Steady-State Strategy Synthesis for Swarms of Autonomous Agents

Martin Jonáš, **Antonín Kučera**, **Vojtěch Kůr** and **Jan Mačák**
Faculty of Informatics, Masaryk University, Czechia
martin.jonas@mail.muni.cz, tony@fi.muni.cz, vojtech.kur@mail.muni.cz, macak.jan@mail.muni.cz

Abstract

Steady-state synthesis aims to construct a policy for a given MDP D such that the long-run average frequencies of visits to the vertices of D satisfy given numerical constraints. This problem is solvable in polynomial time, and memoryless policies are sufficient for approximating an arbitrary frequency vector achievable by a general (infinite-memory) policy.

We study the steady-state synthesis problem for *multiagent systems*, where multiple autonomous agents jointly strive to achieve a suitable frequency vector. We show that the problem for multiple agents is computationally hard (PSPACE or NP hard, depending on the variant), and memoryless strategy profiles are *insufficient* for approximating achievable frequency vectors. Furthermore, we prove that even *evaluating* the frequency vector achieved by a given memoryless profile is computationally hard. This reveals a severe barrier to constructing an efficient synthesis algorithm, even for memoryless profiles. Nevertheless, we design an *efficient and scalable* synthesis algorithm for a subclass of *full* memoryless profiles, and we evaluate this algorithm on a large class of randomly generated instances. The experimental results demonstrate a significant improvement against a naive algorithm based on strategy sharing.

1 Introduction

Steady-state policy synthesis is the problem of computing a suitable decision-making policy (strategy) in a given Markov decision process (MDP) D satisfying given constraints on the limit frequencies of visits to the states of D. More precisely, we say that a strategy σ in D *achieves* a frequency vector μ if for almost every infinite run w in the Markov chain D^σ induced by the strategy and every vertex v of D we have that the limit frequency of visits to v along w is equal to $\mu(v)$.

The existing works concentrate on the steady-state synthesis problem for a *single* agent, where the task is to construct a strategy σ achieving a frequency vector μ where $\vec{v}_\ell \leq \mu \leq \vec{v}_u$ for given lower and upper bounds $\vec{v}_\ell$ and $\vec{v}_u$. The existence of such a strategy is decidable in polynomial time; and if it exists, it can be also computed in polynomial time by linear programming (see *Related work*). Although some frequency vectors are only achievable by infinite-memory strategies, the subclass of *memoryless* strategies is sufficient for producing frequency vectors *arbitrarily close* to each achievable frequency vector. If the underlying graph of D is strongly connected, then the same holds even for a special type of *full* memoryless strategies assigning a positive probability to every edge of D (one can show that for every μ achievable by a memoryless strategy, there is a vector arbitrarily close to μ achievable by a full memoryless strategy). These properties are illustrated in Fig.1 on a trivial MDP with three non-deterministic states. Hence, in the single agent setting, memoryless strategies are sufficient for practical applications. Since we can safely assume that D is a disjoint union of finitely many strongly connected MDPs, the same holds even for *full* memoryless strategies (see Section 4 for details).

Our contribution. In this paper, we extend the scope of steady-state policy synthesis problem to *multiple autonomous agents*. More precisely, the task is to construct a strategy profile for $k \geq 1$ agents in a given MDP D so that the frequencies of visits to the vertices of D (or, more generally, to pre-defined classes of vertices represented by *colors*) by some agent are above a given threshold vector. As a simple example, consider an MDP where the vertices represent devices requiring regular maintenance, and the threshold frequency vector specifies the minimal required frequency of inspections for each device. The steady-state policy synthesis for k agents then corresponds to the problem of designing appropriate schedules for k independent technicians such that the required frequency of inspections is observed.[1] Our main results are twofold.

I. Fundamental properties of the problem. We analyze the role of memory and randomization in constructing (sub)optimal strategy profiles, and we also classify the computational complexity of the steady-state policy synthesis problem. The obtained results demonstrate that the steady-

[1] If two or more technicians meet at the same vertex at the same time, only one of them does the maintenance job. Hence, optimal strategy profiles tend to minimize the frequency of such redundant simultaneous visits. However, this redundancy cannot be avoided completely in general.

state policy synthesis for multiple agents is (perhaps even surprisingly) *more complex* than for a single agent. Consequently, a different algorithmic approach is required. More concretely, we prove the following:

I(a). For two or more agents, the power of full memoryless, memoryless, finite-memory, and general strategy profiles increases strictly. To explain this, we need to introduce one extra notion. Let $k \geq 1$, and let A, B be sets of strategy profiles for an MDP D and k agents. Furthermore, let $\mathcal{F}(A)$ and $\mathcal{F}(B)$ be the sets of frequency vectors achievable by the profiles in A and B. We say that B is *more powerful* than A if $\mathcal{F}(A) \subseteq \mathcal{F}(B)$, and there exists $\mu \in \mathcal{F}(B)$ that cannot be approximated by the vectors of $\mathcal{F}(A)$ (i.e., there is $\delta > 0$ such that the distance between μ and every $\nu \in \mathcal{F}(A)$ is at least δ).

We show that for $k \geq 2$, the sets of full memoryless profiles, memoryless profiles, finite-memory profiles, and general profiles are increasingly more powerful even for strongly connected *graphs*, i.e., MDPs without stochastic vertices. This contrasts sharply with the single agent scenario where full memoryless profiles approximate general profiles.

I(b) The existence of an achievable μ such that $\mu \geq \vec{v}_\ell$ for a given $\vec{v}_\ell$ is a computationally hard problem. Recall that for a single agent, the problem is solvable in polynomial time. For two or more agents, the problem is NP-hard even if D is a strongly connected graph and the set of profiles is restricted to full memoryless profiles, memoryless profiles, or finite-memory profiles with m memory states. For the "colored" variant of the problem, we obtain even PSPACE-hardness.

