Multi-Environment POMDPs with Finite-Horizon Objectives

Léonard Brice
Institute of Science and Technology Austria
Klosterneuburg, Austria
leonard.brice@ist.ac.at

Filip Cano
Institute of Science and Technology Austria
Klosterneuburg, Austria
filip.cano@ist.ac.at

Krishnendu Chatterjee
Institute of Science and Technology Austria
Klosterneuburg, Austria
krishnendu.chatterjee@ist.ac.at

Thomas A. Henzinger
Institute of Science and Technology Austria
Klosterneuburg, Austria
tah@ist.ac.at

Stefanie Muroya
Institute of Science and Technology Austria
Klosterneuburg, Austria
stefanie.muroyalei@ist.ac.at

Abstract

Partially Observable Markov Decision Processes (POMDPs) are systems in which one agent interacts with a stochastic environment, and receives only partial information about the current state. In a multi-environment POMDP (MEPOMDP), the initial state is unknown, and assumed to be adversarially chosen. In this work we focus on computing the optimal value and policy in MEPOMDPs with finite-horizon objectives. That problem is known to be PSPACE-complete in POMDPs. The extension is nontrivial: MEPOMDPs require reasoning about beliefs and payoffs across environments, and deterministic policies are no longer enough for optimal payoff. While the lower bound transfers immediately, the upper bound requires new techniques: we prove that mixtures of at most n deterministic policies suffice for n environments and show how to explore them without storing exponential-size policy descriptions. Our main results are as follows: (1) we establish that it is also PSPACE-complete in the more general setting of MEPOMDPs; (2) we present a practical algorithm and evaluate it on classical benchmarks, significantly outperforming the only previously known algorithm.

1 Introduction

Sequential decision-making under uncertainty is a central challenge in artificial intelligence research. *Markov Decision Processes* (MDPs) constitute a standard theoretical model to capture settings in which an agent interacts with a stochastic environment. An MDP consists of a finite set of *states* and a finite set of *actions*; given a state, each action choice stochastically updates the state and gives per-stage rewards. MDPs have already shown a wide range of applications, from management science to robotics [15, 20, 17]. They also have many extensions: we discuss two prominent ones.

MDPs assume known dynamics. In practice, several models may be plausible without a defensible probability distribution over them. Optimising average performance under an arbitrary prior can conceal poor behaviour in one candidate model; a robust formulation instead seeks one policy that performs well across all candidates.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

The first extension of MDPs is known as *partial-observation MDPs* (POMDPs). While in MDPs the agent always has perfect observation about the state, in POMDPs, the agent only receives some information about the state. POMDPs are remarkably expressive and have found applications in robotics [17], healthcare [8], and autonomous planning more broadly. POMDPs capture state uncertainty within one model, not uncertainty over the model itself.

The second extension is to *multi-environment MDPs* (MEMDPs), where instead of a single environment, there are multiple possible environments that the agent is unaware of. This setting was studied for two environments in [16] and later for many environments [5, 21]. MEMDPs have also been studied in several applications such as recommander systems and robot navigation [4]. In an MEMDP, one environment is selected before execution and remains fixed. The agent may learn about it from the trajectory, but must act robustly before it has been identified.

Combining the above two aspects leads to *multi-environment POMDPs* (MEPOMDPs). Those were introduced in [1], where it was established that they constitute a robust model that is equivalent to various natural settings, such as adversarial-belief POMDPs or multi-objective POMDPs.

MEPOMDPs combine partial state information with discrete model uncertainty, as when a robot has one of finitely many actuator models or experts provide incompatible models. The environment remains fixed, while observations refine state estimates and eliminate incompatible environments. Finite-horizon computation is relevant for tasks with deadlines or resource budgets, and for approximating discounted objectives by truncation. Exact infinite-horizon value computation is undecidable already for POMDPs [12], making finite horizon the natural setting for exact guarantees.

Contributions. In this work, we consider MEPOMDPs with finite-horizon objectives, where the agent's goal is to maximise the reward sum for a finite-horizon length k. We consider both exact and approximate computation of the optimal value the agent can achieve. Our results are as follows.

Complexity. We establish that both the exact and approximate problems are in PSPACE, and therefore PSPACE-complete, using the lower bound known for POMDPs.

Space-efficient algorithm. While in the complexity result we present a non-deterministic algorithm, whose determinisation using standard techniques leads to an algorithm with substantial (albeit polynomial) space usage, we present a deterministic space-efficient algorithm.

Practical algorithm and experiments. We then complement the above theoretical results with a time-efficient practical algorithm, and test its performance and scalability empirically. Our experiments show that our method obtains better performance than the previously existing state of the art tool.

Related works. *Complexity of POMDPs.* The landmark paper of Papadimitriou and Tsitsiklis [14] establishes that the decision version of value computation of POMDPs, under finite horizon, is PSPACE-complete. In [12], Madani, Hanks, and Condon show undecidability for the infinite-horizon case, with undiscounted and discounted payoff, a result that motivates the focus on finite horizon throughout our work. Exact and approximate finite-horizon POMDP solvers are studied extensively in [2, 19], but they optimise one scalar expectation for one known model. Our result shows that adversarial discrete model uncertainty preserves the classical PSPACE boundary.

Multi-environment MDPs. The main works about MEMDPs have been described above [16, 5, 21, 4]. None of those papers consider partial observation. [16] introduces robust synthesis for fully observable environments, [5] studies efficient analysis and applications, and [21, 4] give complexity results for reachability and parity objectives. We consider quantitative finite-horizon rewards under partial observation.

Robust POMDPs. In [13], Osogami introduces *robust POMDPs*, i.e. POMDPs equipped with a continuous uncertainty set for the transition and/or observation probabilities. Multi-environment POMDPs can be seen as a subcase of robust POMDPs, where the uncertainty set is finite. [13] develops value-iteration and point-based methods for continuous uncertainty. In [9], Krale et al. study evaluation of a fixed robust-POMDP policy, reducing it to an equivalent MDP and deriving bounds on its optimality gap. These results do not determine the complexity of exact MEPOMDP value computation. Many robust-POMDP methods also rely on convexity, rectangularity, or dynamic uncertainty. An MEPOMDP instead represents finitely many globally coupled environments, one of which remains fixed throughout the execution; allowing stepwise model changes can therefore be unnecessarily conservative.

Multi-environment POMDPs. Under the name MEPOMDPs, the model was studied by Bovy et al. [1], who propose an exact algorithm for the finite-horizon case and an approximate one for the infinite-horizon discounted case. However, that paper does not provide any complexity result, and the algorithm proposed is significantly slower than the one we present here, as we detail in Section 6. The method in [1] combines POMDP alpha-vector backups with linear programming over adversarial initial beliefs, adapts heuristic search value iteration for the model. We add tight complexity bounds, a deterministic polynomial-space algorithm, and an algorithm with exponential worst-case complexity that proves to be more performant in practical benchmarks adapted from the POMDP literature.

More recently, Galesloot et al. [6] study the same fixed-hidden-model principle under the name hidden-model POMDPs. Their method restricts policies to finite-state controllers and combines deductive worst-case policy evaluation with projected subgradient optimisation. It addresses a complementary, scalable optimisation setting, but does not compute the unrestricted exact finite-horizon value or establish its computational complexity.

Finally, MEPOMDPs can also be seen as a special case of partially observable stochastic games [10, 7], where the opponent has a non-trivial choice to make at the first round only. To the best of our knowledge, the exact complexity of the equivalent of our problem, when an opponent is added to the picture and selects an action at each round, is still open.

2 Background and problem statement

Given a set S, we denote the set of probability distributions over S as $\mathcal{D}(S)$, and the support of a distribution d as $\mathsf{Supp}(d)$. The set of natural numbers (with 0) is $\mathbb{N}$. Given $n \in \mathbb{N}$, we write $[n] = \{1, \dots, n\}$. For a set S, and an indexing set I, when we have defined an element $x_i \in S$ for every $i \in I$, we use the bar notation $\bar{x}$ for the tuple $(x_i)_{i \in I} \in X^I$. Conversely, when we introduce a tuple $\bar{x}$, the notation x_i refers to the element of index $i \in I$. We write $\bar{x} \leq \bar{y}$ when the tuple $\bar{y}$ *dominates* $\bar{x}$, i.e., when we have $x_i \leq y_i$ for every index i. Given a set $S \subseteq \mathbb{R}^n$, we say that S dominates $\bar{x}$, and write $\bar{x} \leq S$, when there exists $\bar{y} \in S$ such that $\bar{x} \leq \bar{y}$. We denote by $\mathsf{Conv}(S) \subseteq \mathbb{R}^n$ the convex closure of S, i.e., the set of points $\bar{x}$ such that there exist points $\bar{y}_1, \dots, \bar{y}_k \in S$ and coefficients $\alpha_1, \dots, \alpha_k \in [0, 1]$ with $\sum_i \alpha_i = 1$ such that $\sum_i \alpha_i \bar{y}_i = \bar{x}$.

2.1 POMDPs

We now define the model that is considered in this paper.

Definition 1 (Partially Observable Markov Decision Process (POMDP))**.** *A POMDP is a tuple $\mathcal{P} = \langle S, A, \mathcal{O}, \delta : S \times A \to \mathcal{D}(S), \mathsf{obs} : S \to \mathcal{O}, r : S \to \mathbb{Z} \rangle$ that consists of (i) a finite set S of states, (ii) a finite set A of actions, (iii) a finite set $\mathcal{O}$ of observations, (iv) a stochastic transition function δ, (v) an observation function* obs*, and (vi) a reward function r.*

Remark 1 (Equivalence with other models). POMDPs are often described with more general observation and reward functions, that depend on states and actions, and have probabilistic outcome. However, this is equivalent to the simple model that we consider (see Remark 4 in [3]).

A MDP is a special case of POMDPs where each state has a unique observation.

A *trajectory* of length $k \in \mathbb{N}$ in the POMDP $\mathcal{P}$ is a finite sequence $\tau = (s_0, a_0, s_1, \dots, a_{k-1}, s_k) \in (S \cdot A)^* \cdot S$. The protagonist of a POMDP aims at generating trajectories that maximise her *payoff*, i.e., the sum of the rewards collected on the way, which we write $\mathsf{payoff}(\tau) = \sum_{\ell \leq k} r(s_\ell)$.

An *observation sequence* in $\mathcal{P}$ is a finite sequence $(o_0, a_0, o_1, \dots, a_{k-1}, o_k) \in (\mathcal{O} \cdot A)^* \cdot \mathcal{O}$.

The set of trajectories in the POMDP $\mathcal{P}$ is denoted by $\mathsf{Traj}(\mathcal{P})$, and the set of observation sequences by $\mathsf{ObsSeq}(\mathcal{P})$. We define $\mathsf{Traj}_k(\mathcal{P})$ as the set of trajectories of length k. Each trajectory τ is associated with an observation sequence, denoted by $\mathsf{obs}(\tau) = (\mathsf{obs}(s_0), a_0, \mathsf{obs}(s_1), \cdots, a_{k-1}, \mathsf{obs}(s_k))$.

Example 1. Figure 1 depicts an example of a POMDP. In this POMDP, when action c is performed from the state s_1, the state t_3 is reached with probability 0.1, and the state t_4 with probability 0.9. The dashed boxes indicate that all states have the same observation o_1, except the states t_3 and t_4, with observation o_2. The double circles indicate states with reward 1; the other ones have reward 0.

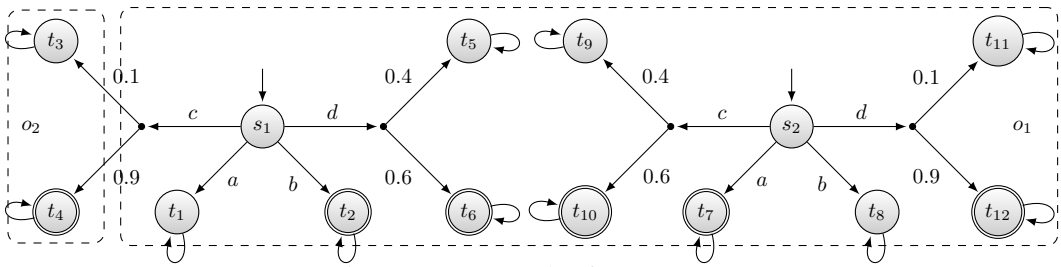


Figure 1: An example of a POMDP

Policies. A *policy* in the POMDP $\mathcal{P}$ is a function $\sigma : \mathsf{ObsSeq}(\mathcal{P}) \to \mathcal{D}(A)$. It is *deterministic* if for every observation sequence $\omega \in \mathsf{ObsSeq}(\mathcal{P})$, there is an action a with $\sigma(\omega)(a) = 1$. We denote the set of all policies on $\mathcal{P}$ as $\mathsf{Pol}(\mathcal{P})$, and the set of deterministic policies as $\mathsf{DetPol}(\mathcal{P})$.

Given a POMDP $\mathcal{P}$ and an initial state $s \in S$, each policy σ defines a probability distribution $\mathbb{P}_s^\sigma$ over trajectories of length k, for each *horizon* $k \in \mathbb{N}$. Thus, we can define the expected payoff obtained when following that policy: we write it $\mathbb{E}_s^k(\sigma)$. Throughout this paper, we focus on policy behaviors within a well-defined finite horizon k: we therefore assume that policies are described only on observations sequences of length bounded by k, which uses space $2^{O(k)}$.

2.2 Beliefs

A classical tool to study POMDPs is the notion of *belief*: from a known initial state, after having performed a sequence of actions and seen a sequence of observations, the agent can keep track of the probability of being in each state.

Definition 2 (Belief)**.** *A* belief *in the POMDP $\mathcal{P}$ is a probability distribution β over states, such that for all s, s' in $\mathsf{Supp}(\beta)$, we have that $\mathsf{obs}(s) = \mathsf{obs}(s')$.*

We sometimes abuse notations and write $\mathsf{obs}(\beta)$ for the unique observation associated to the belief β. When the agent has the belief β, performs an action a and sees the observation o, she must update her belief to reflect her knowledge about which state may have now be reached. We denote that updated belief by $\mathsf{update}(\beta, a, o)$, defined for each state s by:

$$\mathsf{update}(\beta, a, o)(s) = \frac{\sum_{s' \in S} \beta(s') \cdot \delta(s', a)(s)}{\sum_{\substack{s'' \in S \\ \mathsf{obs}(s'')=o}} \sum_{s' \in S} \beta(s') \cdot \delta(s', a)(s'')}$$

if $\mathsf{obs}(s) = o$, and $\mathsf{update}(\beta, a, o)(s) = 0$ otherwise.

2.3 Multi-environment POMDP

We can now define the model we focus on throughout this paper.

Definition 3 (Multi-environment POMDP)**.** *A* multi-environment POMDP *is a pair $\mathcal{M} = \langle \mathcal{P}, (s_i)_{i \in [n]} \rangle$, with $n \in \mathbb{N}$, and where s_i is a state of $\mathcal{P}$ for each i.*

Remark 2 (Equivalence with other models). There are several ways to define ME-POMDPs. A first one would be to consider a single initial state but different transition functions $\delta_1, \ldots, \delta_n$. A second one is to consider different reward functions. A third one is to consider multiple initial states, or more generally initial beliefs. Those three models are introduced in [1], under the names *multi-environment*, *multi-objective*, and *adversarial belief POMDPs*, respectively. The same paper establishes that all of them are equivalent, hence we chose the technically simplest one, with multiple initial states (i.e., adversarial belief) and refer to it as ME-POMDPs for consistency with the literature of MEMDPs.

2.4 Problem statement

Given an MEPOMDP $\mathcal{M}$, we define the *value* of $\mathcal{M}$ as $v = \sup_{\sigma \in \mathsf{Pol}(\mathcal{P})} \min_{i \in [n]} \mathbb{E}_{s_i}^k(\sigma)$.

Problem 1 (Value problem)**.** *Given an MEPOMDP $\mathcal{M}$ and a horizon k, compute the value v.*

We also consider the approximate version of that problem, where we are given a quantity ε, and ask for any quantity in the interval $[v - \varepsilon, v + \varepsilon]$. For complexity purposes, we also consider the decision version of these problems.

Problem 2 (Threshold problem)**.** *Given an MEPOMDP $\mathcal{M}$, a horizon k, and a threshold $\lambda \in \mathbb{Q}$, do we have $v \geq \lambda$?*

The approximate threshold problem is obtained by considering only instances that come with the guarantee that the quantity λ is at distance at least ε from the value—or equivalently, by considering that both answers are acceptable on instances that do not satisfy this guarantee.

Problem encoding. For complexity results, all items in the instance are given with classical encodings, except for the horizon k, assumed to be given in unary. This choice is motivated by the fact that the case $n = 1$ was also studied under this hypothesis [14], and that to our knowledge, the complexity of that problem when k is given in binary remains unknown. In the rest of this paper, we assume that we are given an instance of this problem, whose total size is denoted by $\mathcal{N}$.

The need for randomised policies. While in POMDPs the value coincides with optimising over $\mathsf{DetPol}(\mathcal{P})$, for MEMDPs, randomised policies are more powerful: there exist MEPOMDPs where we have $v > \sup_{\sigma \in \mathsf{DetPol}(\mathcal{P})} \mathbb{E}^k_{s_i}(\sigma)$. This is already known for multi-environment MDPs [16].

Example 2. Consider the POMDP depicted by Figure 1, with horizon 1 and initial states s_1 and s_2. Remember that the double circles indicate a reward 1, and that the other states give reward 0. With a single initial state, a deterministic policy would be sufficient to maximise the expected payoff: action b from s_1, action a from s_2. However, if the initial vertex can be either s_1 or s_2, none of those policies is satisfying. On the other hand, randomising between actions a and b with probability 0.5 each guarantees expected payoff 0.5. Better: the policy that randomises uniformly between actions c and d guarantees an expected payoff 0.75, even though the deterministic policies that play deterministically either c or d are not optimal neither from s_1 nor from s_2.

