Anticipating Oblivious Opponents in Stochastic Games

Shadi Tasdighi Kalat, Sriram Sankaranarayanan and Ashutosh Trivedi

University of Colorado Boulder, USA
Email: `first.lastname @colorado.edu`

Abstract. We present an approach for systematically anticipating the actions and policies employed by *oblivious* environments in concurrent stochastic games, while maximizing a reward function. Our main contribution lies in the synthesis of a finite *information state machine* (ISM) whose alphabet ranges over the actions of the environment. Each state of the ISM is mapped to a belief state about the policy used by the environment. We introduce a notion of consistency that guarantees that the belief states tracked by the ISM stays within a fixed distance of the precise belief state obtained by knowledge of the full history. We provide methods for checking consistency of an automaton and a synthesis approach which, upon successful termination, yields an ISM. We construct a Markov Decision Process (MDP) that serves as the starting point for computing optimal policies for maximizing a reward function defined over plays. We present an experimental evaluation over benchmark examples including human activity data for tasks such as cataract surgery and furniture assembly, wherein our approach successfully anticipates the policies and actions of the environment in order to maximize the reward.

1 Introduction

Concurrent stochastic games [17, 15, 16, 13, 26, 40] offer a natural abstraction for modeling conservative decision-making in the presence of multiple agents in a shared and uncertain environment. In this scenario, the objective of the *Ego* agent—player $\mathcal{P}_1$—is to maximize their desired outcome irrespective of the decisions taken by other agents, represented here as a single agent that we term player $\mathcal{P}_2$ [7]. In a *zero-sum game*, the objective of player $\mathcal{P}_1$ is deemed to be in direct conflict with player $\mathcal{P}_2$. The opposite scenario assumes *cooperation*, wherein $\mathcal{P}_2$'s actions are aimed to maximize the reward for $\mathcal{P}_1$. In this paper, we study another "extreme", wherein $\mathcal{P}_2$ is assumed to be *oblivious*. Their actions are chosen from a predefined set of policies or objectives that are not affected by the actions of $\mathcal{P}_1$. We will show that in such a setting, player $\mathcal{P}_1$ needs to *anticipate* $\mathcal{P}_2$'s moves to maximize their own reward.

Consider a game of Rock-paper-scissors (RPS) against an oblivious adversary. Recall that at each turn, players $\mathcal{P}_1$ and $\mathcal{P}_2$ simultaneously reveal their choice with a show of hands, and both players receive values (Cf. Figure 1) based on straightforward circular-dominance rules (rock defeats scissors, scissors defeats paper, paper defeats rock). The repeated, oblivious RPS can be modeled as a single state concurrent stochastic game, where the goal of player $\mathcal{P}_1$ is to maximize the sequence of rewards

	r_2	p_2	s_2
r_1	0	-1	1
p_1	1	0	-1
s_1	-1	1	0

	$P(r_2 \mid t,\pi)$	$P(p_2 \mid t,\pi)$	$P(s_2 \mid t,\pi)$
π_1	0.5	0.5	0
π_2	0	0.5	0.5
π_3	0.5	0	0.5
π_4	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$

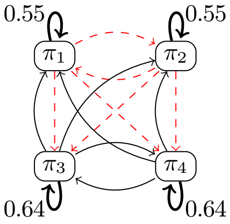


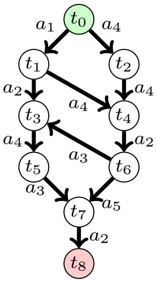
Fig. 1. Rock-paper-scissors (RPS) game arena. Here actions r_i, p_i, and s_i correspond to the choices of "rock", "paper" and "scissors" by player $\mathcal{P}_i$; (**left**) Reward table; (**mid**) player $\mathcal{P}_2$ policies; and (**right**) Markov chain modeling policy change for $\mathcal{P}_2$. The dashed red edges have probability 0.15 whereas the solid edges have probability 0.12.

over a given, potentially infinite, horizon. Considering the conventional interpretation of an adversarial opponent, the expected value of the game remains at 0.

The oblivious RPS "game" is illustrated in Figure 1, where the set of policies (π_1,π_2,π_3, and π_4) used by player $\mathcal{P}_2$ is presented in the table to the right. In the proposed scenario, we assume the following: (a) player $\mathcal{P}_1$ observes the *past* actions of player $\mathcal{P}_2$ but the *current* action of one player is not observable by the other; (b) player $\mathcal{P}_2$ is restricted to playing one of the policies $\{\pi_1,\pi_2,\pi_3,\pi_4\}$ but this choice is *not observable* by $\mathcal{P}_1$; and (c) *policy change:* at each step, player $\mathcal{P}_2$ may shift from the current policy to a new one. This shift is modeled by a Markov chain wherein each state of the chain is labeled by a policy. From player $\mathcal{P}_1$'s perspective, although the policies of player $\mathcal{P}_2$ are known, they are unobservable. Consequently, the problem can be framed as a partially observable MDP (POMDP). This POMDP is the result of merging the original arena with player $\mathcal{P}_2$'s policy set. Framing this as a POMDP permits the use of standard POMDP solution approaches [9]. However, "exact" POMDP planning is undecidable [28]. Furthermore, translating from oblivious games to POMDPs obscures the specialized structure of the problem.

Action/Tool Anticipation in Human-Robot Cooperative Tasks: In scenarios involving humans working with autonomous agents, the ability to "guess" the intent of the human can be critical in ensuring the success of the overall task. Consider a scenario where $\mathcal{P}_2$ is engaged in a complex task involving a sequence of steps such as assembling a piece of furniture [5] or performing a cataract surgery [1].

The task execution is captured by a task graph whose nodes model different states encountered during task execution and edges are labeled with the tool/action that is needed to move from one stage to another. Fig. 2 shows such a graph: the states $S_1{=}\{t_0,\ldots,t_8\}$ represent assembly stages for the corresponding component, while $A{=}\{a_1,\ldots,a_5\}$ represents the actions taken. Multiple edges from the same node represent possible choices that can be made by $\mathcal{P}_2$. The policies of $\mathcal{P}_2$ dictate the choices made by $\mathcal{P}_2$ for each non-terminal state. For some of the states with just one outgoing edge, there is just one choice to be made. However, for states with multiple outgoing edges, the policy dictates the probability distribution of the choice. The policies allow us to model "correlations" in $\mathcal{P}_2$'s action: For instance, policy π_1 models the rule: $\mathcal{P}_2$ chooses tool a_1 at state t_0 and they will choose a_2 at state t_1 with 90% probability. The goal of $\mathcal{P}_1$ is to accurately anticipate $\mathcal{P}_2$'s choice of the



$\pi_1 : t_0 \mapsto a_1, t_1 \mapsto \{a_2 : 0.9, a_4 : 0.1\}$
$\quad t_6 \mapsto \{a_3 : 0.5, a_5 : 0.5\}, \cdots$

$\pi_2 : t_0 \mapsto a_4, t_6 \mapsto \{a_3 : 0.5, a_5 : 0.5\}, \cdots$

$\pi_3 : t_0 \mapsto \{a_1 : 0.5, a_4 : 0.5\}, t_6 \mapsto \{a_3 : 0.5, a_5 : 0.5\},$
$\quad t_1 \mapsto a_2, \ldots$

$\pi_4 : t_0 \mapsto a_1, t_1 \mapsto \{a_2 : 0.5, a_4 : 0.5\},$
$\quad t_6 \mapsto \{a_3 : 0.5, a_5 : 0.5\}, \ldots$

Fig. 2. States of a furniture assembly task and policies for task completion.

next tool in order to perform a cooperative action (eg., pre-fetch the tool to help $\mathcal{P}_2$ or automatically take steps to protect $\mathcal{P}_2$ against a known hazard). We model this using the reward structure: if $\mathcal{P}_1$ correctly predicts the next action of $\mathcal{P}_2$ they obtain a positive reward. However, failure to do so incurs a negative reward. By assuming a set of policies for $\mathcal{P}_2$, our approach moves the prediction problem from one of simply predicting action sequences to first predicting the policy (or the internal logic behind $\mathcal{P}_2$'s actions) and then predicting the action given the policy. Section 7 demonstrates how we can use actual observation data from real-life cataract surgeries and furniture assembly tasks to not just learn the task graph model but also infer policies. In doing so, our approach can produce policies for $\mathcal{P}_1$ that predict the next action with upto 40% accuracy even when there are more than 30 tools/actions to choose from at each step.

Contributions. We introduce the framework of anticipation games (Section 2). 1. *Consistent Information State Machines:* We define the notion of a finite information state machine (ISM) over an alphabet consisting of states and $\mathcal{P}_2$ actions (Section 3). We introduce the concept of $\lambda-$consistency that is similar to an approximate bisimuation relation and show how to check if a given state machine is λ consistent using linear arithmetic SAT-Modulo Theory (SMT) solvers. Next, we provide a semi-algorithm that upon success can synthesize such a machine (Section 4). We provide simple conditions that guarantee the successful termination of our algorithm with a finite state consistent ISM (Section 6). 2. *Policy Synthesis for $\mathcal{P}_1$:* Next we show that a composition of a finite state ISM with the game yields a MDP that forms the basis of finding a policy for $\mathcal{P}_1$ (Section 5). We bound the distances between the transition probabilities and reward functions of the infinite state belief MDP and the finite state approximation. By leveraging a recent result by Subramanian et al. [44], we bound the gap between the optimal belief MDP value function and that of our finite approximation. 3. *Robustness:* In Section 6, we establish bounds on the performance degradation, if $\mathcal{P}_2$ deviates from the assumptions. 4. *Empirical Evaluation:* Finally, we present an empirical evaluation of our work against some challenging benchmarks (Section 7). We show that our approach can clearly anticipate the policies and actions of the other player to maximize the overall reward. In particular, we use two datasets – an IKEA furniture assembly dataset consisting of sequence of actions taken by human assemblers for different furniture models [5] and a sequence of tools used in 25 different cataract surgeries [1]. We use an automata learning tool flexfringe [46] to learn the task model and a simple edge set based clustering to learn policies. We demonstrate how our approach computes policies for $\mathcal{P}_1$ that maximize the ability to predict the next tool choice of $\mathcal{P}_2$.

Related Work. Partially observable stochastic games (POSGs) are a subset of stochastic games where agents have partial information about the state of the environment. Within this paradigm, agents are allowed to have conflicting, or similar objectives, reward structures, and strategies [9, 8, 6]. Solution techniques developed for POSGs are build upon approaches to solve POMDPs such as value iteration and policy iteration [4]. Solving finite-horizon POMDP is PSPACE-complete [36], and solving infinite-horizon POMDPs have been shown to be undecidable [29]. A variety of approximate solution techniques have been introduced for general POMDPs including Point-Based Value iteration [37, 39, 42, 25, 38], grid-based belief MDP approximations [19], semi-MDP approximations [45, 43] and compressing belief states using features [22]. In addition, methods such as POMCP (Partially Observable Monte Carlo Planning [41, 27]), leverage sampling-based approaches to estimate belief states and approximate the value function. These approaches are not easy to compare to the approach in this paper since our approach is tailored explicitly to POMDPs derived from anticipation games for oblivious adversaries. Our approach is closely related to those that group belief states together using bisimulation quotients [10, 11, 20]. A key distinction is that the approach presented here is an approximate notion of bisimulation wherein we guarantee that our information state machines track the precise belief state within a distance of λ in a suitable norm. Thus, we exploit the special structure of the games studied here and prove that finite approximate bisimulations always exist for suitable choice of the parameters.

While traditional POMDP solvers often work with the belief space, there have been approaches that leverage historical information to make decisions, either by directly maintaining a history or by approximating it. The complexity of solving this problem grows exponentially with the length of history [24]. The results in [3] discuss this issue and address the trade-offs between memory usage and solution quality. To overcome this issue, [23], introduces the concept of finite-memory controllers. In another work, [30] investigates an instance-based learning approach for POMDPs, maintaining a set of histories to guide action selection. Similarly, [21, 14] use looping suffix trees to represent the hidden state in deterministic finite POMDPs. This work is later extended to [31], which fixes the size of the policy graph to find the best policy of this size, and [32], that performs stochastic gradient descent on finite-state controller parameters, which guarantees local optimality of the solution. However, note that none of these techniques provide guarantees on the quality of the approximation or the solution so obtained. In this paper, we obtain such guarantees but for the limited case of POMDPs arising from the anticipation games and oblivious adversaries.

Our approach is an instance of the approximate information state introduced by Subramanian et al [44], as a compression of history which is sufficient to evaluate approximate performance, and predict itself. Yang et al [48] specialize this framework to discrete approximate information states but their work learns the automaton from finite samples by solving an expensive nonlinear optimization problem. In this paper, we assume knowledge of the underlying game and opponent policies to construct a finite state machine that is guaranteed to be an approximation information state generator.

The problem of anticipating moves of an oblivious opponent has similarities to the well-studied problem of intent inference or goal recognition [2, 12, 49]. Our approach models the other player's policies which makes the intent inference problem quite simple. On the other hand, our approach allows intents to change in a stochastic manner and more significantly, it folds in the intent inference with planning in a single algorithm.

2 Problem Definition

A *probability distribution* d: $X \to [0,1]$ over a finite set X satisfies $\sum_{s \in X} d(s) = 1$. Let $\mathcal{D}(X)$ represent the set of all probability distributions over X. The distribution d over $X = \{x_1, x_2, \cdots, x_m\}$ is written $\{x_1 : p_1, ..., x_m : p_m\}$ where $p_i = d(x_i)$ for $i \in [m]$. For a natural number $n \geq 1$, let $[n] = \{1, 2, ..., n\}$. Bold case letters denote vectors $\mathbf{b} \in \mathbb{R}^n$. The i^{th} component of $\mathbf{b}$ is denoted as b_i.

A *Markov decision process* (MDP) $\mathcal{M}$ is a tuple $\langle S, A, P, R \rangle$ where S is a finite set of states, A is a finite set of actions, $P : S \times A \to \mathcal{D}(S)$ is the probabilistic transition function, and $R : S \times A \to \mathbb{R}$ is a scalar valued reward function. We write $P(s'|s,a)$ for the probability of state s' if action a is applied to state s. In a two player concurrent game, the set of actions are partitioned between player $\mathcal{P}_1$ and $\mathcal{P}_2$. Transitions of the game are determined by joint actions of both players.

