

TensorPlan and the Few Actions Lower Bound for Planning in MDPs under Linear Realizability of Optimal Value Functions

Gellért Weisz

DeepMind, London, UK

University College London, London, UK

Csaba Szepesvári

DeepMind, London, UK

University of Alberta, Edmonton, Canada

András György

DeepMind, London, UK

Editors: Sanjoy Dasgupta and Nika Haghtalab

Abstract

We consider the minimax query complexity of online planning with a generative model in fixed-horizon Markov decision processes (MDPs) with linear function approximation. Following recent works, we consider broad classes of problems where either (i) the optimal value function v^* or (ii) the optimal action-value function q^* lie in the linear span of some features; or (iii) both v^* and q^* lie in the linear span when restricted to the states reachable from the starting state. Recently, [Weisz et al. \(2021b\)](#) showed that under (ii) the minimax query complexity of any planning algorithm is at least exponential in the horizon H or in the feature dimension d when the size A of the action set can be chosen to be exponential in $\min(d, H)$. On the other hand, for the setting (i), [Weisz et al. \(2021a\)](#) introduced TensorPlan, a planner whose query cost is polynomial in all relevant quantities when the number of actions is fixed. Among other things, these two works left open the question whether polynomial query complexity is possible when A is subexponential in $\min(d, H)$. In this paper we answer this question in the negative: we show that an exponentially large lower bound holds when $A = \Omega(\min(d^{1/4}, H^{1/2}))$, under either (i), (ii) or (iii). In particular, this implies a perhaps surprising exponential separation of query complexity compared to the work of [Du et al. \(2021\)](#) who prove a polynomial upper bound when (iii) holds for all states. Furthermore, we show that the upper bound of TensorPlan can be extended to hold under (iii) and, for MDPs with deterministic transitions and stochastic rewards, also under (ii).

Keywords: Reinforcement Learning, Planning, Online Planning, Linear Function Approximation, Information Theoretic Lower Bound, Sample Complexity

1. Introduction

We are concerned with the query complexity of **online planning** in fixed-horizon Markov decision processes (MDPs) with large state spaces and finite action sets when the planner is used in a **closed-loop** configuration. In each step of the closed-loop process, the planner is **called** with the state of the process, after which it is allowed to query a simulator of the MDP until it decides to stop. Then it needs to return an action, which is used to move the state of the process. This is repeated until

the time for the episode runs out. The goal of the planner is to maximize the total expected reward incurred in the episode.

To deal with large state spaces, the planner is helped by giving it access to features (elements of \mathbb{R}^d , the d -dimensional Euclidean space). The features are associated with the states, but the planner is only given the features of that states that it encounters either in response to a query or when the planner is called with a state. The features are assumed to be such that the optimal value $v^*(s)$ at any state s is equal to the linear combination of the features at the state, where the vector $\theta^* \in \mathbb{R}^d$ formed by the coefficients is fixed (regardless of the state) but unknown. An MDP together with a feature-map (regardless of whether v^* is realizable or not) is called a **featurized MDP**. We also consider the analogous case when the linear combination of features, which can now also depend on the actions, gives the optimal action-value function q^* , as well as the case when the planner has access to two sets of features, one for representing the optimal value function and another one for representing the optimal action-value function.

The model described so far, regardless of which value functions are realizable, is called planning with **local access** as the planner can query the simulator for transitions and associated features only at states previously encountered. We also consider planning with **global access** to the features where the planner is given the set of all the states and associated features in advance (no queries required), and the option to query the simulator for transitions at any state of its choice.

Fix a set of featurized MDPs \mathcal{M} and a positive real δ . We say that a planner P is δ -**sound** for \mathcal{M} if for any $M \in \mathcal{M}$ and any start state s of the underlying MDP, the total expected value the planner achieves in an episode when used in M from s is at most δ worse than the optimal value associated with that start state. The set of δ -sound planners for \mathcal{M} under the global access (local access) model is denoted by $\mathcal{P}_{\text{GA}}(\mathcal{M}, \delta)$ (respectively, by $\mathcal{P}_{\text{LA}}(\mathcal{M}, \delta)$).

A planner's **query cost** is the worst-case expected number of queries it ever uses in a call. For the global access model and a featurized MDP M this is denoted by $q_{\text{GA}}(P, M)$, while for the local access model this is denoted by $q_{\text{LA}}(P, M)$.

Our main results are concerned with the **minimax query cost** that δ -sound planners incur. In particular, for a class of featurized MDPs \mathcal{M} and $\delta > 0$, we denote by $C_{\text{GA}}^*(\mathcal{M}, \delta)$ the minimax query cost (in short, query complexity) of δ -sound planners over \mathcal{M} given the global access model:

$$C_{\text{GA}}^*(\mathcal{M}, \delta) = \inf_{P \in \mathcal{P}_{\text{GA}}(\mathcal{M}, \delta)} \sup_{M \in \mathcal{M}} q_{\text{GA}}(P, M).$$

Similarly, we define

$$C_{\text{LA}}^*(\mathcal{M}, \delta) = \inf_{P \in \mathcal{P}_{\text{LA}}(\mathcal{M}, \delta)} \sup_{M \in \mathcal{M}} q_{\text{LA}}(P, M).$$

Local access is more demanding in that any planner that is δ -sound for \mathcal{M} under the local access model is automatically δ -sound for \mathcal{M} under the global access model. It follows that

$$C_{\text{GA}}^*(\mathcal{M}, \delta) \leq C_{\text{LA}}^*(\mathcal{M}, \delta) \tag{1}$$

no matter the choice of \mathcal{M} and δ .

In this paper we are concerned mainly with three classes of featurized MDPs. For $B \geq 0$ and positive integers d, H, A , these are defined as follows:

- v^* -realizable class: $\mathcal{M}_{B,d,H,A}^{v^*}$ is the class of finite-state-space featurized MDPs with A actions, where the feature-vectors are d -dimensional, the length of the episodes is H . For any (M, φ)

in this class, M is an MDP with some state space \mathcal{S} and random rewards confined to (say) $[0, 1]$, the associated feature-map $\varphi : \mathcal{S} \rightarrow \mathbb{R}^d$ with $\sup_{s \in \mathcal{S}} \|\varphi(s)\|_2 \leq 1$ is such that for some $\theta^* \in \mathbb{R}^d$ with $\|\theta^*\|_2 \leq B$, $v_M^*(s) = \varphi(s)^\top \theta^*$ holds for all $s \in \mathcal{S}$ where v_M^* is the optimal value function in M .¹

- q^* -realizable class: $\mathcal{M}_{B,d,H,A}^{q^*}$ is the class of featurized MDPs as above except that here for any (M, φ) in the class, for $[A] := \{1, \dots, A\}$, $\varphi : \mathcal{S} \times [A] \rightarrow \mathbb{R}^d$ with $\sup_{s \in \mathcal{S}, a \in [A]} \|\varphi(s, a)\|_2 \leq 1$ and $\theta^* \in \mathbb{R}^d$ with $\|\theta^*\|_2 \leq B$, we now require that $q_M^*(s, a) = \varphi(s, a)^\top \theta^*$ holds for all states $s \in \mathcal{S}$ and actions $a \in [A]$, where $q_M^*(s, a)$ is the optimal action-value at (s, a) .
- Reachable- v^*/q^* -realizable class: $\mathcal{M}_{B,d,H,A}^{v^*/q^*\text{reach}}$ is the class of featurized MDPs as above except that here the MDPs M are associated with two feature-maps, $\varphi_v : \mathcal{S} \rightarrow \mathbb{R}^d$ and $\varphi_q : \mathcal{S} \times [A] \rightarrow \mathbb{R}^d$ (their 2-norms bounded by 1 as before), and it is assumed that there exists some $\theta^* \in \mathbb{R}^d$ with $\|\theta^*\|_2 \leq B$ such that $v_M^*(s) = \varphi_v(s)^\top \theta^*$ and $q_M^*(s, a) = \varphi_q(s, a)^\top \theta^*$ hold for any action a and any state s of the MDP that is reachable from the initial states.²

Our main results are a lower and an upper bound for the query complexity of planning for \mathcal{M} where \mathcal{M} is one of the above classes. The lower bound is for the case when the number of actions grows polynomially with $d \wedge H$, where we define $a \wedge b := \min(a, b)$:

Theorem 1.1 (Lower bound with global access, at least $\text{poly}(d \wedge H)$ actions). *For δ, B, d, H sufficiently large, $A \geq d^{1/4} \wedge H^{1/2}$,*

$$C_{\text{GA}}^*(\mathcal{M} \cap \mathcal{M}^{\text{Pdet}}, \delta) = 2^{\Omega(d^{1/4} \wedge H^{1/2})},$$

where $\mathcal{M}^{\text{Pdet}}$ is the class of featurized MDPs with deterministic transitions and

$$\mathcal{M} \in \{\mathcal{M}_{B,d,H,A}^{v^*}, \mathcal{M}_{B,d,H,A}^{q^*}, \mathcal{M}_{B,d,H,A}^{v^*/q^*\text{reach}}\}.$$

Together with (1) this result also gives a lower bound on the query complexity of planning when only local access is available to the featurized MDP.

We also prove a result that provides a polynomial upper bound on the corresponding query complexity for a fixed number of actions.

Theorem 1.2 (Upper bound with local access). *For $\mathcal{M} \in \{\mathcal{M}_{B,d,H,A}^{v^*}, \mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{Pdet}}, \mathcal{M}_{B,d,H,A}^{v^*/q^*\text{reach}}\}$, arbitrary positive reals δ, B and arbitrary positive integers d, H ,*

$$C_{\text{LA}}^*(\mathcal{M}, \delta) = O\left(\text{poly}\left(\left(\frac{dH}{\delta}\right)^A, B\right)\right).$$

The rest of the paper is organized as follows: In the next section, we discuss these result and their relationship to existing work. This is followed by Section 3 that introduces our notation and gives the precise problem description. The proof of Theorem 1.1 is presented in Section 4: after a brief

-
1. For the sake of simplifying the notation, we assume that the states encode the stage index that the process can be at within an episode and we also add a final absorbing state where all actions incur a zero reward. This allows us to use the total expected reward criterion and thus a notation where the dependence on the stage index of values can be suppressed, and also means that we can talk about the initial states in an MDP.
 2. It is without loss of generality that we use that same θ^* in the inner products that yield $v_M^*(s)$ and $q_M^*(s, a)$: if these parameters are not shared, we can concatenate them with only a factor 2 increase in d and B .

intuitive overview in Section 4.1, for modularity, a simplified “abstract game” is introduced and is shown to be hard to solve in Section 4.2; the remaining proof turns the abstract game into a featurized MDP with a low action count that is similarly hard to solve for a planner; this is summarized in Section 4.3 before the arguments are presented in detail. Finally, the proof of Theorem 1.2 is given in Section 5.

2. Discussion and related works

A great many problems of interest can be formulated as optimal sequential decision making in a stochastic environment. If the model of the environment is given or learned with a sufficient accuracy, one only has to figure out how to use the model to find good actions. This is the problem addressed in planning. An elegant, minimalist approach to describe stochastic controlled environments is to adopt the language of MDPs. The price of simplicity (and thus generality) is that efficient planning in large state-spaces is intractable, a phenomenon pointed out by (Bellman, 1957) and today informally referred to as Bellman’s curse of dimensionality. While dynamic programming methods in MDPs with S states, A actions and a horizon of H can solve the planning problem with $\text{poly}(S, A, H)$ resources (Tseng, 1990; Ye, 2011; Scherrer, 2016), in the lack of extra information, the query cost of the easier problem of online planning (Chapter 6 Mausam and Kolobov, 2012) is at least $\Omega(A^H)$ when the number of states is unbounded (Kearns et al., 2002). An intriguing approach to avoid intractability when both S and H are large is the use of “function approximation” which promises to empower planners to extrapolate beyond the states that the planner has encountered. This approach has been proposed shortly after MDPs have been introduced when it was observed that in various problems of practical interest, value functions that the dynamic programming algorithms aim to compute can be well approximated with the linear combination of only a few basis functions, which themselves can be guessed by studying the structure of the problem to be solved (Bellman et al., 1963; Schweitzer and Seidmann, 1985). This raises the question of whether under such a favorable condition a provably efficient planner exist, i.e., whether the curse can be sidestepped.

While this question was arguably one of the main driving forces behind much of the research in operations research and reinforcement learning since the beginnings, most of the early results focused on the case when the function space underlying the features have a certain completeness property when dynamic programming algorithms can be successfully adopted (e.g., Bertsekas and Tsitsiklis, 1996; Tsitsiklis and Van Roy, 1996; Munos, 2003, 2005; Szepesvári and Munos, 2005). For more recent works in this, and some other related directions, see, e.g., (Du et al., 2019; Lattimore et al., 2020; Du et al., 2021) and the references therein.

While interesting, these works left open the question of whether efficient planners exist in the case when the function space may lack the completeness property but is still able to represent the optimal value function. The first results in this direction are quite recent (Wen and Roy, 2013; Weisz et al., 2021b,a; Du et al., 2021). Of these, the closest to our results is a result of Weisz et al. (2021b) who proved an exponential lower bound:

Theorem 2.1 (Weisz et al., 2021b, Theorem 9, lower bound for exponentially many actions). *For any $\delta > 0$ sufficiently small, and H, d, B sufficiently large, if $A = 2^{\Omega(d \wedge H)}$ then*³

$$C_{\text{GA}}^*(\mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{Pdet}}, \delta) = 2^{\Omega(d \wedge H)}.$$

3. Recall that $a \wedge b = \min(a, b)$.

According to this result, as long as there are exponentially many actions, planning remains intractable even for featurized MDPs where the features provided realize the optimal action-value function of the associated MDP and even if the MDPs are deterministic. Note that the exponential lower bound in this theorem is nontrivial since the query complexity of finding a good approximation to a function that lies in the span of d features from input-output examples is polynomial in the number of features regardless the cardinality of the input domain of the function, hence, the intractability in the above result cannot be solely attributed to the presence of a large action set.

Thus, an intriguing question is whether planning for the same setting as considered by that of Theorem 2.1 but with subexponential number of actions is tractable.

This question is partially answered by Theorem 1.1, which states that even with an action count that is polynomial in d and H , planning remains intractable for the same class of MDPs. This theorem also extends the result to two additional settings. The remaining problem then is whether planning is tractable when $2 \leq A = o(d^{1/4} \wedge H^{1/2})$. For a constant number of actions, Theorem 1.2 answers this question in the positive, though part of this theorem when $\mathcal{M} = \mathcal{M}_{B,d,H,A}^{v^*}$ has been proved earlier by Weisz et al. (2021a) and in fact the proof of Theorem 1.2 for the remaining two cases follows closely their proof. The importance of these extensions we give here is that they complement the lower bounds we prove. (Sadly, the featurized MDP classes used in these results are incomparable in the sense that we know of no general way of transforming results from one class to another, hence the need for the separate proofs.)

Intriguingly, even the new results leave open whether online planning with local access is tractable under q^ realizability when the MDPs involved have **stochastic transition dynamics and rewards** while the number of actions is fixed.*

In fact, even though our upper bound holds generally for the v^* -realizable and reachable- v^*/q^* -realizable classes of MDPs, for the q^* -realizable class our upper bound only holds for MDPs with deterministic transitions. If in addition to the transitions, the rewards are also deterministic, the result of Wen and Roy (2013) can be used to show a polynomial query (and even computational) complexity for online planning with local access. While the compute cost would depend linearly on the number of actions, the number of actions would not even appear in the query cost. (This result can also be proved directly by arguing that the subspace that contains q^* is either trivial, or a rollout with any action sequence will provide new information that can be used to decrease the dimension of this subspace by one.)

Given that our upper bound is polynomial when the number of actions is fixed, one may speculate that when the number of actions is large, perhaps one should replace each stage of an episode with $\log_2(A)$ stages, where actions would be chosen by determining their bits one by one, in a sequential fashion. The difficulty then is that this calls for an extension of the state space and a new, suitable feature-map. If one could derive a new, suitable feature-map given the old one and other information available during planning, this would result in a planner with query complexity $O(\text{poly}(\tilde{d}H \log_2(A)/\delta, B))$, showing a very mild dependence on the number of actions provided that \tilde{d} , the dimensionality of the new feature-map can be kept small. Sadly, our lower bound tells us that this **action binarization** approach cannot work when the number of actions is at least $\Omega(d^{1/4} \wedge H^{1/2})$.

Recently, the topic of online learning with good features has also seen many new results. As opposed to planning, here there is no simulator that can reset the state, and the only reset possible is

Publications	Action count	MDP class	poly(\cdot) sample complexity?
Wen and Roy (2013)	any	$\mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{det}}$	✓
Du et al. (2021)	any	$\mathcal{M}_{B,d,H,A}^{v^*/q^*}$	✓
Weisz et al. (2021a)	$O(1)$	$\mathcal{M}_{B,d,H,A}^{v^*}$	✓
Weisz et al. (2021b)	$2^{\Omega(d \wedge H)}$	$\mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{Pdet}}$	✗
This work	$\Omega(d^{1/4} \wedge H^{1/2})$	$\mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{Pdet}}$	✗
— " —	— " —	$\mathcal{M}_{B,d,H,A}^{v^*} \cap \mathcal{M}^{\text{Pdet}}$	✗
— " —	— " —	$\mathcal{M}_{B,d,H,A}^{v^*/q^* \text{reach}} \cap \mathcal{M}^{\text{Pdet}}$	✗
— " —	$O(1)$	$\mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{Pdet}}$	✓
— " —	— " —	$\mathcal{M}_{B,d,H,A}^{v^*}$	✓
— " —	— " —	$\mathcal{M}_{B,d,H,A}^{v^*/q^* \text{reach}}$	✓

Table 1: Comparison of various query complexity results for online planning with global access, and features realizing the optimal value or action-value function. The symbol \mathcal{M}^{det} stands for the class of finite MDPs with deterministic transitions and rewards. ✓ indicates the existence of a sound planner with query cost polynomial in relevant parameters (excluding S and A); ✗ indicates that such a planner does not exist.

to the start state of the MDP. As such, this is a harder setting than online planning. We would like to emphasize two results in this topic closely related to Theorem 1.2. First, the work of Du et al. (2021) implies that as long as all the features are given in advance (as in global access), regardless the number of actions A , the class $\mathcal{M}_{B,d,H,A}^{v^*/q^*}$ enjoys a minimax query complexity of order $\text{poly}(B, d, H)$. For this result, we define the class $\mathcal{M}_{B,d,H,A}^{v^*/q^*}$ like the class $\mathcal{M}_{B,d,H,A}^{v^*/q^* \text{reach}}$, except that realizability is required to hold over the **entire state-space**, and not only for states reachable from the initial states. Note that while according to Theorem 1.1, online planning with **global (and thus also local) access** over $\mathcal{M}_{B,d,H,A}^{v^*/q^* \text{reach}}$ is **intractable**, the result just mentioned implies that online planning with **global access** over $\mathcal{M}_{B,d,H,A}^{v^*/q^*}$ is **tractable**. Thus, while it is immediate from the definitions that

$$\mathcal{M}_{B,d,H,A}^{v^*/q^*} \subset \mathcal{M}_{B,d,H,A}^{v^*/q^* \text{reach}},$$

the two results together imply that the class on the right-hand side (RHS) is substantially larger than the one on the left-hand side (LHS). However, this difference disappears if in addition to the feature-maps, the planner working with the class on the RHS is also provided with the reachable subset of the state-space. Indeed, in this case, the MDP that the planner works with can be redefined by removing all the unreachable states, leaving a featurized MDP that belongs to the class on the LHS. The two results together thus indicate that this extra piece of knowledge is hard to obtain. Indeed, as this information is not provided in the **local access** model when the state space is redefined to reachable states only, Theorem 1.1 implies that online learning with **local access** is **intractable**, which proves the first exponential information theoretic separation result between local and global access that the authors are aware of (cf. Du et al., 2021).