I(c) Evaluating the frequency vector achieved by a given profile is computationally hard, even for strongly connected graphs and memoryless profiles. Intuitively, the reason is that each strategy in the profile may induce a Markov chain with a different period. The complexity of the evaluation procedure depends on the least common multiple of these periods whose size can be exponential in k. Note that for *full* memoryless profiles, all of the induced Markov chains have the *same* period. Consequently, full memoryless profiles can be evaluated in polynomial time on strongly connected MDPs. These observations have important algorithmic consequences explained in the subsection *II. Efficient synthesis algorithm.*

I(d) The existence of a finite-memory profile with m memory states achieving a frequency vector μ such that $\mu \geq \vec{v}_\ell$ for a given $\vec{v}_\ell$ is decidable in polynomial space for every fixed number of agents. This holds also for the "colored" variant of the problem. The algorithm is based on encoding the problem as a formula of first order theory of the reals and applying the results of [Canny, 1988]. The size of the formula is exponential in k, which shows that the number of agents is a key parameter negatively influencing the computational costs.

II. Efficient synthesis algorithm. Since general (infinite-memory) strategies are not algorithmically workable, the scope of algorithmic synthesis is naturally limited to *finite-memory* profiles. The synthesis of a finite-memory profile for an MDP D where every strategy in the profile uses at most m memory states is equivalent to the synthesis of a *memoryless* profile for an MDP D' obtained from D by augmenting its vertices with memory states (see Section 2 for details). Hence, the algorithmic core of the problem is the construction of *memoryless* profiles. However, here we face the obstacle of I(c), saying that even *evaluating* memoryless profiles is computationally hard. This is a severe barrier, because every synthesis algorithm is driven by the objective involving the frequency vector of the constructed profile. Hence, a natural starting point is to explore the constructability of *full* memoryless profiles that can be evaluated in polynomial time (see I(c)). This is challenging, despite the limitations identified in I(a). According to I(b), the associated decision problem is NP-hard even for two agents, and the synthesis can be seen as a non-linear optimization problem whose size increases with the number of agents (see Section 4).

We propose an efficient algorithm for synthesizing full memoryless profiles based on *incremental agent inclusion*. The main idea is the following: Suppose that we already constructed a full memoryless profile for k agents, and we wish to extend the profile to $k+1$ agents. Our algorithm constructs several linear programs depending only on the threshold vector (the objective) and numerical parameters extracted from the previously computed profile for k agents. Hence, the size of these programs is *independent of* k. A full memoryless strategy for the newly included agent is extracted from the solutions of these linear programs. Thus, we prevent the blowup in k, and the complexity of our synthesis algorithm becomes *linear* in the number of agents k. Thus, we (inevitably) trade efficiency for completeness, i.e., the algorithm does not have to find a suitable full MR profile even if it exists. We evaluate our algorithm experimentally on a series of randomly generated instances, and we show that it clearly outperforms a naive algorithm based on strategy sharing (see Section 5 for details).

Related work. All existing works about steady-state synthesis apply to a single agent scenario. [Akshay *et al.*, 2013] solve the problem for *unichain MDPs*, i.e., a subclass of MDPs where every memoryless deterministic policy induces an ergodic Markov chain, by designing a polynomial-space algorithm. A polynomial-time algorithm for general MDPs is given in [Brázdil *et al.*, 2014]. This algorithm can compute *infinite-memory* strategies, which may be necessary for achieving some frequency vectors (see Fig. 1), and it is applicable to a more general class of multiple mean-payoff objectives. It has been implemented [Brázdil *et al.*, 2015] on top of the PRISM model checker [Kwiatkowska *et al.*, 2011]. In [Velasquez, 2019], the problem of constructing a suitable memoryless policy inducing a recurrent Markov chain consisting of all vertices of a given MDP is solved by linear programming. A generalization of this work is presented in [Atia *et al.*, 2020]. Recent works [Křetínský, 2021; Velasquez *et al.*, 2024] combine steady-state constraints with LTL specifications. There are also works concentrating on steady-state deterministic policy synthesis [Velasquez *et al.*, 2023].

2 The Model

We assume familiarity with basic notions of probability theory and Markov chain theory. We use $\mathbb{N}$ and $\mathbb{N}_+$ to denote the sets of all non-negative and positive integers, respectively, and $\mathcal{D}(A)$ to denote the set of all probability distributions over a finite set A. A directed graph is a pair $G = (V, E)$

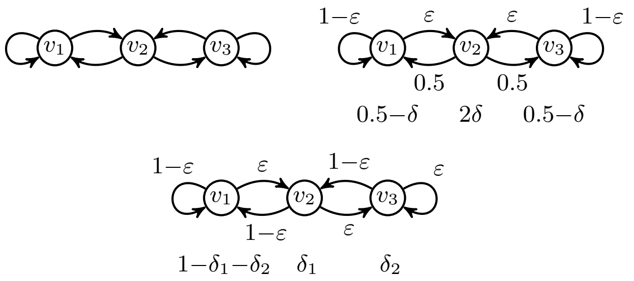

Figure 1: Left: A simple MDP with three non-deterministic vertices v_1, v_2, and v_3. Right: A memoryless strategy for a single agent can achieve the frequency vector $(0.5{-}\delta, 2\delta, 0.5{-}\delta)$ for an arbitrarily small $\delta{>}0$ by choosing a sufficiently small $\varepsilon{>}0$. However, the frequency vector $(0.5, 0, 0.5)$ is achievable only by an infinite-memory strategy where the ε is "progressively smaller" and approaches 0 as the vertices v_1 and v_3 are revisited. Middle: A full memoryless strategy can achieve the frequency vector $(1{-}\delta_1{-}\delta_2, \delta_1, \delta_2)$ where $\delta_1{+}\delta_2 > 0$ is arbitrarily small by choosing a sufficiently small $\varepsilon{>}0$. However, the vector $(1, 0, 0)$ is achievable only by a (non-full) strategy assigning 1 to the self-loop $v_1{\to}v_1$.

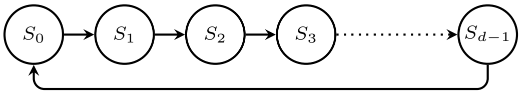

Figure 2: The structure of cyclic classes. For all states s, t we have that $P(s, t) > 0$ only if $s \in S_i$ and $t \in S_{i+1 \bmod d}$ for some $i < d$.

where $E \subseteq V \times V$. For every $v \in V$, we use *In*(v) and *Out*(v) to denote the sets of all in-going and out-going edges of v. We say that G is *strongly connected* if for all $v, u \in V$ there is a finite sequence $v_1, \ldots, v_n$ such that $n \geq 1$, $v_1 = v$, $v_n = u$, and $(v_i, v_{i+1}) \in E$ for all $1 \leq i < n$.