Definition 1 (Concurrent Stochastic Game Arena: Syntax). *A concurrent stochastic game arena $\mathcal{G}$ is a tuple $\langle S, A^{(1)}, A^{(2)}, P, R \rangle$ wherein S is a finite set of states, $A^{(1)}$ and $A^{(2)}$ are disjoint sets of actions for players $\mathcal{P}_1$ and $\mathcal{P}_2$, respectively, $P : S \times A^{(1)} \times A^{(2)} \to \mathcal{D}(S)$ is the joint probabilistic transition function, and $R : S \times A^{(1)} \times A^{(2)} \to \mathbb{R}$ is a reward function for $\mathcal{P}_1$.*

We assume that player $\mathcal{P}_2$ selects their policy from one of n different stochastic policies from the set $\Pi = \{\pi_1, ..., \pi_n\}$, wherein each $\pi_i : S \to \mathcal{D}(A^{(2)})$ represents a map from states to probability distributions over actions in $A^{(2)}$. Let $\pi_i(s,a)$ denote the probability that action a is chosen from state s for policy π_i.

Example 1. Consider the RPS example discussed in the introduction (Figure 1). The state set is a singleton: $S = \{t\}$. We have three actions each for players 1,2: $A^{(1)} = \{r_1, p_1, s_1\}$ and $A^{(2)} = \{r_2, p_2, s_2\}$, corresponding to choices of "rock", "paper" and "scissors", respectively. The transition probabilities are simply $P(t|t,a,b) = 1$ for all $a \in A^{(1)}, b \in A^{(2)}$. The reward for $\mathcal{P}_1$ is the familiar one from the game of rock-paper-scissors, and is shown in Figure 1 (left) $\mathcal{P}_2$ plays one of four possible policies shown in the middle table of Figure 1.

Assumption 1 (Observation and Obliviousness) *We assume that: (a) $\mathcal{P}_1$ observes the* past *actions of $\mathcal{P}_2$ but the* current *action is not observable. (b) $\mathcal{P}_2$ is restricted to playing one of the policies $\{\pi_1, ..., \pi_n\}$ but this choice is* not observable *by $\mathcal{P}_1$.*

Policy Change Model: We assume that $\mathcal{P}_2$ can change policies at each step depending on their current policies according to a Markov chain with n states labeled by the corresponding policies $\pi_1, ..., \pi_n$. Let T represent the transition matrix of this Markov chain such that the entry $T_{ij} = P(\pi_j|\pi_i)$ represents the probability of $\mathcal{P}_2$ switching their policy to π_j given that their current policy is π_i. Returning to Example 1, the Markov chain for switching between the four policies $\pi_1,...,\pi_4$ is shown in Figure 1 (right). A partially observable MDP (POMDP) is a tuple $\langle S,A,P,R,\Omega,O\rangle$ where $\langle S,A,P,R\rangle$ is an MDP, Ω is a finite set of *observations*, and $O:S \to \Omega$ is (deterministic) observation map. The semantics of an OGA under Assumptions 1 can be given as a POMDP.

Definition 2 (OGA: Semantics). *The semantics of an OGA* $\mathcal{G} = \langle S,A^{(1)},A^{(2)},P,R\rangle$ *with player* $\mathcal{P}_2$ *policy set* $\{\pi_1,...,\pi_n\}$ *and policy change given by a Markov chain with transition matrix* T *is a partially observable MDP (POMDP)* $\mathcal{M}' = \langle S',A' = A^{(1)},P',R',\Omega = S,O\rangle$ *where*

- $S' = S \times [n]$ *wherein each state* (s_i,j) *represents a state* $s_i \in S$ *and an index* $j \in [n]$ *representing the current policy being employed by* $\mathcal{P}_2$
- *The probability of a transition* $P((s',j')|(s,j),a)$ *is given as:*

$$P((s',j')|(s,j),a) = T_{jj'} \cdot \sum_{a_2 \in A^{(2)}} (\pi_j(s,a_2) \cdot P(s'|s,a,a_2))$$

- *The reward function is given as:* $R((s,j),a) = \sum_{a_2 \in A^{(2)}} \pi_j(s,a_2) \cdot R(s,a,a_2)$, *and*
- *The observation map* $O:S' \to \Omega$ *is defined as* $(s_i,j) \in S' \mapsto s_i$.

While translating into a POMDP allows us access to a variety of approaches to solving POMDPs [9], they are computationally expensive and ignore the specialized structure of the problem at hand. In this paper, we will work with the original two player game setup to directly exploit the special problem structure at hand.

Our goal is to compute a *finite memory* policy $\pi^{(1)} : S \times M \mapsto A^{(1)}$ that maximizes the expected discounted reward for $\mathcal{P}_1$ with given discount factor $0 < \gamma < 1$. The structure and construction of the required memory M over the states and actions of $\mathcal{P}_2$ is discussed in subsequent sections.

3 Information State Machine and Consistency

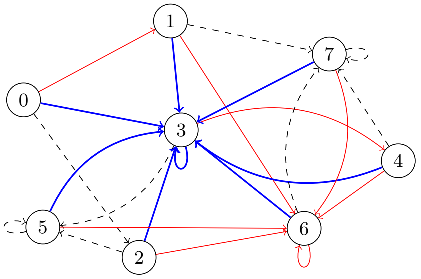
The main approach is to use a sequence of observations of states and $\mathcal{P}_2$ actions to infer a *belief state* $\mathbf{b}$ over the player's policies.

Definition 3 (Belief State). *A belief state* $\mathbf{b} : (b_1,...,b_n) \in \mathbb{R}^n$ *is a vector wherein the* i^{th} *component* b_i *represents* $\mathcal{P}_1$*'s belief that* $\mathcal{P}_2$ *is employing policy* $\pi_i \in \Pi$. *Note that* $b_i \geq 0$ *for all* $i \in [n]$ *and* $\sum_{i=1}^{n} b_i = 1$.

Let $\mathcal{B}_n = \{\mathbf{b} \in \mathbb{R}^n \mid (\forall i \in [n]) \; b_i \geq 0 \; \wedge \; \sum_{i=1}^{n} b_i = 1\}$ denote the set of all belief state vectors in $\mathbb{R}^n$. The uniform belief state $\mathbf{b}_u$ is given by $(\frac{1}{n},...,\frac{1}{n})$. We define two operations over a belief state: (a) conditioning a belief state given some observation and (b) capturing the effect of policy change on a belief state.

Let $\mathbf{b}$ be a belief state and (s,a_2) represent an observation where $s \in S$ and $a_2 \in A^{(2)}$ represent states of the game and actions for $\mathcal{P}_2$. The belief state $\mathbf{b}' = \mathsf{condition}(\mathbf{b},s,a_2)$ is obtained by conditioning $\mathbf{b}$ on the observation (s,a_2):

$$b_i' = \mathsf{condition}(\mathbf{b},s,a_2) = \frac{\pi_i(s,a_2) b_i}{\sum_{j=1}^{n} \pi_j(s,a_2) b_j}. \tag{1}$$



State	Belief
0	(0.25, 0.25, 0.25, 0.25)
1	(0.29, 0.17, 0.29, 0.25)
2	(0.29, 0.29, 0.17, 0.25)
3	(0.17, 0.29, 0.29, 0.25)
4	(0.25, 0.17, 0.32, 0.26)
5	(0.26, 0.32, 0.17, 0.25)
6	(0.31, 0.17, 0.26, 0.26)
7	(0.33, 0.25, 0.17, 0.25)

Fig. 3. Example ISM for the rock-paper-scissors game. Thick blue edges correspond to the observation $(0,p_2)$, dashed edges $(0,s_2)$ and solid red edges $(0,r_2)$.

This expression is obtained as a direct application of Bayes' rule.

Remark 1. The denominator in Eq. (1) needs to be non-zero for $\mathsf{condition}(\mathbf{b},s,a_2)$ to be defined. The denominator being zero means that the current belief states rule out the observation a_2 as having zero probability.

At each step, $\mathcal{P}_2$ switches to a different policy from the one they are currently utilizing according to the Markov chain with transition probabilities given by T. This modifies a belief state $\mathbf{b}$ to a new one $\mathbf{b}'=T^t\mathbf{b}$, wherein T^t denotes the transpose of the matrix T. I.e, $b'_i=\sum_{j=1}^{n} b_j T_{ji}$.Overall, given a sequence $(t_1,a_1)(t_2,a_2)\cdots(t_k,a_k)$ of observations and starting from some initial belief state $\mathbf{b}_0$, we define the sequence of belief states: $\mathbf{b}_0 \xrightarrow{(t_1,a_1)} \mathbf{b}_1 \xrightarrow{(t_2,a_2)} \mathbf{b}_2 \cdots \xrightarrow{(t_k,a_k)} \mathbf{b}_k$, such that $\mathbf{b}_{i+1}=T^t\mathsf{condition}(\mathbf{b}_i,t_{i+1},a_{i+1})$, for $i\in[k-1]$. Recall the *total variation* (tv) distance between two belief states $\mathbf{b}$ and $\mathbf{b}'$, denoted $||\mathbf{b}-\mathbf{b}'||_{tv}=\sum_{i=1}^{n}|b_i-b'_i|$.

We now discuss our model of history in terms of a finite state machine over the states and alphabets of $\mathcal{P}_2$ called the *information state machine*.

Definition 4 (Information State Machine). *An information state machine (ISM) is a deterministic finite state machine that consists of a finite set of states M, alphabet $\Sigma=S\times A^{(2)}$, initial state m_0, transition function $\delta:M\times\Sigma\to M$ and a map that associates state $m\in M$ with a belief state $\mathbf{b}(m)$ with $\mathbf{b}(m_0)=\mathbf{b}_u$.*

Recall that Σ^* denotes a finite sequence of elements from Σ. The transition function can be extended to $\delta:M\times\Sigma^*\to M$ as[1]

$$\delta(m,\langle\text{empty}\rangle)=m,\text{ and }\delta(m,\sigma\circ(t,a))=\delta(\delta(m,\sigma),(t,a))\text{ for }\sigma\in\Sigma^*\text{ and }(t,a)\in\Sigma.$$

The definition requires the state-machine to be deterministic. However, we can relax this requirement to make δ a partial function. We require that for any sequence of observations $\sigma:(t_0,a_0)\cdots(t_l,a_l)$, if σ can occur with non-zero probability (i.e, there exist actions $a'_0,\ldots,a'_{l-1}\in A^{(1)}$, such that $P(t_{j+1}|t_j,a'_j,a_j)>0$ for all $j\in[l-1]$), then (a unique state) $\delta(m_0,\sigma)$ must exist.

[1] We write $\langle$empty$\rangle$ for an empty sequence and use $\circ$ for sequence concatenation.

Example 2. Figure 3 shows an example of an ISM for the rock-paper-scissors problem. Since S has just one state, we do not include the label of this state in our alphabet, but simply label the edges with the actions of $\mathcal{P}_2$. The initial state is 0 and the automaton is deterministic.

We now define the notion of consistency of an ISM. For any sequence of observations $\sigma:\ (t_0,a_0)\cdots(t_k,a_k)$ that can occur with positive probability, and a belief state $\mathbf{b}\in\mathcal{B}_n$, let $\tau(\mathbf{b},\sigma)$ denote the result of *transforming* $\mathbf{b}$ successively based on the observations in σ.

$$\tau(\mathbf{b},\langle\text{empty}\rangle)=\mathbf{b}, \text{ and } \tau(\mathbf{b},\sigma\circ(t_i,a_i))=T^t\mathsf{condition}(\tau(\mathbf{b},\sigma),(t_i,a_i)).$$

Definition 5 (Consistent Information State Machine). *An ISM $\mathcal{M}$ is λ-consistent for $\lambda>0$ iff for every finite, positive probability sequence of $\mathcal{P}_2$ state/action observations $\sigma:(t_0,a_0)\cdots(t_k,a_k)$ such that $m_{k+1}=\delta(m_0,\sigma)$, then the belief state $\mathbf{b}(m_{k+1})$ remains sufficiently close to $\tau(\mathbf{b}_u,\sigma)$, the belief state obtained from the full history: $||\mathbf{b}(m_{k+1})-\tau(\mathbf{b}_u,\sigma)||_{tv}\leq\lambda$.*

The concept of λ-consistency implies that for any history of observations of $\mathcal{P}_2$'s actions, the belief state associated with the information state m reached, remains within total-variation distance λ of the belief state obtained by remembering the entire history.

3.1 Consistency Checking

In this subsection, we describe how to check whether a given ISM $\mathcal{M}$ is consistent for some limit λ using the sufficient condition of edge consistency.

Definition 6 (Edge Consistency). *An edge $e:m\xrightarrow{o}m'$ of the automaton $\mathcal{M}$ (i.e, $m,m'\in M$ and $\delta(m,o)=m'$) is* consistent *for limit λ iff*

$$\forall\ \mathbf{b}\in\mathcal{B}_n:\ \left(\sum_{j=1}^{n}b_j\pi_j(o)>0\ \wedge\ ||\mathbf{b}-\mathbf{b}(m)||_{tv}\leq\lambda\right)\ \Rightarrow\ ||\tau(\mathbf{b},o)-\mathbf{b}(m')||_{tv}\leq\lambda. \quad (2)$$

I.e, any belief state $\mathbf{b}$ that is within a total variation distance λ of $\mathbf{b}(m)$ must, upon updating with observation o, yield a belief state $\tau(\mathbf{b},o)$ that is within λ distance of $\mathbf{b}(m')$.

Notice that we require $\sum_{j=1}^{n}b_j\pi_j(o)=P(o|\mathbf{b})$ to be positive. Failing this condition, the observation o would be zero probability under the belief state $\mathbf{b}$ and thus ruled out.

Theorem 1. *If every edge in an ISM $\mathcal{M}$ is edge consistent for limit λ then the state machine is λ-consistent.*

Proof. Following Def. 5, we need to show that for any finite sequence of observations σ, if $\delta(m_0,\sigma)=m$ then $||\mathbf{b}(m)-\tau(\mathbf{b}_u,\sigma)||_{tv}\leq\lambda$.

Proof proceeds by induction on the length of the sequence σ, denoted $|\sigma|$. When $|\sigma|=0$, we have $m=m_0$ and $\tau(\mathbf{b}_u,\sigma)=\mathbf{b}_u$. Therefore, $||\mathbf{b}(m)-\tau(\mathbf{b}_u,\sigma)||_{tv}=0\leq\lambda$ holds.

