_t \cdot \delta(t)$. In particular, the expected utility of the identity sequence $0, 1, 2, \dots$ is called the *expected time*, denoted by $\mathbb{E}_\delta$.

III. Expected Finite-horizon: Specified Distribution

Given a stopping-time distribution δ with finite support, we show that the optimal expected utility can be computed in polynomial time. This result is straightforward.

Theorem 1. *Let G be a weighted graph. Given a stopping-time distribution $\delta = \{(t_1, p_1), \dots, (t_k, p_k)\} \subseteq \mathbb{N} \times \mathbb{Q}$, with all numbers encoded in binary, the optimal expected utility $\sup_{u \in U_G} \mathbb{E}_\delta(u)$ can be computed in polynomial time.*

In the fixed-horizon problem with $\delta = \{(T, 1)\}$, the optimal plan need not be stationary. The example below shows that in general the transducer for optimal plans

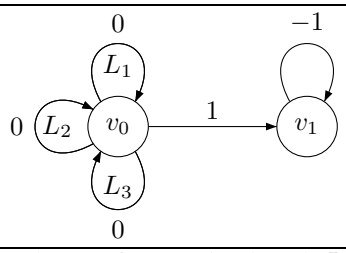


Fig. 2: Three loops of respective length $L_1 = 6 = 2 \cdot 3$, $L_2 = 10 = 2 \cdot 5$, and $L_3 = 15 = 3 \cdot 5$. For $T = 32 = 6 + 10 + 15 + 1$, the optimal plan needs to visit each cycle once.

require $O(T/|V|)$ states as Mealy machine, and $O(T)$ states as Moore machine.

Example 1. *Consider the graph of Fig. 1 with $|V| = n + 1$ vertices, and time bound $T = k \cdot n + 1$ (for some constant k). The optimal plan from v_0 is to repeat k times the cycle $v_0, v_1, \ldots, v_{n-1}$ and then switch to v_n. This path has value 1, and all other paths have lower value: if only the cycle $v_0, v_1, \ldots, v_{n-1}$ is used, then the value is at most 0, and the same holds if the cycle on v_n is ever used before time T. The optimal plan can be represented by a Mealy machine of size $O(T/|V|)$ that counts the number of cycle repetitions before switching to v_n. A Moore machine requires size T as it needs a new memory state at every step of the plan.*

Example 2. *In the example of Fig. 2 the optimal plan needs to visit several different cycles, not just repeating a single cycle and possible switching only at the end. The graph consists of three loops on v_0 with weights 0 and respective length 6, 10, and 15, and an edge to v_1 with weight 1. For expected time $T = 6 + 10 + 15 + 1$, the optimal plan has value 1 and needs to stop exactly when reaching v_1 (to avoid the negative self-loop on v_1). It is easy to show that the remaining length $T - 1 = 31$ can only be obtained by visiting each cycle once: as 31 is not an even number, the path has to visit a cycle of odd length, thus the cycle of length 15; analogously, as 31 is not a multiple of 3, the path has to visit the cycle of length 10, etc. This example can be easily generalized to an arbitrary number of cycles by using more prime numbers.*

We now consider the complexity of computing optimal plans among stationary plans.

Theorem 2. *Let G be a weighted graph and λ be a rational utility threshold. Given a stopping-time distribution δ with finite support, whether $\sup_{u \in S_G} \mathbb{E}_\delta(u) \geq \lambda$ (i.e., whether there is a stationary plan with utility at least λ) is NP-complete. The NP-hardness holds for the fixed-horizon problem $\delta = \{(T, 1)\}$, even when T and all weights are in $O(|V|)$, and thus expressed in unary.*

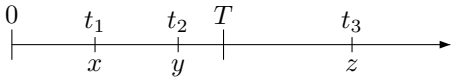


Fig. 3: Timeline.

IV. Expected Finite-horizon: Adversarial Distribution

Our main result is the computation of the following *optimal* values under adversarial distributions[1]. Given a weighted graph G and an expected stopping time $T \in \mathbb{Q}$, we define the following:

- *Optimal values of plans.* For a plan ρ that induces the sequence u of utilities, let
$$val(\rho, T) = val(u, T) = \inf_{\delta \in \Delta : \mathbb{E}_\delta = T} \mathbb{E}_\delta(u).$$
- *Optimal value.* The optimal value is the supremum value over all plans:
$$val(G, T) = \sup_{u \in U_G} val(u, T).$$

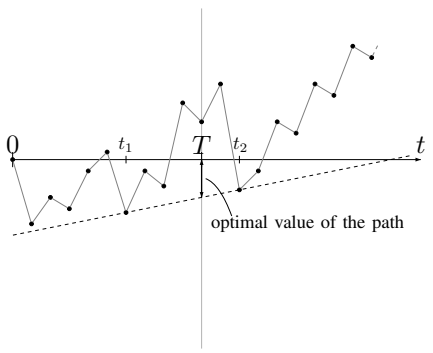
Our two main results are related to the plan complexity and a polynomial-time algorithm.

Theorem 3. *For all weighted graphs G and for all T we have*
$$val(G, T) = \sup_{u \in U_G} val(u, T) = \sup_{u \in S_G} val(u, T),$$
i.e., optimal stationary plans exist for expected finite-horizon under adversarial distribution.

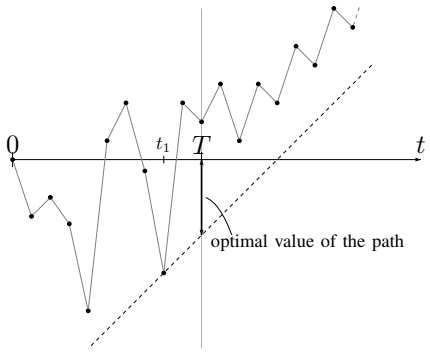
Remark 1. *Note that in contrast to the fixed finite-horizon problem, where stationary plans do not suffice, we show in the presence of an adversary, the simpler class of stationary plans are sufficient for optimality in expected finite-horizon. Moreover, while optimal plans require $O(T/|V|)$-size Mealy (resp., $O(T)$-size Moore) machines for fixed-length plans, our results show that under adversarial distribution optimal plans require $O(1)$-size Mealy (resp., $O(|V|)$-size Moore) machines.*

Theorem 4. *Given a weighted graph G and expected finite-horizon T, deciding whether $val(G, T) \geq 0$ and*

[1] Adversarial distributions may have finite or infinite support.