Markov chains. A *Markov chain* is a triple $C = (S, P, \alpha)$ where S is a finite set of states, $P \colon S \times S \to [0, 1]$ is a stochastic matrix such that $\sum_{s' \in S} P(s, s') = 1$ for every $s \in S$, and $\alpha \in \mathcal{D}(S)$ is an initial distribution.

A state t is *reachable* from a state s if $P^n(s, t) > 0$ for some $n \geq 1$, where P^n denotes the n-th power of P. A *bottom strongly connected component (BSCC)* of C is a maximal $B \subseteq S$ such that B is strongly connected and closed under reachable states, i.e., for all $s, t \in B$ and $r \in S$ we have that t is reachable from s, and if r is reachable from s, then $r \in B$. A Markov chain C is *irreducible* if for all $s, t \in S$ we have that t is reachable from s. We use $\mathbb{I}$ to denote the unique *invariant distribution* of C. Note that every BSCC of C can be seen as an irreducible Markov chain.

For every $s \in S$, let $d(s) = \gcd\{n \in \mathbb{N}_+ \mid P^n(s, s){>}0\}$ be the *period* of s. Recall that if C is irreducible, then $d(s)$ is the same for all $s \in S$ and defines the *period* of C, denoted by d (if C is not clear, we write d_C instead of d). Furthermore, the set S can be partitioned into *cyclic classes* $S_0, \ldots, S_{d-1}$ such that for all $i, j \in \{0, \ldots, d{-}1\}$ and $s, t \in S$ where $s \in S_i$ we have that $t \in S_j$ iff $P^n(s, t) > 0$ for some $n \equiv (j{-}i) \bmod d$. The structure of cyclic classes is shown in Fig. 2. We say that C is *aperiodic* or *periodic* depending on whether $d{=}1$ or not, respectively.

Markov decision processes (MDPs). A *Markov decision process (MDP)*[2] is a triple $D{=}(V, E, p)$ where V is a finite set of *vertices* partitioned into subsets (V_N, V_S) of *non-deterministic* and *stochastic* vertices, $E \subseteq V{\times}V$ is a set of *edges* such that every vertex has at least one outgoing edge, and $p \colon V_S{\to}\mathcal{D}(V)$ is a *probability assignment* s.t. $p(v)(v'){>}0$ iff $(v, v') \in E$. A *run* of D is an infinite sequence $\omega = v_1, v_2, \ldots$ such that $(v_i, v_{i+1}) \in E$ for every $i \in \mathbb{N}$. The i-th vertex v_i visited by ω is denoted by $\omega(i)$. We say D is *strongly connected* if the underlying directed graph (V, E) is strongly connected. D is a *graph* if $V_S = \emptyset$.

Strategies. Outgoing edges in non-deterministic states of an MDP $D = (V, E, p)$ are selected by a *strategy*. The most general type of strategy is a *history-dependent randomized (HR)* strategy where the selection may be randomized and depend on the whole computational history. Since HR strategies require infinite memory, they are not apt for algorithmic purposes.

A strategy is *memoryless randomized (MR)* if the (possibly randomized) decision depends only on the current vertex. Formally, a MR strategy is a pair $\sigma = (v_0, \kappa)$ where $v_0 \in V$ is the *initial* vertex and $\kappa : V \to \mathcal{D}(V)$ is a function such that $\kappa(v)(u) > 0$ implies $(v, u) \in E$, and for all $v \in V_S$ and $u \in V$ we have that $\kappa(v)(u) = p(v)(u)$. We say that σ is *full* if $\kappa(v)(u) > 0$ for all $(v, u) \in E$.

In this paper, we also consider finite-memory randomized strategies with $m \geq 1$ memory states (*FR*$_m$ *strategies*). Intuitively, the memory states are used to "remember" some information about the sequence of previously visited vertices. Formally, let $V' = V \times \{1, \ldots, m\}$ be the set of *augmented vertices*. A FR$_m$ strategy is a pair $((v_0, i_0), \eta)$ where $(v_0, i_0) \in V'$ is an *initial* augmented vertex and $\eta \colon V' \to \mathcal{D}(V')$ such that $\eta(v, i)(u, j) > 0$ implies $(v, u) \in E$. Furthermore, for every (v, i) where $v \in V_S$ and every $(v, u) \in E$ we require $\sum_{j=1}^{m} \eta(v, i)(u, j) = p(v)(u)$. Note that every FR$_m$ strategy can be seen as a *memoryless* strategy for an MDP D' where V' is the set of vertices.

Let ξ be a strategy (HR, FR$_m$, or MR). For every finite path $v_1, \ldots, v_n$ in D, the strategy ξ determines the probability $\mathbb{P}_\xi(v_1, \ldots, v_n)$ of executing the path. By applying the extension theorem (see, e.g., [Rosenthal, 2006]), the function $\mathbb{P}_\xi$ is extended to the probability measure over all runs in D.

Strategy profiles. Let $k \geq 1$. A HR, FR$_m$, MR, or full MR *strategy profile* for k agents is a tuple $\pi = (\xi_1, \ldots, \xi_k)$ where every ξ_i is a HR, FR$_m$, MR, or full MR strategy. A *multi-run* is a tuple $\varrho = (\omega_1, \ldots, \omega_k)$ where each ω_i is a run of D. We use $\mathbb{P}_\pi$ to denote the product measure in the product probability space over the set of all multi-runs.

Steady-state objectives. Let $D{=}(V, E, p)$ be an MDP and *Col* $: V \to \gamma$ a *coloring*, where $\gamma \neq \emptyset$ is a finite set of colors. A coloring is *trivial* if $\gamma{=}V$ and *Col*$(v){=}v$ for all $v \in V$.