The fact that the set of reachable states is hard to obtain should not be too surprising. In robotics and many other problems the reachable part of the state space is known to have a fairly complicated

geometry. In such applications, assuming that the features describe the value functions over the whole state space means that they encode the complicated shape of these reachable sets, as well. As this looks difficult to achieve, the class that requires realizability only for reachable states appears more attractive even though for this class, per our lower bound, tractability only holds when the number of actions is relatively small. For convenience, we summarize the results discussed so far in Table 1.

The second result of interest in online learning with good features is due to Wang et al. (2021), who gave an exponential lower bound in the flavor of Theorem 2.1 by adapting the hard MDP construction of Weisz et al. (2021b) to satisfy a **constant suboptimality gap** between the action values of the best and second-best actions for all states. Instead of exponentially downscaling the values in the more advanced stages, such a result is possible by implementing this reduction effect through zero-reward transitions to the episode-over stage, such that the probability of reaching an advanced stage (instead of the value at such a stage) is exponentially small. While this does not lead to a hardness result in our online planning setup (at least under global access, see e.g., Du et al., 2019), but we note that we expect similar modifications to the hard MDP class underlying our Theorem 1.1 to lead to a similar, constant suboptimality gap version of the theorem in the online learning case.

At this stage, the reader may also wonder about the limitations of the present work. For example, we restricted the MDPs to those that have finite state spaces. As it turns out, this was done only for the sake of convenience: on the one hand, the lower bounds are not impacted by this restriction, while, on the other hand, the upper bounds go through with no change to the proofs apart from the occasional need to switch to a more technical, measure-theoretic language. Similar comments apply to our MDP model which assumes that the same number of actions is available at all states, or changing the problem to allow misspecification errors (when the features can only represent the target functions only with some positive error). If one allows for such misspecification errors, the misspecification level η (measured by the maximum norm) will put a lower bound on the suboptimality gap δ that the planner can achieve. The argument of Weisz et al. (2021a) can be used in this case and gives that with a polynomial query complexity, one can achieve suboptimality gaps of magnitude $\delta = \Omega(\text{poly}(H, d)\eta^{1/A})$. While (similarly) the upper bound of Theorem 1.2 can be adapted to the infinite-horizon discounted MDP setting using the arguments of Weisz et al. (2021a), a more significant limitation of our lower bound (Theorem 1.1) is that it does not immediately apply to the discounted setting, even though the lower bound of Weisz et al. (2021b) (with exponentially many actions) does. Indeed, the linear structure of our MDP construction relies on the rewards counting towards v^* and q^* with the same weight if they are delayed by a few steps. The final limitation is that the results do not apply to problems where a controller is needed to work based on only features of the state. This is due to the limitation of the online planning protocol that requires state information to be passed to the planner (in addition to the features of the state). For settings like this, global planning, which directly aims for arriving at a policy that depends on the states only through the features, is more suitable.

3. Notation and problem setup

The purpose of this section is to introduce the notation we use and the necessary definitions that will allow us to precisely formulate the problems we study. We start with the notation. This is followed

by a quick review of definitions and basic concepts concerning MDPs. The section will be closed by describing the planning problems considered.

3.1. Notation

Let $\mathbb{N}_+ = \{1, 2, \dots\}$ be the set of positive integers, and $\mathbb{N} = \{0\} \cup \mathbb{N}_+$. Let \mathbb{R} denote the set of real numbers, $\mathcal{B}_d(r) = \{x \in \mathbb{R}^d : \|x\|_2 \leq r\}$ the d -dimensional ball of radius r , and let $[i] = \{1, \dots, i\}$ be the set of integers from 1 to i for an integer $i \in \mathbb{N}_+$. For $i, j \in \mathbb{N}$, we use $[i : j] = \{i, i+1, \dots, j\}$ if $i \leq j$, and $[i : j] = \{\}$ otherwise. For vectors a and b of compatible sizes, $\langle a, b \rangle = a^T b$ denotes their inner product. For a True or False statement X (possibly depending on random variables), let $\mathbb{I}\{X\}$ take 1 if X is True, and 0 otherwise. Let $a \wedge b = \min(a, b)$ and $a \vee b = \max(a, b)$. For an event E , let E^C denote its complementary event. Let $()$ denote the empty sequence.

3.2. Episodic Markov decision processes with bounded rewards

A **Markov decision process (MDP)** is defined by a tuple $M = (\mathcal{S}, \mathcal{A}, Q)$ of states, actions, and a transition-reward-kernel, respectively. The structure M defines a discrete time sequential decision making problem where in time step $t = 0, 1, \dots$, an environment responds to an action $A_t (\in \mathcal{A})$ of an agent by transitioning from its current state $S_t (\in \mathcal{S})$ to a new random state $S_{t+1} (\in \mathcal{S})$ while also generating a random reward $R_{t+1} \in \mathbb{R}$ so that the distribution of (R_{t+1}, S_{t+1}) given $S_0, A_0, R_1, S_1, \dots, A_{t-1}, R_t, S_t, A_t$ is given by $Q(\cdot | S_t, A_t)$ regardless of the history before S_t, A_t . Formally, Q is a probability kernel from state-action pairs to reward-state pairs. For simplicity, it is assumed that R_{t+1} above is supported in $[0, 1]$.

To simplify the presentation, we assume that the state space is finite. As noted beforehand, the definitions and results can be naturally translated to infinite state spaces. Similarly, assume that the set of actions is finite and $\mathcal{A} = [A]$ for some integer A .

In this work we focus on the **fixed-horizon undiscounted total expected reward objective**. Denoting the horizon by H , under this objective, the goal is to find a **policy**, a way of choosing actions given the past, such that the total expected reward over H steps is maximized regardless of the initial state of the process. (Formally, a policy is a stochastic kernel from histories to actions.) The H steps of the process is also called an **episode**. As it is well known, the optimal policy, which maximizes the stated objective, depends on the number of steps left before the episode finishes. In this work, we will use an equivalent formulation which avoids this dependence. In this formulation, only the first H rewards can be non-zero, while the process continues indefinitely and the objective is changed to the total undiscounted expected reward. To emulate the fixed-horizon setting, one can then create H disjoint copies of the state space, each corresponding to one step of the process while copying the transition structure to transition from one copy to the next one, and add an extra state (\perp) such that after H steps this state is reached from which point this state is never left while the reward incurred remains zero regardless of the actions taken. This is summarized below:

Assumption 3.1 (Fixed-horizon MDP). *The state space \mathcal{S} satisfies $\mathcal{S} = \cup_{h=0}^H \mathcal{S}_h$ with $\mathcal{S}_H = \{\perp\}$ and Q is such that for any $s \in \mathcal{S}_h$, $h \in [0 : H - 1]$ and $a \in \mathcal{A}$, $Q(\cdot | s, a)$ is supported on $[0, 1] \times (\{\perp\} \cup \mathcal{S}_{h+1})$, while for $h = H$, this support is $\{0\} \times \mathcal{S}_H = \{0\} \times \{\perp\}$. In particular, the sets $\{\mathcal{S}_h\}_{h \in [0:H]}$ are pairwise disjoint.*

Thanks to this assumption, when writing definitions, we consider the infinite horizon total expected reward criterion. This criterion assigns to a policy π used in MDP M from initial state

$s \in \mathcal{S}$ the value $v^\pi(s)$, which is defined as

$$v^\pi(s) = \mathbb{E}_{M,s}^\pi \left[\sum_{t=0}^{\infty} R_{t+1} \right]. \quad (2)$$

Here $\mathbb{E}_{M,s}^\pi$ is the expectation corresponding to the probability distribution $\mathbb{P}_{M,s}^\pi$ over trajectories of infinite length composed of state-action-reward triplets where this probability distribution arises from using policy π in every step, with the first state fixed to s , while next states and rewards are generated according to Q . Under our assumption the value of any policy in any state is well-defined. The value, obviously, depends on M , but to minimize clutter this dependence is suppressed. We will also need the action-value function of a policy. This is defined similarly as above, except that one fixes both the initial state and the initial action. Thus, for $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$q^\pi(s, a) = \mathbb{E}_{M,s,a}^\pi \left[\sum_{t=0}^{\infty} R_{t+1} \right]. \quad (3)$$

where $\mathbb{E}_{M,s,a}^\pi$ is the expectation corresponding to the probability distribution $\mathbb{P}_{M,s,a}^\pi$ over the trajectories as before, except that this time the first state-action pair is fixed to (s, a) instead of just fixing the first state to s . Note that under Assumption 3.1, both v^π (mapping states to reals, the **value function of π**) and q^π (mapping state-action pairs to reals, the **action-value function of π**) are well defined and take values in $[0, H]$ and the infinite sums can be truncated after stage H .

Define $v^\star : \mathcal{S} \rightarrow \mathbb{R}$ and $q^\star : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, the **optimal value** and, respectively, **optimal action-value function** as

$$v^\star(s) = \sup_{\pi} v^\pi(s), \quad q^\star(s, a) = \sup_{\pi} q^\pi(s, a), \quad s \in \mathcal{S}, a \in \mathcal{A}. \quad (4)$$

A policy π is said to be optimal if $v^\star = v^\pi$. It is well known that in our setting an optimal policy always exists and in fact the policy that uses any maximizer of $q^\star(s, \cdot)$ when the state S_t is s is an optimal policy. This policy, as the choice of the action only depends on the last state, is called **memoryless**. Since the choice is also deterministic, the policy is also **deterministic**. A deterministic memoryless policy can be concisely given as a map from states to actions. By slightly abusing notation, in what follows, we will identify such policies with such maps and write $\pi : \mathcal{S} \rightarrow \mathcal{A}$ to denote a memoryless deterministic policy. Given such a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$, its value functions v^π and q^π satisfy the following equations:

$$v^\pi(s) = q^\pi(s, \pi(s)), \quad (5)$$

$$q^\pi(s, a) = r(s, a) + \sum_{s' \in \mathcal{S}} P(s'|s, a) v^\pi(s'), \quad s \in \mathcal{S}, a \in \mathcal{A}, \quad (6)$$

where $P(s'|s, a)$, derived from the transition kernel Q , is the probability of arriving at state s' when the process is in state s and action a is taken while $r(s, a)$ is the expected reward along this transition. Formally, $P(s'|s, a) = Q([0, 1] \times \{s'\} | s, a)$ and $r(s, a) = \int_{[0,1] \times \mathcal{S}} r dQ(r, s' | s, a)$. The coupled equations (6) are known as the Bellman equations for π (Puterman, 1994).

Oftentimes in MDPs the rewards and the next states are independently chosen. In this case, $Q(\cdot | s, a)$ takes the form of the “product” of a probability kernel R mapping state-action pairs to $[0, 1]$ and the probability kernel P mapping state-action pairs to states. In some constructions below, we will thus specify an MDP with the help of two such kernels.

When the dependence of v^π , v^\star , or q^\star on M is important, we will put M in the index of these symbols. For example, for a policy π for M , we will write v_M^π to denote its value function in M .

3.3. Online planning with featurized MDPs

The purpose of this section is to make the earlier definitions more formal. While we repeat some of the definitions (in a more precise way) from the introduction, to avoid unnecessary duplications, some definitions (e.g., the definitions of featurized MDPs) are not repeated.

As described in the introduction, in **online planning** a planner is used in a closed-loop configuration. In our H -horizon setting, given an MDP $M = (\mathcal{S}, \mathcal{A}, Q)$ satisfying Assumption 3.1, in step $t \in [0 : H - 1]$ of using the planner, the planner is given access to state S_t of the process. By convention, $S_0 \in \mathcal{S}_0$ and thus we also have $S_t \in \mathcal{S}_t$, for every t , where $(\mathcal{S}_h)_{0 \leq h \leq H}$ is the decomposition of \mathcal{S} from Assumption 3.1. The planner is given access to a **simulation oracle** that can be queried with state action pairs $(s, a) \in \mathcal{S} \times \mathcal{A}$, to which the oracle responds with a reward-state pair (R', S') generated from Q :

$$(R', S') \sim Q(\cdot | s, a).$$

The planner is given the freedom to decide which queries and how many of them to use whenever it is called. Eventually, the planner needs to stop querying and return an action $A_t \in \mathcal{A}$, which is then used to generate the next state in M and an associated reward:

$$(R_{t+1}, S_{t+1}) \sim Q(\cdot | S_t, A_t).$$

When used in an MDP M , a planner induces a policy π_M . Note that π_M is a stochastic policy (possibly history-dependent, if the planner saves information between its calls), where the stochasticity comes from the randomness of the entire planner-oracle interaction (and possibly some independent randomization). As such, π itself is not a random element. The goal of planner design is to find planners that induce near-optimal policies for large classes of MDPs while controlling the total computational cost of planning. Formally, a **planner is δ -sound** for class \mathcal{M} if for any $M = (\cup_h \mathcal{S}_h, \mathcal{A}, Q) \in \mathcal{M}$ and any $s_0 \in \mathcal{S}_0$, if the planner is used in M while $S_0 = s_0$, it holds that $\mathbb{E}[\sum_{t=1}^H R_t | S_0 = s_0] \geq v_M^*(s_0) - \delta$, or, equivalently,

$$v_M^{\pi_M}(s_0) \geq v_M^*(s_0) - \delta.$$

Above, \mathbb{E} is the expectation operator induced by the probability measure \mathbb{P} induced by the interconnection of the planner with the simulation oracle of M and MDP M over the interaction sequences that contain all information that flows between the planner, the simulation oracle and the MDP. As all queries involve a computation step, a lower bound on the compute cost of planning is the **query cost**, which is defined as the expected number of queries that the planner uses in a planning step (or call). For a planner and MDP, the maximal query cost over all possible calls to the planner is the worst-case query cost of using the planner on the MDP. The notion is naturally extended to the concept of query complexity of classes of MDPs by again taking the worst-case over the possible MDPs in the class.

In the special case when the planner is given access to the state space (and feature-map) of the MDP and can thus call the simulator at any state of the MDP, the problem described so far is known as (online) planning with a **generative model**, or **global access**. As opposed to this, if the planner is not given access to the state space (or the feature-map) and can only query the simulator at states that it has encountered beforehand, we say that the planner has **local access** only to the simulator.

When interacting with a featurized MDP (M, φ_v) in the local access model, with, say, state features (i.e., $\varphi_v : \mathcal{S} \rightarrow \mathbb{R}^d$, where $M = (\mathcal{S}, \mathcal{A}, Q)$), in step t of the planning process, the planner is

called with $(S_t, \varphi_v(S_t))$, while the simulation oracle returns

$$(R', S', \varphi_v(S')) \sim Q(\cdot | s, a)$$

when queried with (s, a) . In an alternate problem setting if the features are defined over state-action pairs, i.e., the planner interacts with a featurized MDP of the form (M, φ_q) with $\varphi_q : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}^d$, in step t of the planning process, the planner is called with $(S_t, \varphi_q(S_t), (\varphi_q(S_t, a))_{a \in \mathcal{A}})$, while the simulation oracle returns

$$(R', S', (\varphi_q(S', a))_{a \in \mathcal{A}}) \sim Q(\cdot | s, a)$$

when queried with (s, a) . When the setting is such that both state and state-action features are available, the interaction is modified accordingly to include both types of features. The definitions of query cost and soundness remain the same for these settings.

4. Lower bound (the proof of Theorem 1.1)

In this section we give the proof of Theorem 1.1. We start by introducing the high level ideas underlying the proof.

4.1. Overview

We prove the lower bound by designing a class of MDPs where by traversing the MDP, the agent effectively has to pick corners of a p -dimensional hypercube, in sequence, until either K picks were made or a pick was sufficiently close to the secret “solution” corner w^* . Here, $p \approx H^{1/2} \wedge d^{1/4}$ (large if both H and d are large) and $K \approx H/p$ (large if H is large). If the agent picks a corner close to the solution, the episode is effectively terminated and the agent receives the highest possible reward achievable from that state. Otherwise, the agent’s next pick has to substantially differ from the previously picked corner. After each choice, the highest reward achievable shrinks by a penalty factor that is governed by how different the subsequent picks are: picking dissimilar corners results in a larger penalty (i.e., a smaller penalty factor). Since subsequent picks need to be substantially different, this means that q^* (or v^*) reduces at an exponential rate throughout the episode until a guess is sufficiently close to the solution or all K picks are exhausted, in which case the agent receives a Bernoulli reward with expectation $\exp(-\Omega(K))$. Without additional information, guessing sufficiently close to the solution is a needle-in-a-haystack problem with an exponentially large haystack: with probability above (say) $3/4$, the secret corner will not be found within $\exp(\Omega(p))$ guesses. Additional information is not provided to the agent as long as the final reward is 0. Since the probability that this Bernoulli outcome is identically zero for the first $\exp(\Omega(K))$ guesses can be made to be $3/4$ or larger, if a planner uses at most $\exp(\Omega(p \wedge K))$ guesses, with probability at least $1/2$, neither blind guessing nor the Bernoulli outcomes will lead to success. Thus, in expectation, any sound planner has to query more than $\exp(\Omega(p \wedge K))$ times.

To achieve realizability of q^* (or v^*), it is sufficient if the value of the optimal policy is a low-order polynomial of the p -dimensional secret solution at any state in the MDP. To achieve this, the mechanics of choosing a guess and the penalty factor are carefully chosen in such a way that the optimal policy has a simple “greedy” structure that moves any guess as close as possible to the solution. The value of this greedy optimal policy is then proved to be a 4th-order polynomial of w^* , which gives rise to a $d \approx p^4$ dimensional feature-map that can realize the optimal values.

For the sake of simplicity and modularity, rather than defining the MDP, we first define a simplified “abstract game” where an “abstract planner” has to guess the above-mentioned secret parameter. This abstract game is essentially what has been described in the previous paragraph. This construction focuses on the information theoretic aspect of the proof, leaving the construction of the MDP with the required realizability properties to the subsequent sections.

4.2. Abstract game

The abstract game has a length parameter $K \in \mathbb{N}_+$ and an integer dimensionality parameter $p \geq 2$, which are known to the abstract planner. Let $W = \{-1, 1\}^p$. Let $\mathbf{0}$ and $\mathbf{1}$ indicate the p -dimensional vectors of all zeros and all ones, respectively. For vectors x and y from W , define $\text{diff}(x, y)$ as the Hamming distance between x and y , i.e., the number of components where x and y are different. We will use the property of the Hamming distance that it can be written as an (affine) bilinear function of its arguments: for $w_1, w_2 \in W$,

$$\text{diff}(w_1, w_2) = \frac{1}{2} (p - \langle w_1, w_2 \rangle) . \quad (7)$$

Note that $\text{diff}(\cdot, \cdot)$ is a metric on the set W . Let

$$W^\star = \{w \in W : p/4 \leq \text{diff}(\mathbf{1}, w) \leq 3p/4\} \quad (8)$$

be the set that will hold the game’s secret parameter: $w^\star \in W^\star$. For any $k \in \mathbb{N}$, let

$$W^{\circ k} = \{(w_i)_{i \in [k]} \in W^k : \text{diff}(w_{i-1}, w_i) \geq p/4 \text{ for } i \in [k]\} , \quad (9)$$

with $w_0 := \mathbf{1}$ defined for convenience, be the subset of k -length sequences of W where the elements are “sufficiently far” from each other.