(a) When an optimal distribution exists



(b) When no optimal distribution exists

Fig. 4: Geometric interpretation of the value of a path.

computing $val(G, T)$ can be done in time polynomial in $|V|$, $\log(T)$, and $\log(W)$ (where W is the largest absolute weight in the graph G).

A. Theorem 3: Plan Complexity

In this section we prove Theorem 3. We start with the notion of sub-distributions. Two sub-distributions δ, δ' are *equivalent* if they have the same probability mass, and the same expected time, that is $p_\delta = p_{\delta'}$ and $\mathbb{E}_\delta = \mathbb{E}_{\delta'}$. The following result is straightforward.

Lemma 1. *If δ_1, δ_1' are equivalent sub-distributions, and $\delta_1 + \delta_2$ is a sub-distribution, then $\delta_1 + \delta_2$ and $\delta_1' + \delta_2$ are equivalent sub-distributions.*

Bi-Dirac distributions are sufficient. By Lemma 1, we can decompose distributions as the sum of two sub-distributions, and we can replace one of the two sub-distributions by a simpler (yet equivalent) one to obtain an equivalent distribution. We show that, given a sequence u of utilities, for all sub-distributions with three points t_1, t_2, t_3 in their support (see Fig. 3), there exists an equivalent sub-distribution with only two points in its support that gives a lower expected value for u. Intuitively, if one has to distribute a fixed probability mass (say 1) among three points with a fixed expected time T, assigning probability p_i at point t_i, then we have $p_3 = 1 - p_1 - p_2$ and $p_1 \cdot t_1 + p_2 \cdot t_2 + p_3 \cdot t_3 = T$, i.e.,

$$\underbrace{p_1 \cdot (t_1 - t_3)}_{p_1'} + \underbrace{p_2 \cdot (t_2 - t_3)}_{p_2'} = T - t_3.$$

The expected utility is

$$p_1 \cdot u_{t_1} + p_2 \cdot u_{t_2} + p_3 \cdot u_{t_3} = p_1' \cdot \frac{u_{t_1} - u_{t_3}}{t_1 - t_3} + p_2' \cdot \frac{u_{t_2} - u_{t_3}}{t_2 - t_3} + u_{t_3}$$

which is a linear expression in variables $\{p_1', p_2'\}$ where the sum $p_1' + p_2'$ is constant. Hence the least expected utility is obtained for either $p_1' = 0$, or $p_2' = 0$. This is the main hint[2] to show that bi-Dirac distributions are sufficient to compute the optimal expected value.

Lemma 2 (Bi-Dirac distributions are sufficient)**.** *For all sequences u of utilities, for all time bounds T, the following holds:*

$$\inf\{\mathbb{E}_\delta(u) \mid \delta \in \Delta \wedge \mathbb{E}_\delta = T\} = \inf\{\mathbb{E}_\delta(u) \mid \delta \in \Delta^{\uparrow\uparrow} \wedge \mathbb{E}_\delta = T\},$$

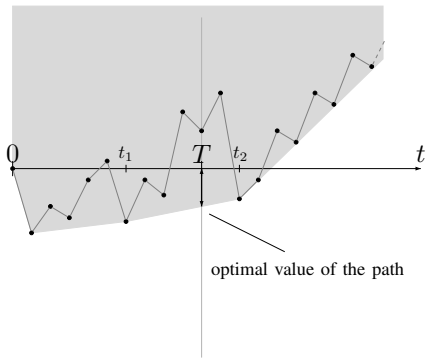
i.e., the set $\Delta^{\uparrow\uparrow}$ of bi-Dirac distributions suffices for the adversary.

Geometric interpretation. It follows from the proof of Lemma 2 that the value of the expected utility of a sequence u of utilities under a bi-Dirac distribution with support $\{t_1, t_2\}$ (where $t_1 < T < t_2$) and expected time T is

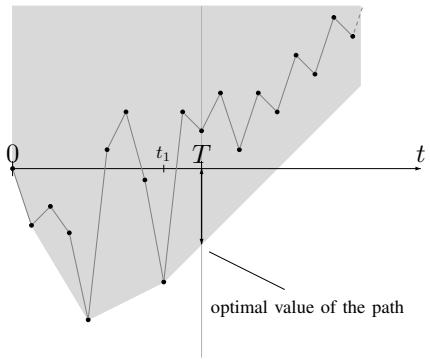
$$u_{t_1} + \frac{T - t_1}{t_2 - t_1} \cdot (u_{t_2} - u_{t_1}).$$

In Fig. 4a, this value is obtained as the intersection of the vertical axis at T and the line that connects the two points (t_1, u_{t_1}) and (t_2, u_{t_2}). Intuitively, the optimal

[2]This argument works here because $T > t_2$, which implies that $0 \leq p_2 \leq 1$ when $p_1 = 0$, and vice versa. A symmetric argument can be used in the case $T < t_2$, to show that then either $p_2 = 0$, or $p_3 = 0$.



(a) For the example of Fig. 4a.



(b) For the example of Fig. 4b.

Fig. 5: Convex hull interpretation of the value of a path.

value of a path is obtained by choosing the two points t_1 and t_2 such that the connecting line intersects the vertical axis at T as down as possible.