Let $\pi = (\xi_1, \ldots, \xi_k)$ be a strategy profile and $\varrho = (\omega_1, \ldots, \omega_k)$ a multi-run. For all $c \in \gamma$ and $n \geq 1$, we use $\#_c^n(\varrho)$ to denote the total number of all $j \in \{1, \ldots, n\}$ such

[2] The adopted MDP definition is standard in the area of graph games. It is equivalent to the "classical" definition of [Puterman, 1994] but leads to simpler notation.

that $\mathit{Col}(\omega_i(j)) = c$ for some $i \in \{1, \dots, k\}$. Furthermore, we define

$$\mathit{Freq}_c(\varrho) \;=\; \lim_{n\to\infty} \frac{\#^n_c(\varrho)}{n}.$$

If the above limit does not exist, we put $\mathit{Freq}_c(\varrho) = \bot$. We use $\mathit{Freq}(\varrho) : \gamma \to [0,1]$ to denote the vector of all $\mathit{Freq}_c(\varrho)$.

Intuitively, $\mathit{Freq}_c(\varrho)$ is the long-run average frequency of visits to a c-colored vertex by some agent. We say that π *achieves* a vector $\mu : \gamma \to [0,1]$ if $\mathbb{P}_\pi[\mathit{Freq}{=}\mu] = 1$. That is, for every color c, the long-run average frequency of visits to a c-colored vertex is defined and equal to $\mu(c)$ for almost all multi-runs.

A *steady-state objective* is a vector $\mathit{Obj} : \gamma \to [0,1]$. The task is to construct a strategy profile π for k agents such that π achieves a vector $\mu \geq \mathit{Obj}$.

3 Fundamental Properties of Multi-Agent Steady-State Synthesis

In this section, we analyze the computational complexity of multi-agent steady-state synthesis. We also investigate the relative power of HR, FR$_m$, MR, and full MR strategy profiles. Proofs of the presented theorems are non-trivial and can be found in [Jonáš *et al.*, 2025].

Let A and B be sets of strategy profiles for an MDP D and $k \geq 1$ agents. Furthermore, let $\mathcal{F}(A)$ and $\mathcal{F}(B)$ be the sets of all frequency vectors achievable by the profiles of A and B, where *Col* is the trivial coloring (see Section 2). We say that A *approximates* B if for every $\mu \in \mathcal{F}(B)$ and every $\varepsilon > 0$, there is $\nu \in \mathcal{F}(A)$ such that $L_\infty(\mu{-}\nu) < \varepsilon$, where $L_\infty(\mu{-}\nu) = \max_c(|\mu(c){-}\nu(c)|)$ is the standard L_∞ norm. Furthermore, we say that B is *more powerful* than A, written $A \prec B$, if $\mathcal{F}(A) \subseteq \mathcal{F}(B)$ and A does *not* approximate B.

Slightly abusing our notation, we use $\mathit{HR}(D,k)$, $\mathit{FR}_m(D,k)$, $\mathit{MR}(D,k)$, and $\mathit{FMR}(D,k)$ to denote the sets of all HR, FR$_m$, MR, and full MR strategy profiles for an MDP D and $k \geq 1$ agents. The next theorem says that the relative power of HR, FR$_m$, MR, and full MR profiles *strictly decreases for $k \geq 2$ agents*, even if D is a strongly connected graph. Since the proof reveals important differences from the single agent scenario, we give a brief sketch.

Theorem 1. *There exist strongly connected graphs D_1, D_2, and D_3 such that*

- *$\mathit{FMR}(D_1,2) \prec \mathit{MR}(D_1,2)$;*
- *$\mathit{MR}(D_2,2) \prec \mathit{FR}_2(D_2,2)$;*
- *$\mathit{FR}_m(D_3,2) \prec \mathit{HR}(D_3,2)$ for all $m \geq 1$.*

The graphs D_1, D_2, and D_3 are shown in Fig. 3, together with the frequency vectors achievable by the more powerful strategy profiles that cannot be approximated by the weaker strategy profiles (for 2 agents).

In D_1, the vector $(1,1)$ is achievable by a MR profile where both agents "walk around the loop" connecting v_1 and v_2, but they start in different vertices. However, for every vector ν achievable by a FMR profile we have that $\nu(v_2) \leq 0.75$. That is, the L_∞-distance to $(1,1)$ is at least $\delta = 0.25$. Intuitively, this is because the self-loop $v_1{\to}v_1$ has to be performed with a fixed positive probability, and even if this probability is very small, the two agents spend a *significant* proportion of time by "walking together", regardless of their initial positions.

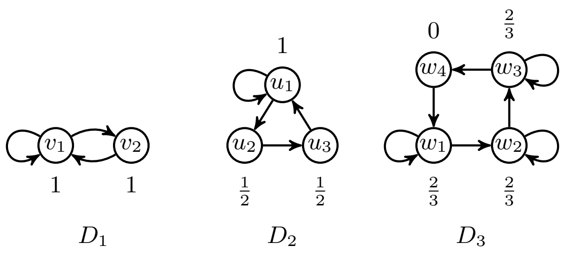

Figure 3: The graphs D_1, D_2, and D_3.

In D_2, a FR$_2$ profile achieving $(1, 0.5, 0.5)$ consists of strategies where both agents walk around the triangle, performing the self-loop on u_1 *exactly once* (this is where two memory states are needed). The first agent starts in u_1 by performing the self-loop, and the other agent starts in u_2. Thus, the agents never meet, and together they produce the frequency vector $(1, 0.5, 0.5)$. However, for every vector ν achievable by a MR profile we have that the L_∞-distance to $(1, 0.5, 0.5)$ is at least $1/9$. Observe that if both MR strategies assign zero probability to the self-loop on u_1, then the frequency of visits to u_1 achieved by the profile is at most $2/3$. If at least one of the MR strategies assigns a positive probability to the self-loop, then the two agents spend a significant proportion of time by "walking together", similarly as in D_1. This leads to the aforementioned gap of $1/9$.

The D_3 scenario requires deeper analysis. It is easy to show that the vector $(2/3, 2/3, 2/3, 0)$ is achievable by a HR profile where both agents "walk around the square" performing each self-loop exactly n times in the n-th cycle. Again, the agents are positioned so that they never meet in the same vertex. Furthermore, we show that for every ν achievable by a FR$_m$ profile, the L_∞ distance to $(2/3, 2/3, 2/3, 0)$ is at least $f(m)$ where $f : \mathbb{N}_+ {\to} (0,1]$ is a suitable function.