The union of these over $k \leq K$ is the action set of the bandit-like game. Given w^\star , the reward function $f_{w^\star} : \{()\} \cup \bigcup_{k \in [K]} W^{\circ k} \rightarrow \mathbb{R}$ (index dropped when clear from context) is defined as follows (again $w_0 := \mathbf{1}$):⁴ $f(\cdot) = g(\text{diff}(w_0, w^\star))$, and for $k \in [K]$,

$$f_{w^\star}((w_i)_{i \in [k]}) = \left(\prod_{i \in [k]} g(\text{diff}(w_{i-1}, w_i)) \right) g(\text{diff}(w_k, w^\star)) \quad \text{where}$$

$$g(x) = 1 - \frac{x}{p} + \frac{(x-1)x}{2p^2} .$$

The game is sequential. It proceeds in steps where the abstract planner performs a query and receives a corresponding response (both the query and the response may be randomized). At each step $t \in \mathbb{N}_+$, the abstract planner randomly chooses whether to continue or not, and what its output or next query (correspondingly) is. If it continues, it chooses a sequence length $L_t \in [K]$, and a sequence $S_t = (w_i^t)_{i \in [L_t]} \in W^{\circ L_t}$. Otherwise, if it returns, it chooses its output $S_t = (w_i^t)_{i \in [8]} \in W^{\circ 8}$.

4. The reason for this form of f will become clear only when the MDP corresponding to the abstract game is defined. For now, let us only note that (1) as the input sequence grows in size, their elements being sufficiently far ensures an exponential rate of reduction of f , and (2) $g(x)$ is the second-order Taylor expansion of $(1 - 1/p)^x$, which ensures through some inequalities that the optimal strategy for maximizing f is to greedily move towards w^\star in the MDP as fast as possible. A simple optimal policy with a low-order polynomial expression for f allows deriving linear features for the MDP’s value function.

Note that the output is confined to have the fixed length of 8.⁵ To distinguish this from the case when the planner continues, we let $L_t = 0$ denote that the planner wants to return an output. Let $N = \min \{t \in \mathbb{N}_+ : L_t = 0\}$ indicate the step at which the planner returns. Thus, the planner's output is S_N .

At step t , denote the choice of the planner by $X_t = (L_t, S_t)$. If the planner is not done yet ($L_t > 0$, and thus $t < N$) then, in response to the planner's query, a random response $Y_t = (U_t, V_t, Z_t) \in \{0, 1\} \times \{0, 1\} \times [0, 1]$ is generated as follows:

- U_t indicates whether the penultimate component of S_t is close to w^\star (for convenience define $w_0^t = \mathbf{1}$):

$$U_t = \mathbb{I}\{\text{diff}(w_{L_t-1}^t, w^\star) < p/4\}.$$

- V_t indicates whether the last component of S_t is close to w^\star :

$$V_t = \mathbb{I}\{\text{diff}(w_{L_t}^t, w^\star) < p/4\}.$$

- Z_t is distributed as $\text{Ber}(f_{w^\star}(S_t))$ if either $V_t = 1$ (the last component of S_t is close to w^\star) or $L_t = K$ (all components are used in S_t), else $Z_t = 0$. Here, Ber denotes the Bernoulli distribution. This is well-defined as $f_{w^\star}(S_t) \in [0, 1]$ by Lemma 4.2.

If, on the other hand, the planner indicates that it is done ($L_t = 0$, and thus $t = N$) then there is no feedback, but the payoff (reward) to the planner is

$$R = f_{w^\star}\left((w_i^N)_{i \in [k^\star]}\right) \quad (10)$$

where $k^\star = k^\star(S_t; w^\star)$ denotes the first component of $S_t = (w_k^N)_{k \in [8]}$ that is sufficiently close to w^\star , or 8 if none of them are:

$$k^\star = \min\{k \in [8] : k = 8 \text{ or } \text{diff}(w_k^N, w^\star) < p/4\}.$$

For future reference, it will be useful to introduce $\tau_{w^\star}(s)$ to denote the first $k^\star(s, w^\star)$ components of $s = (w_i)_{i \in [8]}$ so that $R = f_{w^\star}(\tau_{w^\star}(S_t))$. While the interaction is over at this stage, for simplifying notation, we introduce Y_t and define it as $Y_t = (0, 0, 0)$.

This finishes the description of the abstract game; for a given value of w^\star we will refer it as ‘‘abstract game w^\star ’’. To summarize, in this game, the planner can choose actions from a combinatorially structured action set to collect information for the final round where it needs to choose an action from a smaller (but still combinatorially large) subset of the action set. The feedback is nonlinear. The essence of the information theoretic argument that will follow will be that good planners essentially need to find w^\star .

For these information theoretic arguments, as well as the statement of the main result of this section, some extra definitions are necessary. For $t \in \mathbb{N}_+$, let $F_t = (X_i, Y_i)_{i \in [t-1]}$. For each step t sequentially, if the game is not over yet, i.e., $t - 1 < N$, the planner \mathcal{A} defines the distribution of X_t given F_t . Given F_t and X_t , the distribution of Y_t is defined as above. Together, \mathcal{A} and w^\star define $\mathbb{P}_{w^\star}^{\mathcal{A}}$, the probability distribution over interaction sequences $(X_t, Y_t)_{t \in [N]}$ between the planner and

5. The constant 8 here is sufficiently small to prove that planners cannot guess close enough to w^\star with any of these 8 attempts, yet large enough so that to achieve a small suboptimality in the MDP problem (that will be derived later), it will be crucial to guess a vector among these 8 vectors that is close to w^\star .

the game, where the sequence needs to satisfy that $L_t > 0$ for $t < N$ and $L_N = 0$.⁶ The planner is well-defined if $\mathbb{P}_{w^\star}^{\mathcal{A}}[N < \infty] = 1$. Let $\mathbb{E}_{w^\star}^{\mathcal{A}}$ be the expectation operator corresponding to $\mathbb{P}_{w^\star}^{\mathcal{A}}$. The abstract planner is sound with worst-case query cost \bar{N} if for all $w^\star \in W^\star$, $\mathbb{E}_{w^\star}^{\mathcal{A}}[N - 1] \leq \bar{N}$, and $\mathbb{E}_{w^\star}^{\mathcal{A}}[R] \geq \max_{s \in W^{\circ 8}} f_{w^\star}(\tau_{w^\star}(s)) - 0.01$. We note in passing that $\max_{s \in W^{\circ 8}} f_{w^\star}(\tau_{w^\star}(s)) = f_{w^\star}(\cdot)$, i.e., the maximizing sequence is the empty sequence.

The main result of this section is the following claim, which states that the abstract game is hard:

Theorem 4.1. *For any abstract planner that is sound with query cost \bar{N} ,*

$$\bar{N} = 2^{\Omega(p \wedge K)}.$$

The proof is given in a number of lemmas. We start with some elementary properties of f_{w^\star} :

Lemma 4.2 (Properties of f_{w^\star}). *For any $w^\star \in W^\star$, $k \in \mathbb{N}_+$, $s = (w_{k'})_{k' \in [k]} \in W^{\circ k}$, the following hold:*

$$\frac{11}{32} \leq f_{w^\star}(\cdot) \leq \frac{25}{32}, \quad (11)$$

$$0 < f_{w^\star}(s) \leq \left(\frac{25}{32}\right)^{k + \mathbb{1}\{\text{diff}(w_k, w^\star) \geq p/4\}} \quad (12)$$

Proof. We prove Eq. 12 by first showing that

$$0 < f_{w^\star}(s) \leq \left(\frac{25}{32}\right)^k. \quad (13)$$

This follows since f is the product of $k + 1$ terms, each defined using the function g . Now, notice that $g(x)$ decreases as x increases in the range $0 \leq x \leq p$, so for all $k' \in [k]$, thanks to $\text{diff}(w_{k'-1}, w_{k'}) \geq p/4$ which holds since by assumption $s \in W^{\circ k}$, we have

$$0 < g(p) \leq g(\text{diff}(w_{k'-1}, w_{k'})) \leq g(p/4) < \frac{25}{32}.$$

This, together with $0 < g(0) \leq 1$ proves Eq. 13. To finish the proof of Eq. 12, note that if $\text{diff}(w_k, w^\star) \geq p/4$ then, similarly to the previous case, we have $0 < g(\text{diff}(w_k, w^\star)) \leq g(p/4) < \frac{25}{32}$, which implies Eq. 12. As $w^\star \in W^\star$, $\frac{1}{4}p \leq \text{diff}(\mathbf{1}, w^\star) \leq \frac{3}{4}p$. Hence, $f_{w^\star}(\cdot) = g(\text{diff}(\mathbf{1}, w^\star)) \geq g(\frac{3}{4}p) \geq \frac{11}{32}$ and $f_{w^\star}(\cdot) \leq g(\frac{1}{4}p) \leq \frac{25}{32}$. \blacksquare

Let

$$n = \left\lfloor \min \left(\frac{e^{\frac{p}{8}}}{16} - 5, \frac{\frac{1}{\varepsilon} - 1}{7.5} \right) \right\rfloor, \quad (14)$$

where

$$\varepsilon = \left(\frac{25}{32}\right)^{K+1}.$$

6. Luckily for us, F_t takes values in a finite set, which makes it trivial to show that $\mathbb{P}_{w^\star}^{\mathcal{A}}$ with the required properties exist.

For any $w^\star \in W^\star$, let $E_n^{w^\star}$ be the event when in the first n steps the planner does not hit on any vector that is close to w^\star :

$$E_n^{w^\star} = \bigcap_{t \in [n]} \left\{ t > N \text{ or } \left(t = N \text{ and } \min_{i \in [8]} \text{diff}(w_i^N, w^\star) \geq \frac{p}{4} \right) \right. \\ \left. \text{or } \left(t < N \text{ and } \text{diff}(w_{L_{t-1}}^t, w^\star) \geq p/4 \text{ and } \text{diff}(w_{L_t}^t, w^\star) \geq p/4 \right) \right\}.$$

We define the ‘‘abstract game 0’’ (and, for any planner \mathcal{A} , the associated probability distribution $\mathbb{P}_0^{\mathcal{A}}$) to be a variant of the game where the responses are $Y_t \equiv (0, 0, 0)$ for all $t \in \mathbb{N}_+$ (irrespective of the choices of the planner).

Our next lemma claims that the bad event $E_n^{w^\star}$ happens with large probability in abstract game w^\star whenever it happens with large probability in abstract game 0. The reason for this is that the probability of ever receiving nonzero feedback on the bad event is a small value, which in fact can be bounded by ε (the only way to receive nonzero feedback is by playing to the end, hence ε appears). From here it will follow that since the number of steps is at most n (bad events are defined for interactions of length at most n), the probability of $E_n^{w^\star}$ in game w^\star is at least the probability of this event in game 0 times $(1 - \varepsilon)^n$, and the latter is lower bounded by an absolute constant because n is chosen to be not too large compared to $1/\varepsilon$.

Lemma 4.3. *Take n as defined in Eq. 14. Then, for any abstract planner \mathcal{A} and for any $w^\star \in W$,*

$$\mathbb{P}_{w^\star}^{\mathcal{A}}(E_n^{w^\star}) \geq \frac{7}{8} \mathbb{P}_0^{\mathcal{A}}(E_n^{w^\star}).$$

Proof. We prove that

$$\mathbb{P}_{w^\star}^{\mathcal{A}}(E_n^{w^\star}) \geq (1 - \varepsilon)^n \mathbb{P}_0^{\mathcal{A}}(E_n^{w^\star}). \quad (15)$$

Since by its choice, n satisfies $n \leq \left(\frac{1}{\varepsilon} - 1\right) / 7.5$, or, equivalently, $1 - \varepsilon \geq 1 - \frac{1}{1+7.5n}$, it follows that

$$(1 - \varepsilon)^n \geq \left(1 - \frac{1}{1+7.5n}\right)^n \geq \lim_{n \rightarrow \infty} \left(1 - \frac{1}{1+7.5n}\right)^n = e^{-1/7.5} > 7/8,$$

which shows that it suffices to prove Eq. 15.

Let $(X_t, Y_t)_{t \in [N]}$ be a complete interaction history and let H denote the first $n \wedge N$ components of this (thus, H is shorter than the complete sequence when $n < N$). Let \mathcal{H} be the set of all possible values that H can take. For $h \in \mathcal{H}$, let $E_h = E_n^{w^\star} \cap \{H = h\}$. Clearly, $E_n^{w^\star}$ is the disjoint union of the sets $\{E_h\}_{h \in \mathcal{H}}$. Let $\mathcal{H}^+ = \{h \in \mathcal{H} : \mathbb{P}_0^{\mathcal{A}}(E_h) > 0\}$. Then, $\mathbb{P}_0^{\mathcal{A}}(E_n^{w^\star}) = \sum_{h \in \mathcal{H}^+} \mathbb{P}_0^{\mathcal{A}}(E_h)$, and we prove Eq. 15 by showing that for any $h \in \mathcal{H}$,

$$\rho = \frac{\mathbb{P}_{w^\star}^{\mathcal{A}}(E_h)}{\mathbb{P}_0^{\mathcal{A}}(E_h)} \geq (1 - \varepsilon)^n. \quad (16)$$

Fix $h \in \mathcal{H}^+$ and let $h = (x_t, y_t)_{t \in [n']}$ for some $0 < n' \leq n$. Further, let $x_t = (l_t, s_t)$. Note that for $t < n'$, $l_t > 0$ and either $n' = n$ or $l_{n'} = 0$.

As $\mathbb{P}_0^{\mathcal{A}}(E_h) > 0$, $y_t = (0, 0, 0)$ for all $t \in [n']$. By definition of $\mathbb{P}_{w^\star}^{\mathcal{A}}$ and $\mathbb{P}_0^{\mathcal{A}}$, both the numerator and denominator factorizes into the product of n' terms. Given the same history, the distribution of X_t under both $\mathbb{P}_{w^\star}^{\mathcal{A}}$ and $\mathbb{P}_0^{\mathcal{A}}$ are identical, so the terms that do not cancel remain:

$$\rho = \prod_{t=1}^{n'} \frac{\mathbb{P}_{w^\star}^{\mathcal{A}}(Y_t = (0, 0, 0) | X_t = x_t)}{\mathbb{P}_0^{\mathcal{A}}(Y_t = (0, 0, 0) | X_t = x_t)} = \prod_{t=1}^{n'} \mathbb{P}_{w^\star}^{\mathcal{A}}(Z_t = 0 | X_t = x_t),$$

where $Y_t = (U_t, V_t, Z_t)$. Here, the last equality follows since $\mathbb{P}_0^{\mathcal{A}}[Y_t = (0, 0, 0) | X_t = x_t] = 1$ by definition and $\mathbb{P}_{w^\star}^{\mathcal{A}}(Y_t = (0, 0, 0) | X_t = x_t) = \mathbb{P}_{w^\star}^{\mathcal{A}}(Z_t = 0 | X_t = x_t)$ because on $E_h \subset E_n^{w^\star}$, $U_t = V_t = 0$ holds $\mathbb{P}_{w^\star}^{\mathcal{A}}$ almost surely. Now, by definition, $\mathbb{P}_{w^\star}^{\mathcal{A}}(Z_t = 1 | X_t = x_t) = f_{w^\star}(s_t) \mathbb{I}\{\text{diff}(w_{l_t}^t, w^\star) \leq p/4 \text{ or } l_t = K\}$. Since $E_h \subset E_n^{w^\star}$, $\text{diff}(w_{l_t}^t, w^\star) \leq p/4$ does not hold. Hence, $\mathbb{P}_{w^\star}^{\mathcal{A}}(Z_t = 1 | X_t = x_t) = f_{w^\star}(s_t) \mathbb{I}\{l_t = K\} \leq (25/32)^{K+1} = \varepsilon$, where the inequality follows from Lemma 4.2 using again that $E_h \subset E_n^{w^\star}$ and thus the last component of s_t must be “far” from w^\star . Putting things together and using that $n' \leq n$ gives that $\rho \geq (1 - \varepsilon)^n$, as required. \blacksquare

We plan to argue that the bad event happens with large probability in game 0. In this game, by definition, the planner needs to guess w^\star blindly (as there is no feedback ever). Hence, the success of the planner depends on whether they can without any feedback stumble upon w^\star . To bound this success rate, it will be useful to bound the number of vectors close to a given vector in the hypercube W :

Lemma 4.4. *For any $\tilde{w} \in W$, let $W_{\text{close}}(\tilde{w}) = \{w \in W \mid \text{diff}(w, \tilde{w}) < p/4\}$. Then,*

$$|W_{\text{close}}(\tilde{w})| \leq 2^p \exp\left(-\frac{p}{8}\right)$$

Proof. By symmetry of the p -dimensional hypercube, without loss of generality, let $\tilde{w} = \mathbf{1}$ and $W_{\text{close}} = W_{\text{close}}(\tilde{w})$. Let $X = (X_i)_i \in W$ be a uniformly distributed random variable on W . Note that the components X_i of X are independent Rademacher random variables. We have

$$\begin{aligned} |W_{\text{close}}| &= \sum_{w \in W} \mathbb{I}\{\langle w, \mathbf{1} \rangle > p/2\} = |W| \mathbb{P}(\langle X, \mathbf{1} \rangle > p/2) \\ &= 2^p \mathbb{P}\left(\sum_{i \in [p]} X_i > p/2\right) \leq 2^p \exp\left(\frac{-2(p/2)^2}{4p}\right) = 2^p \exp\left(-\frac{p}{8}\right), \end{aligned}$$

where the second inequality holds by Hoeffding’s inequality. \blacksquare

Our next lemma shows that for any planner the probability of a bad event has an absolute lower bound. We use the previous lemma to show that for any planner there exists a w^\star such that the probability of the corresponding bad event is lower bounded in game 0, and then we apply Lemma 4.3 to get a lower bound for the same event in game w^\star .

Lemma 4.5. *For any abstract planner \mathcal{A} there exists $w^\star \in W^\star$ such that*

$$\mathbb{P}_{w^\star}^{\mathcal{A}}(E_n^{w^\star}) \geq \left(\frac{7}{8}\right)^2.$$

Proof. For any $\hat{w} \in W^\star$, under event $(E_n^{\hat{w}})^c$, either there exists $t \in [n \wedge (N-1)]$ such that $\text{diff}(w_{L_{t-1}}^t, \hat{w}) < p/4$ or $\text{diff}(w_{L_t}^t, \hat{w}) < p/4$, or for some $i \in [8]$, $\text{diff}(w_i^N, \hat{w}) < p/4$. That is, $(E_n^{\hat{w}})^c \subset \{\hat{w} \in Z\}$ where

$$Z := \bigcup_{t \in [n \wedge (N-1)]} \left(W_{\text{close}}(w_{L_{t-1}}^t) \cup W_{\text{close}}(w_{L_t}^t) \right) \cup \left(\bigcup_{i \in [8]} W_{\text{close}}(w_i^N) \right).$$

By Lemma 4.4,

$$|Z| \leq (2n+8)2^p \exp\left(-\frac{p}{8}\right). \quad (17)$$

We also have that $W^\star = W \setminus W_{\text{close}}(\mathbf{1}) \setminus W_{\text{close}}(-\mathbf{1})$, so $|W^\star| \geq 2^p (1 - 2 \exp(-\frac{p}{8}))$. As $w^\star \in Z$ is the good event for the planner, we define

$$w^\star = \arg \min_{\hat{w} \in W^\star} \mathbb{P}_0^{\mathcal{A}}(\hat{w} \in Z). \quad (18)$$

Putting things together and using that $Z \subseteq W$, we get

$$\begin{aligned} 2^p \left(1 - 2 \exp\left(-\frac{p}{8}\right)\right) \mathbb{P}_0^{\mathcal{A}}(w^\star \in Z) &\leq |W^\star| \mathbb{P}_0^{\mathcal{A}}(w^\star \in Z) \\ &\leq \sum_{\hat{w} \in W^\star} \mathbb{P}_0^{\mathcal{A}}(\hat{w} \in Z) \leq \sum_{\hat{w} \in W} \mathbb{P}_0^{\mathcal{A}}(\hat{w} \in Z) = \mathbb{E}_0^{\mathcal{A}}[|Z|] \leq (2n+8)2^p \exp\left(-\frac{p}{8}\right), \end{aligned}$$

Rearranging and using $(E_n^{w^\star})^c \subset \{w^\star \in Z\}$, we get

$$\mathbb{P}_0^{\mathcal{A}}\left(\left(E_n^{w^\star}\right)^c\right) \leq \mathbb{P}_0^{\mathcal{A}}(w^\star \in Z) \leq \frac{(2n+8)2^p \exp\left(-\frac{p}{8}\right)}{2^p (1 - 2 \exp\left(-\frac{p}{8}\right))} \leq 2(n+5) \exp\left(-\frac{p}{8}\right) \leq \frac{1}{8},$$

where the last two inequalities follow by our choice of n . Combining this with Lemma 4.3 finishes the proof. \blacksquare

With this, we are ready to prove Theorem 4.1. In fact, all that is left to show is that if the planner is sound, then the probability of the bad event cannot be too high. That is, connecting the bad event to poor performance.