Lemma 3. *For all sequences u of utilities, if $u_t \geq a \cdot t + b$ for all $t \geq 0$, then the value of the sequence u is at least $a \cdot T + b$.*

Proof. By Lemma 2, it is sufficient to consider bi-Dirac distributions, and for all bi-Dirac distributions δ with arbitrary support $\{t_1, t_2\}$ the value of u under δ is

$$\begin{aligned} & u_{t_1} + \frac{T - t_1}{t_2 - t_1} \cdot (u_{t_2} - u_{t_1}) \\ = & \frac{u_{t_1} \cdot (t_2 - T) + u_{t_2} \cdot (T - t_1)}{t_2 - t_1} \\ \geq & \frac{(a \cdot t_1 + b) \cdot (t_2 - T) + (a \cdot t_2 + b) \cdot (T - t_1)}{t_2 - t_1} \\ \geq & a \cdot T + b \end{aligned}$$

□

It is always possible to fix an optimal value of t_1 (because $t_1 \leq T$ is to be chosen among a finite set of points), but the optimal value of t_2 may not exist, as in Fig. 4b. The value of the path is then obtained as $t_2 \to \infty$. In general, there exists $t_1 \leq T$ such that it is sufficient to consider bi-Dirac distributions with support containing t_1 to compute the optimal value. We say that t_1 is a *left-minimizer* of the expected value in the path. Given such a value of t_1, let $\nu = \inf_{t_2 \geq T} \frac{u_{t_2} - u_{t_1}}{t_2 - t_1}$, and we show in Lemma 4 that $u_t \geq u_{t_1} + (t - t_1) \cdot \nu$, for all $t \geq 0$. This motivates the following definition.

Line of equation $f_u(t)$. Given a left-minimizer t_1, we define the line of equation $f_u(t)$ as follows:

$$f_u(t) = u_{t_1} + (t - t_1) \cdot \nu.$$

Note that the optimal expected utility is

$$\begin{aligned} \min_{0 \leq t_1 \leq T} \inf_{t_2 \geq T} \; & u_{t_1} + \frac{T - t_1}{t_2 - t_1} \cdot (u_{t_2} - u_{t_1}) = \\ & \min_{0 \leq t_1 \leq T} \; u_{t_1} + (T - t_1) \cdot \nu = f_u(T). \end{aligned}$$

In other words, $f_u(T)$ is the optimal value.

Lemma 4 (Geometric interpretation)**.** *For all sequences u of utilities, we have $u_t \geq f_u(t)$ for all $t \geq 0$, and the expected value of u is $f_u(T)$.*

Proof. The result holds by definition of ν for all $t \geq T$. For $t < T$, assume towards contradiction that $u_t < u_{t_1} + (t - t_1) \cdot \nu$. Let $\varepsilon > 0$ such that $u_t = u_{t_1} + (t - t_1) \cdot \nu - \varepsilon$. We obtain a contradiction by showing that there exists a bi-Dirac distribution under which the expected value of u is smaller than the optimal value of u. Consider a bi-Dirac distribution with support $\{t, t_2\}$ where the value t_2 is defined later.

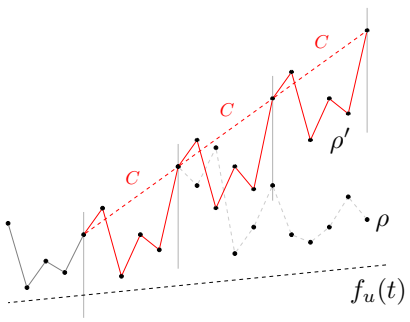
We need to show that

$$u_t + \frac{T - t}{t_2 - t} \cdot (u_{t_2} - u_t) < u_{t_1} + (T - t_1) \cdot \nu,$$

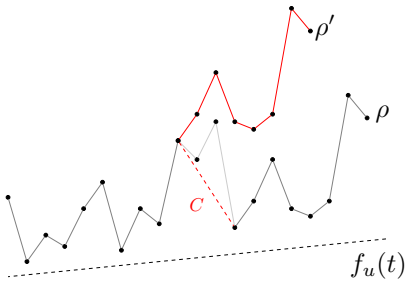
that is

$$\frac{u_t \cdot (t_2 - T) + u_{t_2} \cdot (T - t)}{t_2 - t} < u_{t_1} + (T - t_1) \cdot \nu$$

which, since $u_t = u_{t_1} + (t - t_1) \cdot \nu - \varepsilon$, holds if (successively)



(a) Repeating a good cycle (Lemma 6).



(b) Removing a bad cycle (Lemma 7).

Fig. 6: Constructing a lasso without decreasing the value (Lemma 6 and Lemma 7).

$$u_{t_1} \cdot (t_2 - T) + (t - t_1) \cdot (t_2 - T) \cdot \nu + u_{t_2} \cdot (T - t) \leq \varepsilon \cdot (t_2 - T) + u_{t_1} \cdot (t_2 - t) + (t_2 - t) \cdot (T - t_1) \cdot \nu$$

$$u_{t_1} \cdot (t - T) + u_{t_2} \cdot (T - t) + \nu \cdot (t \cdot t_2 + t_1 \cdot T - t_2 \cdot T - t \cdot t_1) \leq \varepsilon \cdot (t_2 - T)$$

$$(u_{t_2} - u_{t_1}) \cdot (T - t) + \nu \cdot (t_2 - t_1) \cdot (t - T) - \varepsilon \cdot (t_2 - T) \leq 0$$

$$(T - t) \cdot \left(\frac{u_{t_2} - u_{t_1}}{t_2 - t_1} - \nu \right) \cdot (t_2 - t_1) - \varepsilon \cdot (t_2 - T) \leq 0.$$

We consider two cases: (i) if the infimum ν is attained, then we have $\nu = \frac{u_{t_2} - u_{t_1}}{t_2 - t_1}$ for some $t_2 \geq T$, and the inequality holds; (ii) otherwise, we can choose t_2 arbitrarily, and large enough to ensure that $(T - t) \cdot \left(\frac{u_{t_2} - u_{t_1}}{t_2 - t_1} - \nu \right)$ is smaller than $\frac{\varepsilon}{2}$, so that the inequality holds. □

A corollary of the geometric interpretation lemma is that the value of a path can be obtained as the intersection of the vertical line at point T with the boundary of the convex hull of the region above the sequence of utilities, namely $convexHull(\{(t, y) \in \mathbb{N} \times \mathbb{R} \mid y \geq u_t\})$. This result is illustrated in Fig. 5.

