Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Extreme state aggregation beyond Markov decision processes ☆

Marcus Hutter

Research School of Computer Science, Australian National University, Canberra, ACT, 0200, Australia

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
State aggregation
Reinforcement learning
Non-MDP

ABSTRACT

We consider a Reinforcement Learning setup where an agent interacts with an environment in observation–reward–action cycles without any (esp. MDP) assumptions on the environment. State aggregation and more generally feature reinforcement learning is concerned with mapping histories/raw-states to reduced/aggregated states. The idea behind both is that the resulting reduced process (approximately) forms a small stationary finite-state MDP, which can then be efficiently solved or learnt. We considerably generalize existing aggregation results by showing that even if the reduced process is not an MDP, the (q-)value functions and (optimal) policies of an associated MDP with same state-space size solve the original problem, as long as the solution can approximately be represented as a function of the reduced states. This implies an upper bound on the required state space size that holds uniformly for all RL problems. It may also explain why RL algorithms designed for MDPs sometimes perform well beyond MDPs.

1. Introduction

In *Reinforcement Learning* (RL) [25], an *agent* Π takes actions in some *environment* P and observes its consequences and is rewarded for them. A well-understood and efficiently solvable [23] and efficiently learnable [27,14] case is where the environment is (modeled as) a finite-state stationary *Markov Decision Process* (MDP). Unfortunately most interesting real-world problems P are neither finite-state, nor stationary, nor Markov. One way of dealing with this mismatch is to somehow transform the real-world problem into a small MDP: *Feature Reinforcement Learning* (FRL) [10] and U-tree [16] deal with the case of arbitrary unknown environments, while state aggregation assumes the environment is a large known stationary MDP [6,1,5]. The former maps histories into states (Section 2), the latter groups raw states into aggregated states.

Here we follow the FRL approach and terminology, since it is arguably most general: It subsumes the cases where the original process P is an MDP, a k-order MDP, a POMDP, and others (Section 3). Thinking in terms of histories also naturally stifles any temptation of a naive frequency estimate of P (no history ever repeats). Finally we find the history vs state terminologically somewhat neater than raw state vs aggregated state.

More importantly, we consider maps ϕ from histories to states for which the reduced process P_ϕ is not (even approximately) an MDP (Section 4). At first this seems to defeat the original purpose, namely of reducing P to a well-understood and efficiently solvable problem class, namely small MDPs. The main novel contribution of this paper is to show that there

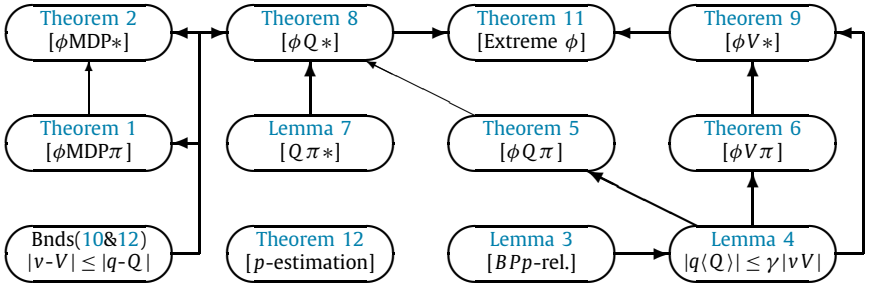
☆ A short version appeared in the proceedings of the ALT 2014 conference [11].

E-mail address: marcus.hutter@anu.edu.au.
URL: http://www.hutter1.net/.

http://dx.doi.org/10.1016/j.tcs.2016.07.032
0304-3975/

is still an associated finite-state stationary MDP p whose solution (approximately) solves the original problem P, as long as the solution can still be represented (Section 5). Indeed, we provide an upper bound on the required state space size that holds uniformly for all P (Section 6). While these are interesting theoretical insights, it is a-priori not clear whether they could by utilized to design (better) RL algorithms. We also show how to learn p from experience (Section 7), and sketch an overall learning algorithm and regret/PAC analysis based on our main theorems (Section 8). We briefly discuss how to relax one of the conditions in our main theorems by permuting actions (Section 9). We conclude with an outlook on future work and open problems (Section 10). A list of notation can be found in Appendix A.