In the next section, we define a few conceptual tool that we use to solve this problem.

3 Multi-belief, multi-expected payoff, and mixtures of policies

3.1 Multi-beliefs

The notion of belief is not sufficient to solve our problem. Indeed, with an adversarially chosen initial state, the agent must consider separately the n *environments* in which the initial state was $s_1, \ldots, s_n$, respectively; and for each of them, construct a different belief. When performing actions and seeing observations, the agent can now get information of two kinds: (i) in each environment i, she must at each step update her belief; but (ii) some observation sequences may also be incompatible with some initial states, in which case the corresponding environment must be eliminated. For example, in the POMDP depicted by Figure 1, if after performing action c, the agent sees the observation o_2, she knows that the initial state was s_1 and must no longer consider the environment 2.

Definition 4 (Multi-belief)**.** *A* multi-belief *in the MEPOMDP $\mathcal{M}$ is a tuple $\bar{\beta} = (\beta_i)_{i \in [n]}$, where each β_i is either a belief or the symbol $\bot$, such that all beliefs $\beta_i \neq \bot$ have the same observation.*

The equality $\beta_i = \bot$ means that the environment i is eliminated, i.e., that the agent learned that s_i was not the initial state. For convenience, we write $\bot(s) = \bot$ for every $s \in S$. The initial multi-belief is $\bar{\beta}^0$, defined by $\beta^0_i(s_i) = 1$ for each i. The update function is extended to multi-beliefs by:

$$\mathsf{update}(\bar{\beta}, a, o) = \left(\begin{cases} \bot & \text{if } \beta_i = \bot \text{ or } \sum_{s \in S} \sum_{\substack{t \in S \\ \mathsf{obs}(t) = o}} \beta_i(s) \cdot \delta(s,a)(t) = 0 \\ \mathsf{update}(\beta_i, a, o) & \text{otherwise} \end{cases} \right)_{i \in [n]}$$

3.2 Multi-expected payoff

Similarly, the notion of expected payoff must be replaced by that of *multi-expected payoff*, defined as tuples of real values. We also allow some coordinates to be set to $\bot$, for policies that are played starting from a multi-belief in which some environments have already been eliminated. For simplicity, we use the conventions $\bot \cdot 0 = 0$ and $\bot + x = \bot$ and $\bot \cdot x = \bot$ for every other x.

Definition 5 (Multi-expected payoff)**.** *A* multi-expected payoff *is a tuple in* $(\mathbb{R} \cup \{\bot\})^n$*. Given a multi-belief* $\bar{\beta}$*, policy* σ*, and horizon* ℓ*, their multi-expected payoff is* $\mathsf{mep}^{\ell}_{\bar{\beta}}(\sigma) = \left(\sum_s \beta_i(s)\mathbb{E}^{\ell}_s(\sigma)\right)_i$.

When $\ell = k$ is clear from context, we write $\mathsf{mep}_{\bar{\beta}}(\sigma)$ instead of $\mathsf{mep}^{\ell}_{\bar{\beta}}(\sigma)$. Problem 2 can now be rephrased as follows: in the POMDP $\mathcal{P}$, does there exists a policy σ such that $\mathsf{mep}_{\bar{\beta}^0}(\sigma) \in [\lambda, +\infty)^n$?

3.3 Mixtures of policies

Finally, we will often need to define policies as a randomised combination of several policies.

Definition 6 (Mixture of policies)**.** *Given a POMDP* $\mathcal{P}$*, some policies* $\sigma_1, \cdots, \sigma_\ell \in \mathsf{DetPol}(\mathcal{P})$*, and some coefficients* $\alpha_1, \cdots, \alpha_\ell \in [0, 1]$ *with* $\sum_{i \leq \ell} \alpha_i = 1$*, we denote by* $\sum_{i \leq \ell} \alpha_i \sigma_i$ *the policy* σ *that, for each* i*, follows the policy* σ_i *with probability* α_i*. The policy* σ *is then a* mixture *of* $\sigma_1, \ldots, \sigma_\ell$*. The* degree *of* σ *is the minimal* $\ell \in \mathbb{N}$ *such that* σ *is a mixture of* ℓ *deterministic policies.*

Remark 3. Every policy has degree at most $|A|^{|\mathcal{O}|^k}$.

Remark 4. Let $\sigma = \sum_{i=1}^{\ell} \alpha_i \sigma_i$ be a mixture of policies and $\bar{\beta}^0$ be an initial product belief. Then, we have that $\mathsf{mep}(\sigma, \bar{\beta}^0) = \sum_i \alpha_i \cdot \mathsf{mep}(\sigma_i, \bar{\beta}^0)$.

4 Complexity

In this section, we show that the threshold problem and its approximate version are PSPACE-complete. The lower bound is a result of [14], but the upper bound requires new techniques.

First, we note that it is sufficient to consider policies with a degree at most the number of initial states. This is a consequence of Carathéodory's theorem [11]: each policy is a mixture of deterministic policies, and its corresponding multi-expected payoff is a convex combination of their multi-expected payoffs. Since the multi-expected payoff of an arbitrary policy is a point in $\mathbb{R}^n$, Carathéodory's theorem states that said convex combination can be written using only $n + 1$ of the original points.

Lemma 1 (App. B)**.** *Let* $\mathcal{P}$ *be a POMDP, let* $\lambda \in \mathbb{Q}$*, and* $s_1, \ldots, s_n$ *be* n *initial states. Let us assume that there exists a policy* σ *such that from every initial state* s_i*, following the policy* σ*, the expected payoff is at least* λ*. Then, there exists a policy of degree at most* n *that offers the same guarantee.*

This result should convey the intuition of why value computation in MEPOMDPs is not fundamentally harder than in POMDPs: although deterministic strategies are not sufficient here, Lemma 1 shows that only a limited amount of randomisation is necessary.

From this result, a first naive algorithm can be proposed. By Lemma 1, to recognize positive instances of Problem 2, it suffices to guess n deterministic policies $\sigma_1, \ldots, \sigma_n$ and to check that they can be mixed into a satisfying policy, i.e., that we have $\mathsf{Conv}\{\bar{x}^1, \ldots, \bar{x}^n\} \cap [\lambda, 1]^n \neq \emptyset$. Every deterministic policy can be described with the list of possible observation sequences, and for each of them, the action that is performed. Such a description requires exponential space. Once those policies are guessed, computing the associated multi-expected payoffs amounts to computing the expected payoffs in n Markov chains, which can be done in time polynomial in the size of the Markov chains, i.e., exponential in the size of the original POMDP. Checking whether we have $\mathsf{Conv}\{\bar{x}^1, \ldots, \bar{x}^n\} \cap [\lambda, +\infty)^n \neq \emptyset$ can then be done in polynomial time. This algorithm shows therefore membership to the class NEXPTIME. To obtain a better upper bound, we note that we do not need the whole description of each deterministic policy, but just their multi-expected payoff, which we can represent using only polynomial space, as formalized in Lemma 2. Therefore, we can guess policies one action at a time, and propagate the multi-expected payoffs instead of storing the whole policy. Such an algorithm also requires the following lemma.

Lemma 2 (App. C)**.** *Let* $R = \max_{s \in S} |r(s)|$*. Let* $X = \{\mathsf{mep}_{\bar{\beta}^0}(\sigma) \mid \sigma \in \mathsf{DetPol}(\mathcal{P})\}$*. There exists a polynomial* π *and an integer* $\mathcal{C} \leq 2^{\pi(\mathcal{N})}$ *such that the set* $N = \{p/\mathcal{C} \mid p \in \{-\mathcal{C}kR, \ldots, \mathcal{C}kR\}\}$ *contains* X*. Thus,* N *can be represented using* $\pi(\mathcal{N}) + 2\mathcal{N} + 1$ *bits.*

Algorithm 1 A non-deterministic polynomial-space algorithm

procedure CONSTRUCTPOLICY($\mathcal{P}, (s_i)_{i\in[n]}, k$)
  $\bar{\beta}^0 \leftarrow \left(\mathbf{1}_{\{s=s_i\}}\right)_{i\in[n]}$
  for $i \in \{1, \dots, n\}$ **do**
    $\bar{x}^i \leftarrow$ CONSTRUCTDETPOLICY($\mathcal{P}, k, \bar{\beta}^0$)
  return $\mathsf{Conv}\{\bar{x}^1, \dots, \bar{x}^n\} \cap [\lambda, +\infty)^n \neq \emptyset$

procedure CONSTRUCTDETPOLICY($\mathcal{P}, \ell, \bar{\beta}$)
  if $\ell = 0$ **then**
    return $\left(\sum_{s\in S} \beta_i(s) r(s)\right)_{i\in[n]}$
  Guess $a \in A$ ▷ (1)
  for $o \in \mathcal{O}$ **do** ▷ (2)
    $\bar{\beta}^o \leftarrow \mathsf{update}(\bar{\beta}, a, o)$ ▷ (3)
    $\bar{x}^o \leftarrow$ CONSTRUCTDETPOLICY($\mathcal{P}, \ell - 1, \bar{\beta}^o$) ▷ (4)
  $\bar{x} \leftarrow \left(\sum_{s\in S} \beta_i(s) \left(r(s) + \sum_{o\in\mathcal{O}} \sum_{t\in S, \mathsf{obs}(t)=o} \delta(s,a)(t) \cdot \bar{x}_i^o\right)\right)_{i\in[n]}$
  return $\bar{x}$ ▷ (5)

We present now a non-deterministic polynomial-space algorithm to solve our problem, illustrated by Figure 2. The core idea is as follows: instead of guessing deterministic policies as a whole, we guess them action by action, in a depth-first fashion. Then, each time a subpolicy (all actions planned after a given observation sequence) has been entirely guessed, the corresponding multi-expected payoff is computed, and the subpolicy is erased. This algorithm is described precisely in Algorithm 1. The following result follows from this algorithm.

Lemma 3 (App. D)**.** *The threshold problem and its approximate version are in* PSPACE.

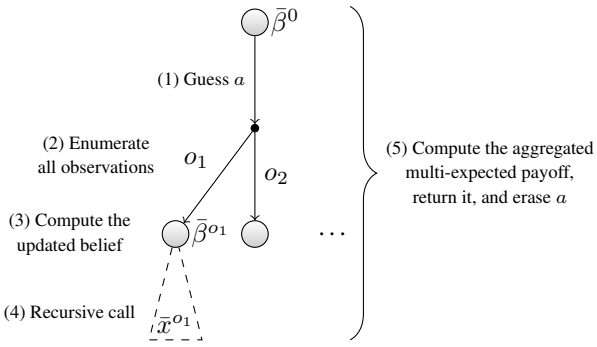


Figure 2: An illustration of Algorithm 1

Finally, the matching lower bound for the exact problem follows from [14] where it was shown for $n = 1$. We briefly show that the proof can be adapted to show the same hardness for the approximate version. We can then conclude on our main theoretical result.

Lemma 4 (App. A)**.** *The approximate threshold problem is* PSPACE*-hard, even when* $n = 1$.

Theorem 1. *The threshold problem and its approximate version are* PSPACE*-complete.*

5 Algorithms

We have proved Problem 2 to be PSPACE-complete by providing a non-deterministic algorithm. In this section, we present two deterministic algorithms, one that optimises space, and another one that optimizes efficiency in practice for tractable instances, even if it may use exponential time and space in the worst case.

5.1 A space-efficient deterministic algorithm

In this section, we assume an enumeration $\mathcal{O} = \{o_1, \dots, o_{|\mathcal{O}|}\}$. Algorithm 2 is a space-efficient determinisation of Algorithm 1. Its rationale is as follows. The MAIN procedure enumerates all possible multi-expected payoff values $\bar{X} = (\bar{x}^1, \dots, \bar{x}^n)$ achievable by n deterministic policies, mimicking the non-deterministic guess of points in CONSTRUCTPOLICY. The CHECKACHIEVABLEVALUE procedure checks whether the value $\bar{x}$ is achievable as a multi-expected payoff in the POMDP $\mathcal{P}$, from a multi-belief $\bar{\beta}$ and a horizon of ℓ steps. Here, achievable means that there exists a deter-

ministic policy σ such that $\mathbb{E}^k_{s_i}(\sigma) = \bar{x}$. The procedure CHECKACHIEVABLEVALUE, in line with CONSTRUCTDETPOLICY, checks whether there is a policy that achieves a multi-expected payoff $\bar{x}$ from the multi-belief $\bar{\beta}$ in ℓ steps by checking, for every pair $(a, o) \in A \times \mathcal{O}$ the multi-expected payoffs that can be achieved with a horizon $\ell - 1$ and a multi-belief $\bar{\beta}^o = \mathsf{update}(\bar{\beta}, a, o)$, and aggregating them according to the Bellman update (Eq. (2)). To do so, it constructs partial sums of Eq. 2 recursively, checking, for every $j \in \{1, \dots, |\mathcal{O}|\}$ and every possible value $\bar{x}_{\mathsf{rem}} \in X^n$, whether $\bar{x}_{\mathsf{rem}}$ can be obtained as the Bellman update partial sum from o_j onward, i.e., whether it can be achieved as a multi-expected payoff in a horizon of $\ell - 1$ steps from the updated multi-belief $\mathsf{update}(\bar{\beta}, a, o_j)$, and for those values where it is possible, whether there is a way to fill up the remaining $|\mathcal{O}| - j + 1$ terms in the partial sum to make it up to $\bar{x}_{\mathsf{rem}}$.[1]

Algorithm 2 A deterministic polynomial-space algorithm

procedure MAIN($\mathcal{P}, (s_i)_{i\in[n]}, k$)
  $\bar{\beta}^0 \leftarrow \left(\mathbf{1}_{\{s=s_i\}}\right)_{i\in[n]}$
  for $\bar{X} = (\bar{x}^1, \dots, \bar{x}^n) \in N^{n^2}$ **do**
    $\mathsf{isXpoint} \leftarrow \forall i \in [n], \text{CHECKACHIEVABLEVALUE}(\mathcal{P}, k, \bar{\beta}^0, \bar{x}^i)$
    if $\mathsf{isXpoint} \wedge (\mathsf{Conv}\{\bar{x}^1, \dots, \bar{x}^n\} \cap [\lambda, +\infty)^n \neq \emptyset)$ **then return** True
  return False

procedure CHECKACHIEVABLEVALUE($\mathcal{P}, \ell, \bar{\beta}, \bar{x}$)
  if $\ell = 0$ **then return** $\bar{x} = \left(\sum_{s\in S} r(s)\beta_i(s)\right)_{i\in[n]}$
  for $a \in A$ **do**
    if PARTIALSUM($1, \ell, \bar{\beta}, a, \bar{x}$) **then return** True
  return False

procedure PARTIALSUM($j, \ell, \bar{\beta}, a, \bar{x}_{\mathsf{rem}}$)
  if $j = |\mathcal{O}| + 1$ **then return** $\bar{x}_{\mathsf{rem}} = \left(\sum_{s\in S} r(s)\beta_i(s)\right)_{i\in[n]}$
  for $\bar{x} \in (N \cup \{\perp\})^n$ **do**
    if CHECKACHIEVABLEVALUE($\mathcal{P}, \ell - 1, \mathsf{update}(\bar{\beta}, a, o_j), \bar{x}$) **then**
      $\bar{y} \leftarrow \left(\sum_{s\in S} \sum_{t\in S,\, \mathsf{obs}(t)=o_j} \beta_i(s) \cdot \delta(s, a)(t) \cdot x_i\right)_{i\in[n]}$
      if PARTIALSUM($j + 1, \ell, \bar{\beta}, a, \bar{x}_{\mathsf{rem}} - \bar{y}$) **then return** True
  return False

Theorem 2 (App. E)**.** *Algorithm 2 decides Problem 2 and uses $O(|S| \cdot |\mathcal{O}| \cdot n \cdot k \cdot \pi(\mathcal{N}))$ space.*

Remark 5. The space requirement of Algorithm 1 is $O(|S| \cdot nk\pi(\mathcal{N}))$, so the space requirement of the standard determinization using Savitch's theorem [18] would be $O(|S|^2 \cdot n^2k^2\pi(\mathcal{N})^2)$, significantly larger than that of Algorithm 2.

Algorithm 2 is thus efficient in terms of space. However, this comes at the cost of several imbricated loops over all values in N, which results in practice in unfeasibly large computation times for most instances. For that reason, we now propose a second method that uses exponential space in the worst case, but is much more time-efficient in practice.

5.2 Practically efficient algorithm

Algorithm 3 implements a different approach: computing the set of multi-expected payoffs bottom-up for increasing horizons, and pruning, at each step, the points that are dominated by other points in the set, corresponding to families of policies that do not need to be explored.

Consider the set $X^\ell(\bar{\beta}) = \{\mathsf{mep}^\ell_{\bar{\beta}}(\sigma) \mid \sigma \in \mathsf{DetPol}(\mathcal{P})\}$. We can construct X^ℓ recursively as:

$$X^\ell(\bar{\beta}) = \left\{ \left(\sum_{s\in S} \beta_i(s) \left(r(s) + \sum_{o\in\mathcal{O}} \sum_{\substack{t\in S,\\ \mathsf{obs}(t)=o}} \delta(s, a)(t) x^o_i \right) \right)_i \;\middle|\; \begin{array}{c} \bar{x}^o \in X^{\ell-1}(\bar{\beta}^{a,o}), \\ \bar{\beta}^{a,o} = \mathsf{update}(\bar{\beta}, a, o), \\ a \in A \end{array} \right\} \quad (1)$$

[1] Note that when $\bar{x}_{\mathsf{rem}}$ is updated, for each i such that $x_i = \perp$, we have $y_i = 0$ and the coordinate $x_{\mathsf{rem}i}$ is left unchanged.