Assume that the result holds for any non-zero probability sequence σ of length m. Let $\sigma' = \sigma \circ (t,a)$ for $t \in S$ and $a \in A^{(2)}$ also of non-zero probability. Let $m = \delta(m_0,\sigma)$ and $m' = \delta(m,(t,a))$. Since the observations, σ and σ' are assumed non-zero probability observations, we note the states m,m' exist and are unique. We know by induction hypothesis that $||\mathbf{b}(m) - \tau(\mathbf{b}_u,\sigma)||_{tv} \leq \lambda$. Note that by edge consistency of the edge $m \xrightarrow{(t,a)} m'$, we have that for all belief states $\mathbf{b} \in \mathcal{B}_n$, we have

$$||\mathbf{b} - \mathbf{b}(m)||_{tv} \leq \lambda \;\Rightarrow\; ||\tau(\mathbf{b},(t,a)) - \mathbf{b}(m')||_{tv} \leq \lambda.$$

Applying this to $\mathbf{b} = \tau(\mathbf{b}_u,\sigma)$, we note that the antecedent holds by induction hypothesis and thus, we conclude that

$$||\underbrace{\tau(\tau(\mathbf{b}_u,\sigma),(t,a)) - \mathbf{b}(m')||_{tv}}_{=\tau(\mathbf{b}_u,\sigma')} \leq \lambda.$$

We now provide an approach to check if a given edge in an automaton $e\colon\, m \xrightarrow{o} m'$ is consistent for a limit λ by checking a formula in linear arithmetic. We will attempt to find a belief state $\mathbf{b}$ that *refutes* (2). I.e, $\mathbf{b} \in \mathcal{B}_n$ that satisfies conditions: (a) $||\mathbf{b} - \mathbf{b}(m)||_{tv} \leq \lambda$; (b) $\sum_{j=1}^{n} \pi(o) b_j > 0$ and (c) $||\tau(\mathbf{b},o) - \mathbf{b}(m')||_{tv} > \lambda$. Note that $\mathbf{b}(m)$ and $\mathbf{b}(m')$ are known belief-vectors while $\mathbf{b}$ is the unknown vector we seek. We will construct a formula Ψ_e in linear arithmetic such that edge e is consistent iff Ψ_e is unsatisfiable. The formula Ψ_e is encoded using variables $\mathbf{b}\colon (b_1,\ldots,b_n)$ representing the unknown belief state and extra variables $\mathbf{x}\colon (x_1,\ldots,x_n)$ and $\mathbf{y}\colon (y_1,\ldots,y_n)$. Let $\alpha_i = \pi_i(o)$ represents the probability of observation o under policy π_i.

(1) Observation o occurs with non-zero probability:

$$\Psi_0(e)\colon\; \sum_{j=1}^{n} \alpha_j b_j > 0 \;\wedge\; \sum_{i=1}^{n} b_i = 1.$$

(2) $||\mathbf{b} - \mathbf{b}(m)||_{tv} \leq \lambda$ must hold.

$$\Psi_1(e)\colon\; \bigwedge_{i=1}^{n} x_i \geq 0 \;\wedge\; \bigwedge_{i=1}^{n} \underbrace{-x_i \leq (b_i - \mathbf{b}(m)_i) \leq x_i}_{\equiv |b_i - \mathbf{b}(m)_i| \leq x_i} \;\wedge\; \sum_{i=1}^{n} x_i \leq \lambda.$$

(3) $||\tau(\mathbf{b},o) - \mathbf{b}(m')||_{tv} > \lambda$. Recall $\tau(\mathbf{b},o) = T^t \times (\mathsf{condition}(\mathbf{b},o)) = T^t \times \left(\frac{b_1\alpha_1}{\sum_{j=1}^{n} b_j\alpha_j}, \ldots, \frac{b_1\alpha_1}{\sum_{j=1}^{n} b_j\alpha_j} \right)$.

$$||\tau(\mathbf{b},o) - \mathbf{b}(m')||_{tv} = \sum_{j=1}^{n} \left| \frac{\sum_{i=1}^{n} T_{ij}\alpha_i b_i}{\sum_{i=1}^{n} \alpha_i b_i} - \mathbf{b}(m')_j \right|.$$

Let e_j denote the expression $\sum_{i=1}^{n} T_{ij}\alpha_i b_i - (\mathbf{b}(m')_j \sum_{i=1}^{n} \alpha_i b_i)$. Since $\sum_{j=1}^{n} \alpha_j b_j > 0$, the condition $||\tau(\mathbf{b},o) - \mathbf{b}(m')||_{tv} > \lambda$ is equivalent to

$$\Psi_2(e)\colon\; \bigwedge_{j=1}^{n} \underbrace{y_j \geq 0 \;\wedge\; (y_j = e_j \vee y_j = -e_j)}_{\equiv\; y_j = |e_j|} \;\wedge\; \left(\sum_{j=1}^{n} y_j > \lambda \sum_{j=1}^{n} \alpha_j b_j \right).$$

Algorithm 1: ConstructConsistentInformationStateMachine()

```
    Data: $\mathcal{G}$,$\Pi$,$T$,$\lambda$
    Result: A finite state machine $\mathcal{M}$.
 1  $m_0 \leftarrow$ newState($\mathbf{b}_u$)                              // create initial state
 2  $\Sigma' = \{(s,a_2) \in S \times A^{(2)} \mid (\exists \pi \in \Pi)\ \pi(s,a_2) > 0\}$   // non-zero prob. observ.
 3  $(\mathcal{M},W) \leftarrow (\emptyset,[m_0])$                      // initialize set of states and worklist
 4  while $W \neq \emptyset$ do
 5      $m \leftarrow pop(W)$                                    // pop a state from the worklist
 6      Add state $m$ to $\mathcal{M}$
 7      for $o \in \Sigma'$                                      // iterate through observations
 8       do
 9          $\mathbf{b}' \leftarrow \tau(\mathbf{b}(m),o)$                             // compute next belief state
10          if not isConsistent($\mathbf{b}(m)$,$o$,$\mathbf{b}'$) then FAIL          // check consistency
11
12          $\hat{m} \leftarrow$ findClosestState($\mathbf{b}'$,$\lambda$)                // search for nearby state
13          if $\hat{m} \neq Nil \wedge$ isConsistent($\mathbf{b}(m)$,$o$,$\mathbf{b}(\hat{m})$)     // existing state found
14           then
15              Add edge $m \xrightarrow{o} \hat{m}$ to $\mathcal{M}$
16          else
17              $m' =$ newState($\mathbf{b}'$)                              // Create new state
18              Add edge $m \xrightarrow{o} m'$ to $\mathcal{M}$
19              push($m'$,$W$)                                   // push new state to worklist
20      return $\mathcal{M}$
```

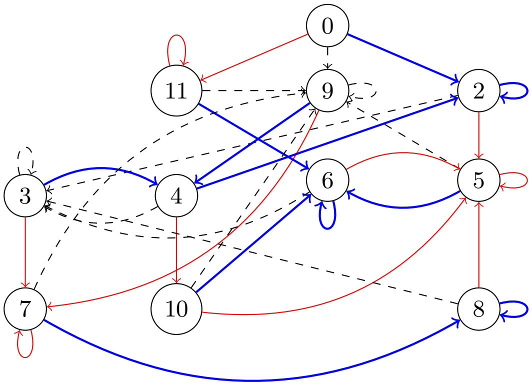
Theorem 2. *An edge e is consistent iff $\Psi(e)$: $\Psi_0(e) \wedge \Psi_1(e) \wedge \Psi_2(e)$ is infeasible.*

Satisfiability Modulo Theory (SMT) solvers such as Z3 can be used to check satisfiability [35]. Alternatively, linear complementarity problem (LCP) solvers [34] be used: the disjunction $y_i = e_i \vee y_i = -e_i$ is equivalent to a complementarity constraint $(y_i - e_i) \perp (y_i + e_i)$.

Example 3. We check the consistency of the automaton from Example 2 for $\lambda = 0.25$. For the edge e: $4 \xrightarrow{r_2} 6$ in the automaton. The formula $\Psi(e)$ is satisfiable with $\mathbf{b} = (0.125,0.17,0.445,0.26)$: $||\mathbf{b} - \mathbf{b}(4)||_{tv} = 0.25$, whereas $||\tau(\mathbf{b},o) - \mathbf{b}(6)||_{tv} \approx 0.337 > 0.25$. The automaton in Figure 3 fails to be consistent.

4 Information State Machine Synthesis Algorithm

Algorithm 1 attempts to synthesize a consistent finite state machine for $\mathcal{P}_2$, given a concurrent game $\mathcal{G}$: $\langle S, A^{(1)}, A^{(2)}, P, R \rangle$, policies Π: $\{\pi_1,...,\pi_n\}$, transition matrix T and $\lambda > 0$ by exploring belief states starting from the initial belief state m_0. Line 2 restricts the alphabet to the set Σ' that has non-zero probability under at least one policy. The algorithm maintains a worklist W that is initialized to contain the initial state m_0 at start. At each iteration, it pops a state from the worklist and adds it to the automaton. Next, the algorithm iterates through all the observations $o \in \Sigma'$ (line



State	Belief
0	(0.25, 0.25, 0.25, 0.25)
2	(0.13, 0.28, 0.33, 0.26)
3	(0.23, 0.33, 0.14, 0.30)
4	(0.13, 0.33, 0.24, 0.29)
5	(0.21, 0.13, 0.39, 0.27)
6	(0.13, 0.20, 0.42, 0.26)
7	(0.30, 0.13, 0.26, 0.31)
8	(0.13, 0.21, 0.35, 0.31)
9	(0.30, 0.24, 0.14, 0.32)
10	(0.22, 0.13, 0.35, 0.30)
11	(0.28, 0.13, 0.33, 0.26)

Fig. 4. (Left) Consistent ISM for $\lambda = 0.25$ the RPS example from Figure 1 obtained by running Algorithm 1. Thick blue edges correspond to the observation $(0,p_2)$, dashed edges $(0,s_2)$ and solid red edges $(0,r_2)$; **(Right)** Beliefs associated with states.

number 7). After computing the next belief state $\mathbf{b}'$ (line 9), it finds the closest state to $\mathbf{b}'$ in the total variation norm and checks that it is closer than the limit λ (line 12). If such a state $\hat{m}$ is found and the edge from m to $\hat{m}$ is consistent (line 13), then the edge is added. Consistency is checked using a SMT or MILP solver as described in Section 3. Otherwise, the algorithm has already checked consistency of the new state and edge that it is about to create (line 10). This is an important operation since a failure of consistency here can result in an overall failure to find a state machine.

Theorem 3. *Any automaton $\mathcal{M}$ returned by Algorithm 1 is λ-consistent.*

Proof. Every edge added to the automaton is consistent, by construction.

Figure 4 shows a consistent ISM with 11 states for the RPS example from Figure 1. Note that Algorithm 1 is not guaranteed to terminate and return a finite ISM. In section 6, we establish a simple condition on the transition matrix T for which the algorithm terminates and yields a finite ISM.

5 Policy Synthesis

Given an ISM $\mathcal{M}$, we will now describe the policy synthesis for $\mathcal{P}_1$ and prove bounds on the optimality of the policy thus obtained w.r.t discounted rewards. We first compose a two player game graph $\mathcal{G}:\langle S,A^{(1)},A^{(2)},P,R\rangle$ with the ISM $\mathcal{M}:\langle M,\Sigma',\delta\rangle$ wherein $\Sigma'\subseteq S\times A^{(2)}$. This MDP serves as a starting point for optimal policy synthesis. Next, for a $\lambda-$consistent information state machine, we show that this MDP is "close" to an infinite state MDP obtained from unbounded histories. We invoke a result on approximate information states (AIS) by Subramanian et al [44] to bound the difference between the optimal value function obtained from finite state histories and that from full histories.

The MDP is given by $\langle S\times M, A^{(1)}, \widehat{P}, \widehat{R}\rangle$ with states (s,m) for $s\in S$ and $m\in M$. Let $\mathbf{b}(m)=(b_1,\ldots,b_n)$. For $a_1\in A^{(1)}$, the probability of transitioning to (s',m') from (s,m) is given by

$$\widehat{P}((s',m')|(s,m),a_1) = \sum_{a_2\in A^{(2)}} \underbrace{1_{\{\delta(m,(s,a_2))=m'\}}}_{\text{indicator function}} \underbrace{\left(\sum_{i=1}^{n} b_i\pi_i(s,a_2)\right)}_{=\mathbb{P}(a_2|\mathbf{b}(m))} P(s'|s,a,a_2). \quad (3)$$

Note that $1_{\{\psi\}}=1$ if ψ holds and 0 otherwise. The reward function is

$$\widehat{R}((s,m),a_1) = \sum_{a_2\in A^{(2)}} \underbrace{P(a_2|\mathbf{b}(m))}_{\text{see eq. (3)}} \underbrace{R(s,a_1,a_2)}_{\text{from } \mathcal{G}}. \quad (4)$$

The composition of a finite ISM with the game yields a finite-state MDP for $\mathcal{P}_1$ that can be solved to yield a policy for $\mathcal{P}_1$. However, since the ISM tracks the belief state approximately, we cannot expect the resulting policy to be optimal when compared to a situation wherein we track the precise belief state. We will bound the loss in value resulting from the belief state approximation in an ISM.

We construct a belief state MDP using the "exact" history of observations up to some time t. The "exact" MDP has as its states $S\times\mathcal{B}_n$ wherein each state is a pair $(s,\mathbf{b}^{(t)})$ for $s\in S$ and $\mathbf{b}^{(t)}=\tau(\mathbf{b}_u,\sigma_t)$ for observation sequence $\sigma_t:(s_1,a_1),\ldots,(s_t,a_t)$. The expected reward obtained for action $a\in A^{(1)}$ in current state $s_{t+1}=s$ is given by

$$R^*((s,\mathbf{b}^{(t)}),a) = \sum_{a_2\in A^{(2)}} P(a_2|\mathbf{b}^{(t)})\ R(s,a,a_2). \quad (5)$$

We define the transition probability P^* as

$$P^*((s',\mathbf{b}^{(t+1)})|(s,\mathbf{b}^{(t)}),a,a_{t+1}) = 1_{\{\mathbf{b}^{(t+1)}=\tau(\mathbf{b}^{(t)},(s,a_{t+1}))\}} P(a_{t+1}|\mathbf{b}^{(t)}) P(s'|s,a,a_{t+1}).$$

Let $m_t=\delta(m_0,\sigma_t)$ be the unique information state from $\mathcal{M}$. Since $\mathcal{M}$ is λ-consistent, we know that $||\mathbf{b}(m_t)-\mathbf{b}^{(t)}||_{tv}\leq\lambda$. We establish bounds on the discrepancies between the rewards obtained and the next state probabilities. Let us define $R_{\max}(s)=\max_{a_1\in A^{(1)},a_2\in A^{(2)}}|R(s,a_1,a_2)|$ and $\alpha_{\max}(s)=\sum_{a_2\in A^{(2)}}\max_{j=1}^{n}\pi_j(s,a_2)$.