Proof of Theorem 4.1. Take a sound abstract planner \mathcal{A} with query cost \bar{N} . Let w^\star be the vector whose existence is guaranteed by the previous lemma. By Markov's inequality,

$$\mathbb{P}_{w^\star}^{\mathcal{A}}[N-1 \geq n] \leq \frac{1}{n} \bar{N}.$$

Let E' be the event under which both $N-1 < n$ and $E_n^{w^\star}$ hold: $E' = \{N-1 < n\} \cap E_n^{w^\star}$. By the union bound and Lemma 4.5,

$$\mathbb{P}_{w^\star}^{\mathcal{A}}[E'] \geq \left(\frac{7}{8}\right)^2 - \frac{1}{n} \bar{N}. \quad (19)$$

Under the event E' , the output of the planner $(w_i^N)_{i \in [8]}$ satisfies $\text{diff}(w_i^N, w^\star) \geq p/4$ for $i \in [8]$, and therefore $k^\star = 8$ and, by Lemma 4.2, the reward R of the game satisfies $R < \left(\frac{25}{32}\right)^9$. Therefore, combined with the soundness of \mathcal{A} , we get

$$\begin{aligned} \frac{11}{32} - 0.01 \leq f_{w^\star}(\cdot) - 0.01 \leq \mathbb{E}_{w^\star}^{\mathcal{A}} [R] &\leq \left(\frac{25}{32}\right)^9 + (1 - \mathbb{P}_{w^\star}^{\mathcal{A}} [E']) \frac{25}{32} \\ &\leq \left(\frac{25}{32}\right)^9 + \left(1 - \left(\frac{7}{8}\right)^2\right) \frac{25}{32} + \frac{\bar{N}}{n} \frac{25}{32}, \end{aligned}$$

where we used Lemma 4.2 to bound $f_{w^\star}(\cdot)$, and the maximum value of R (maximum value of f) by $\frac{25}{32}$. To satisfy this inequality, we must have $\bar{N} > 0.05n$, and thus by substituting Eqs. 14 and simplifying we get

$$\bar{N} = \Omega \left(\min \left(\frac{e^{\frac{p}{8}}}{16} - 5, \frac{1}{7.5} - 1 \right) \right) = \min \left(2^{\Omega(p)}, \Omega \left(\left(\frac{32}{25} \right)^{K+1} \right) \right) = 2^{\Omega(p \wedge K)}.$$

■

4.3. Description of the hard MDP class

Given a large enough horizon H and a large enough dimension d , in this section we construct a class of featurized MDPs with horizon H and feature-space dimension d , such that (i) each featurized MDP in the class corresponds to an abstract game with parameters (K, p) such that $H \approx Kp$, $A = p \approx d^{1/4} \wedge H^{1/2}$ (ii) each MDP M_{w^\star} is associated with some abstract game $w^\star \in W^\star \subset W = \{-1, 1\}^p$; (iii) the feature-maps associated with the MDPs do not depend on w^\star ; (iv) the respective realizability assumptions are satisfied by the featurized MDPs in the class; (v) a planner that is guaranteed to achieve a high value in the MDPs can be used to achieve high values in the associated abstract game, which also means that (vi) for every $w^\star \in W^\star$, one should be able to emulate the queries in the featurized MDP associated with w^\star using queries that are available in the abstract game with w^\star , while the MDP planner should not get any information about w^\star by any other means than through these queries.

In the abstract game, at the end the planner needs to choose a sequence $(w_i)_{i \in [8]} \in W^{\circ 8}$. This will correspond to the first $8p$ steps of the path that the MDP planner traverses in the MDP, which will have deterministic dynamics. To guarantee that the number of actions is small, choosing such a weight sequence will be implemented in the MDP by first choosing w_1 in p steps, then choosing w_2 in another p steps, etc. In each of the p steps of these rounds, choosing an action $a \in [p]$ will allow the MDP planner to flip component a of the weight associated with the round. In particular, in the first p steps, the components of w_1 are chosen this way, starting from the weight vector $w_0 = \mathbf{1}$. In the next p steps, the components of w_2 are chosen this way, but this time starting with w_1 . The process is identical for choosing w_k based on w_{k-1} , where we let $1 \leq k \leq K$ go up to K to support arbitrary queries in the abstract game. To guarantee that the path chosen is in $\cup_k W^{\circ k}$, further rules are necessary. In particular, since we need to guarantee that w_k differs from w_{k-1} by at least $p/4$ positions, the dynamics is chosen so that in the first $\lceil p/4 \rceil$ steps within the k th round, if an action is repeated then it is called illegal, and leads to the end-state \perp , while in the remaining $p - \lceil p/4 \rceil \approx 3p/4$ steps an action repeat is called legal and leads to a “frozen” weight, i.e., starting

from the first such repeated action the weight associated with the path cannot be changed until the round is over. These rules guarantee that if a path of length kp does not end up in \perp , the path uniquely determines an element of $W^{\circ k}$ (in fact, the last state alone uniquely determines such an element). We associate with every action sequence subject to the constraints just described a unique state, which can be seen as a node on the action tree. We will say that a state $s \neq \perp$ belongs to some round $k \in [0 : K - 1]$, if the length l of the associated action sequence $(a_i)_{i \leq l}$ satisfies $kp \leq l < (k + 1)p$. We say that the state is in step i of round k if also $l = kp + i$.

Normally, the transitions of the MDP follow the path in the action tree just described, and the rewards are zero. However, there are two exceptions that depend on w^\star . To describe them, note that a state $s \neq \perp$ that is in step $p - 1$ of some round k is one step away from finalizing the choice of weight vector w_{k+1} . Indeed, such a state, together with the action performed in that state, defines the weight sequence $(w_i)_{i \in [k+1]} \in W^{\circ k+1}$, while a shorter sequence $(w_i)_{i \in [k]} \in W^{\circ k}$ is defined by all states $s \neq \perp$ that are in any step i of some round k .

For a state that is in some step i of some round k , the aforementioned exceptions to the MDP dynamics are: (i) if $k > 0$ and $\text{diff}(w_k, w^\star) < p/4$; (ii) else if $i = p - 1$, and either $k = K - 1$ (last step of episode), or $\text{diff}(w_{k+1}, w^\star) < p/4$. In case (i), the next state is \perp , and the reward is deterministically set to $\langle \varphi, \theta^\star \rangle$, where φ is the feature-vector associated with the state or the state-action pair (depending on which class of featurized MDPs are considered), and θ^\star is a hidden weight vector corresponding to w^\star . In case (ii), a Bernoulli reward with parameter $f_{w^\star}((w_i)_{i \in [k+1]})$ is generated, while also transitioning to \perp . Note that the states associated with case (i) are unreachable from the initial state as any path to such state goes through a state that satisfies (ii). While there is much information to be gained from any query where the state is of this type, planners with local access can never issue such queries, while planners with global access still have very little chance of encountering such a state (the proportion of these states is exponentially small as can be seen from, e.g., the result of Lemma 4.4). We refer the reader to Figure 1 for an illustration of the MDP dynamics and the associated reward structure, and to Eq. 23 for a more precise definition.

The next step is to show that one can define appropriate feature-maps such that the respective realizability conditions hold, which also means that we will need to compute the optimal value (or action-value) functions and then we will also need to show that a sound MDP planner for the appropriate class of MDPs can be used to derive a sound planner for the abstract game.

Here, the main idea is that the optimal value corresponding to a state s that is in some step $0 \leq i \leq p - 1$ of round $0 \leq k \leq K - 1$ takes the form

$$v^\star(s) = f_{w^\star}((w_i)_{i \in [k]}) = g(\text{diff}(w_0, w_1)) \dots g(\text{diff}(w_{k-1}, w_k))g(\text{diff}(w_k, w^\star)),$$

where $w_0 = \mathbf{1}$ by convention, w_i for $1 \leq j < k$ is the weight vector for the corresponding round, while w_k is obtained by performing the component manipulations on w_{k-1} prescribed by the action in round k until step i , after which, the weight obtained is moved as much as possible towards w^\star . Note that w_k here depends on both s and w^\star , while the other weight vectors only depend on s . In fact, one can write $w_k = A(s)w^\star + b(s)$ for some matrix $A(s)$ and vector $b(s)$ that depend on s . Therefore,

$$v^\star(s) = h(s)g(\text{diff}(w_{k-1}(s), A(s)w^\star + b(s)))g(\text{diff}(A(s)w^\star + b(s), w^\star)) \quad (20)$$

where $h(s) = g(\text{diff}(w_0, w_1)) \dots g(\text{diff}(w_{k-2}, w_{k-1}))$ is a scalar that depends only on s . The expression in Eq. 20 is a fourth-order expression of w^\star since g is a quadratic function, and while

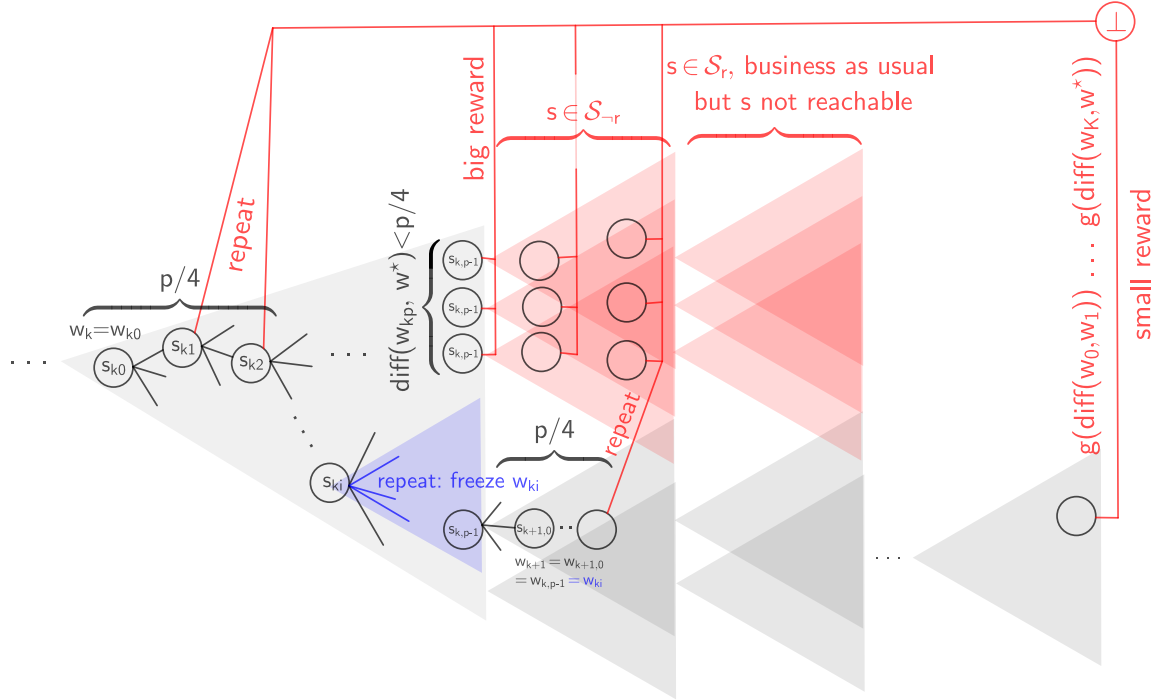


Figure 1: Illustration of an MDP associated with a weight vector w^* . The nodes represent states, which are members of the action tree. Subtrees are illustrated with triangles. Edges represent actions, red edges transit to the episode-over state \perp . Unless the action was illegal, there are next-states that an MDP with some other w^* would have transitioned to. These states still exist in M , but are unreachable, and illustrated with a red triangle. The blue triangle represents a part of the action tree where a legal repeated action freezes the weight corresponding the round. Unless written on the edge, there is no reward for the action. In the figure, $(s_{ki})_{ki}$ represents a path through the state space, while for k, i fixed, w_{ki} represents the weight vector of step i of round k .

diff is (affine) bilinear in its two arguments and so it appears that $\text{diff}(A(s)w^* + b(s), w^*)$ could be quadratic itself, due to the special structure, this expression is still linear in w^* . As such, $v^*(s)$ is (roughly⁷) a linear function of $(w^*)^{\otimes 4}$, the fourth-order tensor product of this vector with itself, which gives rise to the definition of φ_v and θ^* , which is (roughly) the flattening of $(w^*)^{\otimes 4}$. Of course, it remains to verify that $\varphi_v(s)$ and θ^* have small norms as required and also that this definition extends to states that are not reachable from the initial state (to prove the result with global accessibility). In fact, it is exactly this second requirement that made us define the deterministic rewards of $\langle \varphi_v(s), \theta^* \rangle$ and the associated transitions to \perp . (In this case it will be necessary to show that this reward is indeed in the $[0, 1]$ interval.)

A similar argument can be used for q^* realizability, and also for v^*/q^* reachable realizability (in which case the reward at unreachable states could be arbitrary). To finish, one needs to show that a sound MDP planner can be used to implement a sound abstract planner. For this, note that the steps that an MDP planner makes in the first 8 rounds of an episode can be directly translated into an admissible weight sequence of length 8. Further, by construction, the value achieved with this weight sequence is at least as high as the value that the MDP planner would achieve by completing

7. The precise argument will also include lower-order tensor products.

the episode (the function f_{w^\star} and the MDP are such that cutting short a weight sequence obtained from a path in the MDP increases the value of the sequence).

In the remainder of this section, we fill in the gaps of this argument.

4.4. The MDP construction

We start with defining A , p and K as a function of the horizon $H \geq 81$ and dimension $d \geq 31$:

$$\begin{aligned} A = p &= \min \left(\max \{x \in \mathbb{N}_+ : x^4 + x^3 + x^2 + x + 1 \leq d\}, \left\lfloor H^{1/2} \right\rfloor \right), \\ K &= \lfloor H/p \rfloor, \\ H' &= Kp. \end{aligned} \tag{21}$$

By our definition of H -horizon MDPs, any H' -horizon MDP for $H' \leq H$ is also a H -horizon MDP (cf. Assumption 3.1). Hence, we shall construct a H' -horizon MDP with H' defined above. For future reference, it will be useful to note that

$$\begin{aligned} A(=p) &= \Theta \left(H^{1/2} \wedge d^{1/4} \right), & K &= \Theta(H^{1/2} \vee H/(d^{1/4})), \\ p &\geq 2 & \text{and} & K \geq 9. \end{aligned} \tag{22}$$

Similarly to the abstract game, we fix some $w^\star \in W^\star$ (Eq. 8). In what follows, we define two MDPs $M_{w^\star}^v = (\mathcal{S}, \mathcal{A}, Q_{w^\star}^v)$ and $M_{w^\star}^q = (\mathcal{S}, \mathcal{A}, Q_{w^\star}^q)$.

The state and action spaces are the same for all these MDPs. The superscript v and q indicates which realizability setting the MDP is tailored for. Together with the indices, we drop them and just use M and Q to minimize clutter. The difference between $M_{w^\star}^v$ and $M_{w^\star}^q$ is minuscule (see Case 23a). As noted beforehand, $\mathcal{A} = [p]$.

Apart from \perp , states in \mathcal{S} are uniquely identifiable with an action sequence of length at most $Kp-1$. Of all action sequences, we need to remove any action sequence that has a ‘‘repeated’’ action in the critical first $\lceil p/4 \rceil$ steps of any round. For $k \geq 0$, let $U_k \subset \mathcal{A}^k$ be those sequences of in \mathcal{A}^k which do not have any repeated elements. Then, letting $r = \lceil p/4 \rceil$, $V = (\bigcup_{i \in [r]} U_i) \cup (\bigcup_{i \in [p-r]} U_r \times \mathcal{A}^i)$, we define

$$\mathcal{S} = \{\perp, ()\} \cup \bigcup_{0 \leq k \leq K-1} (U_r \times \mathcal{A}^{p-r})^k \times V,$$

where $()$ denotes the empty sequence. The elements of \mathcal{S} (other than \perp) can thus be uniquely identified with a sequence of actions $(a_{00}, \dots, a_{0,p-1}, \dots, a_{k0}, \dots, a_{ki})$ with $0 \leq k \leq K-1$ and $0 \leq i \leq p-1$, where the double indexing emphasizes that the steps are grouped into rounds of length p , and commas between indices are often dropped to minimize clutter. For convenience, we let $\langle k, i \rangle = \{(n, m) : n \in [0 : K-1], m \in [0 : p-1], np + m < kp + i\}$ denote the index set in this double indexing, so that we can write $(a_{nm})_{(n,m) \in \langle k, i \rangle}$ for the above action sequence. Here, we can think of a_{nm} as the action performed in step m of round n .

As described beforehand, we associate a ‘‘weight’’, an element of W , to each state $s \neq \perp$ that corresponds to all the ‘‘flips’’ described by the action sequence for s . Let $w : \mathcal{S} \rightarrow W$ be the corresponding map, where we let $w(\perp) = \mathbf{1}$. We will also find it useful to introduce $w : \mathcal{S} \times \mathcal{A} \rightarrow W$, where for $(s, a) \in \mathcal{S} \times \mathcal{A}$, $w(s, a)$ is the weight sequence where component a of the last weight vector of $w(s)$ is flipped, except when s is a frozen state or $s = \perp$, in which case $w(s, a) = w(s)$ (s is

a frozen state when there is a legal repeated action in the actions that correspond to the current round of the state).

In what follows, we will often find it useful to fix a path, i.e., a complete action sequence of the form $(a_{ki})_{(k,i) \in [0:K-1] \times [0:p-1]} \in \mathcal{A}^{Kp}$. Note that here we allow all action sequences. We then describe the behavior of the MDP in terms of its transitions and rewards encountered during this fixed action sequence. Notationally, we refer to the state (deterministically) reached in round k , step i for the fixed action sequence as s_{ki} . This means that $s_{00} = ()$, and for $0 \leq i \leq p-1$, $s_{k,i+1} = \gamma_{w^*}(s_{ki}, a_{ki})$ and for $0 \leq k \leq K-1$, $s_{k+1,0} = s_{kp}$, where γ_{w^*} is the transition function of the MDP. Note that the state sequence has extra elements, to help with the notation. In particular, $s_{K0} = s_{K-1,p} = \perp$. By a slight abuse of notation, for the fixed action sequence, we also let $w_{ki} = w(s_{k-1,i}, a_{k-1,i})$ (if $s_{ki} \neq \perp$, $w_{ki} = w(s_{ki})$). To disambiguate, the notation w_{ki} always uses two indices for w , while the notation in the abstract game always uses one. To match the weight values of the MDP with those of the abstract game, we introduce the shorthand $w_k = w_{k0}$. To complete the definition of w_{ki} , we define $w_{00} = \mathbf{1}$ (similarly to the abstract game's definition of $w_0 = \mathbf{1}$). We will also find it useful to introduce the function $w_{\text{last}} : \mathcal{S} \rightarrow W$ which to a given state $s = s_{ki} \neq \perp$ at step i or round k assigns the ‘‘last complete weight’’ $w_k = w_{k0}$ while $w_{\text{last}}(\perp) = \mathbf{1}$.