Our next result says that solving the steady-state objectives for $k \geq 2$ agents is computationally hard, even for graphs.

Theorem 2. *Let D be a graph, Col a coloring, and Obj a frequency vector. We have the following:*

(a) The problem whether there exists a HR profile for a given number of agents that achieves $\mu \geq \mathit{Obj}$ is PSPACE*-hard. This holds even under the assumption that if such a μ exists, it can be achieved by a FR$_m$ profile for a sufficiently large m.*

(b) The problem whether there exists a FMR profile for two agents achieving μ such that $\mu \geq \mathit{Obj}$ is NP*-hard, even if D is strongly connected and Col is the trivial coloring. This holds also for MR and FR$_m$ profiles (for every m).*

The following theorem reveals a severe obstacle for designing efficient steady-state synthesis algorithms.

Theorem 3. *Let D be a (strongly connected) graph, Col the trivial coloring, v a vertex of D, and π a MR profile such that π achieves some (unknown) frequency vector μ. The problem whether $\mu(v) = 1$ is* coNP*-hard.*

According to Theorem 3, MR strategy profiles are not only hard to construct, but they are also hard to *evaluate*.

Finally, we give upper complexity bounds on the steady-state synthesis problem.

Theorem 4. *Let $k \geq 1$ be a fixed constant. Given an MDP D, a coloring Col, a frequency vector Obj, and $m \geq 1$, the problem whether there exists an FR_m strategy profile for k agents achieving $\mu \geq$ Obj is in* PSPACE *(assuming the unary encoding of m).*

4 Steady-State Synthesis Algorithm

MDP Normal Form. We start by observing that in the context of steady-state synthesis, we can safely assume that the input MDP D takes the form $\bigcup_{q=1}^{m} D_q$ where $D_1, \ldots, D_m$ are strongly connected MDPs with pairwise disjoint sets of vertices (we say that D is in *normal form*).

To see this, consider (some) MDP D. A *maximal end component (MEC)* of D is a maximal strongly connected sub-MDP of D. The set $\{D_1, \ldots, D_m\}$ of all MECs of D is computable efficiently [Chatterjee and Henzinger, 2014], and $D_1, \ldots, D_m$ can be seen as strongly connected MDPs with pairwise disjoint sets of vertices. It can be shown that for an arbitrary (HR) strategy on D, almost all runs eventually enter and stay in some MEC. Since the finite prefix of a run executed before entering the MEC does not influence the achieved frequency vector, we can safely assume that all runs are initiated in some D_q and never leave it. Thus, the steady-state synthesis problem for D can be reformulated as the steady-state synthesis problem for $\bigcup_{q=1}^{m} D_q$. Full details of this argument are somewhat subtle and they are presented in [Jonáš *et al.*, 2025].

Suppose that π is a strategy profile for an MDP $\bigcup_{q=1}^{m} D_q$ in normal form. To compute the frequency vector μ achieved by π, one is tempted to compute all frequency vectors μ_q achieved in D_q by the agents assigned to D_q, and then put $\mu = \sum_{q=1}^{m} \mu_q$. However, this simple method works only under the assumption that vertices in different MECs have different colors (we say that *Col* is *well-formed*). For example, this condition is satisfied when *Col* is the trivial coloring or when $m = 1$. If *Col* is not well-formed, we can still conclude $\mu \leq \sum_{q=1}^{m} \mu_q$, but the precise computation of μ may require *exponential* time, even for full MR profiles. For simplicity, we consider only well-formed colorings in the rest of this section (this condition is not too restrictive and it does not influence the hardness results of Section 3).

Let us also note that for MDPs in normal form and one agent, full MR profiles approximate MR profiles, which explains the remark in the second paragraph of Section 1.

Evaluating Full MR Profiles. Let $D = (V, E, p)$ be a strongly connected MDP, $\mathit{Col} : V \to \gamma$ a coloring, and $\pi = (\sigma_1, \ldots, \sigma_k)$ a full MR profile for D. We show how to compute the frequency vector achieved by π. Note that based on the previous discussion, this procedure can also be used to evaluate a full MR profile for an MDP $\bigcup_{q=1}^{m} D_q$ in normal form where the underlying coloring is well-formed (we compute the frequency vector μ_q for each D_q and the agents assigned to D_q, and then return the sum of all μ_q).

For every $i \in \{1, \ldots, k\}$, let $D^{\sigma_i} = (V, P_i, \alpha_i)$ be the Markov chain induced by D and $\sigma_i = (v_i, \kappa_i)$. That is, $P_i(v, u)$ is either $\kappa_i(v)(u)$ or $p(v)(u)$ depending on whether $v \in V_N$ or $v \in V_S$, and $\alpha_i(v_i) = 1$. Since D is strongly connected and every σ_i is full, each D^{σ_i} is irreducible and determines the *same* partition of V into $d \geq 1$ cyclic classes $V_0, \ldots, V_{d-1}$. We use $\mathbb{I}_i$ to denote the unique *invariant* distribution of D^{σ_i} satisfying $\mathbb{I}_i(v) = \sum_{u \in V} \mathbb{I}_i(u) \cdot P_i(u, v)$ for every $v \in V$. Furthermore, for every $c \in \gamma$, we use $\mathit{Col}^{-1}(c)$ to denote the pre-image of c (i.e., $\mathit{Col}^{-1}(c)$ is the set of all $v \in V$ such that $\mathit{Col}(v) = c$).

For simplicity, let at first consider the case when $d = 1$. Then, π achieves the frequency vector μ where

$$\mu(c) \quad = \quad 1 - \prod_{i=1}^{k} \left(1 - \sum_{v \in \mathit{Col}^{-1}(c)} \mathbb{I}_i(v)\right) \tag{1}$$

for every $c \in \gamma$. This follows directly from basic results about aperiodic irreducible Markov chains (see, e.g., [Chung, 1967]). More concretely, for every $u \in V$, we have that $\lim_{n\to\infty} P_i^n(v_i, u) = \mathbb{I}_i(u)$. Hence, $\sum_{v \in \mathit{Col}^{-1}(c)} \mathbb{I}_i(v)$ is the limit probability that agent i visits a c-colored vertex after n steps as $n \to \infty$. Since the agents are independent, the product on the right-hand side of (1) is the limit probability that *none* of the k agents visits a c-colored vertex. Consequently, the right-hand side of (1) is the limit probability (and hence also the frequency) that *at least one agent* visits a c-colored vertex. Note that (1) is independent of the initial vertices of the strategies $\sigma_1, \ldots, \sigma_k$.