As a result of Lemma 1, solving Problem 2 is equivalent to checking whether $\mathsf{Conv}(X^k(\bar{\beta}^0)) \cap [\lambda, \infty)^n \neq \emptyset$. Therefore, it is sufficient to keep only a representation of the convex hull of $X^k(\bar{\beta}^0)$. Since we are interested in the intersection with $[\lambda, \infty)^n$, we can also do without any point convexly dominated by points in $X^\ell(\bar{\beta}^0)$. Also, when building $X^\ell(\bar{\beta})$ from corresponding $X^{\ell-1}(\bar{\beta}')$ following Eq. 1, we can prune all dominated points in an $\ell - 1$ depth before building the set for depth ℓ.

Algorithm 3 An efficient algorithm

procedure BUILDFRONTIER($\ell, \bar{\beta}$)
  if $\ell = 0$ **then return** $\{\left(\sum_{s \in S} r(s)\beta_i(s)\right)_{i \in [n]}\}$
  $V \leftarrow \emptyset$
  for $a \in A$ **do**
    for $o \in \mathcal{O}$ **do**
      $V_o \leftarrow$ BUILDFRONTIER($\ell - 1, \mathsf{update}(\bar{\beta}, a, o)$)
    for $(\bar{x}^o)_{o \in \mathcal{O}} \in \prod_{o \in \mathcal{O}} V_o$ **do**
      $\bar{y} \leftarrow \left(\sum_{s \in S} \beta_i(s) \left(r(s) + \sum_{o \in \mathcal{O}} \sum_{t \in S, \mathsf{obs}(t) = o} \delta(s, a)(t) \cdot \bar{x}_i^o\right)\right)_{i \in [n]}$
      $V \leftarrow$ PRUNE($V, \bar{y}$)
  return V

procedure PRUNE($V, \bar{y}$)
  for $\bar{x} \in V$ **do**
    if $\bar{y}$ is dominated by $\bar{x}$ **then return** V
    if $\bar{x}$ is dominated by $\bar{y}$ **then** $V \leftarrow V \setminus \{\bar{x}\}$
  return $V \cup \{\bar{y}\}$

6 Experimental Evaluation

In our experimental evaluation, we study the practical efficiency and scalability of our efficient method (Algorithm 3) and compare it to the state of the art. We tackle two research questions.

(RQ1) Baseline comparison. How efficient is our algorithm compared to the state of the art [1]?

(RQ2) Scalability. How does the computational cost of our method scale with different parameters, such as model size and horizon?

In both the tables and the figures, we report $|S|$ to be only the number of reachable states within the set horizon. All the experiments were executed using 16GB of RAM and a processor Intel Xeon CPU E5-2680 v3 2.50GHz. Timeout is set at $3600s$.

Benchmarks. We consider three benchmarks in our experiments. The first one is the extension of *Rock Sample* to ME-POMDPs introduced in [1]. For the second and third benchmarks, we extend two classical problems from [2] to ME-POMDPs: the *Robot Navigation* and the *Identification (Friend or Foe)* problem. Instances of the Rock Sample problem are denoted by $\mathsf{RS}_{m,g,t}$, where m is the size of the grid, g is the number of good rocks, and t is the total number of rocks. Instances of the Robot Navigation problem are denoted by $\mathsf{RN}_{\text{MAP},d}$, where MAP indicates the map or layout, and d indicates the maximum distance of initial states to a goal. Instances of the Identification problem are denoted by $\mathsf{IFF}_{d_1,d_2,v_1,v_2}$, where d_i and v_i indicate the distance and visibility, respectively, of the two initial foe states. The full description of the three benchmarks is in Appendix F.

Comparison to baseline. To address RQ1, we compare our method with the one proposed in [1], using their own tool. Since [1] is an approximate algorithm, we set it to have no approximation error, to match our exact algorithm. In the appendix, Sec. F.4.1, we show that relaxing the approximation error of [1] has no significant effect in runtime. We report the results on a representative subset of the three benchmarks in Tables 1 and 2. Observe that our method is in general faster than [1], and manages to successfully solve problems with up to 574 states for horizon up to $k = 5$.

Table 1: Rock Sampling.

Model	$\lvert S\rvert$	$\lvert A\rvert$	$\lvert\mathcal{O}\rvert$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RS}_{3,1,2}$	26	7	3	2	4	1.736	0.2335
$\mathsf{RS}_{3,1,3}$	34	8	3	3	3	2.434	0.05812
$\mathsf{RS}_{4,1,2}$	32	7	3	2	7	TO	154.2
$\mathsf{RS}_{6,1,2}$	35	7	3	2	6	43.17	18.68
$\mathsf{RS}_{3,2,7}$	246	12	3	21	5	TO	459.9
$\mathsf{RS}_{3,4,7}$	574	12	3	35	5	TO	783.8
$\mathsf{RS}_{3,6,7}$	192	12	3	7	6	TO	3477

Table 2: Robot Navigation and Identification.

Model	$\lvert S\rvert$	$\lvert A\rvert$	$\lvert\mathcal{O}\rvert$	n	k	[1] (s)	Ours (s)
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	2	304.2	0.0223
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	6	TO	914.1
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	5	TO	70.02
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	6	TO	902.1
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	6	TO	271
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	1	TO	0.1388
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	3	213.5	0.6066

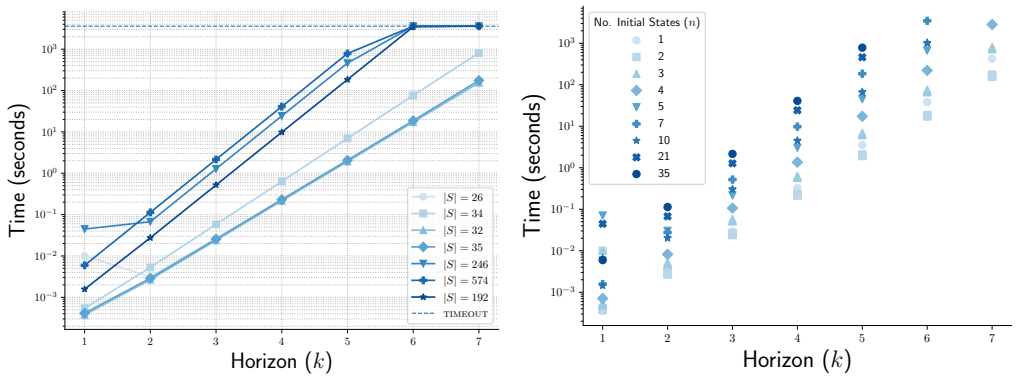


Figure 3: Horizon vs. Computation time in the RockSample benchmark, for different number of states (left), and different number of initial states (right).

Scalability. To investigate RQ2, we compare computation times in terms of horizon, number of states and number of initial states. In Fig. 3 (left), we show computation time for increasing horizons for the models in Table 1. In Fig. 3 (right), we report a scatter plot of computation time and horizon for the whole RockSample benchmark (see Appendix F for description), color-coded by number of initial states. While there is a general trend that computation time increases slightly both with number of states and number of initial states, the strongest predictor of computation time (and thus, the most salient challenge for scalability) is horizon (k).

7 Conclusion

We have considered the value computation problem in MEPOMDPs with finite-horizon objectives, providing tight complexity bounds, a space efficient algorithm, and a time efficient algorithm that proves to be empirically more performant than previously existing tools.

Limitations. From a theoretical perspective, our methods present a usual tradeoff between theoretical and practical performance: Algorithm 2, while asymptotically optimal, is unfeasibly slow in practice; and Algorithm 3, while efficient in practice, uses exponential space in the worst case. From a practical perspective, the main challenge for Algorithm 3 is scalability. In particular, the dependence on the horizon is quite strong, with the solved instances with longest horizon at $k = 7$.

Future work. Along with finite-horizon objectives, one can also consider discounted-sum objectives. The exact problem for discounted-sum objectives is already undecidable for POMDPs [12], but its approximate version can be reduced to finite-horizon objectives: our algorithm can therefore easily be applied (up to adding a discount factor) to solve that problem as well. We leave the theoretical and practical complexities of this problem as future work.

Acknowledgments and Disclosure of Funding

This work was supported by the European Research Council under Grant No.: ERC-2020-AdG 101020093 (VAMOS), by ERC CoG 863818 (ForMSMArt), and Austrian Science Fund (FWF) 10.55776/COE12. This work was also supported by the ISTA Responsible AI Program, made possible through the support of Garrett Camp and the Camp Foundation.

References

[1] Eline Bovy, Caleb Probine, Marnix Suilen, Ufuk Topcu, and Nils Jansen. Multi-environment POMDPs: Discrete model uncertainty under partial observability. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[2] Anthony Rocco Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Brown University, USA, 1998.

[3] Krishnendu Chatterjee, Martin Chmelík, and Mathieu Tracol. What is decidable about partially observable Markov decision processes with ω-regular objectives. *Journal of Computer and System Sciences*, 82(5):878–911, 2016. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2016.02.009. URL https://www.sciencedirect.com/science/article/pii/S0022000016000246.

[4] Krishnendu Chatterjee, Martin Chmelík, Deep Karkhanis, Petr Novotný, and Amélie Royer. Multiple-environment Markov decision processes: Efficient analysis and applications. In *Proceedings of the 30th International Conference on Automated Planning and Scheduling (ICAPS)*, pages 48–56. AAAI Press, 2020.

[5] Krishnendu Chatterjee, Laurent Doyen, Jean-François Raskin, and Ocan Sankur. The value problem for multiple-environment MDPs with parity objective. In Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis, editors, *52nd International Colloquium on Automata, Languages, and Programming, ICALP 2025, Aarhus, Denmark, July 8-11, 2025*, LIPIcs, pages 150:1–150:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. doi: 10.4230/LIPICS.ICALP.2025.150. URL https://doi.org/10.4230/LIPIcs.ICALP.2025.150.

[6] Maris FL Galesloot, Roman Andriushchenko, Milan Češka, Sebastian Junges, and Nils Jansen. Robust finite-memory policy gradients for hidden-model pomdps. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2025.

[7] Eric Hansen, Daniel Bernstein, and Shlomo Zilberstein. Dynamic programming for partially observable stochastic games. *AAAI Workshop - Technical Report*, 06 2004.

[8] Milos Hauskrecht and Hamish S. F. Fraser. Planning treatment of ischemic heart disease with partially observable Markov decision processes. *Artif. Intell. Medicine*, 18(3):221–244, 2000. doi: 10.1016/S0933-3657(99)00042-1. URL https://doi.org/10.1016/S0933-3657(99)00042-1.

[9] Merlijn Krale, Eline M Bovy, Maris FL Galesloot, Thiago D Simão, and Nils Jansen. On evaluating policies for robust POMDPs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[10] H. Kuhn. Extensive games and the problem of information. In H. Kuhn and A. Tucker, editors, *Contributions to the Theory of Games*, pages 193–216. Princeton University Press, 1953.

[11] I. E. Leonard and J. E. Lewis. *Geometry of Convex Sets*. Wiley-Blackwell, Hoboken, New Jersey, United States, 1st edition, 2016.

[12] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1):5–34, 2003. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(02)00378-8. URL https://www.sciencedirect.com/science/article/pii/S0004370202003788. Planning with Uncertainty and Incomplete Information.

[13] Takayuki Osogami. Robust partially observable Markov decision process. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 106–115, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/osogami15.html.

[14] Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987. ISSN 0364765X, 15265471. URL http://www.jstor.org/stable/3689975.

[15] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York, 1994. ISBN 978-0-471-72782-8. doi: 10.1002/9780470316887.

[16] Jean-François Raskin and Ocan Sankur. Multiple-environment Markov decision processes. In Venkatesh Raman and S. P. Suresh, editors, *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014, New Delhi, India, December 15-17, 2014*, LIPIcs, pages 531–543. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014. doi: 10.4230/LIPICS.FSTTCS.2014.531. URL https://doi.org/10.4230/LIPIcs.FSTTCS.2014.531.

[17] Nicholas Roy, Geoffrey J. Gordon, and Sebastian Thrun. Finding approximate POMDP solutions through belief compression. *J. Artif. Intell. Res.*, 23:1–40, 2005. doi: 10.1613/JAIR.1496. URL https://doi.org/10.1613/jair.1496.

[18] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. ISSN 0022-0000. doi: https://doi.org/10.1016/S0022-0000(70)80006-X.

[19] Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 520–527, 2004.

[20] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, Massachusetts, 2nd edition, 2018. ISBN 978-0-262-03924-6. URL http://incompleteideas.net/book/the-book-2nd.html.

[21] Marck van der Vegt, Nils Jansen, and Sebastian Junges. Robust almost-sure reachability in multi-environment MDPs. In Sriram Sankaranarayanan and Natasha Sharygina, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 508–526, Cham, 2023. Springer Nature Switzerland. ISBN 978-3-031-30823-9.

A Proof of Lemma 4

Lemma 4 (App. A)**.** *The approximate threshold problem is* PSPACE*-hard, even when* $n = 1$.

Proof. The proof in [14] goes by reducing the problem QBF to Problem 2 in polynomial time. From a formula of the form $\varphi = \exists x_1 \forall x_2 \dots \forall x_p \psi$, where ψ is a quantifier-free formula in disjunctive normal form, it constructs a POMDP in which getting expected payoff at least 0 is possible if and only if the formula φ is true. A careful reading of this construction also shows that if φ is not true, then every policy gives expected payoff not only negative, but at most $-(1/2)^{p/2}$. By defining $\varepsilon = (1/2)^{p/2}$ as equal to this quantity, we have a polynomial-time reduction to the approximate threshold problem. □

B Proof of Lemma 1

Lemma 1 (App. B)**.** *Let $\mathcal{P}$ be a POMDP, let $\lambda \in \mathbb{Q}$, and $s_1, \dots, s_n$ be n initial states. Let us assume that there exists a policy σ such that from every initial state s_i, following the policy σ, the expected payoff is at least λ. Then, there exists a policy of degree at most n that offers the same guarantee.*

Proof. By Remark 3, the policy σ is a mixture of a finite number of deterministic policies. Using Remark 4, we can therefore write:

$$\mathsf{mep}(\sigma) \in \mathsf{Conv}\left(\mathsf{mep}\left(\mathsf{DetPol}(\mathcal{P})\right)\right).$$

Now, by Carathéodory's theorem, there exist $n+1$ deterministic policies $\sigma'_1, \ldots, \sigma'_{n+1} \in \mathsf{DetPol}(\mathcal{P})$ and coefficients $\beta_1, \ldots, \beta_{n+1}$ such that $\sum_{i=1}^{n+1} \beta_i = 1$ and $\sum_{i=1}^{n+1} \beta_i \mathsf{mep}(\sigma'_i) = \mathsf{mep}(\sigma)$.

Consider now the polyhedron $X = \mathsf{Conv}\{\mathsf{mep}(\sigma'_1), \ldots, \mathsf{mep}(\sigma'_{n+1})\}$. The mapping $f : \bar{x} \mapsto \min_i x_i$ finds its maximum on a facet of X, i.e., the convex hull of at most n of its vertices. We can therefore choose n deterministic policies $\sigma_1, \ldots, \sigma_n \in \{\sigma'_1, \ldots, \sigma'_{n+1}\}$, and n coefficients $\alpha_1, \ldots, \alpha_n$ with $\sum_i \alpha_i = 1$, such that we have $\min_i \sum_{i=1}^{n} \alpha_i \mathsf{mep}(\sigma_i) \geq \lambda$. By Remark 4, the policy $\sigma^\star = \sum_i \alpha_i \sigma_i$ offers then the desired guarantees. □

C Proof of Lemma 2

Lemma 2 (App. C)**.** *Let $R = \max_{s \in S} |r(s)|$. Let $X = \{\mathsf{mep}_{\bar{\beta}^0}(\sigma) \mid \sigma \in \mathsf{DetPol}(\mathcal{P})\}$. There exists a polynomial π and an integer $\mathcal{C} \leq 2^{\pi(\mathcal{N})}$ such that the set $N = \{p/\mathcal{C} \mid p \in \{-\mathcal{C}kR, \ldots, \mathcal{C}kR\}\}$ contains X. Thus, N can be represented using $\pi(\mathcal{N}) + 2\mathcal{N} + 1$ bits.*

Proof. For each triple $(s, a, s') \in S \times A \times S$ with a non-zero transition probability, consider the standard representation of the transition probability $\delta(s, a)[s'] = p_{s,a,s'}/q_{s,a,s'}$, where $p_{s,a,s'}$ and $q_{s,a,s'}$ are positive coprime integers. Let $Q = \{q_{s,a,s'} \mid (s, a, s') \in S \times A \times S\}$, $M = \max Q$ and C be the least common multiple of Q. Since the size of the input is $\mathcal{N}$, M has at most $\mathcal{N}$ bits, so it is at most $2^{\mathcal{N}}$. Note that

$$C \leq \prod_{q \in Q} q \leq M^{|S|^2 \cdot |A|} \leq 2^{\mathcal{N} \cdot |S|^2 \cdot |A|} \leq 2^{\mathcal{N}^4}.$$

Every transition probability can be rewritten as $\delta(s, a)(s') = \tilde{p}_{s,a,s'}/C$, for suitable integers $\tilde{p}_{s,a,s'} \leq C$.