Lemma 1. *For any history σ_t, $|R^*((s,\mathbf{b}^{(t)}),a)-\widehat{R}((s,m_t),a)|\leq R_{max}(s)\alpha_{max}(s)\lambda$.*

Proof. We expand the LHS using Eq. (4) and Eq. (5).

$$\begin{aligned}
|R^*((s,\mathbf{b}^{(t)}),a)-\widehat{R}((s,m_t),a)| &\leq \sum_{a_2\in A^{(2)}} |P(a_2|\mathbf{b}^{(t)})R(s,a_1,a_2)-P(a_2|\mathbf{b}(m_t))R(s,a_1,a_2)| \\
&\leq \sum_{a_2\in A^{(2)}} |R(s,a_1,a_2)||\sum_{i=1}^{n}\mathbf{b}_i^{(t)}\pi_i(s,a_2)-\mathbf{b}(m_t)_i\pi_i(s,a_2)| \\
&\leq |R_{\max}(s)| \sum_{a_2\in A^{(2)}} (\max_{j=1}^{n}\pi_j(s,a_2))\sum_{i=1}^{n}|\mathbf{b}_i^{(t)}-\mathbf{b}(m_t)_i| \\
&\leq |R_{\max}(s)| \sum_{a_2\in A^{(2)}} (\max_{j=1}^{n}\pi_j(s,a_2))||\mathbf{b}^{(t)}-\mathbf{b}(m)||_{tv} \\
&\leq |R_{\max}(s)| \; \alpha_{\max}(s) \; \lambda
\end{aligned}$$

Next, we prove that the next-state distributions $\hat{P}$ and P^* are "close" in the total-variation distance $d_{tv}(\sigma_t,s,a)$ given by the formula:

$$\sum_{a_{t+1}\in A^{(2)}}\sum_{s'\in S}\Big|P^*((s',\mathbf{b}^{(t+1)})|(s,\mathbf{b}^{(t)}),a,a_{t+1})-\widehat{P}((s',m_{t+1})|(s,m_t),a,a_{t+1})\Big|.$$

Lemma 2. *For any history σ_t and action $a\in A^{(1)}$, $d_{tv}(\sigma_t,s,a)\leq\alpha_{\max}(s)\lambda$.*

Proof. We wish to bound the summation.

$$\sum_{a_{t+1}\in A^{(2)}}\sum_{s'\in S}\underbrace{|P^*((s',\mathbf{b}^{(t+1)})|(s,\mathbf{b}^{(t)}),a,a_{t+1})-\widehat{P}((s',m_{t+1})|(s,m_t),a,a_{t+1})|}_{D}.$$

Let D denote the term inside the summation.

$$\begin{aligned}
D &\leq |P(a_{t+1}|\mathbf{b}^{(t)})P(s'|s,a,a_{t+1})-P(a_{t+1}|\mathbf{b}(m_t))P(s'|s,a,a_{t+1})| \\
&\leq P(s'|s,a,a_{t+1})\max_{i=1}^{n}(\pi_i(s,a_{t+1}))||\mathbf{b}^{(t)}-\mathbf{b}(m_t)||_{tv} \\
&\leq P(s'|s,a,a_{t+1})\max_{i=1}^{n}(\pi_i(s,a_{t+1}))\lambda
\end{aligned}$$

Using this, we can bound $d_{tv}(\sigma_t,s,a)$ as

$$\begin{aligned}
d_{tv}(\sigma_t,s,a) &\leq \sum_{a_{t+1}\in A^{(2)}}\sum_{s'\in S}P(s'|s,a,a_{t+1})\max_{i=1}^{n}(\pi_i(s,a_{t+1}))\lambda \\
&\leq \lambda\sum_{a_{t+1}\in A^{(2)}}\max_{i=1}^{n}(\pi_i(s,a_{t+1}))\underbrace{\sum_{s'\in S}P(s'|s,a_{t+1},a)}_{=1} \\
&\leq \lambda\alpha_{\max}(s)
\end{aligned}$$

For some discount factor ν, let V^* be the optimal value for the (infinite state) "exact" MDP with state-space $S \times \mathcal{B}_n$, actions $A^{(1)}$, transition relation P^* and expected reward R^*. Let $\hat{V}$ be the optimal value function for the MDP with state space $S \times M$, transition map $\hat{P}$ and reward $\hat{R}$.

Theorem 4. *There exists K such that for each history σ_t leading to belief $\mathbf{b}^{(t)}$, ISM state m_t and for every game state s, we have $|V^*(s,\mathbf{b}^{(t)}) - \widehat{V}(s,m_t)| \leq K\lambda$.*

This follows from Theorem 27 of Subramanian et al [44] where the constant K equals $\frac{|R_{\max}(s)|\alpha_{\max}(s)+\gamma\rho}{1-\gamma}$, where ρ is the "Lipschitz constant" for the function V. We conclude that a $\lambda-$consistent information state machine can be used in lieu of an exact belief state with a loss in value proportional to λ.

6 Completeness and Robustness

In this section, we first provide a sufficient condition on the transition matrix T that governs how $\mathcal{P}_2$ switches between policies so that Algorithm 1 is guaranteed to terminate successfully and yield a finite ISM. Let t^* be such that for all $i,j \in [n]$, $T_{ij} \geq t^*$. I.e, t^* is the smallest entry in the matrix T. We assume that $t^* > 0$: i.e, the transition matrix T is strictly positive. Note that the entries for each row of T sum up to 1. Therefore, $t^* \leq \frac{1}{n}$. Let $\mathbf{b} = \tau(\mathbf{b}_0, \sigma)$ be the exact belief state obtained starting from the uniform initial belief state $\mathbf{b}_0$ and a sequence of non-zero probability observations σ.

Lemma 3. *Each entry of $\mathbf{b}$ satisfies $b_j \geq t^*$.*

Proof. Proof is by induction on the length of the sequence σ. The base case holds for $\mathbf{b} = \mathbf{b}_0$ since $b_{0,j} = \frac{1}{n} \geq t^*$. Let $\mathbf{b} = \tau(\mathbf{b}_0, \sigma)$ for $|\sigma| = n$. Let o be an observation such that $\mathbf{b}' = \tau(\mathbf{b}, o)$. By induction hypothesis, $b_j \geq t^*$. We have $\mathbf{b}' = T^t \hat{\mathbf{b}}$ where $\hat{\mathbf{b}} = \mathsf{condition}(\mathbf{b}, o)$ is a belief vector. $b'_j = \sum_{i=1}^n T_{ij} \hat{b}_i \geq t^* \sum_{i=1}^n \hat{b}_i \geq t^*$.

For observation o, let $\alpha_j = \pi_j(o)$, $\alpha_{\max}(o) = \max_{j=1}^n \alpha_j$ and $\alpha_{sum}(o) = \sum_{j=1}^n \alpha_j$. We define $\kappa(o) = \frac{\alpha_{\max}(o)}{\alpha_{sum}(o) + n\alpha_{\max}(o)}$. Let $\kappa_{\max} = \max_{o \in \Sigma \times A^{(2)}} \kappa(o)$.

Theorem 5. *If $t^* > \kappa_{\max}$, then for any parameter $\lambda > 0$, Algorithm 1 terminates successfully to yield a finite state consistent ISM.*

We first provide a sketch of the proof. (a) We first establish that the function $\mathbf{b} \mapsto \tau(\mathbf{b}, o)$ is *contractive* in the total variation norm whenever $t^* > \kappa(o)$. Therefore, the consistency check in line 10 will always succeed, or equivalently, Algorithm 1 will not return **FAIL**. It remains to show that the Algorithm will terminate. (b) Next, we show that whenever the call to $\mathsf{findClosestState}(\mathbf{b}', \lambda)$ in line 12 yields a state $\hat{m}$ such that $\mathbf{b}(\hat{m})$ is within distance $(1-\kappa_{\max})\lambda$ of $\mathbf{b}'$, then the edge $m \xrightarrow{o} \hat{m}$ will be consistent. Therefore, we show that for any new state created by Algorithm 1 line 17, the total variation distance from any previously created state is at least $(1-\kappa_{\max})\lambda$. (c) The number of states in the ISM is therefore bounded by the *packing number of the compact set $\mathcal{B}_n$* with L_1 norm balls of radius $(1-\kappa_{\max})\lambda$ [33].

Proof. Let us assume that $T_{i,j} \geq t^*$ for all $i,j \in [n]$. We will first derive conditions for the map $\mathbf{b} \rightarrow \tau(\mathbf{b},o)$ for a given observation $o \in \Sigma \times A^{(2)}$ to be contractive: $||\tau(\mathbf{b}_1,o) - \tau(\mathbf{b}_2,o)||_{tv} \leq \gamma ||\mathbf{b}_1 - \mathbf{b}_2||_{tv}$ for constant $\gamma < 1$.

Definition 7 (Induced Matrix Norm). *Given a $n \times n$ matrix Q, its induced p-norm for $p \geq 1$ is defined as:*

$$||Q||_p = \mathsf{sup}_{\mathbf{x} \in \mathbb{R}^n, \mathbf{x} \neq 0} \; \frac{||Q\mathbf{x}||_p}{||\mathbf{x}||_p}.$$

Also note that for a matrix Q, the induced L_1-norm $||Q||_1$ is defined as

$$||Q||_1 = \max_{j=1}^{n} \sum_{i=1}^{n} |A_{i,j}|,$$

the maximum over all the sum of absolute values of entries along each column of the matrix (Cf. [47] for further details).

Lemma 4. *For any belief vectors $\mathbf{b}_1, \mathbf{b}_2 \in \mathcal{B}_n$, we have*

$$||T^t \mathbf{b}_1 - T^t \mathbf{b}_2||_{tv} \leq (1 - nt^*) ||\mathbf{b}_1 - \mathbf{b}_2||_{tv}.$$

Proof. Let $\mathbb{1}_{n \times n}$ be the $n \times n$ matrix with all 1 entries and $\mathbb{1}_n$ be the $n \times 1$ vector of all 1s. Let $Q = T^t - t^* \mathbb{1}_{n \times n}$. Note that $||Q||_1$ is the maximum among the column sums of Q. Each column of Q corresponds to a row of T with t^* subtracted from each entry. Therefore, each column of Q sums to $1 - nt^*$.

We can write $T^t \mathbf{b} = (T^t - t^* \mathbb{1}_{n \times n})\mathbf{b} + t^* \mathbb{1}_n \mathbf{b} = Q\mathbf{b} + t^* \mathbb{1}_n$. Thus,

$$\begin{aligned} ||T^t \mathbf{b}_1 - T^t \mathbf{b}_2||_{tv} &\leq ||Q\mathbf{b}_1 - Q\mathbf{b}_2 + \cancel{t^* \mathbb{1}_n} - \cancel{t^* \mathbb{1}_n}||_{tv} \\ &\leq ||Q||_1 ||\mathbf{b}_1 - \mathbf{b}_2||_{tv} \\ &\leq (1 - nt^*) ||\mathbf{b}_1 - \mathbf{b}_2||_{tv} \end{aligned}$$

Let $\alpha_j(o)$ denote $\pi_j(o)$, $\alpha_{\max}(o) = \max_{j=1}^{n} \alpha_j(o)$ and $\alpha_{sum}(o) = \sum_{j=1}^{n} \alpha_j(o)$. If the observation o is clear from the context, we will simply write $\alpha_{\max}$ and α_{sum} to denote $\alpha_{\max}(o)$ and $\alpha_{sum}(o)$, respectively.

Let $\mathcal{D}_n = \{\mathbf{b} \in \mathcal{B}_n \mid b_j \geq t^*, \forall\, j \in [n]\}$. Following lemma 3, we can restrict our attention to just those belief vectors in $\mathcal{D}_n$ since every belief state obtained through a non-zero probability sequence of observations will belong to $\mathcal{D}_n$.

Lemma 5. *For a non-zero probability observation $o \in \Sigma \times A^{(2)}$ and belief states $\mathbf{b}_1, \mathbf{b}_2 \in \mathcal{D}_n$, we have*

$$||\tau(\mathbf{b}_1,o) - \tau(\mathbf{b}_2,o)||_{tv} \leq \frac{(1 - nt^*)\alpha_{\max}(o)}{t^* \alpha_{sum}(o)} ||\mathbf{b}_1 - \mathbf{b}_2||_{tv}.$$

Proof. We have

$$\begin{aligned} ||\tau(\mathbf{b}_1,o) - \tau(\mathbf{b}_2,o)||_{tv} &= ||T^t \mathsf{condition}(\mathbf{b}_1,o) - T^t \mathsf{condition}(\mathbf{b}_2,o)||_{tv} \\ &\leq (1 - nt^*) ||\mathsf{condition}(\mathbf{b}_1,o) - \mathsf{condition}(\mathbf{b}_2,o)||_{tv} \text{ applying Lemma 4} \\ &\leq (1 - nt^*) \textstyle\sum_{j=1}^{n} \left| \frac{b_{1,j}\alpha_j}{\sum_{i=1}^{n} b_{1,i}\alpha_i} - \frac{b_{2,j}\alpha_j}{\sum_{i=1}^{n} b_{2,i}\alpha_i} \right| \end{aligned}$$

Note that $\sum_{i=1}^{n} b_{1,i}\alpha_i \geq t^* \sum_{i=1}^{n} \alpha_i = t^* \alpha_{sum}$ since each entry of $\mathbf{b}_1$ is at least t^*. Similarly, we note that $\sum_{i=1}^{n} b_{2,i}\alpha_i \geq t^*\alpha_{sum}$. Therefore,

$$\begin{aligned} ||\tau(\mathbf{b}_1,o) - \tau(\mathbf{b}_2,o)||_{tv} &\leq \tfrac{(1-nt^*)}{t^*\alpha_{sum}} \textstyle\sum_{j=1}^{n} |\alpha_j b_{1,j} - \alpha_j b_{2,j}| \\ &\leq \tfrac{1-nt^*}{t^*\alpha_{sum}} \alpha_{\max} \textstyle\sum_{j=1}^{n} |b_{1,j} - b_{2,j}| \\ &\leq \tfrac{(1-nt^*)\alpha_{\max}(o)}{t^*\alpha_{sum}(o)} ||\mathbf{b}_1 - \mathbf{b}_2||_{tv} \end{aligned}$$

Let us define $\kappa(o) = \frac{\alpha_{\max}}{\alpha_{sum} + n\alpha_{\max}}$.

Lemma 6. *The map* $\mathbf{b} \mapsto \tau(\mathbf{b},o)$ *is contractive if* $t^* > \kappa(o)$.