The (k, i) -indexed notation, such as s_{ki} and w_{ki} (along with other similarly indexed quantities introduced later) is designed to avoid clutter by hiding the implicit dependence on the action sequence, which is assumed to be fixed whenever we use such notations. The action sequence that is fixed should always be clear from the context. Whenever we state a result concerning these symbols, the result is meant to hold for an arbitrary action sequence.

For a state $s \in \mathcal{S}$, $s \neq \perp$ that is in step i of round k , and an action $a \in [A]$, the transition and reward of taking action a in state s leads to the following reward-next state pair (R', S') (which specifies the kernel Q of the MDP):

$$(R', S') = \begin{cases} (\langle \varphi, \theta^* \rangle, \perp), & \text{if } k > 0 \text{ and } \text{diff}(w_k, w^*) < p/4 & (23a) \\ (Z, \perp), & \text{else if } i = p-1, \text{diff}(w_{k+1}, w^*) < p/4 & (23b) \\ (Z, \perp), & \text{else if } k = K-1, i = p-1 \text{ (last step)} & (23c) \\ (0, s_{k,i+1}), & \text{otherwise,} & (23d) \end{cases}$$

Here, the symbols not yet introduced beforehand are defined as follows: (i) $(w_{k'})_{k' \in [k+\mathbb{I}\{i=p-1\}]}$ is the sequence of round-start weights $(w_{k'}, 0)_{k' \in [k+\mathbb{I}\{i=p-1\}]}$ that correspond to state s and action a . If $i = p-1$, this sequence also includes the newly ‘‘compiled’’ weight $w_{k+1,0} = w(s, a)$. (ii) Z has distribution $\text{Ber}(f_{w^*}((w_{k'})_{k' \in [k+\mathbb{I}\{i=p-1\}]})$. (iii) θ^* will be defined in Eq. 31. (iv) for feature-maps φ_v and φ_q (defined in Eqs. 32, 37), $\varphi = \varphi_v(s_{ki})$ if we are in the v^* -realizable setting (MDP $M_{w^*}^v$) and $\varphi = \varphi_q(s_{ki}, a)$ otherwise. In either case, the reward in Case 23a is in $[0, 1]$ by Eq. 35 and Eq. 39.

Later in the proof, the following lemma will be useful to convert a sound planner for the MDP into a sound planner for the abstract game:

Lemma 4.6. *We can simulate an outcome of (R', S') in the MDP using at most one query to the abstract game, if the length, dimensionality, and secret parameters of the game are K , p , and w^* , respectively.*

Proof. For $k = 0, i < p - 1$, we fall under Case 23d and no query to the abstract game is required. Otherwise, let $l = k + \mathbb{I}\{i = p - 1\} > 0$, and query the abstract game with $(l, (w_{k'})_{k' \in [l]})$. This is a valid query as $(w_{k'})_{k' \in [l]} \in W^{\circ k}$. The result to this query allows to determine which case the transition falls under, and it also contains Z (with the required distribution) when the case calls for it. \blacksquare

As alluded to before, Case 23a is somewhat pathological: the transitions are such that if at the end of round k the newly “compiled” weight $w_{k+1,0}$ is close to w^\star ($\text{diff}(w_{k+1,0}, w^\star) < p/4$) then the next state is \perp . This means that by following the transitions, it is impossible to arrive at a state $s \in \mathcal{S}$, where Case 23a would apply.

Lemma 4.7 (Case 23a is unreachable in M). *In MDP M , for all $s \in \mathcal{S}_r$ and $s' \in \mathcal{S}_{-r}$,*

$$s' \notin \text{Reach}_M(s)$$

where

$$\mathcal{S}_{-r} = \{s \in \mathcal{S} : s \neq \perp \text{ and } \text{diff}(w_{\text{last}}(s), w^\star) < p/4\}, \quad \mathcal{S}_r = \mathcal{S} \setminus \mathcal{S}_{-r}. \quad (24)$$

We will find some further notation useful to describe essential properties of the MDP states. Take any path in the MDP and the corresponding states (s_{ki}) . Pick k and i such that $s_{ki} \neq \perp$. Let the “bit mask” $\text{fix}_{ki} \in \{0, 1\}^p$ indicate for each component of w_{ki} whether it is fixed (1) in round k at step i or not (0). Recall that a component is fixed if either the corresponding action is performed in round k before step i , or there was a legal repeated action, in which case all the components are frozen. Let $\text{ct}_{ki}^{\text{flip}}$ be the number of components flipped in round k by step i . Because each component can only be flipped at most once in a round, this satisfies

$$\text{ct}_{ki}^{\text{flip}} = \text{diff}(w_{k0}, w_{ki}).$$

Let e_{ki}^{fix} (and $e_{ki}^{-\text{fix}}$) be the number of components that are fixed (and not fixed, respectively) at step i and have the opposite sign of the respective components of w^\star . These are “error counts”. (As opposed to $\text{ct}_{ki}^{\text{flip}}$ and fix_{ki} , the error counts obviously depend on w^\star). Let the operator $\cdot : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ return the componentwise product of its inputs. For $i \in \{0, 1\}$, let $\neg i = 1 - i$, which is also extended to binary-valued vectors in a componentwise manner. The definitions imply the following identities:

$$e_{ki}^{\text{fix}} = \frac{1}{2} (\langle \mathbf{1}, \text{fix}_{ki} \rangle - \langle \text{fix}_{ki} \cdot w_{ki}, w^\star \rangle), \quad (25)$$

$$e_{ki}^{-\text{fix}} = \frac{1}{2} (\langle \mathbf{1}, \neg \text{fix}_{ki} \rangle - \langle \neg \text{fix}_{ki} \cdot w_{ki}, w^\star \rangle). \quad (26)$$

Consider the case when $\text{fix}_{ki} \neq \mathbf{1}$. Thanks to $s_{ki} \neq \perp$, the first i actions of round k are unique. Therefore, in this case, $\text{ct}_{ki}^{\text{flip}} = i$. Furthermore, each unique action adds 1 to $\langle \mathbf{1}, \text{fix}_{ki} \rangle$, thus $e_{ki}^{\text{fix}} \leq \langle \mathbf{1}, \text{fix}_{ki} \rangle = i = \text{ct}_{ki}^{\text{flip}}$. Similarly, $e_{ki}^{-\text{fix}} \leq \langle \mathbf{1}, \neg \text{fix}_{ki} \rangle = p - i = p - \text{ct}_{ki}^{\text{flip}}$. If on the other hand, $\text{fix}_{ki} = \mathbf{1}$, then $e_{ki}^{-\text{fix}} = 0$. This leads to the following result, which will be useful for our calculations:

Lemma 4.8. *Assuming $s_{ki} \neq \perp$, $e_{ki}^{-\text{fix}} \leq p - \text{ct}_{ki}^{\text{flip}}$, and $e_{ki}^{\text{fix}} \leq p - i$. Furthermore, if $\text{fix}_{ki} \neq \mathbf{1}$, then the following also hold: $\text{ct}_{ki}^{\text{flip}} = i = \langle \mathbf{1}, \text{fix}_{ki} \rangle$, and $e_{ki}^{\text{fix}} \leq \text{ct}_{ki}^{\text{flip}}$.*

4.5. Defining a policy and calculating its value function

We now define a deterministic policy $\pi_{w^\star} : \mathcal{S} \rightarrow [A]$, which later will be shown to be the optimal policy. The purpose of the current section is merely to compute the value function of this policy. The policy is defined as follows: Let $s_{ki} \in \mathcal{S}_r$ be a state along step i of round k and assume that $s_{ki} \neq \perp$. Then π_{w^\star} greedily flips all the components of w_{ki} that have the wrong sign and are not fixed yet. Once this is done, π_{w^\star} freezes the round by repeating an action. Ties are resolved in a systematic fashion.

More formally, let \mathcal{A}_1 be the set of actions where the component of w_{ki} has not been fixed yet and where w_{ki} disagrees in sign with w^\star ; let \mathcal{A}_2 be the set of actions where the component has been fixed:

$$\begin{aligned} \mathcal{A}_1 &= \{a \in [A] : (\text{fix}_{ki})_a = 0 \text{ and } (w_{ki})_a \neq w_a^\star\} \\ \mathcal{A}_2 &= \{a \in [A] : (\text{fix}_{ki})_a = 1\} \end{aligned} \quad (27)$$

Then,

$$\pi_{w^\star}(s_{ki}, \cdot) = \begin{cases} 1, & \text{if } s_{ki} = \perp; & (28a) \\ \arg \max_{a \in [p]} \langle \varphi_q(s_{ki}, a), \theta^\star \rangle, & \text{else if } \text{diff}(w_{k0}, w^\star) > p/4; & (28b) \\ \min \mathcal{A}_1, & \text{else if } |\mathcal{A}_1| = e_{ki}^{-\text{fix}} > 0; & (28c) \\ \min \mathcal{A}_2, & \text{else if } |\mathcal{A}_2| = \text{ct}_{ki}^{\text{flip}} > 0, & (28d) \end{cases}$$

where φ_q is the state-action feature-map defined in Eq. 37, and θ^\star is defined in Eq 31. Note that $s_{ki} \in \mathcal{S}_r$ and $s_{ki} \neq \perp$ implies that either Case 28c or 28d must apply.

With this, the promised result of the section is as follows.

Lemma 4.9. *Assuming $s_{ki} \in \mathcal{S}_r$ and $s_{ki} \neq \perp$, we have*

$$v^{\pi_{w^\star}}(s_{ki}) = \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}}) g(e_{ki}^{\text{fix}}).$$

The high level argument underlying this lemma is that the policy reaches the end state \perp , either after reaching the last step of the current, or the next round. In either cases, the only reward incurred from the current state to the end is when the transition to the end state happens. The definition of this reward can then be invoked to show the result. The detailed proof is as follows:

Proof. Starting from round k step i and letting \mathcal{A}_1 be as in Eq. 27, the policy π_{w^\star} flips all the components in \mathcal{A}_1 (that have the wrong sign and are not fixed yet). We note that $\mathcal{A}_1 = \{\}$ if there was a repeated action in this round (which freezes the components). In this case, $e_{ki}^{-\text{fix}} = 0$ and $s_{ki} \neq \perp$ implies the repeated action was legal, i.e., $i = i + e_{ki}^{-\text{fix}} \geq \lceil p/4 \rceil$, and therefore w_{ki} is frozen, thus regardless of π_{w^\star} , $w_{k+1,0} = w_{ki}$, so $\text{diff}(w_{k0}, w_{k+1,0}) = \text{ct}_{ki}^{\text{flip}} = \text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}}$.

Otherwise, by definition the first $i + |\mathcal{A}_1| = i + e_{ki}^{-\text{fix}} = \text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}} \leq p$ actions in round k are unique (noting the inequality comes from Lemma 4.8). Furthermore, in this case observe that all components where w_{k0} differs in sign from w^\star are flipped in round k by step $i + e_{ki}^{-\text{fix}}$: either because it was flipped in the first i steps (and thus setting the relevant component of fix_{ki} to 1), or because the action corresponding to the component is in \mathcal{A}_1 , and thus flipped by π_{w^\star} .

Therefore $i + e_{ki}^{-\text{fix}} \geq \text{diff}(w_{k0}, w^\star)$. As $\perp \neq s_{ki} \in \mathcal{S}_r$, $\text{diff}(w_{k0}, w^\star) \geq p/4$. As $i + e_{ki}^{-\text{fix}}$ is an integer, $i + e_{ki}^{-\text{fix}} \geq \lceil p/4 \rceil$. At step $i + e_{ki}^{-\text{fix}} \geq \lceil p/4 \rceil$, all the actions in \mathcal{A}_1 are exhausted, and if there are any remaining steps in the round, π_{w^\star} freezes the round by repeating an action (Case 28d). This is a legal action as $i + e_{ki}^{-\text{fix}} \geq \lceil p/4 \rceil$. Therefore $w_{k+1,0} = w_{k,i+e_{ki}^{-\text{fix}}}$.

Regardless of whether w_{ki} is fixed at step i , the number of components that have the wrong sign that are not flipped in round k is exactly e_{ki}^{fix} , and therefore

$$\begin{aligned} \text{diff}(w_{k0}, w_{k+1,0}) &= \text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}} \\ \text{diff}(w_{k+1,0}, w^\star) &= e_{ki}^{\text{fix}} \end{aligned}$$

At the end of round k , at step $p - 1$, either Case 23b or 23c applies and the expectation of the reward is

$$\begin{aligned} f_{w^\star}((w_{k'0})_{k' \in [k+1]}) &= \left(\prod_{k' \in [k+1]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{diff}(w_{k+1,0}, w^\star)) \\ &= \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}}) g(e_{ki}^{\text{fix}}), \end{aligned}$$

or Case 23d applies and the episode continues with round $k + 1$. In this latter case, $\text{fix}_{k+1,0} = \mathbf{0}$, $\text{ct}_{k+1,0}^{\text{flip}} = 0$, $e_{k+1,0}^{-\text{fix}} = \text{diff}(w_{k+1,0}, w^\star) = e_{ki}^{\text{fix}}$, and so in round $k + 1$, π_{w^\star} sets all the remaining components to match w^\star , i.e., $w_{k+2,0} = w^\star$. The transition at the end of round $k + 1$, at step $p - 1$, then falls either under Case 23b or 23c, and the expectation of the reward is the same as before as $g(0) = 1$:

$$\begin{aligned} f_{w^\star}((w_{k'0})_{k' \in [k+2]}) &= \left(\prod_{k' \in [k+2]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{diff}(w_{k+2,0}, w^\star)) \\ &= \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}}) g(e_{ki}^{\text{fix}}) g(0), \end{aligned}$$

As in MDP M any transition with a positive reward expectation transitions to state \perp , the value of π_{w^\star} , the expected sum of rewards along the episode, reduces to the expectation of this single reward in the episode. ■

4.6. Showing that π_{w^\star} is an optimal policy

We start with a lemma that will be used to optimize the attainable reward, given the constraints of the MDP.

Lemma 4.10. *For $p \geq 2$, $l \geq 2$ integer, let $(x_j)_{j \in [l]}$ be integers $0 \leq x_j \leq p$, and let $0 \leq c_1, c_2, c_3 \leq p$ be further integers such that the following all hold:*

- $c_2 \leq c_1$ or $c_3 = 0$;

- $c_1 + c_3 \leq p$;
- $c_1 + c_2 + c_3 \leq \sum_{j \in [l]} x_j$;
- $c_2 \leq \sum_{j \in [2:l]} x_j$;
- $c_1 \leq x_1$.

Then,

$$\prod_{j \in [l]} g(x_j) \leq g(c_1 + c_3)g(c_2).$$

Proof. Note that $g(x) > 0$ and decreases monotonically for $x \in [0, p]$. First we prove for integers $x \geq y$ such that $1 \leq x, y \leq p - 1$, it holds that

$$g(x)g(y) \leq g(x+1)g(y-1). \quad (29)$$

Note that $g(x)g(y) - g(x+1)g(y-1) = -\frac{x-y+1}{2p^4} (p(x-2) + y(p-x) + x)$, and as $x \geq y$, it only remains to prove that $p(x-2) + y(p-x) + x \geq 0$. If $x = 1$ then $y = 1$ and the above holds with equality. Otherwise $x \geq 2$ and all terms are non-negative, finishing the proof of Eq. 29.

We now claim that for any $0 \leq y \leq x \leq p$ integers, $g(x)g(y) \leq g((x+y) \wedge p)$. Since over $[0, p]$, g takes values in $[0, 1]$, this clearly holds when either $y = 0$ or when $x = p$. Furthermore, if $1 \leq y \leq x \leq p-1$, then from Eq. 29 it follows that $g(x)g(y) \leq g(x+1)g(y-1) \leq g(x+2)g(y-2) \leq g((x+y) \wedge p)g((x+y-p) \vee 0) \leq g((x+y) \wedge p)$ where the last inequality follows again because $g(u) \in [0, 1]$ when $u \in [0, p]$.

Now, $g(x_2)g(x_3)g(x_4) \leq g((x_2+x_3) \wedge p)g(x_4) \leq g(((x_2+x_3) \wedge p) + x_4) \wedge p) = g((x_2+x_3+x_4) \wedge p)$. Continuing this way, letting $x_{\geq 2} = \sum_{j \in [2:l]} x_j$, we get

$$\prod_{j \in [2:l]} g(x_j) \leq g(x_{\geq 2} \wedge p).$$

Thus, $\prod_{j \in [l]} g(x_j) \leq g(x_1)g(x_{\geq 2} \wedge p)$.

Consider first the case when $c_3 = 0$. Then, by monotonicity of g , as $x_1 \geq c_1 = c_1 + c_3$ and $c_2 \leq x_{\geq 2}$, $\prod_{j \in [l]} g(x_j) \leq g(c_1 + c_3)g(c_2)$ and we are done.

Now, if $c_3 > 0$, by assumption $c_2 \leq c_1$. In this case, $c_1 + c_2 + c_3 - (x_1 \wedge (c_1 + c_3)) \leq x_{\geq 2} \wedge p$, as (1) $c_1 \leq (x_1 \wedge (c_1 + c_3))$ and thus $c_1 + c_2 + c_3 - (x_1 \wedge (c_1 + c_3)) \leq c_2 + c_3 \leq c_1 + c_3 \leq p$, while (2) by our assumptions, $x_{\geq 2} \geq c_2$ and $x_1 + x_{\geq 2} \geq c_1 + c_2 + c_3$, and therefore $(x_1 \wedge (c_1 + c_3)) + x_{\geq 2} \geq c_1 + c_2 + c_3$. By the monotonicity of g , we can then conclude that

$$\prod_{j \in [l]} g(x_j) \leq g(x_1 \wedge (c_1 + c_3))g(c_1 + c_2 + c_3 - (x_1 \wedge (c_1 + c_3))).$$

Let x'_1 and x'_2 be the above arguments of g in decreasing order, i.e., $x'_1 = (x_1 \wedge (c_1 + c_3)) \vee (c_1 + c_2 + c_3 - (x_1 \wedge (c_1 + c_3)))$ and $x'_2 = (x_1 \wedge (c_1 + c_3)) \wedge (c_1 + c_2 + c_3 - (x_1 \wedge (c_1 + c_3)))$, so that we have $\prod_{j \in [l]} g(x_j) \leq g(x'_1)g(x'_2)$ with $x'_1 \leq c_1 + c_3$ and $x'_1 + x'_2 = c_1 + c_2 + c_3$. Applying Eq. 29 on this product $c_1 + c_3 - x'_1$ times, we get that

$$\prod_{j \in [l]} g(x_j) \leq g(x'_1)g(x'_2) \leq g(c_1 + c_3)g(c_2).$$

■

We now show that π_{w^\star} is an optimal policy by arguing that its value function matches the optimal value function.

Lemma 4.11 (π_{w^\star} is an optimal policy). *In MDP M ,*

$$\forall s \in \mathcal{S}, a \in [A], \quad v^{\pi_{w^\star}}(s) = v^\star(s).$$

Proof. For $s = \perp$, the claim holds by definition as $v^{\pi_{w^\star}}(\perp) = v^\star(\perp) = 0$. Otherwise, let $s = s_{ki}$ be a state along step i of round k . Let us first consider the case when $s_{ki} \in \mathcal{S}_{-r}$. For any action a performed, the transition will happen under Case 23a, and the deterministic reward given equals $q^\star(s_{ki}, a)$. If we are in the v^\star -realizable setting (for MDP $M_{w^\star}^v$), this reward does not depend on the action and therefore $v^{\pi_{w^\star}}(s) = v^\star(s)$ regardless of π_{w^\star} . Otherwise, π_{w^\star} chooses an action under Case 28b, which by definition maximizes the reward, so again $v^{\pi_{w^\star}}(s) = v^\star(s)$ in this case as well.