Consider a deterministic policy $\sigma \in \mathsf{DetPol}(P)$, a belief β, and a trajectory $\tau = (s_0, a_0, \ldots, s_k) \in \mathsf{Traj}_k(\mathcal{P})$. The probability of τ is $\mathbb{P}^\sigma_\beta(\tau) = \beta(s_0) \prod_{i=0}^{k-1} \delta(s_i, a_i)(s_{i+1})$ if τ is compatible with σ, and 0 otherwise. If $\beta = \beta^0_i$ for some $i \in [n]$, then $\beta(s_0) \in \{0, 1\}$, and since all factors can be expressed with denominator C, the probability of τ is of the form

$$\mathbb{P}^\sigma_\beta(\tau) = \frac{m_\tau}{C^k},$$

for some integer $m_\tau \leq C^k$, and the associated payoff can be expressed as $\mathsf{payoff}(\tau)\frac{m_\tau}{C^k}$. Since all rewards are integers, $\mathsf{payoff}(\tau)$ is an integer, $-kR \leq \mathsf{payoff}(\tau) \leq kR$.

Let $\mathcal{C} = C^k$. The expected payoff $\mathbb{E}^k_\beta(\sigma)$ is the sum of probabilities of trajectories times their payoff, so it can be expressed as a rational $p/\mathcal{C}$, where p is an integer between $-kRC^k$ and kRC^k. Since $\mathcal{C} \leq 2^{k \cdot \mathcal{N}^4} \leq 2^{\mathcal{N}^5}$, and $kR \leq 2^{2\mathcal{N}}$ we have that $X \subseteq N$ with $\pi(\mathcal{N}) = \mathcal{N}^5$, and all numbers in N are expressed by an integer in $[0, \mathcal{C}]$, and integer in $[0, kR]$, and a sign, for a total of $\pi(\mathcal{N}) + 2\mathcal{N} + 1$. □

D Proof of Lemma 3

Lemma 3 (App. D)**.** *The threshold problem and its approximate version are in* PSPACE.

Proof. We show that Algorithm 1 is correct, and that it uses only polynomial memory.

If we have a positive instance of Problem 2, then by Lemma 1, there exists a policy σ of degree at most n such that $\mathsf{mep}(\sigma) \in [\lambda, +\infty)^n$; in other words, there exist n deterministic policies $\sigma_1, \ldots, \sigma_n$ such that $\mathsf{Conv}\{\mathsf{mep}(\sigma_1), \ldots, \mathsf{mep}(\sigma_n)\} \cap [\lambda, +\infty)^n \neq \emptyset$. Consider the run of Algorithm 1 where, in the main procedure, the ith call to the procedure CONSTRUCTDETPOLICY constructs the policy σ_i, in the following sense. Each recursive call corresponds to a given observation sequence ω: consider that the action guessed is then $a = \sigma(\omega)$. Then, in the main procedure, the tuples $\bar{x}_1, \ldots, \bar{x}_n$ that are obtained

are such that $\bar{x}_i = \mathsf{mep}(\sigma_i)$ for each i. Since we have $\mathsf{Conv}\{\mathsf{mep}(\sigma_1), \ldots, \mathsf{mep}(\sigma_n)\} \cap [\lambda, +\infty)^n \neq \emptyset$, that run of the algorithm returns True.

Conversely, if there is a run of Algorithm 1 that returns True, then the actions guessed along that run define n deterministic policies $\sigma_1, \ldots, \sigma_n$ such that we have $\mathsf{Conv}\{\mathsf{mep}(\sigma_1), \ldots, \mathsf{mep}(\sigma_n)\} \cap [\lambda, +\infty)^n \neq \emptyset$. As a consequence, we have a positive instance of Problem 2.

Finally, this algorithm uses polynomial space. Indeed, the size of the recursion stack is k (remember that k is given in unary), and each recursive call uses a polynomial amount of information: the POMDP $\mathcal{P}$ and the integer n, an integer $\ell \leq k$, and a multi-belief $\bar{\beta}$ which grows linearly with ℓ.

We therefore have a non-deterministic polynomial-space algorithm that decides Problem 2, and consequently also its approximate version. Since NPSPACE = PSPACE, the conclusion follows. □

E Proof of Theorem 2

Theorem 2 (App. E)**.** *Algorithm 2 decides Problem 2 and uses $O(|S| \cdot |\mathcal{O}| \cdot n \cdot k \cdot \pi(\mathcal{N}))$ space.*

Proof. The correctness of Algorithm 2 follows from the correctness of Algorithm 1, as it uses the same approach, enumerating multi-expected payoffs instead of guessing them.

The MAIN procedure enumerates all possible multi-expected payoff values $\bar{X} = (\bar{x}^1, \ldots, \bar{x}^n)$ achievable by n deterministic policies, and returns true whenever one is found that is achievable by the POMDP (i.e., it corresponds with a list of n deterministic strategies), and it satisfies the condition of Lemma 1. The CHECKACHIEVABLEVALUE procedure checks whether the value $\bar{x}$ is achievable as a multi-expected payoff in the POMDP $\mathcal{P}$, from a multi-belief $\bar{\beta}$ and a horizon of ℓ steps. Here, achievable means that there exists a deterministic strategy σ such that $\mathbb{E}^k_{s_i}(\sigma) = \bar{x}$. A key difference between with respect to Algorithm 1 is that for every action, Algorithm 1 can build $\bar{x}^o$ by guessing a policy in the next recursive level, while Algorithm 2 cannot use non-determinism, so it has to enumerate all possible values of $\bar{x}^o$ obtained by following the recursion. Consider the update in line 14 of Algorithm 1, where we enumerate $\mathcal{O} = \{o_1, \ldots, o_{|\mathcal{O}|}\}$:

$$\bar{x} \leftarrow \sum_{s \in S} \left(r(s)\beta_i(s) + \sum_{j=1}^{|\mathcal{O}|} \sum_{t \in S,\, \mathsf{obs}(t) = o_j} \beta_i(s) \cdot \delta(s,a)(t) \cdot x_i^{o_j} \right)_{i \in [n]}. \tag{2}$$

The procedure CHECKACHIEVABLEVALUE, in line with CONSTRUCTDETPOLICY, checks whether there is a strategy that achieves a multi-expected payoff $\bar{x}$ from the multi-belief $\bar{\beta}$ in ℓ steps by checking, for every pair $(a, o) \in A \times \mathcal{O}$ the multi-expected payoffs that can be achieved with a horizon $\ell - 1$ and a multi-belief $\bar{\beta}^o = \mathsf{update}(\bar{\beta}, a, o)$, and aggregating them according to the Bellman update (Eq. (2)). To do so, it constructs partial sums of Eq. 2 recursively. The PARTIALSUM$(j, \ell, \bar{\beta}, a, \bar{x}_{\mathsf{rem}})$ procedure checks whether there is a strategy that starts with action a and such that

$$\bar{x}_{\mathsf{rem}} = \left(\sum_{j'=j}^{|\mathcal{O}|} \sum_{s \in S} r(s)\beta_i(s) + \sum_{t \in S,\, \mathsf{obs}(t) = o_{j'}} \beta_i(s) \cdot \delta(s,a)(t) \cdot x_i^{o_{j'}} \right)_{i \in [n]}.$$

To do so, it checks, for every possible value in N, whether it can be achieved as a multi-expected payoff in a horizon of $\ell - 1$ steps from the updated multi-belief $\mathsf{update}(\bar{\beta}, a, o_j)$, and for those values where it is possible, whether there is a way to fill up the remaining $|\mathcal{O}| - j + 1$ terms in the partial sum to make it up to $\bar{x}_{\mathsf{rem}}$.

To check the space requirements, we need to check the space used by each procedure, and the size of the recursion stack. Let $D = \pi(\mathcal{N})$ (recall Lemma 2).

First, observe that the main procedure just requires storing n multi-expected payoff points, plus a high-level call to the CHECKACHIEVABLEVALUE procedure.

Second, consider the space requirement for a single frame of both recursive procedures. In a single frame of the CHECKACHIEVABLEVALUE procedure, the space requirement is that required to store the current multi-belief $\bar{\beta}$ ($|S| \cdot nD$) and the current multi-expected payoff value $\bar{x}$ ($n^2 D \leq |S| \cdot nD$), plus the space required for a high-level call to PARTIALSUM. In a single frame of PARTIALSUM, the space requirement is the same as for CHECKACHIEVABLEVALUE: a multibelief and a multi-expected payoff.

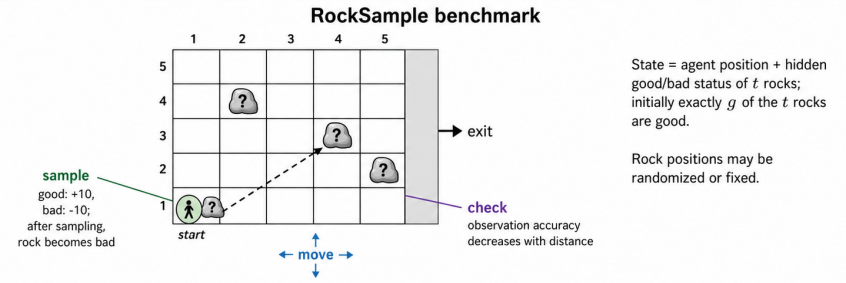


Figure 4: RockSample problem.

Third, consider the size of the recursion stack. If we consider the index (ℓ, j) shared between the procedures CHECKACHIEVABLEVALUE and PARTIALSUM (considering that CHECKACHIEVABLEVALUE has always $j = 0$), at every recursive call, either j increases with a call to PARTIALSUM (lines 15 and 29), keeping ℓ fixed, or ℓ decreases with a call to CHECKACHIEVABLEVALUE (line 27), resetting j to 0. Therefore, the recursion follows all the values of (ℓ, j) in lexicographical order (descending for ℓ and ascending for j), so the size of the recursion stack is always at most $|\mathcal{O}| \cdot k$.

Finally, the space requirement is the space requirement of each frame, times the size of the recursion stack, plus the space required for the main procedure: $|\mathcal{O}| \cdot k|S| \cdot n \cdot D + n^2 \cdot D$. □

F Experimental Details

In this appendix, we provide further details on our experimental evaluation.

F.1 Algorithm choice

As noted in Sec. 5, Algorithm 2 is very inefficient in terms of computation time, so we only consider Algorithm 3 for our empirical evaluation.

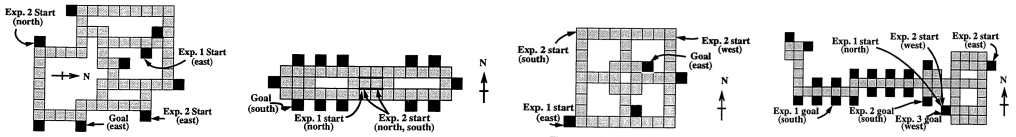
As for the comparison with [1], a feature in their algorithm is that the user can specify a *gap* or *tolerance*, explicitly indicating how precise the approximation of the actual result is allowed to be. In our experience with [1]'s tool, making this gap more or less restrictive had almost no effect in practice, both in the obtained value of the MEPOMDP, as well as in the computation time required. We explain this phenomenon as a consequence of [1] being optimized for the infinite horizon discounted problem, where, according to the results presented in [1], the gap does play a significant role.

F.2 Benchmark description

As benchmarks, we used modified versions of standard problems in the POMDP literature: RockSample, introduced in [19], with the modifications in [1], and Robot Navigation and Aircraft Identification, both introduced in [2]. For consistency with the sources of our models, we consider POMDPs with stochasticity in the observation function, that is, $\mathsf{obs}\colon S \times A \to \mathcal{D}(\mathcal{O})$. As mentioned in Remark 1, these models are equivalent, in the sense that one can systematically build a POMDP with deterministic observation from one with stochastic observation, that has the same value for any horizon.

F.2.1 Rock Sample

The RockSample problem was introduced in [19], and extended to the ME-POMDP setting in [1]. An instance consists of an $m \times m$ grid with t rocks at known positions. The agent starts in the bottom-left cell, moves in the four cardinal directions, can exit through the right boundary, and may sample a rock when located on its cell. Rock qualities are hidden: initially exactly g of the t rocks are good and the remaining $t - g$ rocks are bad. Sampling a good rock gives reward $+10$ and changes that rock to bad; sampling a bad rock gives reward -10. For each rock, a check action returns a noisy good/bad observation whose correctness decreases with the distance between the agent and



(a) CIT (281) (b) MIT (201) (c) PENTAGON (209) (d) SUNYSB (297)

Figure 5: Robot Navigation Maps. NAME (No. states). Figures from [2, Figures 6.1–6.4].

the queried rock. We denote instances of the problem as $\mathsf{RS}_{m,g,t}$. Note that, in our description, we use the instances corresponding to $\mathsf{RS}^{c}_{m,g,t}$ in [1], that is, with fixed rock positions. We illustrate the problem in Figure 4. The initial states are those states in which the agent is located in the bottom-left corner and g out of t rocks are good. The multiple initial states, i.e., the position of the rocks, are sampled in the same way as [1].

F.2.2 Robot navigation

The Robot Navigation problem [2, Appendix H.5] describes a robot moving through a room, that has to reach different goals. We use the CIT, MIT, PENTAGON, and SUNYSB robot-navigation maps, illustrated in Figure 5. Each map is a discretized indoor environment. A nonterminal state is a pair $s = (x, \theta)$, where x is a traversable map location and θ is one of the four cardinal orientations. The source construction also includes an absorbing terminal state, entered after the robot declares that it is at the goal. The concrete four-action instances use $A_{\mathrm{nav}} = \{\mathsf{forward}, \mathsf{left}, \mathsf{right}, \mathsf{declare}\}$. The first three actions are noisy motion actions, and declare moves the process to the absorbing terminal state. The reward is zero except for declare: declaring at the goal gives reward $+1$, while declaring away from the goal gives reward -1.

The motion model is specified by primitive outcomes. Let N denote no movement, F a one-cell forward move, L a 90° left turn, R a 90° right turn, and A absorption. The noisy outcomes are

Action	Primitive outcome distribution		
forward	$N : 0.11,$	$F : 0.88,$	$FF : 0.01$
left	$N : 0.05,$	$L : 0.90,$	$LL : 0.05$
right	$N : 0.05,$	$R : 0.90,$	$RR : 0.05$
declare	$A : 1.00.$		

For each nonterminal state s, let

$$\sigma(s) = (\sigma_{\mathrm{front}}(s), \sigma_{\mathrm{left}}(s), \sigma_{\mathrm{right}}(s)) \in \{\mathsf{open}, \mathsf{wall}, \mathsf{doorway}\}^3$$

be the true local observation in the directions in front of the robot, to its left, and to its right. We replace the source observation table by the alphabet

$$\mathcal{O}_{\mathrm{nav}} = \{\mathsf{open}, \mathsf{wall}, \mathsf{doorway}\}^3 \cup \{(\mathsf{undet}, \mathsf{undet}, \mathsf{undet})\},$$

so $|\mathcal{O}_{\mathrm{nav}}| = 28$. Observations are independent of the action, and depend only on how many sides of the robot have open space, wall, or doorway. Let

$$\alpha_{\mathrm{ow}}(s) = \#\{i \in \{\mathrm{front}, \mathrm{left}, \mathrm{right}\} : \sigma_i(s) \in \{\mathsf{open}, \mathsf{wall}\}\},$$

and let $\alpha_{\mathrm{door}}(s) = 3 - \alpha_{\mathrm{ow}}(s)$. The probability of observing the correct local signature is

$$p_{\mathrm{corr}}(s) = 0.9^{\alpha_{\mathrm{ow}}(s)} 0.7^{\alpha_{\mathrm{door}}(s)}.$$

Thus, for every action a, the probability to observe o is $p_{\mathrm{corr}}(s)$ if $o = \sigma(s)$, $1 - p_{\mathrm{corr}}(s)$ if $o = (\mathsf{undet}, \mathsf{undet}, \mathsf{undet})$, and 0 otherwise.

For each map we choose a pair of initial states that are at a distance at most d to the goal and maximize the observational diversity. Let G_E be the set of goal states for map E, and let d_E be shortest-path distance in the nonterminal state graph induced by positive-probability motion transitions, with the declare transition excluded. The candidate set is

$$C_E = \{s \in S_E \setminus \{\perp\} : d_E(s, G_E) \le d\}.$$

In our test cases, the parameter d is either 1, 2, or 3. As initial states, we select the set $I_E = \{s_E^{(1)}, s_E^{(2)}\}$ defined as $\{s_E^{(1)}, s_E^{(2)}\} \in \arg\max_{\{s,t\} \subseteq C_E,\, s \neq t} D_{\mathrm{JS}}(\mathsf{obs}(s), \mathsf{obs}(t))$, where

$$D_{\mathrm{JS}}(P, Q) = \frac{1}{2} D_{\mathrm{KL}}\left(P \middle\| \frac{P+Q}{2}\right) + \frac{1}{2} D_{\mathrm{KL}}\left(Q \middle\| \frac{P+Q}{2}\right)$$

is the Jensen-Shannon divergence over $\mathcal{O}_{\mathrm{nav}}$. In our test cases, the largest Jensen-Shannon divergence we observed is ≈ 0.204.

F.2.3 Identification, Friend or Foe

The IFF problem (also referred to as *aircraft-identification* problem) models an incoming aircraft that may be a friend or a foe. A nonterminal state is $s = (\tau, d, v)$, where $\tau \in \{\mathsf{friend}, \mathsf{foe}\}$ is the aircraft type, $d \in \{0, \ldots, D-1\}$ is the discretized distance to the base, and $v \in \{0, \ldots, 4\}$ is the visibility of the base to the aircraft, i.e., how well does the aircraft see the base. We use $D = 10$, so there are $2D \cdot 5 = 100$ nonterminal states. As in the source benchmark, there are four absorbing states:

$$\mathsf{base\text{-}safe}, \quad \mathsf{base\text{-}destroyed}, \quad \mathsf{foe\text{-}destroyed}, \quad \mathsf{friend\text{-}destroyed},$$

for a total of 44 states. The action set is $A_{\mathrm{air}} = \{\mathsf{active}, \mathsf{passive}, \mathsf{noop}, \mathsf{attack}\}$. The actions active and passive are sensing actions; active is more accurate but tends to increase visibility more than passive. The action noop uses no sensor, and attack attempts to destroy the aircraft.