Proof. Using Lemma 5, we note that $\mathbf{b} \mapsto \tau(\mathbf{b},o)$ is contractive if $\frac{(1-nt^*)\alpha_{\max}(o)}{t^*\alpha_{sum}(o)} < 1$.

$$\begin{aligned} &\tfrac{(1-nt^*)\alpha_{\max}(o)}{t^*\alpha_{sum}(o)} < 1 \\ &\Leftrightarrow\ (1-nt^*)\alpha_{\max} < t^*\alpha_{sum} \quad \because\ t^*\alpha_{sum} > 0 \\ &\Leftrightarrow\ t^*(\alpha_{sum} + n\alpha_{\max}) > \alpha_{\max} \ \text{ rearranging terms} \\ &\Leftrightarrow\ t^* > \tfrac{\alpha_{\max}}{\alpha_{sum} + n\alpha_{\max}}. \end{aligned}$$

Having established these results, we proceed with the proof of Theorem 5. Let us assume that $t^* > \kappa_{\max} \geq \kappa(o)$ for all non-zero probability observations o.

First, we conclude that Algorithm 1 will never return FAIL (line 10). This is because, any edge $m \xrightarrow{o} m'$ wherein $\mathbf{b}(m') = \tau(\mathbf{b}(m),o)$ will be consistent due to the contractivity of τ. In other words, for any $\mathbf{b} \in \mathcal{D}_n$ such that $||\mathbf{b} - \mathbf{b}(m)||_{tv} \leq \lambda$, we have

$$\begin{aligned} ||\tau(\mathbf{b},o) - \mathbf{b}(m')||_{tv} &\leq \tfrac{(1-nt^*)\alpha_{\max}(o)}{t^*\alpha_{sum}(o)} ||\mathbf{b} - \mathbf{b}(m)||_{tv} \\ &\leq \gamma(o)\lambda \end{aligned}$$

wherein $\gamma(o) = \frac{(1-nt^*)\alpha_{\max}(o)}{t^*\alpha_{sum}(o)} < 1$. Let $L^* = \max_{o \in O} \gamma(o)$. Clearly, $L^* < 1$, as well.

For a given $\delta > 0$, let $\mathcal{B}_\delta(\mathbf{b}) = \{\widehat{\mathbf{b}} \in \mathcal{D}_n \mid ||\mathbf{b} - \widehat{\mathbf{b}}||_{tv} \leq \delta\}$ be a ball of size δ in the total-variation norm over belief states. Next consider a "packing" of the belief space $\mathcal{D}_n$.

Definition 8 (Minimal Packing with balls of size δ). *A minimal packing of* $\mathcal{D}_n$ *using balls of size* δ *is a family of* N *sets* $\mathcal{F} = \{\mathcal{B}(\mathbf{b}_i,\delta)\}$ *for* $i \in [N]$ *that (a) covers the entire belief space* $\bigcup_{S \in \mathcal{F}} S \supseteq \mathcal{D}_n$ *and (b) minimizes the size of the family* N *over all such covers.*

Let $\mathcal{F}$ be a minimal packing of the belief space $\mathcal{D}_n$ with balls of size $(1-L^*)\lambda$. By the compactness of $\mathcal{D}_n$, we note that $\mathcal{F}$ is finite. We now prove that for the automaton constructed by Algorithm 1, we cannot have two states m,m' such that $\mathbf{b}(m),\mathbf{b}(m')$ belong to the same ball in $\mathcal{F}$.

Lemma 7. *Let* $\mathcal{F}$ *represent a family of sets that form a minimal* $\delta = (1-L^*)\lambda$ *packing of the belief space* $\mathcal{D}_n$. *Algorithm 1 during its run cannot produce two states* $m,m' \in M$ *such that* $\mathbf{b}(m) \in S$ *and* $\mathbf{b}(m') \in S$ *for* $S \in \mathcal{F}$.

Proof. We will prove this by contradiction. Let m,m' be two states created such that $||\mathbf{b}(m)-\mathbf{b}(m')||_{tv}\leq(1-L^*)\lambda$. In fact, let us assume that (m,m') are the very first pair of states constructed during the execution of Algorithm 1 with this property.

Let us assume that m is the first state constructed, followed by m'. Let $\mathbf{b}'=\mathbf{b}(m')$. We create the state m' because of we have $\mathbf{b}'=\tau(\mathbf{b}(m_1),o)$ for some previously added state m_1 and observation $o\in\Sigma'$ (line 9 of Algorithm 1).

The call to findClosestState (line 12) must return the state m since if it returned some other state m_2 then $||\mathbf{b}(m_2)-\mathbf{b}(m')||_{tv}\leq||\mathbf{b}(m)-\mathbf{b}(m')||_{tv}\leq(1-L^*)\lambda$. This means that (m,m_2) are a pair of already created states that contradicts the statement of this theorem. However, this goes against our assumption that (m,m') is the very first pair created. Therefore, $m_2=m$.

By assumption,

$$||\mathbf{b}(m)-\mathbf{b}(m')||_{tv}\leq(1-L^*)\lambda.$$

Also, since $\mathbf{b}(m')=\tau(\mathbf{b}(m_1),o)$ and τ is contractive, we know that any belief state $\mathbf{b}\in\mathcal{D}_n$ such that $||\mathbf{b}-\mathbf{b}(m_1)||_{tv}\leq\lambda$,

$$||\tau(\mathbf{b},o)-\mathbf{b}(m')||_{tv}\leq L^*||\mathbf{b}-\mathbf{b}(m_1)||_{tv}\leq L^*\lambda.$$

Therefore,

$$\begin{aligned}||\tau(\mathbf{b},o)-\mathbf{b}(m)||_{tv}&\leq||\tau(\mathbf{b},o)-\mathbf{b}(m')||_{tv}+||\mathbf{b}(m')-\mathbf{b}(m)||_{tv}\\&\leq L^*\lambda+(1-L^*)\lambda\\&\leq\lambda\end{aligned}$$

We thus know that the edge from m_1 to m will be consistent. Therefore, the state m' is never created by our algorithm because the then-branch of the condition in line 13 in Algorithm 1 is executed, yielding a contradiction.

As a result, we have proven a finite upper bound on the number of possible states Algorithm 1 can produce which happens to be the *packing number* of the minimum cardinality family of balls of radius $(1-L^*)\lambda$ in the total variation norm that covers the belief space $\mathcal{D}_n$.

This concludes the proof of Theorem 5.

Example 4. For all observations o in the RPS example from Figure 1, $\alpha_{\max}(o)=0.5$, $\alpha_{sum}(o)=\frac{4}{3}$. We have $\kappa_{\max}=\kappa(o)=\frac{0.5}{4/3+4(0.5)}=\frac{3}{20}=0.15$. Using Theorem 5, for any matrix T all of whose entries exceed 0.15, we are guaranteed a finite state ISM for any $\lambda>0$. Interestingly, the matrix in Figure 1 *does not* satisfy this condition and nevertheless yields finite ISM for $\lambda=0.25$ (Figure 4).

Robustness: Suppose we designed an ISM $\mathcal{M}$ that is consistent for $\lambda>0$ assuming matrix $T=T_D$, whereas in reality $\mathcal{P}_2$ switches policies according to $T=T_A$, wherein $T_A\neq T_D$. We will prove that the ISM $\mathcal{M}$ which is consistent for $T=T_D$ and $\lambda>0$ will remain consistent for $T=T_A$ for a different value $\lambda=\overline{\lambda}$. Let $t_d^*=\min_{i,j\in[n]}T_{D,i,j}$ and

$t_a^*=\min_{i,j\in[n]} T_{A,i,j}$ be the minimum entries in the matrices T_D and T_A respectively. Let us define the function

$$L(T_A,T_D,\mathcal{G},\Pi)=\max_{o\in O}\frac{(1-n\ \max(t_a^*,t_d^*))\alpha_{\max}(o)}{\min(t_a^*,t_d^*)\alpha_{sum}(o)}$$

Theorem 6. *If $t_a^*>0$, $t_d^*>0$ and $L(T_A,T_D,\mathcal{G},\Pi)<1$ then the ISM $\mathcal{M}$ is consistent under the matrix T_D with the consistency parameter $\overline{\lambda}=\frac{\lambda+||(T_A-T_D)^t||_1}{1-L(T_A,T_D,\mathcal{G},\Pi)}$.*

$||T||_1$ refers to the induced $1-$norm of matrix T [47].

Proof. Let us assume that T_A is the actual matrix used by $\mathcal{P}_2$ whereas T_D is the matrix assumed during the design of the consistent ISM $\mathcal{M}$. Let each entry of T_A be at least t_a^* whereas t_d^* is the minimal entry in the matrix T_D. We assume that $t_a^*>0$ and $t_d^*>0$.

For a belief state $\mathbf{b}\in\mathcal{B}_n$, let $\tau_D(\mathbf{b},o)$ denote the updated belief state using the design assumption T_D:

$$\tau_D(\mathbf{b},o)=T_D^t\times\mathsf{condition}(\mathbf{b},o).$$

Likewise, let τ_A be the updated belief state using the actual play matrix T_A:

$$\tau_A(\mathbf{b},o)=T_A^t\times\mathsf{condition}(\mathbf{b},o).$$

Let $t_{\max}=\max(t_a^*,t_d^*)$ and $t_{\min}=\min(t_a^*,t_b^*)$. Recall the definitions: $\alpha_j=\pi(o)$, $\alpha_{\max}(o)=\max_{j=1}^n\alpha_j$ and $\alpha_{sum}(o)=\sum_{j=1}^n\alpha_j$.

Lemma 8. *Let $\mathbf{b}_1,\mathbf{b}_2$ be two belief states such that for all $j\in[n]$, $b_{1,j}\geq t_a^*$ and $b_{2,j}\geq t_d^*$; and o be a non-zero probability observation.*

$$||\tau_A(\mathbf{b}_1,o)-\tau_D(\mathbf{b}_2,o)||_{tv}\leq||(T_A-T_D)^t||_1+\frac{(1-nt_{\max})\alpha_{\max}(o)}{t_{\min}\alpha_{sum}(o)}||b_1-b_2||_{tv}.$$

Proof.

$$\begin{aligned}
&||\tau_A(\mathbf{b}_1,o)-\tau_D(\mathbf{b}_2,o)||_{tv}\\
&=||T_A^t\mathsf{condition}(\mathbf{b}_1,o)-T_D^t\mathsf{condition}(\mathbf{b}_2,o)||_{tv}\quad(\text{* let }\ \mathbf{b}':=\mathsf{condition}(\mathbf{b},o)\text{*})\\
&=||T_A^t(\mathbf{b}_1'-\mathbf{b}_2')||_{tv}+||(T_A-T_D)^t\mathbf{b}_2'||_{tv}\\
&\leq(1-nt_a^*)||\mathbf{b}_1'-\mathbf{b}_2'||_{tv}+||(T_A-T_D)^t||_1\times\underbrace{||\mathbf{b}_2'||_1}_{=1}\ (\text{*Cf. Lemma 4*})
\end{aligned}$$

Consider another derivation that proceeds as follows:

$$\begin{aligned}
&||\tau_A(\mathbf{b}_1,o)-\tau_D(\mathbf{b}_2,o)||_{tv}\\
&=||T_A^t\mathsf{condition}(\mathbf{b}_1,o)-T_D^t\mathsf{condition}(\mathbf{b}_2,o)||_{tv}\quad(\text{* let }\ \mathbf{b}':=\mathsf{condition}(\mathbf{b},o)\text{*})\\
&=||T_D^t(\mathbf{b}_1'-\mathbf{b}_2')||_{tv}+||(T_A-T_D)^t\mathbf{b}_1'||_{tv}\\
&\leq(1-nt_d^*)||\mathbf{b}_1'-\mathbf{b}_2'||_{tv}+||(T_A-T_D)^t||_1\times\underbrace{||\mathbf{b}_1'||_1}_{=1}\ (\text{*Cf. Lemma 4*})
\end{aligned}$$

Combining, we obtain:

$$||\tau_A(\mathbf{b}_1,o)-\tau_D(\mathbf{b}_2,o)||_{tv}\leq\underbrace{\min(1-nt_a^*,1-nt_d^*)}_{=1-nt_{\max}}||\mathbf{b}_1'-\mathbf{b}_2'||_{tv}+||(T_A-T_D)^t||_1.$$

We will now calculate bounds on $||\mathbf{b}'_1 - \mathbf{b}'_2||_{tv}$.

$$||\mathbf{b}'_1 - \mathbf{b}'_2||_{tv} = \sum_{j=1}^{n} \left| \frac{\alpha_j b_{1,j}}{\sum_{i=1}^{n} \alpha_i b_{1,i}} - \frac{\alpha_j b_{2,j}}{\sum_{i=1}^{n} \alpha_i b_{2,i}} \right|$$

Note that $\sum_i \alpha_i b_{1,i} \geq t_a^* \alpha_{sum} \geq t_{\min} \alpha_{sum}$ since $b_{1,i} \geq t_a^*$. Likewise, $\sum_i \alpha_i b_{2,i} \geq t_{\min} \alpha_{sum}$. Therefore,

$$\begin{aligned} ||\mathbf{b}'_1 - \mathbf{b}'_2||_{tv} &= \sum_{j=1}^{n} \left| \frac{\alpha_j b_{1,j}}{\sum_{i=1}^{n} \alpha_i b_{1,i}} - \frac{\alpha_j b_{2,j}}{\sum_{i=1}^{n} \alpha_i b_{2,i}} \right| \\ &\leq \frac{1}{t_{\min} \alpha_{sum}} \sum_{j=1}^{n} |\alpha_j b_{1,j} - \alpha_j b_{2,j}| \leq \frac{\alpha_{\max}}{t_{\min} \alpha_{sum}} ||\mathbf{b}_1 - \mathbf{b}_2||_{tv} \end{aligned}$$

Combining, we obtain,

$$\begin{aligned} &||\tau_A(\mathbf{b}_1, o) - \tau_D(\mathbf{b}_2, o)||_{tv} \\ &\leq (1 - n t_{\max}) ||\mathbf{b}'_1 - \mathbf{b}'_2||_{tv} + ||(T_A - T_D)^t||_1 \\ &\leq \frac{(1 - n t_{\max}) \alpha_{\max}}{t_{\min} \alpha_{sum}} ||\mathbf{b}_1 - \mathbf{b}_2||_{tv} + ||(T_A - T_D)^t||_1. \end{aligned}$$

This completes the proof of this lemma.