The diagram below depicts the dependencies between our results:



2. Feature Markov decision processes (ΦMDP)

This section formally describes the setup of [10]. It consists of the agent–environment framework and maps ϕ from observation–reward–action histories to MDP states. This arrangement is called "Feature MDP" or short ΦMDP. We use upper-case letters P, Q, V, and Π for the Probability, (Q-)Value, and Policy of the original (agent–environment interactive) Process, and lower-case letters p, q, v, and π for the probability, (q-)value, and policy of the (reduced/aggregated) MDP.

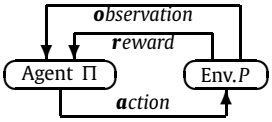
Agent–environment setup [10]. We start with the standard agent–environment setup [24] in which an agent Π interacts with an environment P. The agent can choose from actions $a \in \mathcal{A}$ and the environment provides observations $o \in \mathcal{O}$ and real-valued rewards $r \in \mathcal{R} \subseteq [0;1]$ to the agent. This happens in cycles $t = 1, 2, 3, ...$: At time t, after observing o_t and receiving reward r_t, the agent takes action a_t based on history

$$h_t \;:=\; o_1 r_1 a_1 ... o_{t-1} r_{t-1} a_{t-1} o_t r_t \;\in\; \mathcal{H}_t := (\mathcal{O}\times\mathcal{R}\times\mathcal{A})^{t-1}\times\mathcal{O}\times\mathcal{R}$$

Then the next cycle $t+1$ starts. The agent's objective is to maximize its long-term reward. To avoid integrals and densities, we assume spaces $\mathcal{O}$ and $\mathcal{R}$ are finite. They may be huge, so this is not really restrictive. Indeed, the ΦMDP framework has been specifically developed for huge observation spaces. Generalization to continuous $\mathcal{O}$ and $\mathcal{R}$ is routine [8]. Furthermore we assume that $\mathcal{A}$ is finite and smallish, which is restrictive. Potential extensions to continuous $\mathcal{A}$ are discussed in Section 10.

The agent and environment may be viewed as a pair of interlocking functions of the history $\mathcal{H} := (\mathcal{O}\times\mathcal{R}\times\mathcal{A})^*\times\mathcal{O}\times\mathcal{R} \cup \{\epsilon\}$, where ϵ is the empty history.

$$\text{Env. } P : \mathcal{H}\times\mathcal{A} \rightsquigarrow \mathcal{O}\times\mathcal{R}, \qquad P(o_{t+1}r_{t+1}|h_t a_t),$$
$$\text{Agent } \Pi : \mathcal{H} \rightsquigarrow \mathcal{A}, \qquad \Pi(a_t|h_t) \quad \text{or} \quad a_t = \Pi(h_t),$$



where $\rightsquigarrow$ indicates that mappings $\rightarrow$ are in general stochastic. We make no (stationarity or Markov or other) assumption on environment P. For most parts, environment P is assumed to be fixed, so dependencies on P will be suppressed. For convenience and since optimal policies can be chosen to be deterministic (see Eq. (4)), we consider deterministic policies $a_t = \Pi(h_t)$ only.

Value functions, optimal Policies, and history Bellman equations. We measure the performance of a policy Π in terms of the P-expected γ-discounted reward sum ($0 \le \gamma < 1$), called (Q-)Value of Policy Π at history h_t (and action a_t)

$$V^\Pi(h_t) := \mathbb{E}^\Pi[R_t|h_t], \qquad Q^\Pi(h_t,a_t) := \mathbb{E}^\Pi[R_t|h_t a_t], \qquad R_t := \sum_{\tau=t+1}^{\infty}\gamma^{\tau-t}r_\tau$$

The optimal Policy and (Q-)Value functions are

$$\begin{aligned} V^*(h_t) &:= \max_\Pi V^\Pi(h_t) \\ Q^*(h_t,a_t) &:= \max_\Pi Q^\Pi(h_t,a_t) \\ \Pi^* &:\in \arg\max_\Pi V^\Pi(\epsilon) \end{aligned} \tag{1}$$

Please cite this article in press as: M. Hutter, Extreme state aggregation beyond Markov decision processes, Theoret. Comput. Sci. (2016), http://dx.doi.org/10.1016/j.tcs.2016.07.032