Let us turn to the case where $s_{ki} \in \mathcal{S}_r$. There is at most one reward with positive expectation in any round (or none, if an illegal action is taken). As no state in \mathcal{S}_{-r} is reachable from s_{ki} (by Lemma 4.7), this reward is collected at the end of some round $K' \in [0 : K - 1]$, at step $p - 1$, and has expectation

$$\begin{aligned} f_{w^\star}((w_{k'0})_{k' \in [K'+1]}) &= \left(\prod_{k' \in [K'+1]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{diff}(w_{K'+1,0}, w^\star)) \\ &= \prod_{k' \in [K+1]} g(\text{diff}(w_{k'-1,0}, w_{k',0})), \end{aligned}$$

where, for convenience, we let $w_{k'0} = w^\star$ for $k' \geq K' + 2$ (as $g(0) = 0$). This reward expectation is strictly positive (by Lemma 4.2), so the optimal policy will never take an illegal action.

At round k , $g(\text{diff}(w_{k'-1,0}, w_{k',0}))$ is fixed for $k' \in [k]$, and the policy can only influence the terms $g(\text{diff}(w_{k'-1,0}, w_{k',0}))$ for $k' \in [k + 1 : K + 1]$. We have by definition that $0 \leq \text{diff}(\cdot, \cdot) \leq p$. In any round, once a component is flipped it cannot be flipped back in the same round. This implies that

$$\text{diff}(w_{k0}, w_{k+1,0}) = \text{diff}(w_{k0}, w_{ki}) + \text{diff}(w_{ki}, w_{k+1,0}) \geq \text{diff}(w_{k0}, w_{ki}) = \text{ct}_{ki}^{\text{flip}}.$$

On top of this, $e_{ki}^{\text{fix}} + e_{ki}^{-\text{fix}}$ components differ in sign between w_{ki} and w^\star . By the triangle inequality, as $w_{K+1,0} = w^\star$, this implies that

$$\sum_{k' \in [k+1:K+1]} \text{diff}(w_{k'-1,0}, w_{k',0}) \geq \text{diff}(w_{k0}, w_{ki}) + \text{diff}(w_{ki}, w^\star) = \text{ct}_{ki}^{\text{flip}} + e_{ki}^{\text{fix}} + e_{ki}^{-\text{fix}}.$$

Finally, e_{ki}^{fix} of these have already been flipped in round k by step i . These cannot be flipped again in the same round k , so they need to be included in some future round, i.e., in $\text{diff}(w_{k'-1,0}, w_{k',0})$ for $k' \geq k + 2$:

$$\sum_{k' \in [k+2:K+1]} \text{diff}(w_{k'-1,0}, w_{k',0}) \geq \text{diff}(w_{k+1,1}, w^\star) \geq e_{ki}^{\text{fix}}.$$

By Lemma 4.8,

$$e_{ki}^{-\text{fix}} \leq p - \text{ct}_{ki}^{\text{flip}},$$

and either $\text{fix}_{ki} = \mathbf{1}$, implying $e_{ki}^{-\text{fix}} = 0$, or $e_{ki}^{\text{fix}} \leq \text{ct}_{ki}^{\text{flip}}$:

$$e_{ki}^{\text{fix}} \leq \text{ct}_{ki}^{\text{flip}} \quad \text{or} \quad e_{ki}^{-\text{fix}} = 0.$$

Therefore, we can apply Lemma 4.10 with $c_1 = \text{ct}_{ki}^{\text{flip}}$, $c_2 = e_{ki}^{\text{fix}}$, $c_3 = e_{ki}^{-\text{fix}}$ to optimize the parameters $(x_j)_{j \in [K-k+1]}$ where $x_j = \text{diff}(w_{j+k-1}, w_{j+k})$, to get that

$$\prod_{k' \in [k+1:K+1]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \leq g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}})g(e_{ki}^{\text{fix}}).$$

Therefore, the optimal policy's expected value (which equals the expectation of the only reward in the episode) is upper bounded as:

$$v^*(s_{ki}) \geq \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}})g(e_{ki}^{\text{fix}}) = v^{\pi_{w^*}}(s_{ki}),$$

by Lemma 4.9. Therefore $v^{\pi_{w^*}}(s_{ki}) = v^*(s_{ki})$. ■

4.7. Defining θ^* , φ_v , and φ_q , and showing realizability

For (column) vectors M_1, M_2, \dots , let us denote by $[M_1, M_2, \dots]$ their concatenation $(M_1^\top, M_2^\top, \dots)^\top$. Let $b(M)$ map a tensor of any rank m and any shape $d_1 \times d_2 \times \dots \times d_m$ to the vector of dimension $\prod_{i \in [m]} d_i$ by laying out its elements in a canonical order. Let \otimes denote the tensor product.

We will use the following result to linearize products of vectors:

Lemma 4.12. *For any positive integer n and any vectors a_1, a_2, \dots, a_n and b_1, b_2, \dots, b_n of equal dimension:*

$$\langle a_1, b_1 \rangle \langle a_2, b_2 \rangle \dots \langle a_n, b_n \rangle = \langle b(a_1 \otimes a_2 \otimes \dots \otimes a_n), b(b_1 \otimes b_2 \otimes \dots \otimes b_n) \rangle.$$

By Lemma 4.7, and because $M_{w^*}^v$ and $M_{w^*}^q$ have the same transitions and rewards for any state $s \in \mathcal{S}_r$, we do not notationally distinguish between $M_{w^*}^v$ and $M_{w^*}^q$ when describing the value or action-value functions of these MDPs on states $s \in \mathcal{S}_r$, as these are the same in the two MDPs.

We define the feature-map $\varphi_v : \mathcal{S} \rightarrow \mathcal{B}_d(1)$ and $\varphi_q : \mathcal{S} \times [A] \rightarrow \mathcal{B}_d(1)$. For state \perp , let $\varphi_v(\perp) = \mathbf{0}$ and for all actions $a \in [A]$, $\varphi_q(\perp, a) = \mathbf{0}$. Realizability immediately holds as $v^*(\perp) = q^*(\perp, a) = 0 = \langle \mathbf{0}, \theta^* \rangle$. For any state $s \in \mathcal{S}$, $s \neq \perp$, let $s = s_{ki}$ be a state along step i of round k . Let us introduce the function

$$v'(s_{ki}) = \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}})g(e_{ki}^{\text{fix}}). \quad (30)$$

By Lemmas 4.11 and 4.9, it holds that $v'(s_{ki}) = v^*(s_{ki})$ if $s_{ki} \in \mathcal{S}_r$. Observe that out of the terms above, only $e_{ki}^{-\text{fix}}$ and e_{ki}^{fix} depends on w^* , and this dependence is linear. In particular, recall that $\text{ct}_{ki}^{\text{flip}}$ depends only on the actions, and not on w^* . Combined with the fact that g is a second-order polynomial, $v'(s_{ki})$ is a fourth-order expression in w^* , which can thus be linearized in $1 + p + p^2 + p^3 + p^4 \leq d$ dimensions. Let $\bar{w}^* = w^* / \|w^*\|_2 = w^* / \sqrt{p}$, and

$$\theta^* = 63 \left[1, \bar{w}^*, (\bar{w}^*)^{\otimes 2}, (\bar{w}^*)^{\otimes 3}, (\bar{w}^*)^{\otimes 4}, \mathbf{0}^{d-(1+p+p^2+p^3+p^4)} \right], \quad (31)$$

where $\mathbf{0}^{d-(1+p+p^2+p^3+p^4)}$ is a vector of zeros of dimensionality $d - (1 + p + p^2 + p^3 + p^4)$, serving the purpose to pad the vector to exactly d dimensions, as required by the definition. As $\|\bar{w}^\star\|_2 = 1$, we have that

$$\|\theta^\star\|_2 \leq 63 \cdot 5 = 315 := B.$$

Finally, for $Z_{(0)}, Z_{(1)}, Z_{(2)}, Z_{(3)}, Z_{(4)}$ calculated in Appendix A.1, if we let

$$\varphi_v(s_{ki}) = \frac{1}{63} \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) \left[Z_{(0)}, Z_{(1)}, Z_{(2)}, Z_{(3)}, Z_{(4)}, \mathbf{0}^{d-(1+p+p^2+p^3+p^4)} \right], \quad (32)$$

then by Eq. 46,

$$\langle \varphi_v(s_{ki}), \theta^\star \rangle = \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}}) g(e_{ki}^{\text{fix}}) = v'(s_{ki}) \quad (33)$$

$$(34)$$

Eq. 32 completes the definition of φ_v , while Eq. 33 implies that

$$0 \leq \langle \varphi_v(s_{ki}), \theta^\star \rangle \leq 1, \quad (35)$$

as $v'(s_{ki})$ is a product of $g(\cdot) \in [0, 1]$ terms (as $\text{diff}(\cdot, \cdot) \in [0, p]$). Furthermore, combining this with $\| [Z_{(0)}, Z_{(1)}, Z_{(2)}, Z_{(3)}, Z_{(4)}] \|_2 \leq 63$ (by Eq. 46), we have that

$$\|\varphi_v(s)\|_2 \leq 1 \quad \text{for all } s \in \mathcal{S},$$

which ensures that $\varphi_v : \mathcal{S} \rightarrow \mathcal{B}_d(1)$. We stress that, as required, $\varphi_v(s_{ki})$ does not depend on w^\star .

To show v^\star -realizability with these features, i.e., that $v^\star(s_{ki}) = \langle \varphi_v(s_{ki}), \theta^\star \rangle$, we start by pointing out that if $s_{ki} \in \mathcal{S}_{-r}$ then this immediately holds:

Lemma 4.13. *For any state $s \in \mathcal{S}_{-r}$ and action $a \in [A]$, regardless of the values of $\varphi_v(s)$, $\varphi_q(s, a)$, and θ^\star , v^\star -realizability for $M_{w^\star}^v$ and q^\star -realizability for $M_{w^\star}^q$ immediately holds as the transition falls under Case 23a:*

$$\begin{aligned} v_{M_{w^\star}^v}^\star(s) &= \langle \varphi_v(s), \theta^\star \rangle \\ q_{M_{w^\star}^q}^\star(s, a) &= \langle \varphi_q(s, a), \theta^\star \rangle \end{aligned}$$

Otherwise $s_{ki} \in \mathcal{S}_r$, and v^\star -realizability follows from Eq. 33 by recalling that $v'(s_{ki}) = v^\star(s_{ki})$ in this case. We conclude the following lemma from this:

Lemma 4.14. *$M_{w^\star}^v$ is v^\star -realizable with features $\varphi_v : (M_{w^\star}^v, \varphi_v) \in \mathcal{M}_{B,d,H,A}^{v^\star} \cap \mathcal{M}^{\text{Pdet}}$.*

We move on to defining φ_q and showing q^\star -realizability for $M_{w^\star}^q$. For any state $s \in \mathcal{S}$, $s \neq \perp$ and action $a \in [A]$, let $s = s_{ki}$ be a state along step i of round k . Let $s_{k,i+1}^a$ denote the value taken by $s_{k,i+1}$ if $a_{ki} = a$, and similarly for $w_{k,i+1}^a$. For $i = p - 1$ only, let us introduce

$$q'(s_{k,p-1}, a) = \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{diff}(w_{k0}, w_{k+1,0}^a)) g(\text{diff}(w_{k+1,0}^a, w^\star)). \quad (36)$$

Let

$$c(s_{ki}, a) = \frac{1}{63} \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{diff}(w_{k0}, w_{k+1,0}^a)),$$

which is a scalar that does not depend on w^\star . The only remaining term in q' has a second-order dependence on w^\star . For $X_{(0)}, X_{(1)}, X_{(2)}$ calculated in Appendix A.2, we let

$$\varphi_q(s_{ki}, a) = \begin{cases} \varphi_v(s_{k,i+1}^a) & \text{else if } i < p; \\ c(s_{ki}, a) [X_{(0)}, X_{(1)}, X_{(2)}, \mathbf{0}^{d-(1+p+p^2)}] & \text{otherwise,} \end{cases} \quad (37a)$$

where $\mathbf{0}^{d-(1+p+p^2)}$ is a vector of zeros of dimensionality $d - (1 + p + p^2)$. Then by Eq. 49, for θ^\star set according to Eq. 31,

$$\langle \varphi_q(s_{k,p-1}, a), \theta^\star \rangle = q'(s_{k,p-1}, a). \quad (38)$$

Eq 37 completes the definition of φ_q , while Eq. 38 together with Eq. 35 implies that

$$0 \leq \langle \varphi_q(s_{ki}, a), \theta^\star \rangle \leq 1 \quad \text{for all } a \in [A], s_{ki} \in \mathcal{S}, s_{ki} \neq \perp, \quad (39)$$

as $q'(s_{k,p-1}, a)$ is a product of $g(\text{diff}(\cdot, \cdot)) \in [0, 1]$ terms. Furthermore, combining this with $\| [X_{(0)}, X_{(1)}, X_{(2)}] \|_2 \leq 8$ (by Eq. 49), we have that

$$\| \varphi_q(s, a) \|_2 \leq 1 \quad \text{for all } s, a \in \mathcal{S} \times [A],$$

which ensures that $\varphi_q : \mathcal{S} \times [A] \rightarrow \mathcal{B}_d(1)$, as required. Again we stress that $\varphi_v(s_{ki})$ does not depend on w^\star .

To show q^\star -realizability, we first consider the case when $s_{ki} \in \mathcal{S}_r$ and $i = p - 1$ i.e., $\varphi_q(s_{ki}, a)$ falls under Case 37b. In this case,

$$\begin{aligned} \langle \varphi_q(s_{k,p-1}, a), \theta^\star \rangle &= q'(s_{k,p-1}, a) \\ &= \left(\prod_{k' \in [k]} g(\text{diff}(w_{k'-1,0}, w_{k',0})) \right) g(\text{diff}(w_{k0}, w_{k+1,0}^a)) g(\text{diff}(w_{k+1,0}^a, w^\star)) \\ &= q^\star(s_{k,p-1}, a), \end{aligned}$$

where the first equality comes from Eq 38. The last equality holds by definition if the transition and reward follows Case 23b or 23c; otherwise under Case 23d, it holds since

$$q^\star(s_{k,p-1}, a) = v^\star(s_{k+1,0}^a) = q'(s_{k,p-1}, a),$$

where the second equality follows from Lemmas 4.9 and 4.11.

Turning to the case where $s_{ki} \in \mathcal{S}_r$ and $i < p - 1$, we note that $\varphi_q(s_{ki}, a)$ falls under Case 37a, while the transition and reward follows Case 23d. Therefore

$$q^\star(s_{ki}, a) = v^\star(s_{k,i+1}^a) = \langle \varphi_v(s_{k,i+1}^a), \theta^\star \rangle = \langle \varphi_q(s_{k,i+1}, a), \theta^\star \rangle,$$

where the second equality follows from Lemma 4.14.

Together with Lemma 4.13 that proves q^\star -realizability for the case of $s_{ki} \in \mathcal{S}_{-r}$, we conclude that the following holds:

Lemma 4.15. $M_{w^*}^q$ is q^* -realizable with features φ_q : $(M_{w^*}^q, \varphi_q) \in \mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{Pdet}}$.

Recall that $\text{Reach}(s_{00}) \subseteq \mathcal{S}_r$ under either MDP $M_{w^*}^v$ or $M_{w^*}^q$ (by Lemma 4.7), and that value and action-value functions on such states take the same value for the two MDPs. Then, combining Lemmas 4.14 and 4.15, we have the following result:

Lemma 4.16. $M_{w^*}^v$ is reachable- v^*/q^* -realizable with features φ_v and φ_q : $(M_{w^*}^v, \varphi_v, \varphi_q) \in \mathcal{M}_{B,d,H,A}^{v^*/q^*\text{reach}} \cap \mathcal{M}^{\text{Pdet}}$.

4.8. Reduction to planning in the abstract game

Proof of Theorem 1.1. Let $\delta \geq 0.01$, $B \geq 315$, $d \geq 31$, $H \geq 81$. In what follows, we prove the theorem only for A (and p) set according to Eq. 21. This is sufficient to prove the result for $A \geq \lfloor \sqrt{H} \rfloor \wedge 0.8d^{14} \geq p$ (Eq. 21) as soundness with a lower action count cannot be harder to achieve, since it is always possible to duplicate some actions without changing the difficulty of the problem.

Let \mathcal{P} be any δ -sound planner with worst-case query cost \bar{N} for some class class $\mathcal{M} \cap \mathcal{M}^{\text{Pdet}}$, where

$$\mathcal{M} \in \{ \mathcal{M}_{B,d,H,A}^{v^*}, \mathcal{M}_{B,d,H,A}^{q^*}, \mathcal{M}_{B,d,H,A}^{v^*/q^*\text{reach}} \}.$$

We show that \mathcal{P} gives rise to a sound abstract planner for the abstract game of Section 4.2 and therefore, by Theorem 4.1, it must use exponentially many queries.