If $d > 1$, then the frequency vector μ depends on the initial positioning of the agents into the cyclic classes, and the above reasoning must be applied to the d-step matrices P_i^d. For every $i \in \{1, \ldots, k\}$ and $j \in \{0, \ldots, d-1\}$, let $V(i, j)$ be the cyclic class visited by agent i after traversing precisely j edges from the initial vertex v_i (in particular, $V(i, 0)$ is the cyclic class containing the initial vertex v_i). Furthermore, for every $c \in \gamma$, let $V^c(i, j) = V(i, j) \cap \mathit{Col}^{-1}(c)$. Equation (1) is generalized into the following:

$$\mu(c) = \frac{1}{d} \sum_{j=0}^{d-1} \left(1 - \prod_{i=1}^{k} \left(1 - d \cdot \sum_{v \in V^c(i,j)} \mathbb{I}_i(v)\right)\right). \tag{2}$$

Note that (2) is computable in polynomial time.

The Algorithm. For a given MDP $D = \bigcup_{q=1}^{m} D_q$ in normal form, a well-formed coloring *Col*, $k \geq 1$, and a frequency vector *Obj*, we wish to compute a full MR profile π for k agents achieving a frequency vector μ such that $\mathit{Dist}(\mu, \mathit{Obj})$ is *minimized*, where

$$\mathit{Dist}(\mu, \mathit{Obj}) = \sum_{c \in \gamma} \max\{0, \mathit{Obj}(c) - \mu(c)\}. \tag{3}$$

A natural idea is to construct a mathematical program minimizing $\mathit{Dist}(\mu, \mathit{Obj})$. Each full MR strategy σ_i in the desired profile π can be encoded by variables representing the edge probabilities, and the invariant distribution $\mathbb{I}_i$ can then be encoded by simple linear constraints. However, computing the frequency vector μ involves the *non-linear* right-hand side of (2), which makes the resulting program non-linear.

To overcome this difficulty, Algorithm 1 constructs the profile π *incrementally* by adding the agents one-by-one. Suppose that we already constructed a profile for ℓ agents, and

Algorithm 1 Incremental Steady-State Synthesis Algorithm

```
Inputs:
    MDP $D = \bigcup_{q=1}^{m} D_q$ in normal form
    Well-formed coloring $Col : V \to \gamma$
    Objective $Obj : \gamma \to [0, 1]$
    Number of agents $k \geq 1$
Outputs:
    A full MR strategy profile $\pi$ for $D$ and $k$ agents
Initialize:
    $\pi \leftarrow \emptyset$
for all $i \in \{1, \ldots, k\}$ do
    BestDistance $\leftarrow \infty$
    for all $q \in \{1, \ldots, m\}$ do
        for all cyclic classes $C \in \{C_0, \ldots, C_{d_q - 1}\}$ of $D_q$ do
            $\sigma \leftarrow$ StrategyOfLP$(Obj, \pi, C, D_q)$
            $\nu \leftarrow$ Evaluate$(\pi{+}\sigma, D)$
            if $Dist(\nu, Obj) <$ BestDistance then
                BestDistance $\leftarrow Dist(\nu, Obj)$
                BestStrategy $\leftarrow \sigma$
    $\pi \leftarrow \pi +$ BestStrategy
return $\pi$
```

we wish to compute a suitable full MR strategy $\sigma_{\ell+1} = (v_{\ell+1}, \kappa_{\ell+1})$ for another agent. The algorithm examines all possible allocations for $v_{\ell+1}$, i.e., all cyclic classes C in all D_q. For given C and D_q, the procedure StrategyOfLP constructs the linear program of Fig. 4 and returns the full MR strategy $\sigma = (v_0, \kappa)$, where $v_0 \in C$ and $\kappa(u)(v)$ is the *normalized* value of $x_{u,v}$ attained by solving the program. Note that the $x_{u,v}$ variable in the LP represents the *frequency* of the edge $(u, v) \in E^q$, not the probability of the edge. The key observation is that since the strategies $\sigma_1, \ldots, \sigma_\ell$ are *fixed*, the right-hand side of (2) becomes *linear*. In Fig. 4, we use $\mathcal{X}_j^c$ to denote the *constant* value of the product $\prod_{i=1}^{\ell}(1 - d_c \cdot \sum_{v \in V^c(i,j)} \mathbb{I}_i(v))$, where d_c denotes the period of the MEC containing the vertices of color c (if there is no such vertex, we put $d_c = 1$), $V^c(C, j)$ denotes the set of all c-colored vertices in the cyclic class of D_q visited after traversing precisely j edges from a vertex of C.

After computing the strategy σ, Algorithm 1 proceeds by evaluating the profile $\pi + \sigma$ obtained by appending σ to π. If the frequency vector achieved by this profile is better than the frequency vectors achieved for all σ's computed so far, the current σ is set as a new candidate for $\sigma_{\ell+1}$. Algorithm 1 terminates after constructing a profile for all k agents.

5 Experimental Evaluation

The main goal of our experiments is to evaluate the quality of the strategy profiles constructed by Algorithm 1. We also assess the efficiency of Algorithm 1. Additional analyses of the results and some additional plots are in [Jonáš *et al.*, 2025]. The reproduction package for the evaluation is available from Zenodo [Jonáš *et al.*, 2025].