Let σ_t be a sequence of observations with non-zero probability and $\mathbf{b} = \tau_D(\mathbf{b}_0, \sigma_t)$ and $\hat{\mathbf{b}} = \tau_A(\mathbf{b}_0, \sigma_t)$ for the uniform initial belief state $\mathbf{b}_0$. Let m be the state in the ISM $m = \delta(m_0, \sigma_t)$ with associated belief state $\mathbf{b}(m)$.

We will now proceed to the proof of the original theorem. Let us consider an edge $m \xrightarrow{o} m'$ in the automaton. We will show that the edge is $\overline{\lambda}$ consistent under τ_A. Let $\hat{\mathbf{b}}$ be any belief state such that $\hat{b}_j \geq t_a^*$ and

$$||\hat{\mathbf{b}} - \mathbf{b}(m)||_{tv} \leq \overline{\lambda},$$

wherein

$$\overline{\lambda} = \frac{\lambda + ||(T_A - T_D)^t||_1}{1 - L(T_A, T_D, \mathcal{G}, \Pi)},$$

and $L(T_A, T_D, \mathcal{G}, \Pi) = \max_{o \in O} \frac{(1 - n t_{\max}) \alpha_{\max}}{t_{\min} \alpha_{sum}}$ wherein $L(T_A, T_D, \mathcal{G}, \Pi) < 1$ by assumption.

We wish to prove that

$$||\tau_A(\hat{\mathbf{b}}, o) - \mathbf{b}(m')||_{tv} \leq \overline{\lambda}.$$

First, we note that for any belief state $\mathbf{b}$ such that $b_j \geq t_d^*$

$$||\tau_D(\mathbf{b}, o) - \tau_A(\hat{\mathbf{b}}, o)||_{tv} \leq \frac{(1 - n t_{\max}) \alpha_{\max}}{t_{\min} \alpha_{sum}} ||\mathbf{b} - \hat{\mathbf{b}}||_{tv} + ||(T_A - T_D)^t||_1.$$

Therefore,

$$\begin{aligned} &||\tau_A(\hat{\mathbf{b}}, o) - \mathbf{b}(m')|| \leq ||\tau_A(\hat{\mathbf{b}}, o) - \tau_D(\mathbf{b}(m), o)|| + ||\tau_D(\mathbf{b}(m), o) - \mathbf{b}(m')|| \\ &\leq \frac{(1 - n t_{\max}) \alpha_{\max}(o)}{t_{\min} \alpha_{sum}(o)} ||\hat{\mathbf{b}} - \mathbf{b}(m)||_{tv} + ||(T_A - T_D)^t||_1 + \lambda \\ &\leq L(T_D, T_A, \Pi) \overline{\lambda} + \underbrace{||(T_A - T_D)^t||_1 + \lambda}_{= (1 - L(T_D, T_A, \mathcal{G}, \Pi)) \overline{\lambda}} \\ &\leq \overline{\lambda} \end{aligned}$$

This completes the proof.

7 Experimental Evaluation

We present an experimental evaluation based on an implementation of the ideas mentioned thus far. Our implementation uses the Python programming language and inputs a user-defined game structure, n policies for player $\mathcal{P}_2$, values for parameters $\lambda > 0$. For each case, the policy design Markov chain whose transition system is given by $T(\epsilon)$, such that $T(\epsilon)_{i,i} = \epsilon$ and $T(\epsilon)_{i,j} = \frac{1-\epsilon}{n-1}$ when $i \neq j$. In other words, player $\mathcal{P}_2$ plays the same policy as previous step with probability ϵ and switches to a different policy uniformly with probability $(1-\epsilon)/(n-1)$. Our implementation uses the Gurobi optimization solver [18] to implement the consistency checks described in Section 3 and uses it to implement the consistent information state machine synthesis as described in Section 4.

Performance Evaluation on Benchmark Problems. We consider benchmarks for evaluating our approaches in terms of the ability to construct finite information state machines, the sizes of these machines and the performance of the resulting policies synthesized by our approach.

1. RPS: The rock-paper-scissors game and $\mathcal{P}_2$ policies as described in Example 1.
2. RPS-MEM: The rock-paper-scissors game but with "memory" of the previous move by each player. This game has 9 states that remember the previous move of each player, and the policies for $\mathcal{P}_2$ model behaviors such as "play action now that would have beaten $\mathcal{P}_1$ in the previous turn" or "repeat the previous action of $\mathcal{P}_1$".
3. ANTICIPATE-N-AVOID(N) consists of a circular corridor with N rooms numbered 1,...,N with four designated rooms marked as meeting zones. $\mathcal{P}_2$ chooses one of four policies that navigate them to one of the meeting rooms whereas the rewards for $\mathcal{P}_1$ are negative if they happen to be in the same cell as $\mathcal{P}_2$ or in an adjacent cell while the rewards are positive if they happen to be farther away. The game has N^2 states for N rooms.

The game structures and the policies for $\mathcal{P}_2$ are given in the appendix.

Table 1 shows the performance over these benchmarks. We have four benchmarks as described briefly above and in detail in the Appendices A, and B. For these benchmarks the number of states ranges from 1 for the rock-paper-scissors game to 2080 states for the ANTICIPATE-ACTION game. Similarly, the number of actions of each player and the number of policies employed by $\mathcal{P}_2$ are reported. For each game, we choose various values of $(\lambda, T(\epsilon))$ and report the overall performance in terms of number of states of the information state machine, the time taken to construct it, the size of the MDP and the time taken to compute an optimal policy using policy iteration. Since the transition matrix $T = T(\epsilon)$, we note that $\min(T_{i,j}) = \frac{\epsilon}{n-1}$ provided $\epsilon \leq \frac{n-1}{n}$. We ran two series of experiments for each benchmark by fixing λ and decreasing ϵ for the matrix $T(\epsilon)$ until Algorithm 1 reports a failure or times out after one hour. The first observation is that our approach works for values of $t^* = \frac{\epsilon}{n-1}$ that are smaller than the limit suggested by Theorem 5. At the same time, we note that as ϵ decreases, the size of the automaton $\mathcal{M}$ and the corresponding size of the MDP obtained by composing the automaton with the game all increase, as does the time taken to construct. Also, if Algorithm 1 fails, it happens very quickly, allowing us to increase ϵ until we succeed.

Benchmark	Size	$\lambda = 0.1$					$\lambda = 0.05$				
		ϵ	$\|M\|$	T_{alg1}	|MDP|	T_{PI}	ϵ	$\|M\|$	T_{alg1}	|MDP|	T_{PI}
RPS	(1, 3, 3, 4)	0.5	6	0.34	6	<0.01	0.5	10	0.4	10	<0.01
		0.4	20	0.92	20	<0.01	0.4	29	1.1	29	<0.01
		0.3	80	4.6	80	0.01	0.3	115	4.9	115	0.02
		0.2	×	0.02	- Alg. 1 Fail -		0.2	×	0.4	- Alg. 1 Fail -	
RPS-MEM	(9, 3, 3, 9)	0.6	77	28.7	244	0.1	0.6	176	55	526	0.1
		0.55	228	117	688	1.3	0.55	448	215	1342	0.34
		0.5	834	743	2500	4.6	0.5	1516	1101	4546	1.8
		0.45	×	14.2	- Alg. 1 Fail -		0.45	- Timeout $>1hr$-			
ANT.-AVD.	(625, 3, 3, 4)	0.55	7	1.5	1701	1.8	0.55	17	2.6	3526	4.2
		0.5	4.3	12	2726	3.2	0.5	28	6.9	5226	12.9
		0.45	8.8	26	4926	11.5	0.45	61	14.5	8326	19.5
		0.4	19.7	66	10042	26.5	0.4	137	34.6	16882	50.5
		0.35	68.5	194	24592	84	0.35	366	77.6	37770	112.1
		0.3	×	4	- Alg. 1 Fail -		0.3	1289	305.4	126395	431.1

Table 1. Performance results of our approach on various benchmarks and different values of the parameters λ,ϵ. "Size" is a four-tuple consisting of $(|S|,|A^{(1)}|,|A^{(2)}|,|\Pi|)$, T_{alg1} is time taken (seconds) to run Algorithm 1 and T_{PI} is time taken (seconds) for policy iteration to converge (discount factor $\gamma = 0.95$). Experiments were run on Linux server with four 2.4 GHz Intel Xeon CPUs and $64GB$ RAM.

Next Tool Usage Prediction. We study the performance of our approach on two datasets involving human task performance: (a) the IKEA ASM dataset that consists of 371 individual furniture assemblies of four distinct furniture models, wherein the actions performed by the human assembler are labeled using a neural network (CNN) to yield sequences of actions performed by the human [5]; and (b) the CATARACTS dataset consisting of 25 cataract surgery videos, wherein a CNN is used to identify the sequence of tools employed by the surgeon [1].

We first used automata learning tool flexfringe to construct a DFA model from a training set consisting of 75% of the sequences in each dataset [46]. Flexfringe successfully constructed a DFA that includes the sequences of actions/tools used (Cf. Appendix C). The game graph $\mathcal{G}$ consists of the automata states and edges. The transitions between states are governed by the actions of $\mathcal{P}_2$. The actions of $\mathcal{P}_1$ are the same as that of $\mathcal{P}_2$: $A^{(1)} = A^{(2)}$. The goal of $\mathcal{P}_1$ is to predict the next action/tool usage by $\mathcal{P}_2$ based on knowledge of the current state. The reward $R(s,a_1,a_2) = 1$ if $a_1 = a_2$ (i.e, $\mathcal{P}_1$'s action matches that of $\mathcal{P}_2$) and $R(s,a_1,a_2) = -1$ otherwise. The policies of $\mathcal{P}_2$ are also constructed from the training data as well. For each sequence σ in the training data we collect the set of edges (states and actions) in the automaton that are traversed by σ. Each such edge set describes a policy π wherein the player upon reaching a state chooses the action on one of the outgoing edges from the set uniformly at random, or alternatively, if no outgoing edge from the set is present, the player chooses any action uniformly at random. Note that multiple sequences from the training data map can onto the same policy.

Ikea-Shelf-Drawer ($\lvert\mathcal{G}\rvert=18$,$\lvert\mathit{\Pi}\rvert=7$)					Ikea-TV-Bench ($\lvert\mathcal{G}\rvert=18$,$\lvert\mathit{\Pi}\rvert=13$)				
(λ,ϵ)	$\lvert\mathcal{M}\rvert$	T_M	r_{avg}	ap_{avg}	(λ,ϵ)	$\lvert\mathcal{M}\rvert$	T_M	r_{avg}	ap_{avg}
(0.01, 0.5)	344	97.7	0.137	0.407	(0.01, 0.5)	846	245.5	0.203	0.389
(0.01, 0.6)	917	291	0.137	0.418	(0.01, 0.6)	2852	1006	0.21	0.404
(0.01, 0.7)	3547	1324.5	0.137	0.43	(0.01, 0.7)	- timeout $>3600s$			
(0.02, 0.5)	189	65.4	0.137	0.407	(0.02, 0.5)	425	142.8	0.198	0.389
(0.02, 0.6)	472	168.5	0.137	0.418	(0.02, 0.6)	1263	486.32	0.21	0.404
(0.02, 0.7)	1566	615.2	0.137	0.43	(0.02, 0.7)	- Algo. 1 fail -			
Ikea-Coffee-Table ($\lvert\mathcal{G}\rvert=15$,$\lvert\mathit{\Pi}\rvert=12$)					Cataract-Surgery ($\lvert\mathcal{G}\rvert=36$,$\lvert\mathit{\Pi}\rvert=14$)				
(λ,ϵ)	$\lvert\mathcal{M}\rvert$	T_M	r_{avg}	ap_{avg}	(λ,ϵ)	$\lvert\mathcal{M}\rvert$	T_M	r_{avg}	ap_{avg}
(0.01, 0.5)	521	150	0.181	0.408	(0.01, 0.4)	399	236.8	0.287	0.512
(0.01, 0.6)	1441	494	0.181	0.420	(0.01, 0.5)	1360	846.7	0.287	0.518
(0.01, 0.7)	- timeout $>3600s$				(0.01, 0.6)	- timeout $>3600s$			
(0.02, 0.5)	279	115	0.18	0.409	(0.02, 0.4)	207	138	0.287	0.512
(0.02, 0.6)	705	292	0.178	0.420	(0.02, 0.5)	626	371	0.287	0.518
(0.02, 0.7)	- Algo. 1 fail -				(0.02, 0.6)	2404	1642	0.287	0.525

Table 2. Performance data on tool prediction problem for various task sequences. $|\mathcal{G}|$ denotes size of automaton , $|\mathit{\Pi}|$: number of policies for $\mathcal{P}_2$, $|\mathcal{M}|$: ISM size, T_M: time taken by Algo. 1, r_{avg}: average reward per move, ap_{avg}: average probability of $\mathcal{P}_2$'s action at each step using ISM belief state.

Once the game and the policy are constructed from the training data, we use Algorithm 1 to construct an ISM given $T{=}T(\epsilon)$ and λ. This is used to construct an MDP, and thus, a policy π_1 for $\mathcal{P}_1$. The policy is tested by using the held out test sequences consisting of the 25% of the sequences not used in learning the task model or the policies. Using each sequence as the set of actions chosen by the oblivious $\mathcal{P}_2$, we measure the average reward for each episode and the average action prediction score for $\mathcal{P}_1$ for various values of ϵ,λ.

Table 2 shows the size of the ISM, running time of Algo. 1 and the performance of the policy for $\mathcal{P}_1$ on the held out test sequences. First, we note that the performance in terms of running time and size of the ISM shows trends that are similar to the previous benchmarks reported in Table 1. In terms of the held out sequences, we note that our approach is successful in terms of predicting the actions of $\mathcal{P}_2$. Given that the cataract data has 41 actions and Ikea dataset has 32 actions, our approach performs much better than a random guess. At the same time, the action probability score (the average probability ascribed by the ISM belief's state to $\mathcal{P}_2$ action in the current move) is also high given the large space of possible actions. Interestingly, however, we note that changing λ,ϵ has an enormous impact on the running time and size of the ISM but very little impact on the performance on the unseen test sequence. The average probability score shows a small variations across different values of λ,ϵ. We believe that this is a function of the rather small values of λ used since it assures us that the ISM tracks the belief state very precisely.

8 Conclusion

We study concurrent stochastic games against oblivious opponents where the opponent (environment) is not necessarily defined as adversarial or cooperative, but rather oblivious that is bounded to choose from a finite set of policies. We introduce the notion of *information state machine* (ISM) whose states are mapped to a belief state on the environment policy, and provide the guarantee that the belief states tracked by this automaton stay within a fixed distance of the precise belief state obtained by tracking the entire history for the environment. In the future, we would like to better characterize the relationship between the various parameters involved in Algorithm 1 to provide a tighter condition for its termination. We are also interested in understanding the applicability of these ideas to the more general case of partially observable Markov decision processes.