Lemmas 4.14, 4.15, and 4.16 show that MDPs $(M_{w^*}^v)_{w^* \in W^*}$, $(M_{w^*}^q)_{w^* \in W^*}$, and $(M_{w^*}^v)_{w^* \in W^*}$ respectively, together with feature-maps φ_v and φ_q , belong to these classes. Therefore, the δ -sound planner \mathcal{P} satisfies, for any MDP M with parameter w^* in its class:

$$v_M^{\pi_M}(s_{00}) \geq v_{w^*}^*(s_{00}) - 0.01,$$

where s_{00} is the initial state in M and π_M is the policy induced by the interconnection of \mathcal{P} and MDP M . Let \mathbb{P} and \mathbb{E} be the probability measure and expectation, respectively, induced by this interconnection (as defined in Section 3.3). Then,

$$\begin{aligned} v_M^{\pi_M}(s_{00}) &= \mathbb{E} \left[\sum_{t=1}^H R_t \mid S_0 = s_{00} \right] \geq v^*(s_{00}) - 0.01 \\ \mathbb{E} \left[\sum_{t=1}^{8p} R_t + v^*(S_{8 \cdot p}) \mid S_0 = s_{00} \right] &\geq \mathbb{E} \left[\sum_{t=1}^H R_t \mid S_0 = s_{00} \right] \geq v^*(s_{00}) - 0.01, \end{aligned}$$

where we put \cdot in the index of S to signify multiplication: as opposed to s , S only has a single index. It is valid to refer to the state $S_{8 \cdot p}$ as $K \geq 9$ by Eq. 22. Let us map any partial trajectory $S_0, A_0, S_1, A_1, \dots, S_{8 \cdot p-1}, A_{8 \cdot p-1}$ to the sequence $(\tilde{w}_i)_{i \in [8]} \in W^{o8}$ as follows. Let $j \in [8]$ be the smallest index for which $S_{j \cdot p-1} = \perp$, or let $j = 9$ if no such index exist in $[8]$. For $i \in [j-1]$, let $\tilde{w}_i = w(S_{i \cdot p-1}, A_{i \cdot p-1})$; for $i \in [j : 8]$, let \tilde{w}_i be any values such that $(\tilde{w}_i)_{i \in [8]} \in W^{o8}$ (which is always possible as $(\tilde{w}_i)_{i \in [j-1]} \in W^{o_{j-1}}$ when $j > 1$). Let

$$k^* = \min\{i \in [8] : i = 8 \text{ or } \text{diff}(\tilde{w}_i, w^*) < p/4\}.$$

Let R be the final reward of an abstract game (with the same parameters K, p, w^\star) for this sequence $(\tilde{w}_i)_{i \in [8]}$. By Eq. 10,

$$R = f_{w^\star}(w(S_{i \cdot p}))_{i \in [k^\star]}.$$

Observe that if there is an illegal action in the sequence $A_0, \dots, A_{8 \cdot p - 1}$, then $\sum_{t=1}^{8p} R_t + v^\star(S_{8 \cdot p}) = 0$. Otherwise, if $\text{diff}(\tilde{w}_i, w^\star) \geq p/4$ for all $i \in [8]$, then all transitions leading to $S_{8 \cdot p}$ fall under Case 23d as $K \geq 9$, and by Lemmas 4.9 and 4.11, $R = v^\star(S_{8 \cdot p}) = \sum_{t=1}^{8p} R_t + v^\star(S_{8 \cdot p})$. Finally, if $\text{diff}(\tilde{w}_{k^\star}, w^\star) < p/4$, then $S_{k^\star \cdot p} = \perp$, $v^\star(S_{k^\star \cdot p}) = 0$, $R = R_{k^\star \cdot p}$, and the rest of the rewards are zero. Therefore, either way,

$$\mathbb{E}[R] \geq \mathbb{E} \left[\sum_{t=1}^{8p} R_t + v^\star(S_{8 \cdot p}) \mid S_0 = s_{00} \right] \geq v^\star(s_{00}) - 0.01 = f_{w^\star}(). \quad (40)$$

Recall that each response to \mathcal{P} 's query to the MDP's simulator, as well as the transitions $(R_{t+1}, S_{t+1}) \sim \mathcal{Q}(\cdot \mid S_t, A_t)$ (for $t \in [0 : H - 1]$) can be implemented with at most one simulator call (respectively) to the abstract game (with the same parameters K, p, w^\star ; see Lemma 4.6). In expectation, this results in at most $8p\bar{N} + 8p$ such queries to the abstract game simulator. Together with Eq. 40, and noting that the choice of $w^\star \in W^\star$ was arbitrary, we see that \mathcal{P} can be used to construct an abstract planner \mathcal{A} that is sound with worst-case query cost $8p\bar{N} + 8p$. Therefore, by Theorem 4.1, and using Eq. 22,

$$\begin{aligned} 8p\bar{N} + 8p &= 2^{\Omega(p \wedge K)} \\ \bar{N} &= 2^{\Omega(H^{1/2} \wedge d^{1/4})}. \end{aligned}$$

■

5. Extending the guarantees of TensorPlan: the Proof of Theorem 1.2

The purpose of this section is to provide a proof of Theorem 1.2. As this theorem has three parts depending on the choice of the class of featurized MDPs \mathcal{M} , we proceed based on this choice.

As mentioned beforehand, Weisz et al. (2021a) already proved the theorem for $\mathcal{M} = \mathcal{M}_{B,d,H,A}^{v^\star}$. Hence, it remains to show the theorem for

$$\mathcal{M} = \mathcal{M}_{B,d,H,A}^{v^\star/q^\star \text{ reach}} \quad \text{and} \quad \mathcal{M} = \mathcal{M}_{B,d,H,A}^{q^\star} \cap \mathcal{M}^{\text{Pdet}}.$$

We start with the former case.

First, for easy reference, we include the pseudocode of TENSORPLAN, the planner from Weisz et al. (2021a), which establishes the claim for $\mathcal{M} = \mathcal{M}_{B,d,H,A}^{v^\star}$. The pseudocode, which can be found in Appendix B, is adjusted in minor ways to fit our conventions.

In the pseudocode, the simulator oracle is represented through the function SIMULATE, which returns a reward, next-state, associated-feature triplet as the response to a call (query) of a state and action tuple, as defined in Section 3.3. TENSORPLAN (TP), as a planner, is defined through the call to function GetAction. An input to this function indicates whether this function is called for the first state of an episode. The significance of this is that for the first state, TENSORPLAN runs a more expensive planning step, the result of which is reused in subsequent calls to GetAction *within* the episode.

From Weisz et al. (2021a), we have the following result:

Theorem 5.1 (Weisz et al. (2021a), Theorem 4.2 with Corollary 4.3). *For arbitrary positive reals δ, B and arbitrary positive integers d, H , it holds that*

$$C_{\text{LA}}^{\star}(\mathcal{M}_{B,d,H,A}^{v^{\star}}, \delta) = O\left(\text{poly}\left(\left(\frac{dH}{\delta}\right)^A, B\right)\right).$$

and TENSORPLAN is a planner that achieves this.

Let us now show that the same result also holds for $\mathcal{M} = \mathcal{M}_{B,d,H,A}^{v^{\star}/q^{\star}\text{reach}}$. We actually state and show this result for the case when q^{\star} realizability over the reachable states is dropped, leading to the class $\mathcal{M}_{B,d,H,A}^{v^{\star}\text{reach}}$. Clearly, it suffices to show the polynomial query complexity bound for $\mathcal{M}_{B,d,H,A}^{v^{\star}\text{reach}}$.

Lemma 5.2. *For arbitrary positive reals δ, B and arbitrary positive integers d, H , it holds that*

$$C_{\text{LA}}^{\star}(\mathcal{M}_{B,d,H,A}^{v^{\star}\text{reach}}, \delta) = O\left(\text{poly}\left(\left(\frac{dH}{\delta}\right)^A, B\right)\right).$$

and TENSORPLAN is a planner that achieves this.

Proof. This result is based on the observation that when TENSORPLAN is used with an MDP M from some initial state s_0 of the MDP, it only collects data from transitions for states in $\text{Reach}_M(s_0)$, the set of states that are reachable from s_0 . As such, running TENSORPLAN in M from s_0 generates the same joint distribution over queries and transitions as running it in an MDP M' whose state space is restricted to $\text{Reach}_M(s_0)$. Since in M there are no transitions from $\text{Reach}_M(s_0)$ to outside of this set, MDP M' is well-defined and its optimal value function matches that of M on the states in $\text{Reach}_M(s_0)$. Therefore $(M', \varphi_v|_{\text{Reach}_M(s_0)}) \in \mathcal{M}_{B,d,H,A}^{v^{\star}}$. Since TENSORPLAN is δ -optimal with the required polynomial complexity over the latter class, it induces a δ -optimal policy in M' with polynomial query cost, while, based on the relationship between M and M' , this policy is also δ -optimal in M . \blacksquare

To prove Theorem 1.2, it remains to show that the query complexity of $\mathcal{M}_{B,d,H,A}^{q^{\star}} \cap \mathcal{M}^{\text{Pdet}}$ is also polynomial.

Lemma 5.3. *For arbitrary positive reals δ, B and arbitrary positive integers d, H , it holds that*

$$C_{\text{LA}}^{\star}(\mathcal{M}_{B,d,H,A}^{q^{\star}} \cap \mathcal{M}^{\text{Pdet}}, \delta) = O\left(\text{poly}\left(\left(\frac{dH}{\delta}\right)^A, B\right)\right).$$

and TENSORPLAN can be adjusted to achieve this.

Proof. Take any featurized MDP $(M, \varphi_q) \in \mathcal{M}_{B,d,H,A}^{q^{\star}} \cap \mathcal{M}^{\text{Pdet}}$. Let $M = (\mathcal{S}, [A], Q)$ for some set of states \mathcal{S} and reward-transition kernel Q and let q^{\star} be the optimal action-value function in M . Let P and R be the transition and reward kernels respectively, corresponding to Q . Let $f : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ be such that $P(f(s, a)|s, a) = 1$. This function exists because M has deterministic transitions. Let $\mathcal{S} = \{\perp\} \cup \bigcup_{h=0}^{H-1} \mathcal{S}_h$ be the decomposition of the state-space of M from Assumption 3.1, noting that \mathcal{S}_h are pairwise disjoint. Fix an initial state $s_0 \in \mathcal{S}_0$.

The idea of the proof is to construct a new MDP \bar{M} from M such that (i) acting near-optimally in the new MDP implies acting near-optimally in M ; (ii) the optimal value function \bar{v}^{\star} of \bar{M} is linearly realizable with some feature-map $\bar{\varphi}_v$ with a ‘‘small parameter vector’’ $\bar{\theta}^{\star}$; (iii) transitions in the new MDP can be simulated by using a simulator of M , while this simulator can also provide access to $\bar{\varphi}_v$.

Then, one can use TENSORPLAN with the new MDP to get a good policy in M while keeping the query cost under control.

The new MDP $\bar{M} = (\bar{\mathcal{S}}, [A], \bar{Q})$ is an $H+1$ -horizon MDP which ‘delays’ rewards and transitions by one step. This will allow us to satisfy all three requirements listed above. In particular, the key to this is that states in the new MDP will be of the form (s, a) where (s, a) is a “generic” state-action pair in M and we will ensure that $\bar{v}^*((s, a)) = q^*(s, a)$. The state features in the new MDP can then be essentially chosen to be $\bar{\varphi}_v((s, a)) = \varphi_q(s, a)$.

The details are as follows: Apart from some special cases, the states in this new MDP are pairs of the original MDP’s states and actions, of the form (s, a) , with the action component corresponding to an action taken in the “previous step” in the new MDP. Then, when action a' is used in state (s, a) , unless $f(s, a) = \perp$ the next state is $(f(s, a), a')$, while the reward incurred comes from $R(\cdot|s, a)$ where R is the reward kernel underlying Q . When $f(s, a) = \perp$, the next state is simply \perp , while the reward still comes from $R(\cdot|s, a)$. At state \perp , as before, any action still transitions to \perp with no reward incurred. Finally, we add another state, $(s_0, 0)$, to the state of the new MDP so that $\{(s_0, 0)\}$ becomes the set of initial states. The transitions here are as follows: if action a is taken in state $(s_0, 0)$, the next state becomes (s_0, a) with no reward incurred. (The reason for not adding all states from $\mathcal{S}_1 \times \{0\}$ to \bar{M} will become clear later.) In summary, in the new MDP, when an action is taken, the action chosen in the previous time step (and stored as part of the state) is carried out and the new action is stored to be used in the next step.

Let $r(s, a)$ be the expected immediate reward when action a is taken in state s . We claim that by the construction of \bar{M} ,

$$\bar{v}^*((s, a)) = q^*(s, a), \quad (s, a) \in \mathcal{S}_h \times [A], 0 \leq h \leq H-1. \quad (41)$$

Indeed, this is trivial for $h = H-1$ as here as $q^*(s, a) = \bar{v}^*((s, a)) = r(s, a)$, as the final step of the episode transitions to \perp . Now, recall that the optimal value functions satisfy the **Bellman-optimality equations**. In the case of M , these take the form

$$v^*(s) = \max_a q^*(s, a), \quad (42)$$

$$q^*(s, a) = r(s, a) + v^*(f(s, a)), \quad (s, a) \in \mathcal{S} \times [A]. \quad (43)$$

Denote the same reward for \bar{M} and state (s, a) and action a' by $\bar{r}((s, a), a')$. Of course, the Bellman-optimality equations also hold for \bar{v}^* . With this, if Eq. 41 holds up to $h+1$, for $(s, a) \in \mathcal{S}_h \times [A]$ we have

$$\bar{v}^*((s, a)) = \max_{a' \in [A]} \bar{r}((s, a), a') + \bar{v}^*((f(s, a), a')) \quad (\text{Bellman optimality equations for } \bar{v}^*)$$

$$= \max_{a' \in [A]} r(s, a) + q^*(f(s, a), a') \quad (\text{definition of } \bar{r} \text{ and induction hypothesis})$$

$$= r(s, a) + v^*(f(s, a)) \quad (\text{Eq. 42})$$

$$= q^*(s, a), \quad (\text{Eq. 43})$$

finishing the proof of Eq. 41.

Now, Eq. 41 suggests just to define $\bar{\varphi}_v((s, a)) = \varphi_q(s, a)$ and $\bar{\varphi}_v(\perp) = \mathbf{0}$. This almost works except that we also need to define $\bar{\varphi}_v$ at $(s_0, 0)$. To deal with this case, we extend the dimension

of the feature space by one, setting $\bar{\varphi}_v((s_0, 0)) = [1, \mathbf{0}]$.⁸ Now, using that $\bar{r}((s_0, 0), a) = 0$ for all $a \in [A]$, we have

$$\bar{v}^*((s_0, 0)) = \max_a \bar{r}((s_0, 0), a) + \bar{v}^*((s_0, a)) = \max_a q^*(s_0, a) = v^*(s_0), \quad (44)$$

where the second equality used Eq. 41 with $(s, a) = (s_0, a)$. Using this, we set $\bar{\theta}^* = (v^*(s_0), \theta^*)$ so that \bar{v}^* is linearly realizable with the new features. (This is the point where we exploit that in \bar{M} there is only a single new initial state: this is why it suffices to add a single extra dimension to the feature space.)

As $\varphi_q \in \mathcal{S} \times [A] \rightarrow \mathcal{B}_d(1)$ and $\theta^* \in \mathcal{B}_d(B)$ we have that $v^*(s_0) \leq B$, therefore, by the triangle inequality, $\bar{\theta}^* \in \mathcal{B}_{d+1}(2B)$. Using $(M, \varphi_q) \in \mathcal{M}_{B,d,H,A}^{q^*}$, it follows that

$$(\bar{M}, \bar{\varphi}_v) \in \mathcal{M}_{2B,d+1,H+1,A}^{v^*}.$$

The planning method TP' for the class $\mathcal{M}_{B,d,H,A}^{q^*} \cap \mathcal{M}^{\text{pdet}}$ is designed as follows: When TP' is called with $s_0, \varphi_q(s_0, \cdot)$ at the beginning of an episode, TP' calls the `GetAction` method of TENSORPLAN with

$$((s_0, 0), (1, \mathbf{0}), \text{true}, A, H + 1, d + 1, \text{SIMULATE}', \delta, 2B)$$

where the pseudocode of `SIMULATE'` is given in Algorithm 1.

Algorithm 1 SIMULATE'

```

1: Inputs:  $\bar{s}, a'$ ; returns: rewards, next-states, associated features
2: if  $\bar{s} = \perp$  then
3:   return  $(0, \perp, \mathbf{0})$ 
4: else if  $\bar{s} = (s, 0)$  for some  $s \in \mathcal{S}$  then
5:   return  $(0, (s, a'), (0, \varphi_q(s, a')))$  ▷ Note:  $\bar{\varphi}_v((s, a')) = (0, \varphi_q(s, a'))$ 
6: else
7:    $(s, a) \leftarrow \bar{s}$ 
8:    $(R, S') \leftarrow \text{SIMULATE}(s, a)$ 
9:   if  $S' = \perp$  then
10:    return  $(R, \perp, \mathbf{0})$ 
11:   else
12:    return  $(R, (S', a'), (0, \varphi_q(S', a')))$  ▷ Note:  $\bar{\varphi}_v((S', a')) = (0, \varphi_q(S', a'))$ 
13:   end if
14: end if
    
```

The action A_0 returned by `GetAction` is returned by TP', which is then executed in M , transitioning to state S_1 while incurring reward R_1 (we set $S_0 = s_0$). More generally, for $1 \leq t \leq H$ let S_t be the t th state of MDP M , R_t be the reward associated with transitioning to S_t , and A_{t-1} be the action that led to this transition. Let $\bar{S}_0 = (s_0, 0)$ be the corresponding state in \bar{M} , and for $1 \leq t \leq H - 1$ let $\bar{S}_t = (S_t, A_{t-1})$ when $S_t \neq \perp$, and $\bar{S}_t = \perp$ when $S_t = \perp$. We also let $\bar{S}_H = \perp$.

⁸ $\mathbf{0}$ is a d -dimensional vector of zeros, and recall that $[1, \mathbf{0}] = (1, \mathbf{0}^T)^T$.

As the interaction between \bar{M} and TP' continues, for $1 \leq t \leq H$, TP' is called with $S_t, \varphi_q(S_t, \cdot)$. TP' then calls `GetAction` of TENSORPLAN with

$$(\bar{S}_t, \bar{\varphi}_v(\bar{S}_t), \text{false}, A, H + 1, d + 1, \text{SIMULATE}', \delta, 2B),$$

where \bar{S}_t is constructed as described above, from S_t and from A_{t-1} , the action returned by the last call to `TENSORPLAN.GetAction`, which is stored by TP' in the global memory. For $1 \leq t \leq H - 1$, the action A_t returned by `GetAction` is used in M .

Note that $(R_{t+1}, \bar{S}_{t+1}) \sim \bar{Q}(\cdot | \bar{S}_t, A_t)$ holds for $0 \leq t \leq H - 1$: Formally, if \mathbb{P} denotes the distribution induced over interaction sequences and \mathcal{F}_t is the smallest σ -algebra that makes the history leading up to the choice of A_t (and including A_t) measurable, then

$$\mathbb{P}(R_{t+1}, \bar{S}_{t+1} \in \cdot | \mathcal{F}_t) = \bar{Q}(\cdot | \bar{S}_t, A_t) \quad (45)$$

holds \mathbb{P} -almost surely for $0 \leq t \leq H - 1$. This relation follows directly from the definitions. Due to Eq. 45, from the perspective of TENSORPLAN, the environment is exactly \bar{M} . Hence, letting $\bar{\pi}$ denote the policy induced in \bar{M} by these calls, and $\bar{v}^{\bar{\pi}}$ denote the corresponding value function in \bar{M} , by Theorem 5.1 and Eq. 44, $\bar{v}^{\bar{\pi}}((s_0, 0)) \geq \bar{v}^*(s_0, 0) - \delta = v^*(s_0) - \delta$, while the total expected number of queries issued to `SIMULATE'` is $O\left(\text{poly}\left(\left(\frac{(d+1)(H+1)}{\delta}\right)^A, 2B\right)\right) = O\left(\text{poly}\left(\left(\frac{dH}{\delta}\right)^A, B\right)\right)$. Since in each call to `SIMULATE'`, `SIMULATE` is called at most once, the total number of queries issued to `SIMULATE` during the course of an episode satisfies the same bound. It follows that the query cost of the new planner also enjoys this bound. Hence, it remains to show that the new planner is also sound.

To see this, let π denote the policy induced by TP', the planner constructed above for M . Then, because rewards after H steps in M are by definition zero,

$$v^\pi(s_0) = \mathbb{E}[\sum_{t=1}^H R_t | S_0 = s_0] = \bar{v}^{\bar{\pi}}((s_0, 0)),$$

where \mathbb{E} is the expectation corresponding to \mathbb{P} and the second equality holds because of Eq. 45 (as noted before, from the point of view of TENSORPLAN, the environment is \bar{M} thanks to this identity). Putting things together thus finishes the proof. \blacksquare

Proof of Theorem 1.2. The theorem follows directly from Theorem 5.1 and Lemmas 5.2 and 5.3. \blacksquare

References

- R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, New Jersey, 1957.
- R. Bellman, R. Kalaba, and B. Kotkin. Polynomial approximation – a new computational technique in dynamic programming: Allocation processes. *Mathematics of Computation*, 17(8):155–161, 1963.
- Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.
- C. S. Chow and J. N. Tsitsiklis. The complexity of dynamic programming. *Journal of Complexity*, 5: 466–488, 1989.
- Simon S Du, Sham M Kakade, Ruosong Wang, and Lin F Yang. Is a good representation sufficient for sample efficient reinforcement learning? In *International Conference on Learning Representations*, 2019.