On nonterminal transitions that do not enter an absorbing state, the aircraft type τ is unchanged. For $d > 0$, the aircraft advances one distance bin toward the base with probability 0.8, and remains at the same distance with probability 0.2. The distance transition is independent of τ, v, and the action, conditional on not entering an absorbing state. We modify the visibility transition by forbidding visibility decreases. Let the source visibility-increment probabilities over $\Delta \in \{-1, 0, +1\}$ be

Action	$q_a(-1)$	$q_a(0)$	$q_a(+1)$
noop	0.25	0.75	0.00
passive	0.00	0.90	0.10
active	0.00	0.05	0.95
attack	0.00	0.20	0.80.

For our benchmark, transitions to lower visibility are removed and the remaining probabilities are renormalized:

$$P(v' = v + \Delta \mid v, a) = \frac{q_a(\Delta)\, \mathbf{1}\{\Delta \geq 0\}\, \mathbf{1}\{0 \leq v + \Delta \leq 4\}}{\sum_{\delta \in \{-1,0,+1\}} q_a(\delta)\, \mathbf{1}\{\delta \geq 0\}\, \mathbf{1}\{0 \leq v + \delta \leq 4\}}.$$

Consequently, for $v < 4$, noop leaves visibility unchanged with probability 1, while passive, active, and attack use the probabilities shown in the 0 and $+1$ columns above. At $v = 4$, all actions leave visibility at 4 with probability 1.

The attack action may enter an absorbing state before the normal distance and visibility update is applied. At distance d, the probability of destroying the aircraft is

$$p_{\mathrm{hit}}(d) = \frac{(D-d)^2}{D^2}.$$

If the attack succeeds, the successor is foe-destroyed when $\tau = \mathsf{foe}$, and friend-destroyed when $\tau = \mathsf{friend}$. If the attack fails, the usual distance and modified visibility transitions are applied, conditional on failure. When a non-attacked friend is already at $d = 0$, the next state is base-safe with probability 1. When a non-attacked foe is already at $d = 0$, the next state is base-destroyed with probability $0.25 + 0.1v$, and base-safe otherwise. The only nonzero rewards are the immediate rewards for entering absorbing states:

Absorbing state	Reward
base-safe	0
base-destroyed	-100
foe-destroyed	20
friend-destroyed	-30.

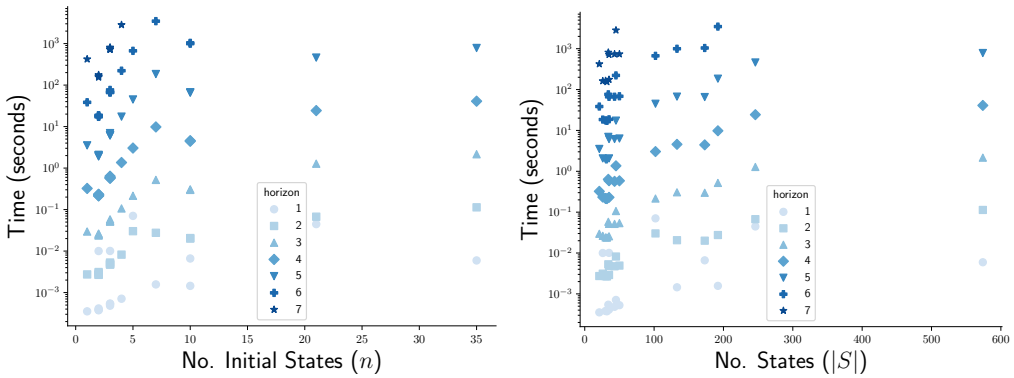


Figure 6: Scalability of Rock Sampling benchmark

The observation alphabet is

$$\mathcal{O}_{\text{air}} = \big(\{\mathsf{friend}, \mathsf{foe}\} \times \{0, \ldots, D-1\}\big) \cup \{\mathsf{nothing}, \mathsf{absorb}\},$$

so $|\mathcal{O}_{\text{air}}| = 2D{+}2 = 22$. The action noop emits nothing with probability 1, and absorbing states emit absorb with probability 1. For sensing, let $\bar{\tau}$ denote the opposite type and let $d^+ = \min\{d{+}1, D{-}1\}$. We use the following modified observation probabilities:

$$\mathsf{obs}((\tau, d) \mid (\tau, d, v), \mathsf{active}) = 0.9, \quad \mathsf{obs}((\bar{\tau}, d^+) \mid (\tau, d, v), \mathsf{active}) = 0.1,$$
$$\mathsf{obs}((\tau, d) \mid (\tau, d, v), \mathsf{passive}) = 0.8, \quad \mathsf{obs}((\bar{\tau}, d^+) \mid (\tau, d, v), \mathsf{passive}) = 0.2.$$

All other type-distance observations have probability zero. For $d = D - 1$, the reported distance d^+ is clipped to the largest distance bin, matching the boundary convention of the discretized observation space. The attack action follows the source-domain convention: successful terminal outcomes emit absorb, while unsuccessful nonterminal outcomes use the active-sensing observation model above.

Initial states in the aircraft-identification experiments are chosen from two foe states and one friend state. For parameters

$$0 \le d_1 < d_2 \le 4, \qquad v_1, v_2 \in \{0, \ldots, 4\},$$

we use the initial-state set

$$I_{d_1,d_2,v_1,v_2} = \{(\mathsf{foe}, d_1, v_1),\ (\mathsf{foe}, d_2, v_2),\ (\mathsf{friend}, d_2, \bar{v})\},$$

where $\bar{v}$ is any fixed visibility level for the friend state. The value of $\bar{v}$ is immaterial for our initialization because the friend's visibility does not affect the modified observation probabilities or the terminal reward associated with the friend/foe distinction.

Problem instances are labelled according to their initial state parameters, i.e., $\mathsf{IFF}_{d_1,d_2,v_1,v_2}$.

F.3 Scalability: Further results

In Figure 6, we extend our study of the scalability of our approach in the RockSample problem by plotting computation time in terms of total number of states and number of initial states. In both cases, we observe that computation time increases slightly with the size of the POMDP, but the major scalability issue is the horizon.

In Figure 7, we show analogous scatter plots for the robot navigation and identification benchmarks. As before, we observe that the dominant factor for scalability is horizon.

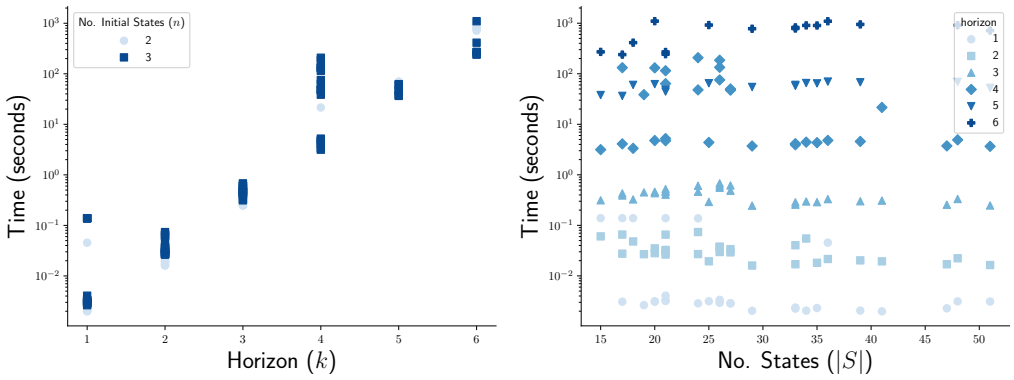


Figure 7: Scalability of the Robot Navigation and Identification Benchmarks

F.4 Comparison to baseline: Further results

In Tables 3 to 6, we extend the results in Table 1 by reporting the whole suite of experiments on the RockSample problem, comparing our runtime with the runtime of [1]. In Tables 8 to 11 we extend the results for the robot navigation problem, and in Tables 12 to 14, for the identification problem. The results show that our method is faster almost all the time, with an average ratio for the non-timeout instances of $1s$ of computation (ours) for $14537s$ of computation ([1]).

F.4.1 Comparison with approximate solution

In this section, we study running times when setting [1]'s tool with a positive tolerance ($\varepsilon > 0$), that is, allowing for approximate solutions with an approximation error below ε. In Table 15 we report the running times for the full benchmark of the Rock sampling problem. For reference, we also include the running time for $\varepsilon = 0$ (last column). Table 16 reports running times for different tolerance values for the robot navigation problem, and in Tables 17 to 21, for the identification problem.

F.5 Ablation study

Table 22 shows a comparison between our method, the ablated method, and the method from [1]. Our results show that pruning is a significantly effective addition, as the ablated method runs quickly into the memory limit and the process gets killed by the operating system. We use a memory limit of 16GB.

In the table TO is timeout (over 3600s), and MO is memory-out (over 16GB).

Table 3: Rock sampling full benchmark results (Part 1 of 5).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RS}_{3,1,2}$	26	7	3	2	1	0.3738	0.01002
$\mathsf{RS}_{3,1,2}$	26	7	3	2	2	0.7799	0.003126
$\mathsf{RS}_{3,1,2}$	26	7	3	2	3	1.229	0.02633
$\mathsf{RS}_{3,1,2}$	26	7	3	2	4	1.736	0.2335
$\mathsf{RS}_{3,1,2}$	26	7	3	2	5	2.281	2.068
$\mathsf{RS}_{3,1,2}$	26	7	3	2	6	TO	18.45
$\mathsf{RS}_{3,1,2}$	26	7	3	2	7	TO	162
$\mathsf{RS}_{3,1,3}$	34	8	3	3	1	0.7684	0.000544
$\mathsf{RS}_{3,1,3}$	34	8	3	3	2	1.532	0.005291
$\mathsf{RS}_{3,1,3}$	34	8	3	3	3	2.434	0.05812
$\mathsf{RS}_{3,1,3}$	34	8	3	3	4	TO	0.636
$\mathsf{RS}_{3,1,3}$	34	8	3	3	5	TO	6.96
$\mathsf{RS}_{3,1,3}$	34	8	3	3	6	TO	76.41
$\mathsf{RS}_{3,1,3}$	34	8	3	3	7	TO	812
$\mathsf{RS}_{3,1,4}$	45	9	3	4	1	1.267	0.000711
$\mathsf{RS}_{3,1,4}$	45	9	3	4	2	2.694	0.008213
$\mathsf{RS}_{3,1,4}$	45	9	3	4	3	4.234	0.1065
$\mathsf{RS}_{3,1,4}$	45	9	3	4	4	TO	1.362
$\mathsf{RS}_{3,1,4}$	45	9	3	4	5	TO	17.37
$\mathsf{RS}_{3,1,4}$	45	9	3	4	6	TO	222.1
$\mathsf{RS}_{3,1,4}$	45	9	3	4	7	TO	2834
$\mathsf{RS}_{3,2,3}$	50	8	3	3	1	2.158	0.000529
$\mathsf{RS}_{3,2,3}$	50	8	3	3	2	14.93	0.004932
$\mathsf{RS}_{3,2,3}$	50	8	3	3	3	7.246	0.05386
$\mathsf{RS}_{3,2,3}$	50	8	3	3	4	TO	0.5853
$\mathsf{RS}_{3,2,3}$	50	8	3	3	5	TO	6.34
$\mathsf{RS}_{3,2,3}$	50	8	3	3	6	TO	68.53
$\mathsf{RS}_{3,2,3}$	50	8	3	3	7	TO	738.4

Table 4: Rock sampling full benchmark results (Part 2 of 5).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RS}_{4,1,2}$	32	7	3	2	1	1.111	0.000375
$\mathsf{RS}_{4,1,2}$	32	7	3	2	2	2.344	0.002675
$\mathsf{RS}_{4,1,2}$	32	7	3	2	3	14.24	0.02388
$\mathsf{RS}_{4,1,2}$	32	7	3	2	4	5.263	0.2143
$\mathsf{RS}_{4,1,2}$	32	7	3	2	5	6.878	1.917
$\mathsf{RS}_{4,1,2}$	32	7	3	2	6	24.33	17.2
$\mathsf{RS}_{4,1,2}$	32	7	3	2	7	TO	154.2

Table 5: Rock sampling full benchmark results (Part 3 of 5).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RS}_{5,1,2}$	30	7	3	2	1	2.659	0.000382
$\mathsf{RS}_{5,1,2}$	30	7	3	2	2	5.628	0.002711
$\mathsf{RS}_{5,1,2}$	30	7	3	2	3	8.921	0.0241
$\mathsf{RS}_{5,1,2}$	30	7	3	2	4	12.58	0.2163
$\mathsf{RS}_{5,1,2}$	30	7	3	2	5	16.55	1.938
$\mathsf{RS}_{5,1,2}$	30	7	3	2	6	20.9	17.61
$\mathsf{RS}_{5,1,2}$	30	7	3	2	7	25.42	164.3
$\mathsf{RS}_{6,1,2}$	35	7	3	2	1	5.432	0.000414
$\mathsf{RS}_{6,1,2}$	35	7	3	2	2	11.58	0.002896
$\mathsf{RS}_{6,1,2}$	35	7	3	2	3	18.32	0.02568
$\mathsf{RS}_{6,1,2}$	35	7	3	2	4	26.05	0.2304
$\mathsf{RS}_{6,1,2}$	35	7	3	2	5	TO	2.066
$\mathsf{RS}_{6,1,2}$	35	7	3	2	6	43.17	18.68
$\mathsf{RS}_{6,1,2}$	35	7	3	2	7	52.68	175.8

Table 6: Rock sampling full benchmark results (Part 4 of 5).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RS}_{3,1,3}$	35	8	3	3	1	0.7359	0.01007
$\mathsf{RS}_{3,1,3}$	35	8	3	3	2	5.004	0.004771
$\mathsf{RS}_{3,1,3}$	35	8	3	3	3	2.441	0.05178
$\mathsf{RS}_{3,1,3}$	35	8	3	3	4	3.436	0.5878
$\mathsf{RS}_{3,1,3}$	35	8	3	3	5	TO	6.156
$\mathsf{RS}_{3,1,3}$	35	8	3	3	6	TO	66.51
$\mathsf{RS}_{3,1,3}$	35	8	3	3	7	TO	716
$\mathsf{RS}_{3,2,3}$	43	8	3	3	1	2.14	0.000483
$\mathsf{RS}_{3,2,3}$	43	8	3	3	2	4.526	0.004735
$\mathsf{RS}_{3,2,3}$	43	8	3	3	3	7.225	0.05159
$\mathsf{RS}_{3,2,3}$	43	8	3	3	4	TO	0.5671
$\mathsf{RS}_{3,2,3}$	43	8	3	3	5	TO	6.15
$\mathsf{RS}_{3,2,3}$	43	8	3	3	6	TO	66.86
$\mathsf{RS}_{3,2,3}$	43	8	3	3	7	TO	736.1
$\mathsf{RS}_{3,3,3}$	21	8	3	1	1	2.797	0.000353
$\mathsf{RS}_{3,3,3}$	21	8	3	1	2	5.977	0.002736
$\mathsf{RS}_{3,3,3}$	21	8	3	1	3	9.338	0.02959
$\mathsf{RS}_{3,3,3}$	21	8	3	1	4	TO	0.3233
$\mathsf{RS}_{3,3,3}$	21	8	3	1	5	TO	3.525
$\mathsf{RS}_{3,3,3}$	21	8	3	1	6	60.94	38.46
$\mathsf{RS}_{3,3,3}$	21	8	3	1	7	TO	421.8
$\mathsf{RS}_{3,2,5}$	133	10	3	10	1	13.97	0.001451
$\mathsf{RS}_{3,2,5}$	133	10	3	10	2	TO	0.02059
$\mathsf{RS}_{3,2,5}$	133	10	3	10	3	46.86	0.3065
$\mathsf{RS}_{3,2,5}$	133	10	3	10	4	200.7	4.567
$\mathsf{RS}_{3,2,5}$	133	10	3	10	5	TO	67.1
$\mathsf{RS}_{3,2,5}$	133	10	3	10	6	TO	997.7
$\mathsf{RS}_{3,2,5}$	133	10	3	10	7	TO	TO