Benchmarks. For simplicity, we perform our experiments on graphs. This does not affect efficiency since stochastic vertices do not add any extra computational costs. Moreover, it does not affect the quality comparison between the baseline and the incremental synthesis procedure of Algorithm 1.

min $Dist(\mu, Obj)$

subject to

$$x_{u,v} \in (0, 1], \qquad (u, v) \in E^q,$$

$$\sum_{(u,v) \in E^q} x_{u,v} = 1,$$

$$\sum_{(v,u) \in Out(v)} x_{v,u} = \sum_{(u,v) \in In(v)} x_{u,v}, \qquad v \in V^q,$$

$$x_{v,w} = p^q(v)(w) \cdot \sum_{(u,v) \in In(v)} x_{u,v}, \qquad v \in V_S^q, (v, w) \in Out(v),$$

$$\mu(c) = \frac{1}{d_c} \cdot \sum_{j=0}^{d_c - 1} \left(1 - \mathcal{X}_j^c \cdot \left(1 - d_c \cdot \sum_{v \in V^c(C,j)} \sum_{(u,v) \in In(v)} x_{u,v}\right)\right)$$

Figure 4: The linear program for $Obj, \pi, C, D_q = (V^q, E^q, p^q)$.

To avoid any systematic bias, we randomly generated two families of strongly connected input graphs: aperiodic and periodic. For aperiodic graphs, we randomly generated graphs with up to 400 vertices and an edge between each pair of vertices with probability 0.01. For periodic graphs, we randomly generated structures of Fig. 2 with $d \in \{5, 10, 15, 20\}$ cyclic classes, at most 20 vertices in each cyclic class, and an edge between each two vertices from neighboring cyclic classes with probability 0.6. We considered only graphs that are strongly connected. For each graph, we randomly generated 5 objectives with at most 30 colors and target values $Obj(c)$ from $\{0, 0.1, 0.2, \ldots, 0.9\}$ and randomly assigned a color to each vertex. In this way, we obtained 2000 aperiodic and 1600 periodic benchmarks (i.e., combinations of a graph and an objective) with at most 400 vertices.

Baseline. Since the steady-state synthesis problem for $k{\geq}2$ agents is computationally hard (see Section 3), we cannot compare the quality of profiles constructed by Algorithm 1 against the optimal solutions as there is no feasible way to determine them. However, we can still compare Algorithm 1 with a *straightforward* synthesis procedure based on sharing the strategy computed for *one* agent. In some cases, this simple method even leads to optimal solutions. For example, if k agents move along a directed ring consisting of $n \geq k$ vertices, they can achieve the frequency vector $(k/n, \ldots, k/n)$ by sharing the same strategy ("walk along the ring") so that each agent starts in a different vertex.

The baseline synthesis procedure works as follows. We start by computing a full MR strategy σ for a single agent, optimizing a suitably defined value. Then, k agents that use the strategy σ are allocated to the cyclic classes $C_0, \ldots, C_{d-1}$ by Round Robin assignment. More specifically, the strategy σ is constructed by a LP *maximizing* $AltDist(Obj, \nu)$ where $AltDist(Obj, \nu) = \min_{c \in Col} \frac{\nu(c)}{Obj(c)}$ and $\nu(c) = \sum_{v \in Col^{-1}(c)} \mathbb{I}(v)$. Intuitively, the goal is to cover all the colors in proportion to their target values. We maximize $AltDist$ instead of minimizing $Dist$ because in our preliminary experiments, the performance of the algorithm based on minimizing $Dist$ was significantly worse. More concretely, when using $Dist$, the strategy σ tended to focus on a subset of colors, which also caused the resulting strategy profile to focus

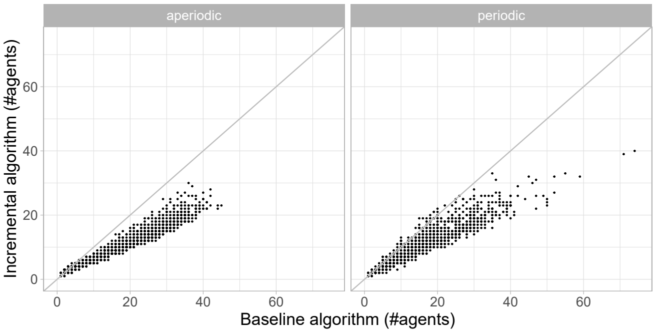

Figure 5: Numbers of agents sufficient to satisfy the objective using each of the algorithms (lower is better). Each point (x, y) is a benchmark for which the objective is satisfied by x agents by the baseline algorithm, and y agents by Algorithm 1. Divided by the type of the graph (aperiodic/periodic).

only on some colors.

Implementation and experimental setup. We implemented both algorithms in a simple open source Python tool that uses Gurobi [Gurobi Optimization, LLC, 2024] to solve the linear programming problems. The tool is available from GitLab[3]. We executed both algorithms on each benchmark with timeout 120 seconds of wall time on a Linux computer with AMD Ryzen 7 PRO 5750G CPU and 32 GB of RAM.

Quality of strategies. For each benchmark, we compared the two algorithms with respect to the number of agents that is necessary to satisfy the objective. The results are presented in Fig. 5. Algorithm 1 often requires significantly fewer agents to satisfy the objective. Numerically, Algorithm 1 required fewer agents on 3163 of the benchmarks and more on 85 benchmarks. It also required only 10.40 agents on average, compared to 16.65 agents needed by the baseline. The improvement occurs both for periodic and aperiodic input graphs, which shows that the main benefit is not due to the smarter initial assignment of agents to cyclic classes but because of the core approach of incremental addition of agents.

We also investigated the distances achieved by the strategy profiles for fewer agents than necessary to satisfy the objective. This is presented in Fig. 6 on a randomly selected subset of benchmarks. The plot again shows that Algorithm 1 satisfies the objective with significantly fewer agents. More importantly, it shows that for most benchmarks, the profiles synthesized by Algorithm 1 are better for all numbers of agents smaller than the ones needed by any of the algorithms.

The experiments show that Algorithm 1 can satisfy objectives with significantly fewer agents, if enough agents are available. Additionally, when the number of available agents is insufficient, Algorithm 1 in most cases achieves a smaller distance to satisfying the objectives than the baseline.

Efficiency. We also measured the runtime for both algorithms. The measured wall times are summarized in Table 1. The table shows that the mean wall time of the baseline algorithm is significantly better than the one of Algorithm 1. This is not surprising as the main bottleneck of both algorithms is LP solving and the baseline algorithm requires only one call of the LP solver per benchmark, whereas the incremental algorithm requires $k \cdot d$ calls, where k is the number of agents and d is the period of the input graph. Nevertheless, all executions of our algorithm finished within 5.08 seconds on aperiodic benchmarks and within 28.88 seconds on periodic benchmarks, which is not prohibitive in practice.