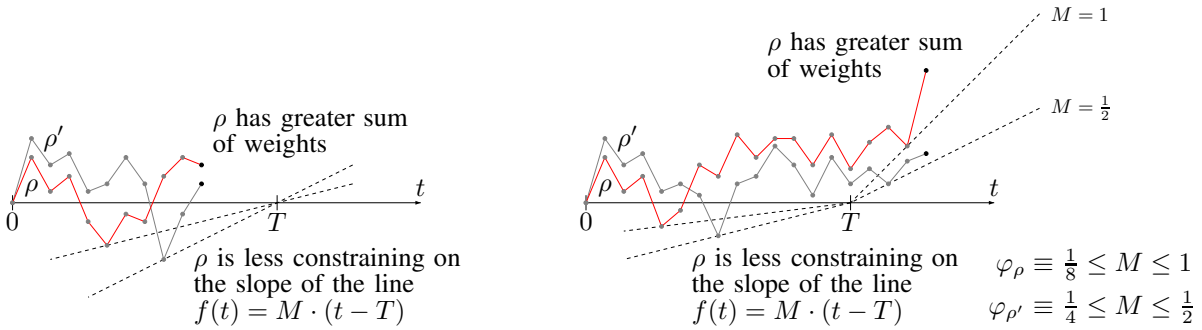
Simple lassos are sufficient. A *lasso* is a path of the form AC^ω where A and C are finite paths (with C a nonempty cycle), where AC^ω is A followed by infinite repetition of the cycle C. A lasso is *simple* if all strict prefixes of the finite path AC are acyclic. In other words, simple lassos correspond to stationary plans.

We show that there is always a simple lasso with optimal value. Our proof has four steps. Given a path ρ that gives the utility sequence u, let ν be the slope of $f_u(t)$. Given a cycle C in the path ρ, let S_C be the sum of the weights in C and let $M_C = \frac{S_C}{|C|}$ be the average weight of the cycle edges. The cycle C is *good* if $M_C \geq \nu$, i.e., the average weight of the cycle is at least ν, and *bad* otherwise.

- First, we show (in Lemma 5) that every path contains a good cycle.
- Second, we show (in Lemma 6) that if the first cycle in a path is good, then repeating the cycle cannot decrease the value of the path.
- Third, we show (in Lemma 7) that removing a bad cycle from a path cannot decrease the value of the path.
- Finally, we show (in Lemma 8) that given any path, using the above two operations of removal of bad cycles and repetition of good cycles, we obtain a simple lasso that does not decrease the value of the original path.

Thus we establish that simple lassos (or stationary plans) are sufficient for optimality. To formalize the ideas we consider the notion of cycle decomposition.

Cycle decomposition. The *cycle decomposition* of a path $\rho = e_0 e_1 \dots$ is an infinite sequence of simple cycles $C_1, C_2, \dots$ obtained as follows: push successively $e_0, e_1, \dots$ onto a stack, and whenever we push an edge that closes a (simple) cycle, we remove the cycle from the stack and append it to the cycle decomposition. Note that the stack content is always a prefix of a path of length at most $|V|$.



(a) The path length is smaller than T.

(b) The path length is greater than T.

Fig. 7: The path ρ is preferred to ρ'.

Lemma 5. *Let $T \in \mathbb{N}$. Given a path ρ that induces a sequence u of utilities, let $\nu = \min_{0\leq t_1\leq T} \inf_{t_2\geq T} \frac{u_{t_2}-u_{t_1}}{t_2-t_1}$. Then, in the cycle decomposition of ρ there exists a simple cycle C with $M_C \geq \nu$.*

Proof. Towards contradiction, assume that all the (finitely many) cycles C in the cycle decomposition of ρ are such that $M_C < \nu$. Let t_1 be a left-minimizer of ρ. Since all cycles in ρ have average weight smaller than ν, we have:

$$\liminf_{t_2\to\infty} \frac{u_{t_2} - u_{t_1}}{t_2 - t_1} < \nu$$

Since the infimum is bounded by the liminf, it follows that

$$\min_{0\leq t_1\leq T} \inf_{t_2\geq T} \frac{u_{t_2} - u_{t_1}}{t_2 - t_1} < \nu$$

which is in contradiction with the definition of ν. □

We show that repeating a good cycle, and removing a bad cycle from a path cannot decrease the value of the path.

Lemma 6. *Let $T \in \mathbb{N}$. If the first cycle C in the cycle decomposition of a path ρ is good, i.e., $M_C \geq \nu$ where $\nu = \min_{0\leq t_1\leq T} \inf_{t_2\geq T} \frac{u_{t_2}-u_{t_1}}{t_2-t_1}$, then there exists a lasso ρ' such that $val(\rho', T) \geq val(\rho, T)$.*

Proof. Let u be the sequence of utilities induced by ρ. Since C is the first cycle in ρ, there is a prefix of ρ of the form AC where A is a finite path. Consider the lasso $\rho' = AC^\omega$ and its induced sequence of utilities u'.

We show that the value of ρ' is at least the value of ρ. By Lemma 4, the optimal value of u is $f_u(T)$, and the sequence u is above the line $f_u(t)$ (which has slope ν), i.e., $u_t \geq f_u(t)$ for all $t \geq 0$. By Lemma 3 it is sufficient to show that u' is above the line $f_u(t)$ to establish that the optimal value of u' is at least $f_u(T)$, that is $val(\rho', T) \geq val(\rho, T)$, and conclude the proof (the argument is illustrated in Fig. 6a).