Table 7: Rock sampling full benchmark results (Part 5 of 5).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RS}_{3,3,5}$	173	10	3	10	1	36.18	0.006625
$\mathsf{RS}_{3,3,5}$	173	10	3	10	2	TO	0.02006
$\mathsf{RS}_{3,3,5}$	173	10	3	10	3	121.7	0.2983
$\mathsf{RS}_{3,3,5}$	173	10	3	10	4	TO	4.436
$\mathsf{RS}_{3,3,5}$	173	10	3	10	5	TO	65.58
$\mathsf{RS}_{3,3,5}$	173	10	3	10	6	TO	1041
$\mathsf{RS}_{3,3,5}$	173	10	3	10	7	TO	TO
$\mathsf{RS}_{3,4,5}$	102	10	3	5	1	51.55	0.07048
$\mathsf{RS}_{3,4,5}$	102	10	3	5	2	356	0.03017
$\mathsf{RS}_{3,4,5}$	102	10	3	5	3	172.6	0.2155
$\mathsf{RS}_{3,4,5}$	102	10	3	5	4	TO	3.038
$\mathsf{RS}_{3,4,5}$	102	10	3	5	5	TO	44.98
$\mathsf{RS}_{3,4,5}$	102	10	3	5	6	TO	669.2
$\mathsf{RS}_{3,4,5}$	102	10	3	5	7	TO	TO
$\mathsf{RS}_{3,2,7}$	246	12	3	21	1	54.27	0.04466
$\mathsf{RS}_{3,2,7}$	246	12	3	21	2	115.9	0.06734
$\mathsf{RS}_{3,2,7}$	246	12	3	21	3	185.3	1.277
$\mathsf{RS}_{3,2,7}$	246	12	3	21	4	TO	24.3
$\mathsf{RS}_{3,2,7}$	246	12	3	21	5	TO	459.9
$\mathsf{RS}_{3,2,7}$	246	12	3	21	6	TO	TO
$\mathsf{RS}_{3,2,7}$	246	12	3	21	7	TO	TO
$\mathsf{RS}_{3,4,7}$	574	12	3	35	1	627.6	0.005945
$\mathsf{RS}_{3,4,7}$	574	12	3	35	2	1339	0.1132
$\mathsf{RS}_{3,4,7}$	574	12	3	35	3	2129	2.157
$\mathsf{RS}_{3,4,7}$	574	12	3	35	4	TO	40.9
$\mathsf{RS}_{3,4,7}$	574	12	3	35	5	TO	783.8
$\mathsf{RS}_{3,4,7}$	574	12	3	35	6	TO	TO
$\mathsf{RS}_{3,4,7}$	574	12	3	35	7	TO	TO
$\mathsf{RS}_{3,6,7}$	192	12	3	7	1	1029	0.001572
$\mathsf{RS}_{3,6,7}$	192	12	3	7	2	TO	0.02758
$\mathsf{RS}_{3,6,7}$	192	12	3	7	3	3493	0.5194
$\mathsf{RS}_{3,6,7}$	192	12	3	7	4	TO	9.806
$\mathsf{RS}_{3,6,7}$	192	12	3	7	5	TO	184.6
$\mathsf{RS}_{3,6,7}$	192	12	3	7	6	TO	3477
$\mathsf{RS}_{3,6,7}$	192	12	3	7	7	TO	TO

Table 8: Robot navigation full benchmark results (Part 1 of 4).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RN}_{\text{CIT},1}$	33	4	28	2	1	151.8	0.002365
$\mathsf{RN}_{\text{CIT},1}$	33	4	28	2	2	TO	0.04038
$\mathsf{RN}_{\text{CIT},1}$	33	4	28	2	3	TO	0.2864
$\mathsf{RN}_{\text{CIT},1}$	33	4	28	2	4	TO	4.136
$\mathsf{RN}_{\text{CIT},1}$	33	4	28	2	5	TO	60.3
$\mathsf{RN}_{\text{CIT},1}$	33	4	28	2	6	TO	826.8
$\mathsf{RN}_{\text{CIT},2}$	33	4	28	2	1	150.9	0.002247
$\mathsf{RN}_{\text{CIT},2}$	33	4	28	2	2	305.3	0.017
$\mathsf{RN}_{\text{CIT},2}$	33	4	28	2	3	TO	0.2568
$\mathsf{RN}_{\text{CIT},2}$	33	4	28	2	4	TO	3.935
$\mathsf{RN}_{\text{CIT},2}$	33	4	28	2	5	TO	57.73
$\mathsf{RN}_{\text{CIT},2}$	33	4	28	2	6	TO	786.3
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	1	150.3	0.003128
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	2	304.2	0.0223
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	3	457.3	0.3325
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	4	TO	4.921
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	5	TO	69.29
$\mathsf{RN}_{\text{CIT},3}$	48	4	28	2	6	TO	914.1

Table 9: Robot navigation full benchmark results (Part 2 of 4).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RN}_{\text{MIT},1}$	47	4	28	2	1	77.93	0.00228
$\mathsf{RN}_{\text{MIT},1}$	47	4	28	2	2	TO	0.01699
$\mathsf{RN}_{\text{MIT},1}$	47	4	28	2	3	TO	0.2556
$\mathsf{RN}_{\text{MIT},1}$	47	4	28	2	4	TO	3.737
$\mathsf{RN}_{\text{MIT},1}$	47	4	28	2	5	TO	TO
$\mathsf{RN}_{\text{MIT},1}$	47	4	28	2	6	TO	TO
$\mathsf{RN}_{\text{MIT},2}$	41	4	28	2	1	77.54	0.001988
$\mathsf{RN}_{\text{MIT},2}$	41	4	28	2	2	155.4	0.01937
$\mathsf{RN}_{\text{MIT},2}$	41	4	28	2	3	TO	0.3086
$\mathsf{RN}_{\text{MIT},2}$	41	4	28	2	4	TO	21.56
$\mathsf{RN}_{\text{MIT},2}$	41	4	28	2	5	TO	TO
$\mathsf{RN}_{\text{MIT},2}$	41	4	28	2	6	TO	TO
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	1	78.39	0.04544
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	2	156.8	0.02158
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	3	238.3	0.3299
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	4	TO	4.802
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	5	TO	70.02
$\mathsf{RN}_{\text{MIT},3}$	36	4	28	2	6	TO	1096

Table 10: Robot navigation full benchmark results (Part 3 of 4).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RN}_{\text{PEN.},1}$	34	4	28	2	1	84.61	0.002054
$\mathsf{RN}_{\text{PEN.},1}$	34	4	28	2	2	TO	0.05489
$\mathsf{RN}_{\text{PEN.},1}$	34	4	28	2	3	TO	0.2966
$\mathsf{RN}_{\text{PEN.},1}$	34	4	28	2	4	TO	4.433
$\mathsf{RN}_{\text{PEN.},1}$	34	4	28	2	5	TO	65.52
$\mathsf{RN}_{\text{PEN.},1}$	34	4	28	2	6	TO	905.4
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	1	85.13	0.002313
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	2	171.3	0.01818
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	3	TO	0.2877
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	4	TO	4.33
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	5	TO	64.54
$\mathsf{RN}_{\text{PEN.},2}$	35	4	28	2	6	TO	902.1
$\mathsf{RN}_{\text{PEN.},3}$	25	4	28	2	1	85.39	0.003148
$\mathsf{RN}_{\text{PEN.},3}$	25	4	28	2	2	170	0.01943
$\mathsf{RN}_{\text{PEN.},3}$	25	4	28	2	3	261.1	0.2908
$\mathsf{RN}_{\text{PEN.},3}$	25	4	28	2	4	TO	4.37
$\mathsf{RN}_{\text{PEN.},3}$	25	4	28	2	5	TO	64.95
$\mathsf{RN}_{\text{PEN.},3}$	25	4	28	2	6	TO	919.8

Table 11: Robot navigation full benchmark results (Part 4 of 4).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{RN}_{\text{SUNYSB},1}$	39	4	28	2	1	170	0.002059
$\mathsf{RN}_{\text{SUNYSB},1}$	39	4	28	2	2	TO	0.02027
$\mathsf{RN}_{\text{SUNYSB},1}$	39	4	28	2	3	TO	0.2996
$\mathsf{RN}_{\text{SUNYSB},1}$	39	4	28	2	4	TO	4.574
$\mathsf{RN}_{\text{SUNYSB},1}$	39	4	28	2	5	TO	67.88
$\mathsf{RN}_{\text{SUNYSB},1}$	39	4	28	2	6	TO	947.4
$\mathsf{RN}_{\text{SUNYSB},2}$	29	4	28	2	1	168.6	0.002042
$\mathsf{RN}_{\text{SUNYSB},2}$	29	4	28	2	2	338.9	0.01603
$\mathsf{RN}_{\text{SUNYSB},2}$	29	4	28	2	3	TO	0.2455
$\mathsf{RN}_{\text{SUNYSB},2}$	29	4	28	2	4	TO	3.717
$\mathsf{RN}_{\text{SUNYSB},2}$	29	4	28	2	5	TO	54.77
$\mathsf{RN}_{\text{SUNYSB},2}$	29	4	28	2	6	TO	778.2
$\mathsf{RN}_{\text{SUNYSB},3}$	51	4	28	2	1	169.7	0.003113
$\mathsf{RN}_{\text{SUNYSB},3}$	51	4	28	2	2	342.6	0.01639
$\mathsf{RN}_{\text{SUNYSB},3}$	51	4	28	2	3	516.6	0.2474
$\mathsf{RN}_{\text{SUNYSB},3}$	51	4	28	2	4	TO	3.66
$\mathsf{RN}_{\text{SUNYSB},3}$	51	4	28	2	5	TO	52.69
$\mathsf{RN}_{\text{SUNYSB},3}$	51	4	28	2	6	TO	714

Table 12: Identification (friend or foe) full benchmark results (Part 1 of 3).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{IFF}_{1,2,0,2}$	20	4	22	3	1	TO	0.003146
$\mathsf{IFF}_{1,2,0,2}$	20	4	22	3	2	error	0.03503
$\mathsf{IFF}_{1,2,0,2}$	20	4	22	3	3	TO	0.4615
$\mathsf{IFF}_{1,2,0,2}$	20	4	22	3	4	TO	4.773
$\mathsf{IFF}_{1,2,0,2}$	20	4	22	3	5	TO	62.33
$\mathsf{IFF}_{1,2,0,2}$	20	4	22	3	6	TO	1099
$\mathsf{IFF}_{1,2,0,4}$	18	4	22	3	1	TO	0.1386
$\mathsf{IFF}_{1,2,0,4}$	18	4	22	3	2	TO	0.04797
$\mathsf{IFF}_{1,2,0,4}$	18	4	22	3	3	TO	0.3258
$\mathsf{IFF}_{1,2,0,4}$	18	4	22	3	4	TO	3.342
$\mathsf{IFF}_{1,2,0,4}$	18	4	22	3	5	TO	60.24
$\mathsf{IFF}_{1,2,0,4}$	18	4	22	3	6	TO	412.6
$\mathsf{IFF}_{1,2,2,0}$	21	4	22	3	1	TO	0.003343
$\mathsf{IFF}_{1,2,2,0}$	21	4	22	3	2	error	0.03263
$\mathsf{IFF}_{1,2,2,0}$	21	4	22	3	3	TO	0.4695
$\mathsf{IFF}_{1,2,2,0}$	21	4	22	3	4	TO	5.215
$\mathsf{IFF}_{1,2,2,0}$	21	4	22	3	5	TO	48.06
$\mathsf{IFF}_{1,2,2,0}$	21	4	22	3	6	TO	269.5
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	1	TO	0.139
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	2	error	0.0605
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	3	TO	0.3132
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	4	TO	3.149
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	5	TO	37.9
$\mathsf{IFF}_{1,2,2,4}$	15	4	22	3	6	TO	271
$\mathsf{IFF}_{1,2,4,0}$	21	4	22	3	1	TO	0.1388
$\mathsf{IFF}_{1,2,4,0}$	21	4	22	3	2	error	0.06559
$\mathsf{IFF}_{1,2,4,0}$	21	4	22	3	3	TO	0.4053
$\mathsf{IFF}_{1,2,4,0}$	21	4	22	3	4	TO	4.727
$\mathsf{IFF}_{1,2,4,0}$	21	4	22	3	5	TO	44.38
$\mathsf{IFF}_{1,2,4,0}$	21	4	22	3	6	TO	244.1
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	1	TO	0.1388
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	2	error	0.06575
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	3	TO	0.39
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	4	TO	4.096
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	5	TO	36.52
$\mathsf{IFF}_{1,2,4,2}$	17	4	22	3	6	TO	239.5

Table 13: Identification (friend or foe) full benchmark results (Part 2 of 3).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	1	TO	0.1376
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	2	141.6	0.07371
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	3	213.5	0.6066
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	4	error	208
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	5	error	TO
$\mathsf{IFF}_{1,3,0,2}$	24	4	22	3	6	error	TO
$\mathsf{IFF}_{1,3,0,4}$	20	4	22	3	1	TO	0.003086
$\mathsf{IFF}_{1,3,0,4}$	20	4	22	3	2	141.9	0.02816
$\mathsf{IFF}_{1,3,0,4}$	20	4	22	3	3	408.1	0.4326
$\mathsf{IFF}_{1,3,0,4}$	20	4	22	3	4	TO	130.8
$\mathsf{IFF}_{1,3,0,4}$	20	4	22	3	5	TO	TO
$\mathsf{IFF}_{1,3,0,4}$	20	4	22	3	6	error	TO
$\mathsf{IFF}_{1,3,2,0}$	26	4	22	3	1	TO	0.002936
$\mathsf{IFF}_{1,3,2,0}$	26	4	22	3	2	141.9	0.03683
$\mathsf{IFF}_{1,3,2,0}$	26	4	22	3	3	514.2	0.6137
$\mathsf{IFF}_{1,3,2,0}$	26	4	22	3	4	error	134
$\mathsf{IFF}_{1,3,2,0}$	26	4	22	3	5	TO	TO
$\mathsf{IFF}_{1,3,2,0}$	26	4	22	3	6	TO	TO
$\mathsf{IFF}_{1,3,2,4}$	17	4	22	3	1	TO	0.003083
$\mathsf{IFF}_{1,3,2,4}$	17	4	22	3	2	142.5	0.02751
$\mathsf{IFF}_{1,3,2,4}$	17	4	22	3	3	515.4	0.4297
$\mathsf{IFF}_{1,3,2,4}$	17	4	22	3	4	TO	131.8
$\mathsf{IFF}_{1,3,2,4}$	17	4	22	3	5	TO	TO
$\mathsf{IFF}_{1,3,2,4}$	17	4	22	3	6	TO	TO
$\mathsf{IFF}_{1,3,4,0}$	26	4	22	3	1	7.666	0.003359
$\mathsf{IFF}_{1,3,4,0}$	26	4	22	3	2	error	0.02967
$\mathsf{IFF}_{1,3,4,0}$	26	4	22	3	3	error	0.5556
$\mathsf{IFF}_{1,3,4,0}$	26	4	22	3	4	error	185.3
$\mathsf{IFF}_{1,3,4,0}$	26	4	22	3	5	error	TO
$\mathsf{IFF}_{1,3,4,0}$	26	4	22	3	6	error	TO
$\mathsf{IFF}_{1,3,4,2}$	21	4	22	3	1	TO	0.004064
$\mathsf{IFF}_{1,3,4,2}$	21	4	22	3	2	165.2	0.03064
$\mathsf{IFF}_{1,3,4,2}$	21	4	22	3	3	error	0.5227
$\mathsf{IFF}_{1,3,4,2}$	21	4	22	3	4	TO	114.9
$\mathsf{IFF}_{1,3,4,2}$	21	4	22	3	5	TO	TO
$\mathsf{IFF}_{1,3,4,2}$	21	4	22	3	6	TO	TO

Table 14: Identification (friend or foe) full benchmark results (Part 3 of 3).

Model	$\|S\|$	$\|A\|$	$\|\mathcal{O}\|$	n	k	Time [1] (s)	Time ours (s)
$\mathsf{IFF}_{2,3,0,2}$	26	4	22	3	1	TO	0.003104
$\mathsf{IFF}_{2,3,0,2}$	26	4	22	3	2	137.3	0.03796
$\mathsf{IFF}_{2,3,0,2}$	26	4	22	3	3	TO	0.6793
$\mathsf{IFF}_{2,3,0,2}$	26	4	22	3	4	TO	75.48
$\mathsf{IFF}_{2,3,0,2}$	26	4	22	3	5	TO	TO
$\mathsf{IFF}_{2,3,0,2}$	26	4	22	3	6	TO	TO
$\mathsf{IFF}_{2,3,0,4}$	24	4	22	3	1	TO	0.002859
$\mathsf{IFF}_{2,3,0,4}$	24	4	22	3	2	136.7	0.02701
$\mathsf{IFF}_{2,3,0,4}$	24	4	22	3	3	TO	0.4624
$\mathsf{IFF}_{2,3,0,4}$	24	4	22	3	4	TO	47.96
$\mathsf{IFF}_{2,3,0,4}$	24	4	22	3	5	TO	TO
$\mathsf{IFF}_{2,3,0,4}$	24	4	22	3	6	TO	TO
$\mathsf{IFF}_{2,3,2,0}$	27	4	22	3	1	TO	0.002836
$\mathsf{IFF}_{2,3,2,0}$	27	4	22	3	2	138.9	0.03382
$\mathsf{IFF}_{2,3,2,0}$	27	4	22	3	3	TO	0.6166
$\mathsf{IFF}_{2,3,2,0}$	27	4	22	3	4	TO	50
$\mathsf{IFF}_{2,3,2,0}$	27	4	22	3	5	TO	TO
$\mathsf{IFF}_{2,3,2,0}$	27	4	22	3	6	TO	TO
$\mathsf{IFF}_{2,3,2,4}$	19	4	22	3	1	TO	0.002635
$\mathsf{IFF}_{2,3,2,4}$	19	4	22	3	2	142.7	0.02685
$\mathsf{IFF}_{2,3,2,4}$	19	4	22	3	3	error	0.4497
$\mathsf{IFF}_{2,3,2,4}$	19	4	22	3	4	TO	38.59
$\mathsf{IFF}_{2,3,2,4}$	19	4	22	3	5	TO	TO
$\mathsf{IFF}_{2,3,2,4}$	19	4	22	3	6	TO	TO
$\mathsf{IFF}_{2,3,4,0}$	27	4	22	3	1	TO	0.002888
$\mathsf{IFF}_{2,3,4,0}$	27	4	22	3	2	138.8	0.02922
$\mathsf{IFF}_{2,3,4,0}$	27	4	22	3	3	TO	0.4841
$\mathsf{IFF}_{2,3,4,0}$	27	4	22	3	4	TO	47.38
$\mathsf{IFF}_{2,3,4,0}$	27	4	22	3	5	TO	TO
$\mathsf{IFF}_{2,3,4,0}$	27	4	22	3	6	TO	TO
$\mathsf{IFF}_{2,3,4,2}$	21	4	22	3	1	TO	0.003189
$\mathsf{IFF}_{2,3,4,2}$	21	4	22	3	2	139.1	0.02644
$\mathsf{IFF}_{2,3,4,2}$	21	4	22	3	3	TO	0.4669
$\mathsf{IFF}_{2,3,4,2}$	21	4	22	3	4	TO	63.63
$\mathsf{IFF}_{2,3,4,2}$	21	4	22	3	5	TO	TO
$\mathsf{IFF}_{2,3,4,2}$	21	4	22	3	6	TO	TO

Table 15: Meaningful tolerances $\varepsilon > 0$ that improve running time on the rock sampling benchmark.