Acknowledgments. This work was supported in part by the US NSF under award numbers CPS-1836900, CPS-1932189, CCF-2146563, and the NSF IUCRC Center for Autonomous Air Mobility and Sensing (CAAMS).

Disclosure of Interests. The authors have no conflicts of interest to disclose.

References

1. Al Hajj, H., Lamard, M., Conze, P.H., Roychowdhury, S., Hu, X., Maršalkaitė, G., Zisimopoulos, O., Dedmari, M.A., Zhao, F., Prellberg, J., Sahu, M., Galdran, A., Araújo, T., Vo, D.M., Panda, C., Dahiya, N., Kondo, S., Bian, Z., Vahdat, A., Bialopetravičius, J., Flouty, E., Qiu, C., Dill, S., Mukhopadhyay, A., Costa, P., Aresta, G., Ramamurthy, S., Lee, S.W., Campilho, A., Zachow, S., Xia, S., Conjeti, S., Stoyanov, D., Armaitis, J., Heng, P.A., Macready, W.G., Cochener, B., Quellec, G.: Cataracts: Challenge on automatic tool annotation for cataract surgery. Medical Image Analysis **52**, 24–41 (2019)
2. Avrahami-Zilberbrand, D., Kaminka, G.A.: Two logical theories of plan recognition. Journal of Logic and Computation **12**(3), 371–412 (2002)
3. Beauquier, D., Burago, D., Slissenko, A.: On the complexity of finite memory policies for Markov decision processes. In: Mathematical Foundations of Computer Science 1995: 20th International Symposium, MFCS'95 Prague, Czech Republic, August 28–September 1, 1995 Proceedings 20. pp. 191–200. Springer (1995)
4. Bellman, R.: A Markovian decision process. Journal of mathematics and mechanics pp. 679–684 (1957)
5. Ben-Shabat, Y., Yu, X., Saleh, F., Campbell, D., Rodriguez-Opazo, C., Li, H., Gould, S.: The ikea asm dataset: Understanding people assembling furniture through actions, objects and pose. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 847–859 (January 2021)
6. Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of Markov decision processes. Mathematics of operations research **27**(4), 819–840 (2002)
7. Bewley, T., Kohlberg, E.: On stochastic games with stationary optimal strategies. Mathematics of Operations Research **3**(2), 104–125 (1978)
8. Boutilier, C., Dean, T., Hanks, S.: Decision-theoretic planning: Structural assumptions and computational leverage. Journal of Artificial Intelligence Research **11**, 1–94 (1999)

9. Cassandra, A.R.: A survey of POMDP applications. In: AAAI 1998 fall symposium on planning with partially observable Markov decision processes. vol. 1724 (1998)
10. Castro, P.S., Panangaden, P., Precup, D.: Equivalence relations in fully and partially observable Markov decision processes. In: IJCAI. vol. 9, pp. 1653–1658 (2009)
11. Castro, P.S., Panangaden, P., Precup, D.: Notions of state equivalence under partial observability. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09). pp. 1653–1658 (2009)
12. Charniak, E., Goldman, R.P.: A bayesian model of plan recognition. Artificial Intelligence **64**(1), 53–79 (1993)
13. Chatterjee, K., Henzinger, T.A.: A survey of stochastic ω-regular games. Journal of Computer and System Sciences **78**(2), 394–413 (2012)
14. Daswani, M., Sunehag, P., Hutter, M.: Feature reinforcement learning using looping suffix trees. In: European Workshop on Reinforcement Learning. pp. 11–24. PMLR (2013)
15. De Alfaro, L., Henzinger, T.A., Mang, F.Y.: The control of synchronous systems. In: International Conference on Concurrency Theory. pp. 458–473. Springer (2000)
16. De Alfaro, L., Henzinger, T.A., Mang, F.Y.: The control of synchronous systems, part II. In: International Conference on Concurrency Theory. pp. 566–581. Springer (2001)
17. Filar, J., Vrieze, K.: Competitive Markov decision processes. Springer Science & Business Media (2012)
18. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2023), https://www.gurobi.com
19. Hauskrecht, M.: Value-function approximations for partially observable Markov decision processes. Journal of artificial intelligence research **13**, 33–94 (2000)
20. Hermanns, H., Krčál, J., Křetínský, J.: Probabilistic bisimulation: Naturally on distributions. In: International Conference on Concurrency Theory. pp. 249–265. Springer (2014)
21. Holmes, M.P., Isbell Jr, C.L.: Looping suffix tree-based inference of partially observable hidden state. In: Proceedings of the 23rd international conference on Machine learning. pp. 409–416 (2006)
22. Horák, K., Bošanskỳ, B., Kiekintveld, C., Kamhoua, C.: Compact representation of value function in partially observable stochastic games. arXiv preprint arXiv:1903.05511 (2019)
23. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial intelligence **101**(1-2), 99–134 (1998)
24. Kearns, M., Mansour, Y., Ng, A.: Approximate planning in large POMDPs via reusable trajectories. Advances in Neural Information Processing Systems **12** (1999)
25. Kim, D., Lee, J., Kim, K.E., Poupart, P.: Point-based value iteration for constrained POMDPs. In: IJCAI. vol. 11, pp. 1968–1974 (2011)
26. Kochenderfer, M.J.: Decision making under uncertainty: theory and application. MIT press (2015)
27. Lim, M.H., Tomlin, C.J., Sunberg, Z.N.: Sparse tree search optimality guarantees in POMDPs with continuous observation spaces. arXiv preprint arXiv:1910.04332 (2019)
28. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In: AAAI/IAAI. pp. 541–548 (1999)
29. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. Artificial Intelligence **147**(1-2), 5–34 (2003)
30. McCallum, R.A.: Instance-based utile distinctions for reinforcement learning with hidden state. In: Machine Learning Proceedings 1995, pp. 387–395. Elsevier (1995)
31. Meuleau, N., Kim, K.E., Kaelbling, L.P., Cassandra, A.R.: Solving POMDPs by searching the space of finite policies. arXiv preprint arXiv:1301.6720 (2013)
32. Meuleau, N., Peshkin, L., Kim, K.E., Kaelbling, L.P.: Learning finite-state controllers for partially observable environments. arXiv preprint arXiv:1301.6721 (2013)

33. Mohri, M., Rostamizadeh, A., Talwalkar, A.: Foundations of Machine Learning. The MIT Press (2012)
34. Murty, K.G., Yu, F.T.: Linear complementarity, linear and nonlinear programming, vol. 3. Citeseer (1988)
35. Nieuwenhuis, R., Oliveras, A., Tinelli, C.: Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL (t). Journal of the ACM (JACM) **53**(6), 937–977 (2006)
36. Papadimitriou, C.H., Tsitsiklis, J.N.: The complexity of Markov decision processes. Mathematics of operations research **12**(3), 441–450 (1987)
37. Pineau, J., Gordon, G., Thrun, S., et al.: Point-based value iteration: An anytime algorithm for POMDPs. In: Ijcai. vol. 3, pp. 1025–1032 (2003)
38. Roy, N., Gordon, G.J.: Exponential family PCA for belief compression in POMDPs. Advances in Neural Information Processing Systems **15** (2002)
39. Shani, G., Brafman, R.I., Shimony, S.E.: Forward search value iteration for POMDPs. In: IJCAI. pp. 2619–2624. Citeseer (2007)
40. Shapley, L.S.: Stochastic games. Proceedings of the national academy of sciences **39**(10), 1095–1100 (1953)
41. Silver, D., Veness, J.: Monte-carlo planning in large POMDPs. Advances in neural information processing systems **23** (2010)
42. Spaan, M.T., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. Journal of artificial intelligence research **24**, 195–220 (2005)
43. Strauch, R.E.: Negative dynamic programming. The Annals of Mathematical Statistics **37**(4), 871–890 (1966)
44. Subramanian, J., Sinha, A., Seraj, R., Mahajan, A.: Approximate information state for approximate planning and reinforcement learning in partially observed systems. The Journal of Machine Learning Research **23**(1), 483–565 (2022)
45. Theocharous, G., Kaelbling, L.: Approximate planning in POMDPs with macro-actions. Advances in neural information processing systems **16** (2003)
46. Verwer, S., Hammerschmidt, C.A.: flexfringe: A passive automaton learning package. In: 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 638–642 (2017). https://doi.org/10.1109/ICSME.2017.58
47. Weisstein, E.W.: Matrix Norm (2002), cf. https://mathworld.wolfram.com/MatrixNorm.html
48. Yang, L., Zhang, K., Amice, A., Li, Y., Tedrake, R.: Discrete approximate information states in partially observable environments. In: 2022 American Control Conference (ACC). pp. 1406–1413. IEEE (2022)
49. Yoon, H., Sankaranarayanan, S.: Predictive runtime monitoring for mobile robots using logic-based bayesian intent inference. In: International Conference on Robotics and Automation (ICRA). pp. 8565–8571. IEEE (2021)

A Rock Paper Scissors with Memory

We describe the RPS-MEM benchmark used in our approach. The state of the game $\mathcal{G}$ is given as $S = A_1 \times A_2$ wherein $A_1 = \{r_1, p_1, s_1\}$ and $A_2 = \{r_2, p_2, s_2\}$ while $A^{(1)} = A_1$ and $A^{(2)} = A_2$. The transition map is given as follows:

$$P((s_1,s_2) \mid s,a_1,a_2) = \begin{cases} 1 & \text{if } s_1 = a_1,\ s_2 = a_2 \\ 0 & \text{otherwise} \end{cases}$$

In other words, the state s “remembers” the previous action of both players. The reward map for each state is identical to that of the RPS game from Example 1.

We define 9 policies for $\mathcal{P}_2$.

Policy π_1 chooses rock/paper with 0.45 probability and scissors with 0.1 probability regardless of the state.

$$\pi_1(a,b) = \{r_2 : 0.45, p_2 : 0.45, s_2 : 0.1\}.$$

Likewise, we define policies π_2, π_3.

$$\pi_2(a,b) = \{r_2 : 0.45, p_2 : 0.1, s_2 : 0.45\}$$

$$\pi_3(a,b) = \{r_2 : 0.1, p_2 : 0.45, s_2 : 0.45\}$$

Policy π_4: mostly repeat what player $\mathcal{P}_1$ played in the previous round.

$$\pi_4(a,b) = \begin{cases} \{r_2 : 0.8, p_2 : 0.1, s_2 : 0.1\} & \text{if } a = r_1 \\ \{r_2 : 0.1, p_2 : 0.8, s_2 : 0.1\} & \text{if } a = p_1 \\ \{r_2 : 0.1, p_2 : 0.1, s_2 : 0.8\} & \text{if } a = s_1 \end{cases}$$

Policy π_5: mostly play what would have beaten player 1 in the previous round.

$$\pi_5(a,b) = \begin{cases} \{r_2 : 0.8, p_2 : 0.1, s_2 : 0.1\} & \text{if } a = s_1 \\ \{r_2 : 0.1, p_2 : 0.8, s_2 : 0.1\} & \text{if } a = r_1 \\ \{r_2 : 0.1, p_2 : 0.1, s_2 : 0.8\} & \text{if } a = p_1 \end{cases}$$

Policy π_6: Mostly play what player 1 did not play in the previous round.

$$\pi_6(a,b) = \begin{cases} \{r_2 : 0.1, p_2 : 0.45, s_2 : 0.45\} & \text{if } a = r_1 \\ \{r_2 : 0.45, p_2 : 0.1, s_2 : 0.45\} & \text{if } a = p_1 \\ \{r_2 : 0.45, p_2 : 0.45, s_2 : 0.1\} & \text{if } a = s_1 \end{cases}$$

Policy π_7: mostly repeat what player $\mathcal{P}_2$ played in the previous round.

$$\pi_4(a,b) = \begin{cases} \{r_2 : 0.8, p_2 : 0.1, s_2 : 0.1\} & \text{if } b = r_2 \\ \{r_2 : 0.1, p_2 : 0.8, s_2 : 0.1\} & \text{if } b = p_2 \\ \{r_2 : 0.1, p_2 : 0.1, s_2 : 0.8\} & \text{if } b = s_2 \end{cases}$$

Policy π_8: mostly play what would have beaten $\mathcal{P}_2$ in the previous round.

$$\pi_5(a,b) = \begin{cases} \{r_2: 0.8, p_2: 0.1, s_2: 0.1\} & \text{if } b = s_2 \\ \{r_2: 0.1, p_2: 0.8, s_2: 0.1\} & \text{if } b = r_2 \\ \{r_2: 0.1, p_2: 0.1, s_2: 0.8\} & \text{if } b = p_2 \end{cases}$$

Policy π_9: mostly play what $\mathcal{P}_2$ did not play in the previous round.

$$\pi_7(a,b) = \begin{cases} \{r_2: 0.1, p_2: 0.45, s_2: 0.45\} & \text{if } b = r_2 \\ \{r_2: 0.45, p_2: 0.1, s_2: 0.45\} & \text{if } b = p_2 \\ \{r_2: 0.45, p_2: 0.45, s_2: 0.1\} & \text{if } b = s_2 \end{cases}$$

B Anticipate and Avoid

Anticipate and Avoid game involves a circular arena with N cells labeled 1,...,N. The state space S encodes joint positions of two players in this arena.

$$S = \{(i,j) \mid 1 \leq i \leq N, 1 \leq j \leq N\}.$$

Let us define $i \oplus 1$ as the same as $i+1$ if $1leqi \leq N-1$ and to be 1 if $i=1$. Likewise, we define $i \ominus 1$ as $i-1$ for $2 \leq i \leq N$ and N if $i=1$.

The actions are $A^{(1)} = A^{(2)} = \{L, R\}$ standing for left and right, respectively. Let us define $p(j|i,a)$ for a single player as follows:

$$p(j|i,a) = \begin{cases} 0.2 & j = i \\ 0.8 & j = i \oplus 1, a = R \\ 0.8 & j = i \ominus 1, a = L \\ 0 & \text{otherwise} \end{cases}$$

In other words, upon moving left, the player may stay in the same cell with 0.2 probability or move to "previous" cell with 0.8 probability and similarly for moving right.