We show that $u'_t \geq f_u(t)$ for all $t \geq 0$:

- either $t \leq |A| + |C|$, and then $u'_t = u_t \geq f_u(t)$,
- or $t > |A| + |C|$, and then let $k \in \mathbb{N}$ such that $|A| \leq t - k \cdot |C| \leq |A| + |C|$, and we have

$$\begin{aligned} u'_t &= u_{t-k\cdot|C|} + k \cdot S_C && (\rho' = AC^\omega) \\ &\geq f_u(t - k \cdot |C|) + k \cdot M_C \cdot |C| \\ &\qquad (u \text{ is above } f_u(t) \text{ and } S_C = M_C \cdot |C|) \\ &\geq f_u(t) - \nu \cdot k \cdot |C| + k \cdot M_C \cdot |C| \\ &\qquad (f_u(t) \text{ is linear with slope } \nu) \\ &\geq f_u(t) + k \cdot |C| \cdot (M_C - \nu) \\ &\geq f_u(t). && (M_C \geq \nu) \end{aligned}$$

□

Lemma 7. *Let $T \in \mathbb{N}$. If a path ρ contains a bad cycle C, that is such that $M_C < \nu$ where $\nu = \min_{0\leq t_1\leq T} \inf_{t_2\geq T} \frac{u_{t_2}-u_{t_1}}{t_2-t_1}$, then removing C from ρ gives a path ρ' such that $val(\rho', T) \geq val(\rho, T)$.*

Proof. Let u, u' be the sequences of utilities induced by respectively ρ and ρ'. By the same argument as in the proof of Lemma 6 (using Lemma 3 and Lemma 4), it is sufficient to show that u' is above the line $f_u(t)$. Since C is a cycle in ρ, there is a prefix of ρ of the form AC

where A is a finite path, and for all $t \geq 0$ we have (the argument is illustrated in Fig. 6b): either $t \leq |A|$, then $u'_t = u_t \geq f_u(t)$, or $t > |A|$, and then

$$\begin{aligned} u'_t &= u_{t+|C|} - S_C \\ &\qquad (C \text{ is removed from } \rho \text{ to get } \rho') \\ &\geq f_u(t+|C|) - M_C \cdot |C| \\ &\quad (u \text{ is above } f_u(t) \text{ and } S_C = M_C \cdot |C|) \\ &\geq f_u(t) + \nu \cdot |C| - M_C \cdot |C| \\ &\qquad (f_u(t) \text{ is linear with slope } \nu) \\ &\geq f_u(t) + |C| \cdot (\nu - M_C) \\ &\geq f_u(t). \qquad (M_C < \nu) \end{aligned}$$

□

Now we can show how to construct a simple lasso with value at least the value of a given arbitrary path, and it follows that simple lassos are sufficient for optimality.

Lemma 8. *Let $T \in \mathbb{N}$. There exists a simple lasso AC^ω such that $val(AC^\omega, T) = val(G, T)$.*

Proof. Given an arbitrary path ρ, we construct a simple lasso with at least the same value as ρ. It follows that the optimal value is obtained by stationary plans. The construction repeats the following steps:

1) Let C be the first cycle in the cycle decomposition of ρ;
2) if C is a bad cycle for the original path ρ, then we remove it to obtain a new path ρ'. We continue the procedure with ρ' (go to step 1.);
3) otherwise C is a good cycle for the original path ρ. Let A be the prefix of ρ until C starts, and we construct the lasso AC^ω.

First, note that if the above procedure terminates, then the constructed lasso has a value at least the value of the original path ρ (by Lemma 6 and Lemma 7), and it is a simple lasso by definition of the cycle decomposition.

Now we show that the procedure always terminates. By Lemma 5, there always exists a good cycle in the cycle decomposition of ρ, and thus eventually a good cycle becomes the first cycle in the path constructed by the above procedure, which then terminates. □

Theorem 3 follows from the above lemmas.

B. Theorem 4: Algorithm and Complexity Analysis

In this section we present our algorithm and then the complexity analysis (Theorem 4).

Algorithm. The key challenges to obtain an algorithm are as follows. First, while for the fixed-horizon problem backward induction or powering of transition matrix leads to an algorithm, for expected time horizon with an adversary, there is no a-priori bound on the number of steps, and hence the backward induction approach is not applicable. Second, stationary optimal plans suffice, and as shown in Theorem 2 computing optimal stationary plans for the fixed horizon problem is NP-hard. We present an algorithm that iteratively constructs the *most promising* candidate paths according to a partial order of the paths, and the key is to define the partial order.

It follows from the geometric interpretation lemmas (Lemma 3 and Lemma 4) that the value of a path is at least 0 if its sequence of utilities is above some line that contains the point $(T, 0)$.

Lemma 9. *The value of a sequence u of utilities is at least 0 if and only if there exists a slope $M \in \mathbb{R}$ such that $u_t \geq M \cdot (t - T)$ for all $t \geq 0$.*

The expression $u_t - M \cdot (t - T)$ that appears in the condition of Lemma 9 corresponds to the sequence of utilities in the graph where M is subtracted from all weights, up to the constant $T \cdot M$. Since M is unknown, we can define the following symbolic constraint on M (associated with a path ρ) that ensures, if it is satisfiable, that the sequence of utilities of $\rho = e_0 e_1 \dots e_k$ is above the line of equation $f(t) = M \cdot (t - T)$:

$$\varphi_\rho \equiv \bigwedge_{0 \leq i \leq k} (u_i \geq M \cdot (i - T))$$

Note that $k = |\rho| - 1$, and the constraint φ_ρ represents an interval (possibly empty, possibly unbounded) of values for M. Intuitively, a finite path is more promising (thus preferred) in order to be prolonged to an infinite path with value at least 0 if the total sum of weights is large and the constraint φ_ρ is weak (see Fig. 7a and Fig. 7b). To each finite path ρ, we associate a pair $\langle z, \psi \rangle$ consisting of the sum u of the weights in ρ, and the constraint $\psi = \varphi_\rho$.