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{RS}_{3,2,3}$	7	10.0	214.734942	TO
$\mathsf{RS}_{3,2,3}$	7	16.666637	162.974267	TO
$\mathsf{RS}_{3,2,3}$	7	20.0	12.911721	TO
$\mathsf{RS}_{3,2,3}$	7	14.0	205.202806	TO
$\mathsf{RS}_{3,2,3}$	7	16.666637	153.575063	TO
$\mathsf{RS}_{3,2,3}$	7	20.0	12.718293	TO
$\mathsf{RS}_{3,3,3}$	7	10.0	65.583179	TO
$\mathsf{RS}_{3,3,3}$	7	20.0	16.556184	TO
$\mathsf{RS}_{3,2,5}$	5	10.0	289.43802	TO
$\mathsf{RS}_{3,2,5}$	5	20.0	60.398253	TO
$\mathsf{RS}_{3,2,5}$	6	9.999977	2850.84934	TO
$\mathsf{RS}_{3,2,5}$	6	10.0	989.714879	TO
$\mathsf{RS}_{3,2,5}$	6	15.0	580.956723	TO
$\mathsf{RS}_{3,2,5}$	6	20.0	73.022158	TO
$\mathsf{RS}_{3,2,5}$	7	10.0	1349.932869	TO
$\mathsf{RS}_{3,2,5}$	7	15.0	1076.938035	TO
$\mathsf{RS}_{3,2,5}$	7	20.0	84.886007	TO
$\mathsf{RS}_{3,3,5}$	6	15.0	1509.046957	TO
$\mathsf{RS}_{3,3,5}$	6	20.0	189.186391	TO
$\mathsf{RS}_{3,3,5}$	7	19.375	3354.298341	TO
$\mathsf{RS}_{3,3,5}$	7	20.0	219.097685	TO
$\mathsf{RS}_{3,4,5}$	5	6.666667	2290.876245	TO
$\mathsf{RS}_{3,4,5}$	5	10.0	222.28128	TO
$\mathsf{RS}_{3,4,5}$	6	16.666667	3404.402459	TO
$\mathsf{RS}_{3,4,5}$	6	18.333333	2406.558198	TO
$\mathsf{RS}_{3,4,5}$	6	20.0	267.271925	TO
$\mathsf{RS}_{3,4,5}$	7	13.333303	3427.793512	TO
$\mathsf{RS}_{3,4,5}$	7	20.0	311.159537	TO

Table 16: Meaningful tolerances $\varepsilon > 0$ that improve running time on the robot navigation benchmark.

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{RN}_{\text{CIT},1}$	2	0.00271	1036.496869	TO
$\mathsf{RN}_{\text{CIT},1}$	2	0.01	298.317198	TO
$\mathsf{RN}_{\text{CIT},1}$	3	0.054964	2651.807021	TO
$\mathsf{RN}_{\text{CIT},1}$	3	0.7943	447.394049	TO
$\mathsf{RN}_{\text{CIT},1}$	4	0.010657	3560.988061	TO
$\mathsf{RN}_{\text{CIT},1}$	4	0.966725	583.901878	TO
$\mathsf{RN}_{\text{CIT},2}$	3	0.00572	3386.211035	TO
$\mathsf{RN}_{\text{CIT},2}$	3	0.009	446.538798	TO
$\mathsf{RN}_{\text{CIT},3}$	4	0.0001	3548.543892	TO
$\mathsf{RN}_{\text{CIT},3}$	4	0.71289	594.832288	TO
$\mathsf{RN}_{\text{MIT},1}$	2	0.00271	518.342156	TO
$\mathsf{RN}_{\text{MIT},1}$	2	0.01	150.553949	TO
$\mathsf{RN}_{\text{MIT},1}$	3	0.054379	2503.895705	TO
$\mathsf{RN}_{\text{MIT},1}$	3	0.054964	1369.080751	TO
$\mathsf{RN}_{\text{MIT},1}$	3	0.7943	230.148927	TO
$\mathsf{RN}_{\text{MIT},1}$	4	0.010592	3373.050409	TO
$\mathsf{RN}_{\text{MIT},1}$	4	0.010657	1862.766506	TO
$\mathsf{RN}_{\text{MIT},1}$	4	0.966725	311.284316	TO
$\mathsf{RN}_{\text{MIT},1}$	5	0.008201	2329.608765	TO
$\mathsf{RN}_{\text{MIT},1}$	5	0.995168	383.502956	TO
$\mathsf{RN}_{\text{MIT},1}$	6	0.008869	2814.605176	TO
$\mathsf{RN}_{\text{MIT},1}$	6	0.99934	460.80324	TO
$\mathsf{RN}_{\text{PEN},1}$	2	0.00271	1858.005867	TO
$\mathsf{RN}_{\text{PEN},1}$	2	0.01	999.201917	TO
$\mathsf{RN}_{\text{PEN},1}$	2	0.89	164.502555	TO
$\mathsf{RN}_{\text{PEN},1}$	3	0.054379	2777.488685	TO
$\mathsf{RN}_{\text{PEN},1}$	3	0.054964	1513.262795	TO
$\mathsf{RN}_{\text{PEN},1}$	3	0.9879	250.414842	TO
$\mathsf{RN}_{\text{PEN},1}$	4	0.069944	2026.016335	TO
$\mathsf{RN}_{\text{PEN},1}$	4	0.998669	327.671769	TO
$\mathsf{RN}_{\text{PEN},1}$	5	0.080113	2548.471477	TO
$\mathsf{RN}_{\text{PEN},1}$	5	0.999854	413.853363	TO
$\mathsf{RN}_{\text{PEN},1}$	6	0.082247	3070.93714	TO
$\mathsf{RN}_{\text{PEN},1}$	6	0.999984	498.117854	TO
$\mathsf{RN}_{\text{PEN},2}$	3	0.0177	1492.731108	TO
$\mathsf{RN}_{\text{PEN},2}$	3	0.801	251.091594	TO
$\mathsf{RN}_{\text{PEN},2}$	4	0.71081	2023.965727	TO
$\mathsf{RN}_{\text{PEN},2}$	4	0.96921	335.280305	TO
$\mathsf{RN}_{\text{PEN},2}$	5	0.938894	2549.003754	TO
$\mathsf{RN}_{\text{PEN},2}$	5	0.995723	418.483646	TO
$\mathsf{RN}_{\text{PEN},2}$	6	0.987306	3085.625036	TO
$\mathsf{RN}_{\text{PEN},2}$	6	0.999441	506.778236	TO
$\mathsf{RN}_{\text{PEN},3}$	4	0.010124	2833.602887	TO
$\mathsf{RN}_{\text{PEN},3}$	4	0.01593	332.856869	TO
$\mathsf{RN}_{\text{SUNYSB},1}$	2	0.00271	1156.637073	TO
$\mathsf{RN}_{\text{SUNYSB},1}$	2	0.01	331.517546	TO
$\mathsf{RN}_{\text{SUNYSB},1}$	3	0.054964	2993.939058	TO
$\mathsf{RN}_{\text{SUNYSB},1}$	3	0.7943	500.289369	TO
$\mathsf{RN}_{\text{SUNYSB},2}$	3	0.000135	2124.813691	TO
$\mathsf{RN}_{\text{SUNYSB},2}$	3	0.009	493.375661	TO

Table 17: Meaningful tolerances $\varepsilon > 0$ that improve running time on the identification (friend of foe) benchmark (Part 1 of 5).

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{IFF}_{1,2,0,2}$	2	0.451222	76.474819	error
$\mathsf{IFF}_{1,2,0,2}$	2	10.335019	11.986798	error
$\mathsf{IFF}_{1,2,0,2}$	3	0.064845	304.878326	TO
$\mathsf{IFF}_{1,2,0,2}$	3	0.065956	208.809091	TO
$\mathsf{IFF}_{1,2,0,2}$	3	0.434824	113.755056	TO
$\mathsf{IFF}_{1,2,0,2}$	3	10.366268	18.006563	TO
$\mathsf{IFF}_{1,2,0,2}$	4	0.057703	372.86874	TO
$\mathsf{IFF}_{1,2,0,2}$	4	0.062779	245.994885	TO
$\mathsf{IFF}_{1,2,0,2}$	4	0.425804	119.642974	TO
$\mathsf{IFF}_{1,2,0,2}$	4	10.406565	23.824224	TO
$\mathsf{IFF}_{1,2,0,2}$	5	0.425539	126.517056	TO
$\mathsf{IFF}_{1,2,0,2}$	5	10.409824	29.661885	TO
$\mathsf{IFF}_{1,2,0,2}$	6	0.05706	418.433589	TO
$\mathsf{IFF}_{1,2,0,2}$	6	0.06602	323.31635	TO
$\mathsf{IFF}_{1,2,0,2}$	6	0.428498	228.396439	TO
$\mathsf{IFF}_{1,2,0,2}$	6	10.410062	35.603996	TO
$\mathsf{IFF}_{1,2,0,4}$	2	2.686185	75.570873	TO
$\mathsf{IFF}_{1,2,0,4}$	2	10.335019	11.990939	TO
$\mathsf{IFF}_{1,2,0,4}$	3	2.625939	177.047695	TO
$\mathsf{IFF}_{1,2,0,4}$	3	10.335019	113.646608	TO
$\mathsf{IFF}_{1,2,0,4}$	3	10.407664	17.965753	TO
$\mathsf{IFF}_{1,2,0,4}$	4	2.639832	151.483333	TO
$\mathsf{IFF}_{1,2,0,4}$	4	10.335019	88.296378	TO
$\mathsf{IFF}_{1,2,0,4}$	4	10.410011	23.893436	TO
$\mathsf{IFF}_{1,2,0,4}$	5	2.639779	157.624318	TO
$\mathsf{IFF}_{1,2,0,4}$	5	10.335019	94.745284	TO
$\mathsf{IFF}_{1,2,0,4}$	5	10.410096	29.889675	TO
$\mathsf{IFF}_{1,2,0,4}$	6	2.639779	164.085923	TO
$\mathsf{IFF}_{1,2,0,4}$	6	10.335019	101.312377	TO
$\mathsf{IFF}_{1,2,0,4}$	6	10.41008	35.72717	TO
$\mathsf{IFF}_{1,2,2,0}$	2	0.451222	76.950439	error
$\mathsf{IFF}_{1,2,2,0}$	2	10.335019	12.051624	error
$\mathsf{IFF}_{1,2,2,0}$	3	0.05911	306.731205	TO
$\mathsf{IFF}_{1,2,2,0}$	3	0.060956	210.168062	TO
$\mathsf{IFF}_{1,2,2,0}$	3	0.434824	114.148018	TO
$\mathsf{IFF}_{1,2,2,0}$	3	10.366268	17.919671	TO
$\mathsf{IFF}_{1,2,2,0}$	4	0.058062	375.554867	TO
$\mathsf{IFF}_{1,2,2,0}$	4	0.058322	247.797874	TO
$\mathsf{IFF}_{1,2,2,0}$	4	0.425804	120.340503	TO
$\mathsf{IFF}_{1,2,2,0}$	4	10.406565	24.000758	TO
$\mathsf{IFF}_{1,2,2,0}$	5	0.058477	254.674578	TO
$\mathsf{IFF}_{1,2,2,0}$	5	0.425439	127.239174	TO
$\mathsf{IFF}_{1,2,2,0}$	5	10.409824	30.169275	TO
$\mathsf{IFF}_{1,2,2,0}$	6	0.05822	229.273519	TO
$\mathsf{IFF}_{1,2,2,0}$	6	0.425478	133.777427	TO
$\mathsf{IFF}_{1,2,2,0}$	6	10.410062	35.914999	TO
$\mathsf{IFF}_{1,2,2,4}$	2	0.451222	76.717783	error
$\mathsf{IFF}_{1,2,2,4}$	2	10.335019	12.056523	error
$\mathsf{IFF}_{1,2,2,4}$	3	0.065162	307.229459	TO
$\mathsf{IFF}_{1,2,2,4}$	3	0.066234	210.491118	TO
$\mathsf{IFF}_{1,2,2,4}$	3	0.434824	114.234266	TO
$\mathsf{IFF}_{1,2,2,4}$	3	10.366268	18.074022	TO
$\mathsf{IFF}_{1,2,2,4}$	4	0.057943	375.2	TO
$\mathsf{IFF}_{1,2,2,4}$	4	0.063019	247.464541	TO
$\mathsf{IFF}_{1,2,2,4}$	4	0.425804	120.09296	TO
$\mathsf{IFF}_{1,2,2,4}$	4	10.406565	23.917012	TO
$\mathsf{IFF}_{1,2,2,4}$	5	0.062817	381.827551	TO
$\mathsf{IFF}_{1,2,2,4}$	5	0.063177	254.211061	TO
$\mathsf{IFF}_{1,2,2,4}$	5	0.425539	126.935491	TO
$\mathsf{IFF}_{1,2,2,4}$	5	10.409824	29.968624	TO

Table 18: Meaningful tolerances $\varepsilon > 0$ that improve running time on the identification (friend of foe) benchmark (Part 2 of 5).

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{IFF}_{1,2,2,4}$	6	0.06292	324.694133	TO
$\mathsf{IFF}_{1,2,2,4}$	6	0.06566	228.996656	TO
$\mathsf{IFF}_{1,2,2,4}$	6	0.425638	133.358436	TO
$\mathsf{IFF}_{1,2,2,4}$	6	10.410062	35.923125	TO
$\mathsf{IFF}_{1,2,4,0}$	2	0.451222	76.964088	error
$\mathsf{IFF}_{1,2,4,0}$	2	10.335019	12.070612	error
$\mathsf{IFF}_{1,2,4,0}$	3	0.05911	306.711583	TO
$\mathsf{IFF}_{1,2,4,0}$	3	0.060956	209.902358	TO
$\mathsf{IFF}_{1,2,4,0}$	3	0.434824	113.962958	TO
$\mathsf{IFF}_{1,2,4,0}$	3	10.366268	17.928714	TO
$\mathsf{IFF}_{1,2,4,0}$	4	0.058062	375.746958	TO
$\mathsf{IFF}_{1,2,4,0}$	4	0.058322	247.673112	TO
$\mathsf{IFF}_{1,2,4,0}$	4	0.425804	119.923301	TO
$\mathsf{IFF}_{1,2,4,0}$	4	10.406565	23.746487	TO
$\mathsf{IFF}_{1,2,4,0}$	5	0.058477	255.065468	TO
$\mathsf{IFF}_{1,2,4,0}$	5	0.425439	127.138802	TO
$\mathsf{IFF}_{1,2,4,0}$	5	10.409824	30.059509	TO
$\mathsf{IFF}_{1,2,4,0}$	6	0.05822	229.04807	TO
$\mathsf{IFF}_{1,2,4,0}$	6	0.425478	133.323256	TO
$\mathsf{IFF}_{1,2,4,0}$	6	10.410062	35.868922	TO
$\mathsf{IFF}_{1,2,4,2}$	2	0.451222	78.266124	error
$\mathsf{IFF}_{1,2,4,2}$	2	10.335019	12.128613	error
$\mathsf{IFF}_{1,2,4,2}$	3	0.064845	311.671375	TO
$\mathsf{IFF}_{1,2,4,2}$	3	0.065956	213.738168	TO
$\mathsf{IFF}_{1,2,4,2}$	3	0.434824	116.052927	TO
$\mathsf{IFF}_{1,2,4,2}$	3	10.366268	18.208854	TO
$\mathsf{IFF}_{1,2,4,2}$	4	0.057703	382.914156	TO
$\mathsf{IFF}_{1,2,4,2}$	4	0.062779	252.617002	TO
$\mathsf{IFF}_{1,2,4,2}$	4	0.425804	123.428096	TO
$\mathsf{IFF}_{1,2,4,2}$	4	10.406565	24.658906	TO
$\mathsf{IFF}_{1,2,4,2}$	5	0.425539	130.144402	TO
$\mathsf{IFF}_{1,2,4,2}$	5	10.409824	31.097878	TO
$\mathsf{IFF}_{1,2,4,2}$	6	0.05706	428.610111	TO
$\mathsf{IFF}_{1,2,4,2}$	6	0.06602	331.335254	TO
$\mathsf{IFF}_{1,2,4,2}$	6	0.428498	234.21308	TO
$\mathsf{IFF}_{1,2,4,2}$	6	10.410062	36.590249	TO
$\mathsf{IFF}_{1,3,0,2}$	2	1.467944	77.059827	141.641348
$\mathsf{IFF}_{1,3,0,2}$	2	10.335019	11.872165	141.641348
$\mathsf{IFF}_{1,3,0,2}$	3	0.381255	116.006194	213.48041
$\mathsf{IFF}_{1,3,0,2}$	3	10.335019	17.900276	213.48041
$\mathsf{IFF}_{1,3,0,2}$	4	0.388136	154.878372	error
$\mathsf{IFF}_{1,3,0,2}$	4	10.335019	23.726062	error
$\mathsf{IFF}_{1,3,0,2}$	5	0.38856	161.283435	error
$\mathsf{IFF}_{1,3,0,2}$	5	10.335019	29.889981	error
$\mathsf{IFF}_{1,3,0,2}$	6	0.20606	136.066608	error
$\mathsf{IFF}_{1,3,0,2}$	6	10.335019	35.956934	error
$\mathsf{IFF}_{1,3,0,4}$	2	0.383022	77.237247	141.917578
$\mathsf{IFF}_{1,3,0,4}$	2	10.335019	11.878174	141.917578
$\mathsf{IFF}_{1,3,0,4}$	3	0.317788	115.205143	408.103823
$\mathsf{IFF}_{1,3,0,4}$	3	10.335019	17.901375	408.103823
$\mathsf{IFF}_{1,3,0,4}$	4	0.005135	799.717785	TO
$\mathsf{IFF}_{1,3,0,4}$	4	0.00579	671.655286	TO
$\mathsf{IFF}_{1,3,0,4}$	4	0.041855	281.414459	TO
$\mathsf{IFF}_{1,3,0,4}$	4	0.362956	152.851509	TO
$\mathsf{IFF}_{1,3,0,4}$	4	10.335019	23.868235	TO
$\mathsf{IFF}_{1,3,0,4}$	5	0.033362	317.795578	TO
$\mathsf{IFF}_{1,3,0,4}$	5	0.360447	158.648246	TO
$\mathsf{IFF}_{1,3,0,4}$	5	10.335019	29.803402	TO

Table 19: Meaningful tolerances $\varepsilon > 0$ that improve running time on the identification (friend of foe) benchmark (Part 3 of 5).