- Simon S Du, Sham M Kakade, Jason D Lee, Shachar Lovett, Gaurav Mahajan, Wen Sun, and Ruosong Wang. Bilinear classes: A structural framework for provable generalization in RL. In *ICML*, volume 139, pages 2826–2836, 2021.
- Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning*, 49:193–208, 2002.
- Tor Lattimore, Csaba Szepesvári, and Gellért Weisz. Learning with good feature representations in bandits and in RL with a generative model. In *ICML*, pages 9464–9472, 2020.
- Mausam and Andrey Kolobov. Planning with Markov decision processes: An ai perspective. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–210, 2012.
- Remi Munos. Error bounds for approximate policy iteration. In *ICML*, pages 560–567, 2003.
- Remi Munos. Error bounds for approximate value iteration. In *AAAI*, pages 1006–1011, 2005.
- Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 1994.
- J Rust, J F Traub, and H Wozniakowski. Is there a curse of dimensionality for contraction fixed points in the worst case? *Econometrica*, 70(1):285–329, January 2002.
- Bruno Scherrer. Improved and generalized upper bounds on the complexity of policy iteration. *Mathematics of Operations Research*, 41(3):758–774, August 2016.
- Paul J. Schweitzer and Abraham Seidmann. Generalized polynomial approximations in Markovian decision processes. *Journal of Mathematical Analysis and Applications*, 110(2):568–582, September 1985.
- Cs. Szepesvári and R. Munos. Finite time bounds for sampling based fitted value iteration. In *ICML*, pages 881–886, 2005.
- Paul Tseng. Solving h -horizon, stationary Markov decision problems in time proportional to $\log(h)$. *Operations Research Letters*, 9(5):287–297, 1990.
- John N Tsitsiklis and Benjamin Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1):59–94, 1996.
- Yuanhao Wang, Ruosong Wang, and Sham Kakade. An exponential lower bound for linearly realizable mdp with constant suboptimality gap. *Advances in Neural Information Processing Systems*, 34, 2021.
- Gellért Weisz, Philip Amortila, Barnabás Janzer, Yasin Abbasi-Yadkori, Nan Jiang, and Csaba Szepesvári. On query-efficient planning in MDPs under linear realizability of the optimal state-value function. In *COLT*, volume 134 of *Proceedings of Machine Learning Research*, pages 4355–4385, 2021a.
- Gellért Weisz, Philip Amortila, and Csaba Szepesvári. Exponential lower bounds for planning in MDPs with linearly-realizable optimal action-value functions. In *ALT*, volume 132 of *Proceedings of Machine Learning Research*, pages 1237–1264, 2021b.

Zheng Wen and Benjamin Van Roy. Efficient exploration and value function generalization in deterministic systems. In *Advances in Neural Information Processing Systems*, pages 3021–3029, 2013.

Yinyu Ye. The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate. *Mathematics of Operations Research*, 36(4):593–603, 2011.

Appendix A. Calculating the linear features

A.1. Calculating feature components of φ_v

We follow the notation of Section 4.7. In particular, for any state $s \in \mathcal{S}$, $s \neq \perp$, let $s = s_{ki}$ be a state along step i of round k . We intend to linearize the expression $g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}})g(e_{ki}^{\text{fix}})$.

Let $x = \text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}}$ and $y = e_{ki}^{\text{fix}}$. Then, x and y can be written according to Eqs. 25, 26 as:

$$\begin{aligned} y &= \frac{1}{2} (\langle \mathbf{1}, \text{fix}_{ki} \rangle - \langle \text{fix}_{ki} \cdot \bar{w}_{ki}, \bar{w}^* \rangle) = \langle y_{(1,0)}, \mathbf{1} \rangle + \langle y_{(1,1)}, \bar{w}^* \rangle \\ &\quad \text{for } y_{(1,0)} = \frac{1}{2} \langle \mathbf{1}, \text{fix}_{ki} \rangle \text{ and } y_{(1,1)} = -\frac{\sqrt{p}}{2} \text{fix}_{ki} \cdot \bar{w}_{ki} \\ &\quad \text{with } \|y_{(1,0)}\|_2, \|y_{(1,1)}\|_2 \leq p \\ x &= \text{ct}_{ki}^{\text{flip}} + \frac{1}{2} (\langle \mathbf{1}, \neg \text{fix}_{ki} \rangle - \langle \neg \text{fix}_{ki} \cdot \bar{w}_{ki}, \bar{w}^* \rangle) = \langle x_{(1,0)}, \mathbf{1} \rangle + \langle x_{(1,1)}, \bar{w}^* \rangle \\ &\quad \text{for } x_{(1,0)} = \text{ct}_{ki}^{\text{flip}} + \frac{1}{2} \langle \mathbf{1}, \neg \text{fix}_{ki} \rangle \text{ and } x_{(1,1)} = -\frac{\sqrt{p}}{2} \neg \text{fix}_{ki} \cdot \bar{w}_{ki} \\ &\quad \text{with } \|x_{(1,0)}\|_2, \|x_{(1,1)}\|_2 \leq p \end{aligned}$$

Notice that $x_{(\cdot, \cdot)}$ and $y_{(\cdot, \cdot)}$ do not depend on w^* , only on the current state s_{ki} . Furthermore, using Lemma 4.12

$$\begin{aligned} x^2 &= \langle x_{(1,0)}^2, \mathbf{1} \rangle + \langle 2x_{(1,0)}x_{(1,1)}, \bar{w}^* \rangle + \langle b(x_{(1,1)} \otimes x_{(1,1)}), b(\bar{w}^* \otimes \bar{w}^*) \rangle \\ &= \langle x_{(2,0)}, \mathbf{1} \rangle + \langle x_{(2,1)}, \bar{w}^* \rangle + \langle x_{(2,2)}, (\bar{w}^*)^2 \rangle \\ &\quad \text{for } x_{(2,0)} = x_{(1,0)}^2, x_{(2,1)} = 2x_{(1,0)}x_{(1,1)}, x_{(2,2)} = b(x_{(1,1)} \otimes x_{(1,1)}), \\ &\quad \text{and } (\bar{w}^*)^{\otimes 2} = b(\bar{w}^* \otimes \bar{w}^*) \\ &\quad \text{with } \|x_{(2,0)}\|_2, \|x_{(2,1)}\|_2, \|x_{(2,2)}\|_2 \leq 2p^2 \\ g(x) &= 1 + x \frac{-2p-1}{2p^2} + x^2 \frac{1}{2p^2} = \langle X_{(0)}, \mathbf{1} \rangle + \langle X_{(1)}, \bar{w}^* \rangle + \langle X_{(2)}, (\bar{w}^*)^2 \rangle \\ &\quad \text{for } X_{(0)} = 1 + \frac{-2p-1}{2p^2} x_{(1,0)} + \frac{1}{2p^2} x_{(2,0)}, \\ &\quad X_{(1)} = \frac{-2p-1}{2p^2} x_{(1,1)} + \frac{1}{2p^2} x_{(2,1)}, \text{ and } X_{(2)} = \frac{1}{2p^2} x_{(2,2)} \\ &\quad \text{with } \|X_{(0)}\|_2 \leq 4, \|X_{(1)}\|_2 \leq 3, \|X_{(2)}\|_2 \leq 1. \end{aligned}$$

and by a similar calculation,

$$\begin{aligned}
 y^2 &= \langle y_{(2,0)}, 1 \rangle + \langle y_{(2,1)}, \bar{w}^\star \rangle + \langle y_{(2,2)}, (\bar{w}^\star)^2 \rangle \\
 &\text{for } y_{(2,0)} = y_{(1,0)}^2, \quad y_{(2,1)} = 2y_{(1,0)}y_{(1,1)}, \quad y_{(2,2)} = b(y_{(1,1)} \otimes y_{(1,1)}) \\
 &\text{with } \|y_{(2,0)}\|_2, \|y_{(2,1)}\|_2, \|y_{(2,2)}\|_2 \leq 2p^2 \\
 g(y) &= \langle Y_{(0)}, 1 \rangle + \langle Y_{(1)}, \bar{w}^\star \rangle + \langle Y_{(2)}, (\bar{w}^\star)^2 \rangle \\
 &\text{for } Y_{(0)} = 1 + \frac{-2p-1}{2p^2}y_{(1,0)} + \frac{1}{2p^2}y_{(2,0)}, \\
 &\quad Y_{(1)} = \frac{-2p-1}{2p^2}y_{(1,1)} + \frac{1}{2p^2}y_{(2,1)}, \quad \text{and } Y_{(2)} = \frac{1}{2p^2}y_{(2,2)} \\
 &\text{with } \|Y_{(0)}\|_2 \leq 4, \|Y_{(1)}\|_2 \leq 3, \|Y_{(2)}\|_2 \leq 1.
 \end{aligned}$$

Therefore, again using Lemma 4.12,

$$\begin{aligned}
 g(\text{ct}_{ki}^{\text{flip}} + e_{ki}^{-\text{fix}})g(e_{ki}^{\text{fix}}) &= g(x)g(y) \\
 &= \langle b(X_{(0)} \otimes Y_{(0)}), 1 \rangle + \langle b(X_{(0)} \otimes Y_{(1)} + X_{(1)} \otimes Y_{(0)}), \bar{w}^\star \rangle \\
 &\quad + \langle b(X_{(0)} \otimes Y_{(2)} + X_{(1)} \otimes Y_{(1)} + X_{(2)} \otimes Y_{(0)}), (\bar{w}^\star)^{\otimes 2} \rangle \\
 &\quad + \langle b(X_{(1)} \otimes Y_{(2)} + X_{(2)} \otimes Y_{(1)}), (\bar{w}^\star)^{\otimes 3} \rangle + \langle b(X_{(2)} \otimes Y_{(2)}), (\bar{w}^\star)^{\otimes 4} \rangle \\
 &= \langle Z_{(0)}, 1 \rangle + \langle Z_{(1)}, \bar{w}^\star \rangle + \langle Z_{(2)}, (\bar{w}^\star)^{\otimes 2} \rangle + \langle Z_{(3)}, (\bar{w}^\star)^{\otimes 3} \rangle + \langle Z_{(4)}, (\bar{w}^\star)^{\otimes 4} \rangle \\
 &\text{for } Z_{(0)} = b(X_{(0)} \otimes Y_{(0)}), \\
 &\quad Z_{(1)} = b(X_{(0)} \otimes Y_{(1)} + X_{(1)} \otimes Y_{(0)}), \\
 &\quad Z_{(2)} = b(X_{(0)} \otimes Y_{(2)} + X_{(1)} \otimes Y_{(1)} + X_{(2)} \otimes Y_{(0)}), \\
 &\quad Z_{(3)} = b(X_{(1)} \otimes Y_{(2)} + X_{(2)} \otimes Y_{(1)}), \\
 &\quad (\bar{w}^\star)^{\otimes 3} = b(\bar{w}^\star \otimes \bar{w}^\star \otimes \bar{w}^\star), \\
 &\quad (\bar{w}^\star)^{\otimes 4} = b(\bar{w}^\star \otimes \bar{w}^\star \otimes \bar{w}^\star \otimes \bar{w}^\star) \\
 &\text{with } \|Z_{(0)}\|_2 \leq 16, \|Z_{(1)}\|_2 \leq 24, \|Z_{(2)}\|_2 \leq 17, \|Z_{(3)}\|_2 \leq 6.
 \end{aligned} \tag{46}$$

A.2. Calculating feature components of φ_q

We follow the notation of Section 4.7. In particular, for any state $s \in \mathcal{S}$, $s \neq \perp$ and action $a \in [A]$, let $s = s_{ki}$ be a state along step i of round k . Let $s_{k,i+1}^a$ denote the value taken by $s_{k,i+1}$ if $a_{ki} = a$, and similarly for $w_{k,i+1}^a$. We intend to linearize the expression $g(\text{diff}(w_{k+1,0}^a, w^\star))$.

Let $x = \text{diff}(w_{k+1,0}^a, w^\star)$. By Eq. 7,

$$\begin{aligned}
 x &= \frac{1}{2} \left(p - \langle w_{k+1,1}^a, w^\star \rangle \right) = \langle x_{(1,0)}, 1 \rangle + \langle x_{(1,1)}, w^\star \rangle \\
 &\text{for } x_{(1,0)} = \frac{1}{2}p \text{ and } x_{(1,1)} = -\frac{1}{2}w_{k+1,1}^a \\
 &\text{with } \|x_{(1,0)}\|_2, \|x_{(1,1)}\|_2 \leq p
 \end{aligned} \tag{47}$$

By a similar calculation to the previous case,

$$\begin{aligned}
 x^2 &= \langle x_{(1,0)}^2, 1 \rangle + \langle 2x_{(1,0)}x_{(1,1)}, w^\star \rangle + \langle b(x_{(1,1)} \otimes x_{(1,1)}), b(w^\star \otimes w^\star) \rangle \\
 &= \langle x_{(2,0)}, 1 \rangle + \langle x_{(2,1)}, w^\star \rangle + \langle x_{(2,2)}, (w^\star)^2 \rangle \\
 &\quad \text{for } x_{(2,0)} = x_{(1,0)}^2, x_{(2,1)} = 2x_{(1,0)}x_{(1,1)}, x_{(2,2)} = b(x_{(1,1)} \otimes x_{(1,1)}), \\
 &\quad \text{and } (w^\star)^{\otimes 2} = b(w^\star \otimes w^\star)
 \end{aligned} \tag{48}$$

$$\begin{aligned}
 &\text{with } \|x_{(2,0)}\|_2, \|x_{(2,1)}\|_2, \|x_{(2,2)}\|_2 \leq 2p^2 \\
 g(x) &= 1 + x \frac{-2p-1}{2p^2} + x^2 \frac{1}{2p^2} = \langle X_{(0)}, 1 \rangle + \langle X_{(1)}, w^\star \rangle + \langle X_{(2)}, (w^\star)^2 \rangle \\
 &\quad \text{for } X_{(0)} = 1 + \frac{-2p-1}{2p^2}x_{(1,0)} + \frac{1}{2p^2}x_{(2,0)}, \\
 &\quad X_{(1)} = \frac{-2p-1}{2p^2}x_{(1,1)} + \frac{1}{2p^2}x_{(2,1)}, \text{ and } X_{(2)} = \frac{1}{2p^2}x_{(2,2)} \\
 &\quad \text{with } \|X_{(0)}\|_2 \leq 4, \|X_{(1)}\|_2 \leq 3, \|X_{(2)}\|_2 \leq 1.
 \end{aligned} \tag{49}$$

Appendix B. Pseudocode and constants of TENSORPLAN

Algorithm 2 TENSORPLAN.GetAction

```

1: Inputs:  $s, \varphi_v, \text{EpisodeStart},$ 
2:            $A, H, d, \text{SIMULATE}, \delta, B$ 
3: if EpisodeStart then           ▶ Initialize global  $\theta^+$ 
4:   TENSORPLAN.Init(
5:      $s, \varphi_v(s), A, H, d, \text{SIMULATE}, \delta$ )
6: end if
7:  $\bar{\Delta} \leftarrow \text{ApproxTD}(s, \varphi_v(s), A, n_2, \text{SIMULATE})$ 
8: return  $\arg \min_{a \in [A]} \left| \left\langle \bar{\Delta}_a, [1, \theta^+] \right\rangle \right|$ 

```

Algorithm 3 APPROXTD

```

1: Inputs:  $s, \varphi_v(s), A, n, \text{SIMULATE}$ 
2: for  $a = 1$  to  $A$  do
3:   for  $l = 1$  to  $n$  do
4:      $(R_l, S'_l, \varphi_v(S'_l)) \leftarrow$ 
5:       SIMULATE( $s, a$ )
6:      $\tilde{\Delta}_l \leftarrow [R_l, (\varphi_v(S'_l) - \varphi_v(s))]$ 
7:   end for
8:    $\Delta_a := \frac{1}{n} \sum_{l \in [n]} \tilde{\Delta}_l$ 
9: end for
10: return  $(\Delta_a)_{a \in [A]}$ 

```

Algorithm 4 TENSORPLAN.Init

```

1: Inputs:  $s_0, \varphi_v(s_0), A, H, d, \text{SIMULATE}, \delta$ 
2:  $X \leftarrow \{\}$  ▷  $X$  is a list
3: Initialize  $\zeta, \varepsilon, n_1, n_2, n_3$  via equations (52), (53), (54), (55), (56), respectively.
4: for  $\tau = 1$  to  $E_d + 2$  do
5:   Choose any  $\theta_\tau \in \arg \max_{\theta \in \text{Sol}(X)} \langle \varphi_v(s_0), \theta \rangle$  ▷ Optimistic choice
6:   CleanTest  $\leftarrow$  true
7:   for  $t = 1$  to  $n_1$  do ▷  $n_1$  rollouts with  $\theta_\tau$ -induced policy
8:      $S_{\tau t 1} = s_0$  ▷ Initialize rollout
9:     for  $j = 1$  to  $H$  do ▷ Stages in episode
10:       $\bar{\Delta}_{\tau t j, \cdot} \leftarrow \text{ApproxTD}(S_{\tau t j}, \varphi_v(S_{\tau t j}), A, n_2, \text{SIMULATE})$ 
11:      if CleanTest and  $\min_{a \in [A]} \left| \left\langle \bar{\Delta}_{\tau t j a}, [1, \theta]_\tau \right\rangle \right| > \frac{\delta}{4H}$  then ▷ Consistency failure?
12:         $\hat{\Delta}_{\tau t j, \cdot} \leftarrow \text{ApproxTD}(S_{\tau t j}, \varphi_v(S_{\tau t j}), A, n_3, \text{SIMULATE})$  ▷ Refined data
13:         $X.\text{append} \left( \otimes_{a \in [A]} \hat{\Delta}_{\tau t j a} \right)$  ▷ Save failure data
14:        CleanTest  $\leftarrow$  false ▷ Not clean anymore
15:      end if
16:       $A_{\tau t j} \leftarrow \arg \min_{a \in [A]} \left| \left\langle \bar{\Delta}_{\tau t j a}, [1, \theta]_\tau \right\rangle \right|$  ▷ Find most consistent action
17:       $(R_{\tau t j}, S_{\tau t j+1}, \varphi_v(S_{\tau t j+1})) \leftarrow \text{SIMULATE}(S_{\tau t j}, A_{\tau t j})$  ▷ Roll forward
18:    end for
19:  end for
20:  if CleanTest then break ▷ Success?
21: end for
22: Save into global memory  $\theta^+ \leftarrow \theta_\tau$ 

```

$$E_d = \left\lceil 3(d+1)^A \frac{e}{e-1} \ln \left\{ 3 + 3 \left(\frac{2(B+1)^A 3^A}{H^A \varepsilon} \right)^2 \right\} + 1 \right\rceil \quad (50)$$

$$\text{Sol}(\Delta_1, \dots, \Delta_\tau) = \left\{ \theta \in \mathbb{R}^d : \|\theta\|_2 \leq B, \forall i \in [\tau] : \left| \left\langle \Delta_i, \otimes_{a \in [A]} \bar{1}\theta \right\rangle \right| \leq \frac{H^A \varepsilon}{2\sqrt{E_d}} \right\} \quad (51)$$

$$\zeta = \frac{1}{4H} \delta \quad (52)$$

$$\varepsilon = \left(\frac{\delta}{12H^2} \right)^A / \left(1 + \frac{1}{2\sqrt{E_d}} \right) \quad (53)$$

$$n_1 = \left\lceil \frac{32(1+2B)^2}{\delta^2} \log \frac{E_d + 1}{\zeta} \right\rceil \quad (54)$$

$$n_2 = \left\lceil \frac{1867H^2(B+1)^2(d+1)}{2\delta^2} \log(4(E_d+1)n_1HA(d+1)/\zeta) \right\rceil \quad (55)$$

$$n_3 = \left\lceil \max \left\{ n_2, \frac{32(H+1)^2 E_d}{\varepsilon^2} \log((2(E_d+1)n_1HA)/\zeta) \right\} \right\rceil \quad (56)$$