For two pairs $\langle z, \psi \rangle$, $\langle z', \psi' \rangle$ (associated with paths ρ and ρ' respectively), we write $\langle z, \psi \rangle \succeq \langle z', \psi' \rangle$ if $z \geq z'$ and ψ' implies ψ, and we say that ρ is *preferred* to ρ' (this is a partial order). Given a set S of such pairs, denote by $\lceil S \rceil = \{s_1 \in S \mid \forall s_2 \in S : s_2 \succeq s_1 \to s_1 \succeq s_2\}$ the set of $\succeq$-maximal elements of S. Note that the elements of $\lceil S \rceil$ are pairwise $\succeq$-incomparable.

Intuitively, if ρ and ρ' end in the same vertex, and ρ is *preferred* to ρ', then it is easier to extend ρ than ρ' to obtain an (infinite) path with expected value at least 0. Formally, for all infinite paths π with $\mathsf{start}(\pi) =$

Algorithm 1 BestPaths(t_0, v_0, z_0, ψ_0)

Input : $t_0 \in \mathbb{N}$ is an initial time point, v_0 is an initial vertex, z_0 is the initial sum of weights, and ψ_0 is the initial constraint on the slope parameter M.

Output: The table of $\succeq$-maximal values of paths from v_0 with initial values t_0, z_0, ψ_0.

begin

```
                                   /* initialization */
1   D[t_0, v_0] ← {⟨z_0, ψ_0⟩}
2   for v ∈ V \ {v_0} do
3       D[t_0, v] ← ∅
                                   /* iterations */
4   for i = 1, ..., |V| do
5       for v ∈ V do
6           D[t_0 + i, v] ← ∅
7           for v_1 ∈ V and ⟨z_1, ψ_1⟩ ∈ D[t_0 + i − 1, v_1]
            do
8               if (v_1, v) ∈ E then
9                   z ← z_1 + w(v_1, v)
10                  t ← t_0 + i − 1
11                  ψ ← ψ_1 ∧ (z ≥ M · (t − T))
12                  D[t_0 + i, v] ←
                    D[t_0 + i, v] ∪ {⟨z, ψ⟩}
13          D[t_0 + i, v] ← ⌈D[t_0 + i, v]⌉
14  return D
```

end

$\mathsf{end}(\rho) = \mathsf{end}(\rho')$ we have $val(\rho \cdot \pi, T) \geq val(\rho' \cdot \pi, T)$. We use this result in the following form.

Lemma 10. *Let ρ_1, ρ_A be two paths of the same length with the same end state, i.e.,* $\mathsf{end}(\rho_1) = \mathsf{end}(\rho_A)$*. If ρ_1 is preferred to ρ_A, then for all paths ρ_C with* $\mathsf{start}(\rho_C) = \mathsf{end}(\rho_A)$*, the path $\rho_1 \cdot \rho_C$ is preferred to the path $\rho_A \cdot \rho_C$.*

Our algorithm uses the procedure BestPaths(t_0, v_0, z_0, ψ_0) (shown as Algorithm 1) that computes the $\succeq$-maximal pairs $\langle z, \psi \rangle$ corresponding to the paths ρ_1 of length $1, 2, \ldots, |V|$ that start at time t_0 in vertex v_0 (see Fig. 8), and that prolong a path $\rho_\sharp$ with sum of weights z_0 and constraint ψ_0 on M (where z is the sum of weights along $\rho_\sharp \cdot \rho_1$, and $\psi \equiv \varphi_{\rho_\sharp \cdot \rho_1}$). We give a precise statement of this result in Lemma 11.

Lemma 11 (Correctness of BestPaths)**.** *Let $\rho_\sharp$ be a finite path of length t_0, that ends in state* $\mathsf{end}(\rho_\sharp) = v_0$ *with sum of weights z_0 and associated constraint ψ_0 on M.*

Algorithm 2 ExistsPositivePath(v_0)

Input : v_0 is an initial vertex.

Output: true iff there exists a path from v_0 with expected utility at least 0.

begin

```
1   A ← BestPaths(0, v_0, 0, true)
2   for i = 0, ..., |V| do
3       for v̂ ∈ V and ⟨z_1, ψ_1⟩ ∈ A[i, v̂] do
4           C ← BestPaths(i, v̂, z_1, ψ_1)
5           for j = 1, ..., |V| − i do
6               for ⟨z_2, ψ_2⟩ ∈ C[i + j, v̂] do
7                   if ψ_2 ∧ (z_2 − z_1)/j ≥ M is satisfiable
                    then return true
8   return false
```

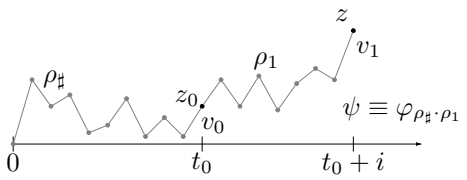
end

Let $D =$ BestPaths(t_0, v_0, z_0, ψ_0)*. Then,*

- *for all $0 \leq i \leq |V|$, for all $v_1 \in V$, for all pairs $\langle z, \psi \rangle \in D[t_0 + i, v_1]$, there exists a path ρ_1 of length i with* $\mathsf{start}(\rho_1) = v_0$ *and* $\mathsf{end}(\rho_1) = v_1$*, such that*
  - *z is the sum of weights of the path $\rho_\sharp \cdot \rho_1$, and*
  - *$\psi \equiv \varphi_{\rho_\sharp \cdot \rho_1}$ is the constraint on M associated with the path $\rho_\sharp \cdot \rho_1$;*
- *for all paths ρ_1 of length $i \leq |V|$ such that* $\mathsf{start}(\rho_1) = v_0$ *and* $\mathsf{end}(\rho_1) = v_1$*, there exists a pair $\langle z', \psi' \rangle \in D[t_0 + i, v_1]$ such that $\langle z', \psi' \rangle \succeq \langle z, \psi \rangle$ where*
  - *z is the sum of weights of the path $\rho_\sharp \cdot \rho_1$, and*
  - *$\psi \equiv \varphi_{\rho_\sharp \cdot \rho_1}$ is the constraint on M associated with the path $\rho_\sharp \cdot \rho_1$.*