The reward map is defined by first defining a state distance function:

$$\rho(i,j) = \begin{cases} \min(j-i, i-j+N) & \text{if } i \leq j \\ \min(i-j, j-i+N) & \text{if } i > j \end{cases}$$

We define the reward for state/actions as

$$R((i,j),a_1,a_2) = \begin{cases} -10 & i = j \\ -5 & i \neq j \wedge \rho(i,j) \leq N/10 \\ 0 & i \neq j \wedge \rho(i,j) \in (N/10, 3N/10] \\ 1 & \text{otherwise} \end{cases}$$

In other words, the reward structure incentivizes i,j positions to be farther apart than $3N/10$.

Player 2 can play one of four policies of the form target_t for $t=1,\lceil N/4\rceil,\lceil 2N/4\rceil,\lceil 3N/4\rceil$, where the policy $\mathsf{target}(j)$ is defined as

$$\mathsf{target}_t(i,j)=\begin{cases}\{L{:}0.8,R{:}0.2\} & j>t \wedge (t-j+N\geq j-t)\\ \{L{:}0.8,R{:}0.2\} & j<t \wedge (j-t+N\leq t-j)\\ \{L{:}0.2,R{:}0.8\} & j>t \wedge (t-j+N\leq t-j)\\ \{L{:}0.2,R{:}0.8\} & j<t \wedge (j-t+N\geq t-j)\\ \{L{:}0.5,R{:}0.5\} & \text{otherwise}\end{cases}$$

C Appendix: Ikea Furniture Assembly and Cataract Surgery Graphs

The ikea furniture assembly dataset was taken from the previous work of Ben-Shabat et al [5]. It involves sequences of tasks for four different furniture types with roughly 90 sequences for each furniture type. We employed a 75%-25% training/testing data split. The tool flexfringe was used to learn an automaton model using sequences in the training data.

0	flip table top	1	pick up leg	2	align leg screw with table thread
3	spin leg	4	other	5	tighten leg
6	rotate table	7	flip table	8	pick up shelf
9	attach shelf to table	10	pick up table top	11	lay down table top
12	push table	13	flip shelf	14	lay down leg
15	lay down shelf	16	push table top	17	pick up side panel
18	align side panel holes with front panel dowels	19	attach drawer side panel	20	pick up bottom panel
21	slide bottom of drawer	22	pick up back panel	23	attach drawer back panel
24	pick up pin	25	insert drawer pin	26	position the drawer right side up
27	pick up front panel	28	lay down bottom panel	29	lay down front panel
30	lay down back panel	31	lay down side panel		

Table 3. Action IDs and their description for the IKEA furniture assembly benchmark.

0	+Bonn forceps	1	+secondary incision knife	2	-Bonn forceps
3	-secondary incision knife	4	+primary incision knife	5	-primary incision knife
6	+viscoelastic cannula	7	-viscoelastic cannula	8	+capsulorhexis cystotome
9	-capsulorhexis cystotome	10	+capsulorhexis forceps	11	-capsulorhexis forceps
12	+hydrodissection canula	13	-hydrodissection canula	14	+phacoemulsifier handpiece
15	+micromanipulator	16	-phacoemulsifier handpiece	17	-micromanipulator
18	+irrigation/aspiration handpiece	19	-irrigation/aspiration handpiece	20	+implant injector
21	-implant injector	22	+Rycroft canula	23	-Rycroft canula
24	+Troutman forceps	25	-Troutman forceps	26	+cotton
27	-cotton	28	+Charleux canula	29	-Charleux canula
30	+suture needle	31	-suture needle	32	+Vannas scissors
33	-Vannas scissors	34	+needle holder	35	-needle holder
36	+vitrectomy handpiece	37	-vitrectomy handpiece	38	+biomarker
39	-biomarker	40	+Mendez ring	41	-Mendez ring

Table 4. Action IDs and their description for the cataract surgery benchmark. A “+” sign before a tool indicates its introduction during a particular step, whereas a “-” sign indicates its removal.

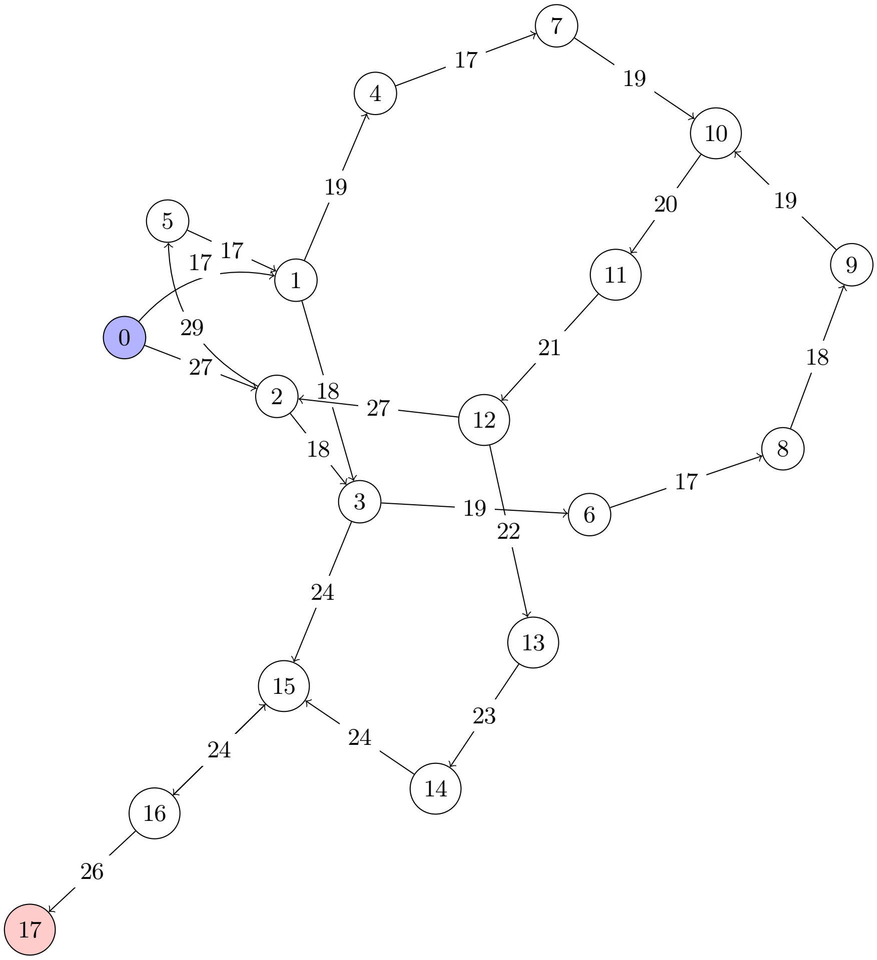


Fig. 5. IKEA Shelf Drawer Assembly Task Machine.

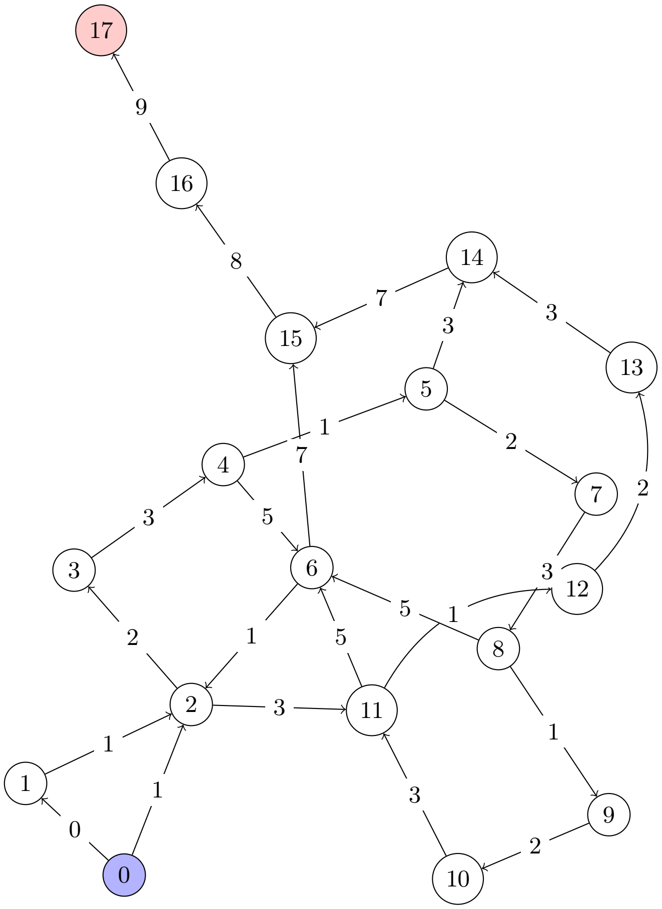


Fig. 6. IKEA TV Bench Assembly Task Machine

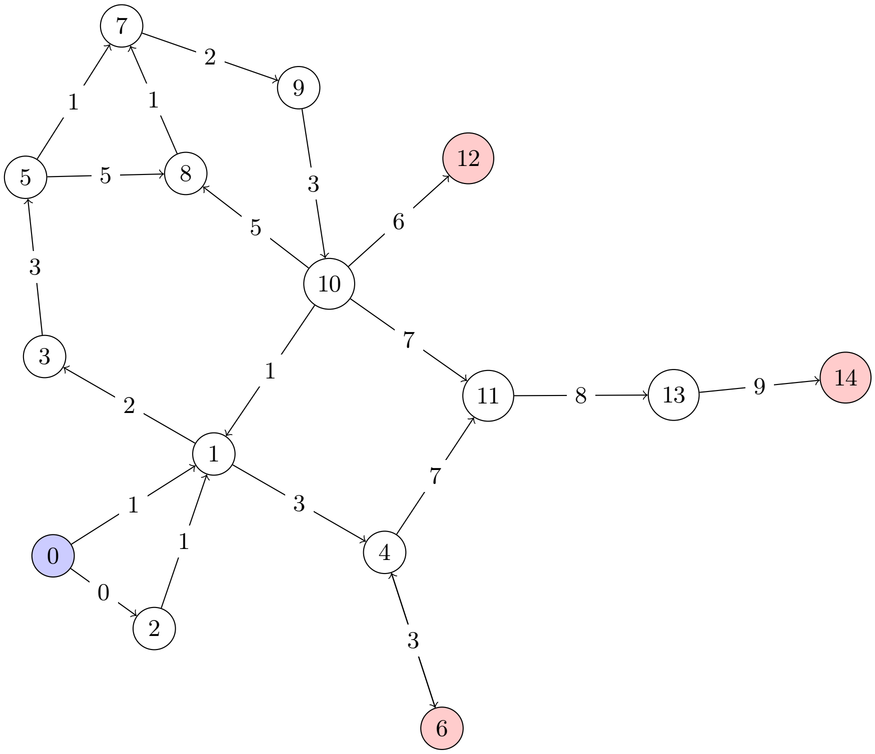


Fig. 7. IKEA Coffee Table Assembly Task Machine

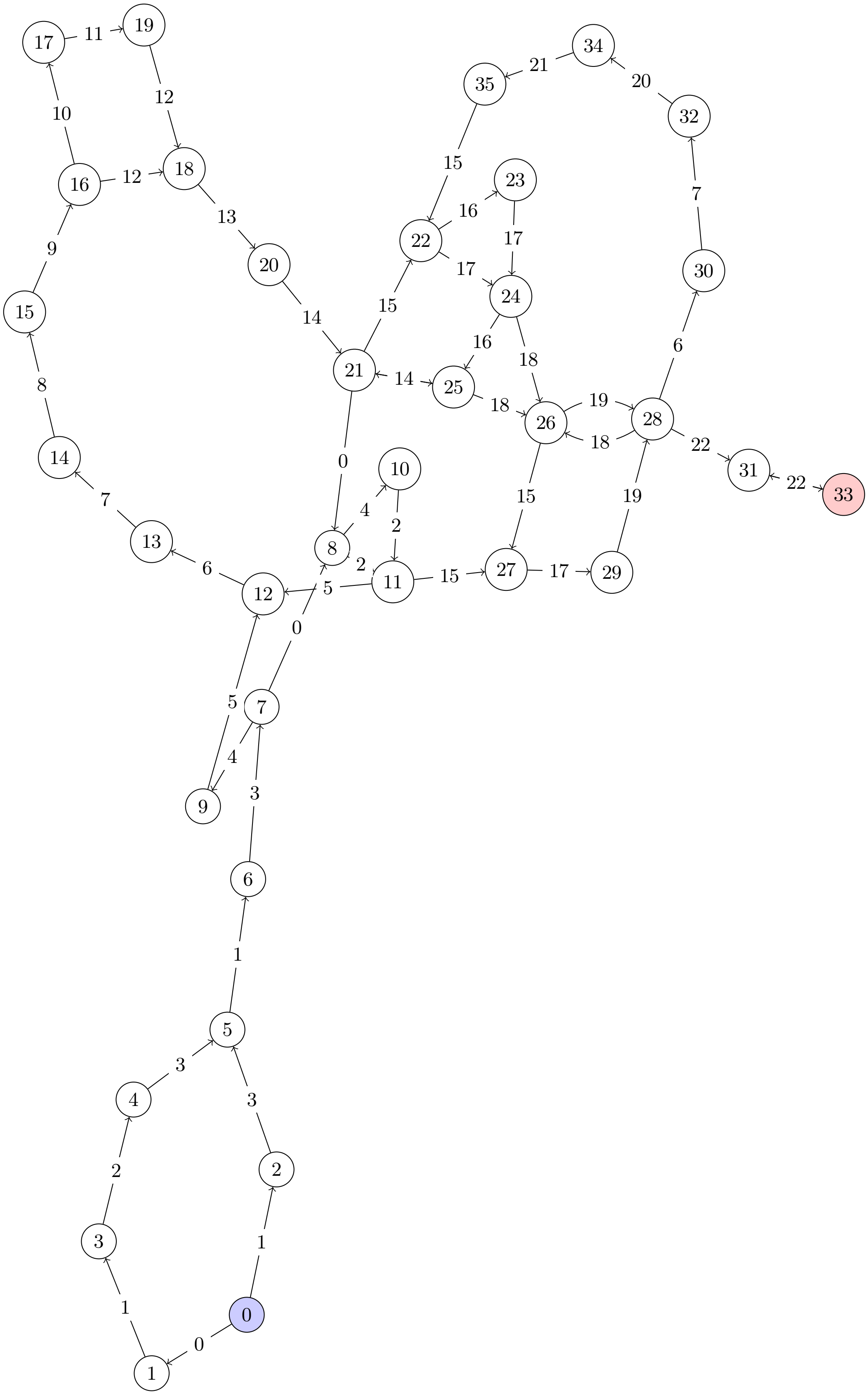


Fig. 8. Cataract Surgery Task Machine