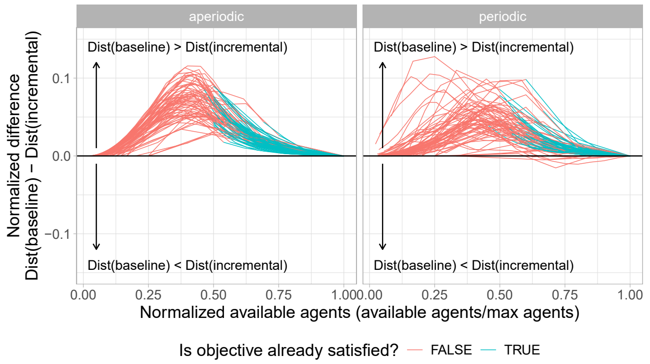

Figure 6: Comparison of distances achieved by the two algorithms on a randomly selected subset of 150 benchmarks. Each line represents a benchmark. The y-axis shows the difference of the normalized distances $\frac{Dist(\pi_{\text{baseline}}, Obj)}{|\gamma|} - \frac{Dist(\pi_{\text{incremental}}, Obj)}{|\gamma|}$ between the obtained profiles π_{baseline} and $\pi_{\text{incremental}}$ for k agents. The x-axis shows the number k of agents between 0 and the number sufficient for both of the algorithms, normalized between $[0, 1]$. The line is colored blue if any of the algorithms has already satisfied the objective. Divided by the type of the graph (aperiodic/periodic).

		Wall time (s)		
Benchmarks	Algorithm	mean	median	max
Aperiodic	Baseline	0.10	0.09	0.25
	Incremental	0.97	0.71	5.08
Periodic	Baseline	0.03	0.03	0.11
	Incremental	3.57	1.88	28.87

Table 1: Wall times of both algorithms, divided by the type of the graph (aperiodic/periodic).

Discussion. Even though Algorithm 1 is less efficient than the naive baseline algorithm, the performance is not prohibitive in practice and it achieves the objectives with significantly fewer agents. In our view, this is more important metric; few additional seconds to synthesize the strategy profile is cheap, whereas each extra agent can be far more costly.

Conclusions

We have extended steady-state synthesis to multiagent setting and presented an efficient synthesis algorithm. The main challenges for future work include tackling the synthesis of (non-full) MR profiles and extending the whole approach to more general classes of infinite-horizon objectives.

[3] https://gitlab.fi.muni.cz/formela/multi-agent-steady-state-synthesis

Acknowledgments

The work received funding from the European Union's Horizon Europe program under the Grant Agreement No. 101087529.

References

[Akshay *et al.*, 2013] S. Akshay, N. Bertrand, S. Haddad, and L. Hélouët. The steady-state control; problem for Markov decision processes. In *Proceedings of 10th Int. Conf. on Quantitative Evaluation of Systems (QEST'13)*, volume 8054 of *Lecture Notes in Computer Science*, pages 290–304. Springer, 2013.

[Atia *et al.*, 2020] G. Atia, A. Beckus, I. Alkhouri, and A. Velasquez. Steady-state policy synthesis in multichain Markov decision processes. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2020)*, pages 4069–4075, 2020.

[Brázdil *et al.*, 2014] T. Brázdil, V. Brožek, K. Chatterjee, V. Forejt, and A. Kučera. Markov decision processes with multiple long-run average objectives. *Logical Methods in Computer Science*, 10(1):1–29, 2014.

[Brázdil *et al.*, 2015] T. Brázdil, K. Chatterjee, V. Forejt, and A. Kučera. MultiGain: A controller synthesis tool for MDPs with multiple mean-payoff objectives. In *Proceedings of TACAS 2015*, volume 9035 of *Lecture Notes in Computer Science*, pages 181–187. Springer, 2015.

[Canny, 1988] J. Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of STOC'88*, pages 460–467. ACM Press, 1988.

[Chatterjee and Henzinger, 2014] K. Chatterjee and M. Henzinger. Efficient and dynamic algorithms for alternating Büchi games and maximal end-component decomposition. *Journal of the Association for Computing Machinery*, 61(1):1–40, 2014.

[Chung, 1967] K.L. Chung. *Markov Chains with Stationary Transition Probabilities*. Springer, 1967.

[Gurobi Optimization, LLC, 2024] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024.

[Jonáš *et al.*, 2025] M. Jonáš, A. Kučera, V. Kůr, and J. Mačák. Steady-state strategy synthesis for swarms of autonomous agents. *arXiv*, 2505.12406 [cs.MA], 2025.

[Jonáš *et al.*, 2025] M. Jonáš, A. Kučera, V. Kůr, and J. Mačák. Reproduction Package for IJCAI 2025 Paper "Steady-State Strategy Synthesis for Swarms of Autonomous Agents" , May 2025.

[Křetínský, 2021] J. Křetínský. LTL-constrained steady-state policy synthesis. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2021)*, pages 4104–4111, 2021.

[Kwiatkowska *et al.*, 2011] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *Proceedings of CAV 2011*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011.

[Puterman, 1994] M.L. Puterman. *Markov Decision Processes*. Wiley, 1994.

[Rosenthal, 2006] J.S. Rosenthal. *A first look at rigorous probability theory*. World Scientific Publishing, 2006.

[Velasquez *et al.*, 2023] A. Velasquez, I. Alkhouri, K. Subramani, P. Wojciechowski, and G. Atia. Optimal deterministic controller synthesis from steady-state distributions. *Journal of Automated Reasoning*, 67(7), 2023.

[Velasquez *et al.*, 2024] A. Velasquez, I. Alkhouri, A. Beckus, A. Trivedi, and G. Atia. Controller synthesis for linear temporal logic and steady-state specifications. *Autonomous Agents and Multi-Agent Systems*, 38(17), 2024.

[Velasquez, 2019] A. Velasquez. Steady-state policy synthesis for verifiable control. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 5653–5661, 2019.