As we know that simple lassos are sufficient for optimal value (Lemma 8), our algorithmic solution is to explore finite paths from the initial vertex, until a loop is formed. Thus it is sufficient to explore paths of length at most $|V|$. However, given a simple lasso $\rho_A \cdot \rho_C^\omega$, it is not sufficient that the finite path $\rho_A \cdot \rho_C$ lies above a line $M \cdot (t - T)$ (where M satisfies the constraint ψ_{AC} associated with $\rho_A \cdot \rho_C$) to ensure that the value of the lasso $\rho_A \cdot \rho_C^\omega$ is at least 0. The reason is that by repeating the cycle ρ_C several times, the path may eventually cross the line $M \cdot (t - T)$. We show (in Lemma 12) that this cannot happen if the average weight M_C of the cycle is greater than the slope of the line (i.e., $M_C \geq M$).

Lemma 12. *Given a lasso $\rho_A \cdot \rho_C^\omega$, let ψ_{AC} be the symbolic constraint on M associated with the finite path*



$\langle z, \psi \rangle \in D[t_0 + i, v_1]$
where $D = \mathsf{BestPaths}(t_0, v_0, z_0, \psi_0)$

Fig. 8: The result of the computation of $\mathsf{BestPaths}(t_0, v_0, z_0, \psi_0)$.

$\rho_A \cdot \rho_C$, and let M_C be the average weight of the cycle ρ_C. The lasso $\rho_A \cdot \rho_C^\omega$ has value at least 0 if and only if the formula $\psi_{AC} \wedge (M_C \geq M)$ is satisfiable.

The algorithm $\mathsf{ExistsPositivePath}(v_0)$ explores the paths from v_0, and keeps the $\succeq$-preferred paths, that is those with the largest total weight and weakest constraint on M. There may be several $\succeq$-incomparable paths of a given length i that reach a given vertex $\hat{v}$, therefore we need to compute a *set* $A[i, \hat{v}]$ of $\succeq$-incomparable pairs (line 1 of Algorithm 2).

Given a pair $\langle z_1, \psi_1 \rangle \in A[i, \hat{v}]$, the algorithm ExistsPositivePath further explores (for-loop at line 3 of Algorithm 2) the paths from $\hat{v}$, until a cycle ρ_C of length j is formed around $\hat{v}$, with average weight $M_C = \frac{z_2 - z_1}{j}$ and associated pair $\langle z_2, \psi_2 \rangle \in C[i + j, \hat{v}]$ (line 7 of Algorithm 2) such that $\psi_2 \wedge (M_C \geq M)$ is satisfiable. We claim that there exists such a cycle if and only if there exists a lasso with value at least 0. The claim is established in the following lemma.

Lemma 13 (Correctness of ExistsPositivePath)**.** *There exists an infinite path from v_0 with value at least 0 if and only if* $\mathsf{ExistsPositivePath}(v_0)$ *returns* true*.*

Optimal value. We can compute the optimal value using the procedure ExistsPositivePath as follows. From Lemma 4, the optimal value is either of the form $\frac{u_{t_1} \cdot (t_2 - T) + u_{t_2} \cdot (T - t_1)}{t_2 - t_1}$, or of the form $u_{t_1} + (T - t_1) \cdot \nu$ where the following bounds hold ($\nu = \inf_{t_2 \geq T} \frac{u_{t_2} - u_{t_1}}{t_2 - t_1}$):

- $0 \leq t_1 \leq t_2 \leq |V|$
- $0 \leq t_2 - t_1 \leq |V|$
- $0 \leq T - t_1 \leq |V|$
- $0 \leq t_2 - T \leq |V|$
- $-W \cdot |V| \leq u_{t_1}, u_{t_2} \leq W \cdot |V|$
- ν is a rational number $\frac{p}{q}$ where $-W \cdot |V| \leq p \leq W \cdot |V|$ and $1 \leq q \leq |V|$

Therefore, in both cases we get the following result.

Lemma 14. *The optimal value belongs to the set*

$$\mathsf{ValueSpace} = \left\{ \frac{p}{q} \mid -2W \cdot |V|^2 \leq p \leq 2W \cdot |V|^2 \text{ and } 1 \leq q \leq |V| \right\}.$$

Given a value $\frac{p}{q}$, we can decide if there exists a path with expected value at least $\frac{p}{q}$ by subtracting $\frac{p}{q \cdot T}$ from all the weights the graphs, and asking if there exists a path with expected value at least 0 in the modified graph. Indeed, if we define $w'(e) = w(e) + \eta$ for all edges $e \in E$ (where η is a constant), then for all paths ρ, if u is the sequence of utilities along ρ according to w, and u' is the sequence of utilities along ρ according to w', then

$$\begin{aligned} \sum_i p_i \cdot u'_i &= \sum_i p_i \cdot (u_i + \eta \cdot i) \\ &= \eta \cdot \sum_i p_i \cdot i + \sum_i p_i \cdot u_i \\ &= T \cdot \eta + \sum_i p_i \cdot u_i, \end{aligned}$$

thus the value of the path is shifted by $T \cdot \eta$. Then it follows from Lemma 14 that the optimal value can be computed by a binary search using $O(|\mathsf{ValueSpace}|) = O(\log(W \cdot |V|))$ calls to ExistsPositivePath.

Optimal path. An optimal path can be constructed by a slight modification of the algorithm. In BestPaths, we can maintain a path associated to each pair in D as follows: the empty path is associated with the pair $\langle z_0, \psi_0 \rangle$ added at line 1 of Algorithm 1, and given the path ρ_1 associated with the pair $\langle z_1, \psi_1 \rangle$ (line 7 of Algorithm 1), we associate the path $\rho_1 \cdot (v_1, v)$ with the pair $\langle z, \psi \rangle$ added to D at line 12 of Algorithm 1. It is easy to see that for every pair $\langle z, \psi \rangle$ in D, the associated path can be used as the path ρ_1 in Lemma 11 (item 1). Therefore, when $\mathsf{ExistsPositivePath}(v_0)$ returns true (line 7 of Algorithm 2), we can output the path $\rho_1 \cdot \rho_2^\omega$ where ρ_i is the path associated with the pair $\langle z_i, \psi_i \rangle$ $(i = 1, 2)$.