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{IFF}_{1,3,0,4}$	6	0.00638	908.686747	error
$\mathsf{IFF}_{1,3,0,4}$	6	0.018979	747.082145	error
$\mathsf{IFF}_{1,3,0,4}$	6	0.044894	393.328231	error
$\mathsf{IFF}_{1,3,0,4}$	6	0.36543	234.580844	error
$\mathsf{IFF}_{1,3,0,4}$	6	10.335019	35.527858	error
$\mathsf{IFF}_{1,3,2,0}$	2	0.383022	77.624215	141.91681
$\mathsf{IFF}_{1,3,2,0}$	2	10.335019	11.977965	141.91681
$\mathsf{IFF}_{1,3,2,0}$	3	0.132984	413.774795	514.208244
$\mathsf{IFF}_{1,3,2,0}$	3	0.317788	115.672065	514.208244
$\mathsf{IFF}_{1,3,2,0}$	3	10.335019	17.909951	514.208244
$\mathsf{IFF}_{1,3,2,0}$	4	0.004996	542.509162	error
$\mathsf{IFF}_{1,3,2,0}$	4	0.005611	413.818409	error
$\mathsf{IFF}_{1,3,2,0}$	4	0.023733	284.820704	error
$\mathsf{IFF}_{1,3,2,0}$	4	0.362956	152.84294	error
$\mathsf{IFF}_{1,3,2,0}$	4	10.335019	23.841004	error
$\mathsf{IFF}_{1,3,2,0}$	5	0.005733	514.046505	TO
$\mathsf{IFF}_{1,3,2,0}$	5	0.010987	352.233895	TO
$\mathsf{IFF}_{1,3,2,0}$	5	0.360666	191.511139	TO
$\mathsf{IFF}_{1,3,2,0}$	5	10.335019	30.038704	TO
$\mathsf{IFF}_{1,3,2,0}$	6	0.005239	1265.199023	TO
$\mathsf{IFF}_{1,3,2,0}$	6	0.005559	1101.958233	TO
$\mathsf{IFF}_{1,3,2,0}$	6	0.005679	486.164358	TO
$\mathsf{IFF}_{1,3,2,0}$	6	0.021117	326.146578	TO
$\mathsf{IFF}_{1,3,2,0}$	6	0.360271	166.268666	TO
$\mathsf{IFF}_{1,3,2,0}$	6	10.335019	36.182132	TO
$\mathsf{IFF}_{1,3,2,4}$	2	0.383022	77.501331	142.514123
$\mathsf{IFF}_{1,3,2,4}$	2	10.335019	11.963282	142.514123
$\mathsf{IFF}_{1,3,2,4}$	3	0.132984	414.983334	515.357185
$\mathsf{IFF}_{1,3,2,4}$	3	0.317788	115.328902	515.357185
$\mathsf{IFF}_{1,3,2,4}$	3	10.335019	17.867992	515.357185
$\mathsf{IFF}_{1,3,2,4}$	4	0.005135	943.774142	TO
$\mathsf{IFF}_{1,3,2,4}$	4	0.00579	814.402909	TO
$\mathsf{IFF}_{1,3,2,4}$	4	0.041855	281.808707	TO
$\mathsf{IFF}_{1,3,2,4}$	4	0.362956	152.863577	TO
$\mathsf{IFF}_{1,3,2,4}$	4	10.335019	23.984778	TO
$\mathsf{IFF}_{1,3,2,4}$	5	0.033362	319.50252	TO
$\mathsf{IFF}_{1,3,2,4}$	5	0.360447	159.380286	TO
$\mathsf{IFF}_{1,3,2,4}$	5	10.335019	29.55917	TO
$\mathsf{IFF}_{1,3,2,4}$	6	0.005099	1212.249705	TO
$\mathsf{IFF}_{1,3,2,4}$	6	0.005179	918.751541	TO
$\mathsf{IFF}_{1,3,2,4}$	6	0.020398	754.95113	TO
$\mathsf{IFF}_{1,3,2,4}$	6	0.044894	398.138069	TO
$\mathsf{IFF}_{1,3,2,4}$	6	0.36543	237.619894	TO
$\mathsf{IFF}_{1,3,2,4}$	6	10.335019	35.887154	TO
$\mathsf{IFF}_{1,3,4,2}$	2	0.383022	90.340224	165.237963
$\mathsf{IFF}_{1,3,4,2}$	2	10.335019	13.809389	165.237963
$\mathsf{IFF}_{1,3,4,2}$	3	0.044519	488.611768	error
$\mathsf{IFF}_{1,3,4,2}$	3	0.317788	135.421368	error
$\mathsf{IFF}_{1,3,4,2}$	3	10.335019	20.737646	error
$\mathsf{IFF}_{1,3,4,2}$	4	0.362956	178.707634	TO
$\mathsf{IFF}_{1,3,4,2}$	4	10.335019	27.602859	TO
$\mathsf{IFF}_{1,3,4,2}$	5	0.004695	1135.059032	TO
$\mathsf{IFF}_{1,3,4,2}$	5	0.005234	942.89021	TO
$\mathsf{IFF}_{1,3,4,2}$	5	0.039595	410.545458	TO
$\mathsf{IFF}_{1,3,4,2}$	5	0.360686	223.484471	TO
$\mathsf{IFF}_{1,3,4,2}$	5	10.335019	34.628805	TO
$\mathsf{IFF}_{1,3,4,2}$	6	0.004599	1252.686854	TO
$\mathsf{IFF}_{1,3,4,2}$	6	0.005719	645.729229	TO
$\mathsf{IFF}_{1,3,4,2}$	6	0.035855	421.755427	TO
$\mathsf{IFF}_{1,3,4,2}$	6	0.360351	193.534618	TO

Table 20: Meaningful tolerances $\varepsilon > 0$ that improve running time on the identification (friend of foe) benchmark (Part 4 of 5).

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{IFF}_{1,3,4,2}$	6	10.335019	41.381999	TO
$\mathsf{IFF}_{2,3,0,2}$	2	1.939408	74.737667	137.343105
$\mathsf{IFF}_{2,3,0,2}$	2	10.335019	11.74911	137.343105
$\mathsf{IFF}_{2,3,0,2}$	3	0.293408	208.601651	TO
$\mathsf{IFF}_{2,3,0,2}$	3	1.884998	113.63439	TO
$\mathsf{IFF}_{2,3,0,2}$	3	10.335019	17.764729	TO
$\mathsf{IFF}_{2,3,0,2}$	4	0.245361	405.335497	TO
$\mathsf{IFF}_{2,3,0,2}$	4	0.283536	278.305535	TO
$\mathsf{IFF}_{2,3,0,2}$	4	1.801674	151.392554	TO
$\mathsf{IFF}_{2,3,0,2}$	4	10.335019	23.751316	TO
$\mathsf{IFF}_{2,3,0,2}$	5	0.274026	286.240798	TO
$\mathsf{IFF}_{2,3,0,2}$	5	1.631319	159.607277	TO
$\mathsf{IFF}_{2,3,0,2}$	5	10.335019	29.65847	TO
$\mathsf{IFF}_{2,3,0,2}$	6	0.283859	260.947321	TO
$\mathsf{IFF}_{2,3,0,2}$	6	1.631274	166.078665	TO
$\mathsf{IFF}_{2,3,0,2}$	6	10.335019	35.601055	TO
$\mathsf{IFF}_{2,3,0,4}$	2	1.939408	74.241271	136.696771
$\mathsf{IFF}_{2,3,0,4}$	2	10.335019	11.72567	136.696771
$\mathsf{IFF}_{2,3,0,4}$	3	0.293408	207.000162	TO
$\mathsf{IFF}_{2,3,0,4}$	3	1.884998	112.703548	TO
$\mathsf{IFF}_{2,3,0,4}$	3	10.335019	17.591871	TO
$\mathsf{IFF}_{2,3,0,4}$	4	0.245361	400.729476	TO
$\mathsf{IFF}_{2,3,0,4}$	4	0.283536	275.260925	TO
$\mathsf{IFF}_{2,3,0,4}$	4	1.801674	150.027988	TO
$\mathsf{IFF}_{2,3,0,4}$	4	10.335019	23.365801	TO
$\mathsf{IFF}_{2,3,0,4}$	5	0.274026	283.948861	TO
$\mathsf{IFF}_{2,3,0,4}$	5	1.631319	158.464371	TO
$\mathsf{IFF}_{2,3,0,4}$	5	10.335019	29.625754	TO
$\mathsf{IFF}_{2,3,0,4}$	6	0.283859	259.304075	TO
$\mathsf{IFF}_{2,3,0,4}$	6	1.631274	164.947204	TO
$\mathsf{IFF}_{2,3,0,4}$	6	10.335019	35.405053	TO
$\mathsf{IFF}_{2,3,2,0}$	2	1.939408	75.681753	138.915399
$\mathsf{IFF}_{2,3,2,0}$	2	10.335019	11.963679	138.915399
$\mathsf{IFF}_{2,3,2,0}$	3	0.318409	211.942822	TO
$\mathsf{IFF}_{2,3,2,0}$	3	1.884998	115.205659	TO
$\mathsf{IFF}_{2,3,2,0}$	3	10.335019	18.052294	TO
$\mathsf{IFF}_{2,3,2,0}$	4	0.274243	409.427494	TO
$\mathsf{IFF}_{2,3,2,0}$	4	0.305782	281.320625	TO
$\mathsf{IFF}_{2,3,2,0}$	4	1.801674	153.524426	TO
$\mathsf{IFF}_{2,3,2,0}$	4	10.335019	24.174249	TO
$\mathsf{IFF}_{2,3,2,0}$	5	0.306825	451.002362	TO
$\mathsf{IFF}_{2,3,2,0}$	5	0.307405	322.661713	TO
$\mathsf{IFF}_{2,3,2,0}$	5	1.650738	194.726674	TO
$\mathsf{IFF}_{2,3,2,0}$	5	10.335019	30.014687	TO
$\mathsf{IFF}_{2,3,2,0}$	6	0.294539	361.6288	TO
$\mathsf{IFF}_{2,3,2,0}$	6	0.305259	265.162844	TO
$\mathsf{IFF}_{2,3,2,0}$	6	1.631694	168.737673	TO
$\mathsf{IFF}_{2,3,2,0}$	6	10.335019	35.979194	TO
$\mathsf{IFF}_{2,3,2,4}$	2	0.383022	77.552645	142.687529
$\mathsf{IFF}_{2,3,2,4}$	2	10.335019	11.985888	142.687529
$\mathsf{IFF}_{2,3,2,4}$	3	0.044519	504.559813	error
$\mathsf{IFF}_{2,3,2,4}$	3	0.280943	407.277901	error
$\mathsf{IFF}_{2,3,2,4}$	3	0.317788	115.656604	error
$\mathsf{IFF}_{2,3,2,4}$	3	10.335019	17.916342	error
$\mathsf{IFF}_{2,3,2,4}$	4	0.005215	540.656027	TO
$\mathsf{IFF}_{2,3,2,4}$	4	0.00579	411.262733	TO
$\mathsf{IFF}_{2,3,2,4}$	4	0.041855	282.378471	TO
$\mathsf{IFF}_{2,3,2,4}$	4	0.362956	153.316284	TO
$\mathsf{IFF}_{2,3,2,4}$	4	10.335019	23.988811	TO

Table 21: Meaningful tolerances $\varepsilon > 0$ that improve running time on the identification (friend of foe) benchmark (Part 5 of 5).

Model	k	ε	Time [1] (s) ($\varepsilon > 0$)	Time [1] (s) ($\varepsilon = 0$)
$\mathsf{IFF}_{2,3,2,4}$	5	0.033362	321.565696	TO
$\mathsf{IFF}_{2,3,2,4}$	5	0.360447	160.516355	TO
$\mathsf{IFF}_{2,3,2,4}$	5	10.335019	30.261006	TO
$\mathsf{IFF}_{2,3,2,4}$	6	0.004699	982.385689	TO
$\mathsf{IFF}_{2,3,2,4}$	6	0.005279	820.318127	TO
$\mathsf{IFF}_{2,3,2,4}$	6	0.005699	689.00884	TO
$\mathsf{IFF}_{2,3,2,4}$	6	0.006159	559.359959	TO
$\mathsf{IFF}_{2,3,2,4}$	6	0.044894	398.010023	TO
$\mathsf{IFF}_{2,3,2,4}$	6	0.36543	237.51286	TO
$\mathsf{IFF}_{2,3,2,4}$	6	10.335019	35.970266	TO
$\mathsf{IFF}_{2,3,4,0}$	2	1.939408	75.577512	138.79073
$\mathsf{IFF}_{2,3,4,0}$	2	10.335019	11.969358	138.79073
$\mathsf{IFF}_{2,3,4,0}$	3	0.319778	210.292622	TO
$\mathsf{IFF}_{2,3,4,0}$	3	1.884998	114.124223	TO
$\mathsf{IFF}_{2,3,4,0}$	3	10.335019	17.86182	TO
$\mathsf{IFF}_{2,3,4,0}$	4	0.275622	408.491556	TO
$\mathsf{IFF}_{2,3,4,0}$	4	0.307001	280.316039	TO
$\mathsf{IFF}_{2,3,4,0}$	4	1.801674	152.52919	TO
$\mathsf{IFF}_{2,3,4,0}$	4	10.335019	23.982122	TO
$\mathsf{IFF}_{2,3,4,0}$	5	0.307265	414.838747	TO
$\mathsf{IFF}_{2,3,4,0}$	5	0.307885	286.888755	TO
$\mathsf{IFF}_{2,3,4,0}$	5	1.799511	159.32949	TO
$\mathsf{IFF}_{2,3,4,0}$	5	10.335019	29.811451	TO
$\mathsf{IFF}_{2,3,4,0}$	6	0.308439	230.064947	TO
$\mathsf{IFF}_{2,3,4,0}$	6	1.782093	134.438962	TO
$\mathsf{IFF}_{2,3,4,0}$	6	10.335019	36.041919	TO
$\mathsf{IFF}_{2,3,4,2}$	2	1.939408	75.688371	139.109953
$\mathsf{IFF}_{2,3,4,2}$	2	10.335019	11.914726	139.109953
$\mathsf{IFF}_{2,3,4,2}$	3	0.319778	210.793983	TO
$\mathsf{IFF}_{2,3,4,2}$	3	1.884998	114.750093	TO
$\mathsf{IFF}_{2,3,4,2}$	3	10.335019	17.852247	TO
$\mathsf{IFF}_{2,3,4,2}$	4	0.275622	409.50455	TO
$\mathsf{IFF}_{2,3,4,2}$	4	0.307001	281.18696	TO
$\mathsf{IFF}_{2,3,4,2}$	4	1.801674	153.032512	TO
$\mathsf{IFF}_{2,3,4,2}$	4	10.335019	23.946691	TO
$\mathsf{IFF}_{2,3,4,2}$	5	0.307265	416.466311	TO
$\mathsf{IFF}_{2,3,4,2}$	5	0.307885	287.952489	TO
$\mathsf{IFF}_{2,3,4,2}$	5	1.799511	160.001563	TO
$\mathsf{IFF}_{2,3,4,2}$	5	10.335019	30.061209	TO
$\mathsf{IFF}_{2,3,4,2}$	6	0.308439	230.52634	TO
$\mathsf{IFF}_{2,3,4,2}$	6	1.782093	134.488155	TO
$\mathsf{IFF}_{2,3,4,2}$	6	10.335019	35.849211	TO

Table 22: Ablation study on the same instances as Table 1.

Model	$\lvert S\rvert$	$\lvert A\rvert$	$\lvert\mathcal{O}\rvert$	n	k	Time ours (s)	Time without prunning (s)	Time[1] (s)
$\mathsf{RS}_{3,1,2}$	26	7	3	2	4	0.233528	0.237096	1.735581
$\mathsf{RS}_{3,1,3}$	34	8	3	3	3	0.058124	0.054495	2.433545
$\mathsf{RS}_{4,1,2}$	32	7	3	2	7	154.249	MO	TO
$\mathsf{RS}_{6,1,2}$	35	7	3	2	6	18.6796	29.628089	43.165601
$\mathsf{RS}_{3,2,7}$	246	12	3	21	5	459.885	MO	TO
$\mathsf{RS}_{3,4,7}$	574	12	3	35	5	783.78	MO	TO
$\mathsf{RS}_{3,6,7}$	192	12	3	7	6	3476.53	MO	TO
$\mathsf{RS}_{3,1,2}$	26	7	3	2	5	2.06783	4.915408	2.280673
$\mathsf{RS}_{4,1,2}$	32	7	3	2	6	17.1989	59.258818	24.329221