Complexity analysis. We show that the algorithm ExistsPositivePath (Algorithm 2) runs in polynomial time. The key challenge is to bound the number of $\succeq$-incomparable pairs computed by BestPaths (Algorithm 1) and enumerated in the 4th for-loop (line 6 of Algorithm 2). The number of such pairs corresponds to the number of simple paths in a graph, and hence could be exponential in general. However, we show that only a polynomial number of paths can correspond to $\succeq$-incomparable pairs, and therefore there is a polynomial bound on the number of $\succeq$-incomparable pairs. Those paths are characterized by a small number of parameters (such as the length, the starting vertex, the ending vertex, etc.) that have a polynomial-size range (namely, $|V|$), and therefore they are at most polynomially many. It follows that the worst-case complexity of BestPaths and ExistsPositivePath, which is bounded by the dominant operations of computing and enumerating over sets of $\succeq$-maximal elements, is polynomial time (Theorem 4).

V. Expected Finite-Horizon: Best-Case Distribution

We now consider the problem of maximizing the value of a plan where the value of a plan is computed as the supremum value (instead of the infimum value) over all distributions with expected stopping time T. The optimization problem is thus to choose a path as well as a stopping-time distribution in order to maximize the value.

Given a weighted graph G and an expected stopping time $T \in \mathbb{Q}$, we define the following:

- *Optimal sup-value of plans.* For a plan ρ that induces the sequence u of utilities, let
$$val_{\sup}(\rho, T) = val_{\sup}(u, T) = \sup_{\delta \in \Delta : \mathbb{E}_\delta = T} \mathbb{E}_\delta(u).$$
- *Optimal sup-value.* The optimal sup-value is the supremum value over all plans:
$$val_{\sup}(G, T) = \sup_{u \in U_G} val_{\sup}(u, T).$$

Since the distribution is chosen by the maximizer and there is no adversary, the optimal sup-value is at least as large as the optimal (inf-)value defined in Section IV. However, while stationary plans suffice against adversarially chosen distributions, it turns out that optimal plans for the sup-value are in general not stationary (i.e., memory is necessary for optimality).

However, we show that after time T memory is no longer necessary. A plan $\rho = e_0 e_1 \dots$ is *stationary after* T if for all $T \leq t_1 < t_2$, if $e_{t_1} = (\cdot, v)$ and $e_{t_2} = (\cdot, v)$, then $e_{t_1+1} = e_{t_2+1}$. We denote by $S_G^{\succeq T}$ the set of all sequences of utilities induced by plans in G that are stationary after T.

Theorem 5. *For all weighted graphs G and for all T we have*
$$val_{\sup}(G, T) = \sup_{u \in U_G} val_{\sup}(u, T) = \sup_{u \in S_G^{\succeq T}} val_{\sup}(u, T),$$
i.e., optimal stationary-after-T plans exist for expected finite-horizon under best-case distribution.

It follows from Theorem 5 that an optimal plan for the sup-value always exists (since there are finitely many stationary-after-T plans).

We show that computing optimal plans among *stationary* plans cannot be done in polynomial time unless P = NP. In contrast, the optimal sup-value for arbitrary paths and best-case distribution can be computed in polynomial time.

Theorem 6. *Given a weighted graph G, an integer T, and a threshold $\lambda \in \mathbb{Q}$, deciding whether $\sup_{u \in S_G} val_{\sup}(u, T)$ is at least λ is NP-complete. The NP-hardness holds for T and all weights expressed in unary.*

We show that optimal plans for best-case distributions have a shape that consists of simple cycles and connecting segments of polynomial length. As we have a polynomial algorithm to compute the best path of a fixed length (Theorem 1) we obtain a polynomial algorithm for the best-case distribution problem by enumerating the possible lengths and end-points of the segments and cycles, and then computing the best utility such segments can have.

Theorem 7. *Given a weighted graph G and expected finite-horizon T, the optimal sup-value can be computed in time polynomial in $|V|$, $\log(T)$, and $\log(W)$ (where W is the largest absolute weight in the graph G).*

VI. Conclusion

In this work we consider the expected finite-horizon problem. Our most interesting results are for worst-case distribution of stopping times, for which we establish stationary plans are sufficient, and present polynomial-time algorithms (in contrast with the case of specified distribution and best-case distribution where memory is necessary and computing an optimal plan among stationary plans is NP-complete). In terms of algorithmic complexity, our main goal was to establish polynomial-time algorithms, and we expect that better algorithms and refined complexity analysis can be obtained.

REFERENCES

[1] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu. Bounded model checking. *Advances in Computers*, 58:117–148, 2003.

[2] B. Courcelle and J. Engelfriet. *Graph Structure and Monadic Second-Order Logic: A Language-Theoretic Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2012.

[3] I. Dew-Becker. *Essays on Time-Varying Discount Rates*. PhD thesis, Harvard University, 2012.

[4] E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. *Real-Time Systems*, 4(4):331–352, 1992.

[5] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer-Verlag, 1997.

[6] E. Grädel, P. G. Kolaitis, L. Libkin, M. Marx, J. Spencer, M. Y. Vardi, Y. Venema, and S. Weinstein. *Finite Model Theory and Its Applications (Texts in Theoretical Computer Science. An EATCS Series)*. Springer-Verlag, 2005.

[7] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[8] H. Howard. *Dynamic Programming and Markov Processes*. MIT Press, 1960.

[9] S. M. LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[10] M. A. Nowak. *Evolutionary dynamics*. Harvard University Press, 2006.

[11] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

[12] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12:441–450, 1987.

[13] S. J. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach (3rd ed.)*. Pearson Education, 2010.