NEURAL INFORMATION PROCESSING SYSTEMS

WORKSHOP ON SELF-SUPERVISED LEARNING: THEORY AND PRACTICE

Self-guided Approximate Linear Programs

Parshan Pakiman[1] | Selva Nadarajah[1] | Negar Soheili[1] | Qihang Lin[2]

[1] University of Illinois at Chicago, College of Business | [2] University of Iowa, Tippie College of Business

Motivation

MDPs

- Markov Decision Processes (MDPs) provide a versatile model of sequential decision-making problems.
- MDPs are extensively used to model various applications arising in autonomous driving, robotics, queuing, marketing, dynamic pricing, etc.
- Solving large-scale MDPs requires tackling the *curses of dimensionalities.*

RL and ADP

- Reinforcement Learning (RL) and Approximate Dynamic Programming (ADP) include a vast collection of techniques to solve challenging MDPs.
- Solving high-dimensional MDPs with most ADP methods require performing value function approximation (VFA).
- *Selecting features* that define VFA typically requires *domain knowledge* and *heuristic hand-engineering.*

ALPs

- Approximate Linear Programs (ALPs) compute VFAs via a linear program with an infinite number of constraints.
- ALPs have been successfully used to tackle challenging applications in multiple application domains.
- Research on reducing ALP constraints is extensive, while there are few works tackling *feature selection* in ALPs, an *implementation hurdle.*

Novelty and Contribution

Random Kitchen Sinks (RKSs) + "Self-guiding" Constrains → Replacing Feature Selection in ALPs with the Sampling of Randomized Features

Application-agnostic Policies: Developing an ALP framework for computing *application-agnostic* policies, VFAs, and bounds.

RKSs in Constrained RL: RKSs are used in data mining and unconstrained RL (i.e., value iteration) while we use them in ALPs, *constrained RL* models.

Simplifying Implementation: Simplifying the implementation of ALPs by replacing the *feature selection* hurdle with the *sampling of randomized features.*

Optimality Gap: Constructing a convergent sequence of *optimality gaps* to assess quality of ALP policies.

Self-guiding Constraints: Developing *self-guiding constraints* to deliver a sequence of policies with monotonically improving worst-case performance.

Numerical Advantages: Our *application-agnostic* policies compete with state-of-the-art *policies tailored* to two challenging applications in inventory control.

ALP through RKSs and Self-guiding Constrains

Exact Linear Program

$$\max_{V'\in\mathcal{C}} \ \mathbb{E}_\nu\left[V'(s)\right] \quad \text{s.t.} \quad V'(s) - \gamma\mathbb{E}[V'(s') \mid s,a] \le c(s,a), \quad \forall (s,a)$$

Dense approximation via RKSs associated with *universal kernels*

Feature-based Exact Linear Program

$$\sup_{b_0,\boldsymbol{b}} \ b_0 + \langle \boldsymbol{b}, \mathbb{E}_\nu[\varphi(s)]\rangle$$
$$\text{s.t.} \quad (1-\gamma)b_0 + \langle \boldsymbol{b}, \varphi(s) - \gamma\mathbb{E}[\varphi(s') \mid s,a]\rangle \le c(s,a), \quad \forall (s,a)$$
$$\|\boldsymbol{b}\|_{\infty,\rho} \le C.$$

Easy-to-compute *sample average approximation* that provides a *near-optimal VFA* for a *finite* number of samples

Feature-based Approximate Linear Program (FALP)

$$\max_{\boldsymbol{\beta}} \ \beta_0 + \sum_{i=1}^{N}\beta_i\mathbb{E}_\nu\left[\varphi(s;\theta_i)\right]$$
$$\text{s.t.} \quad (1-\gamma)\beta_0 + \sum_{i=1}^{N}\beta_i\left(\varphi(s;\theta_i) - \gamma\mathbb{E}\left[\varphi(s';\theta_i) \mid s,a\right]\right) \le c(s,a), \quad \forall (s,a)$$

Self-guiding constraints ensure:

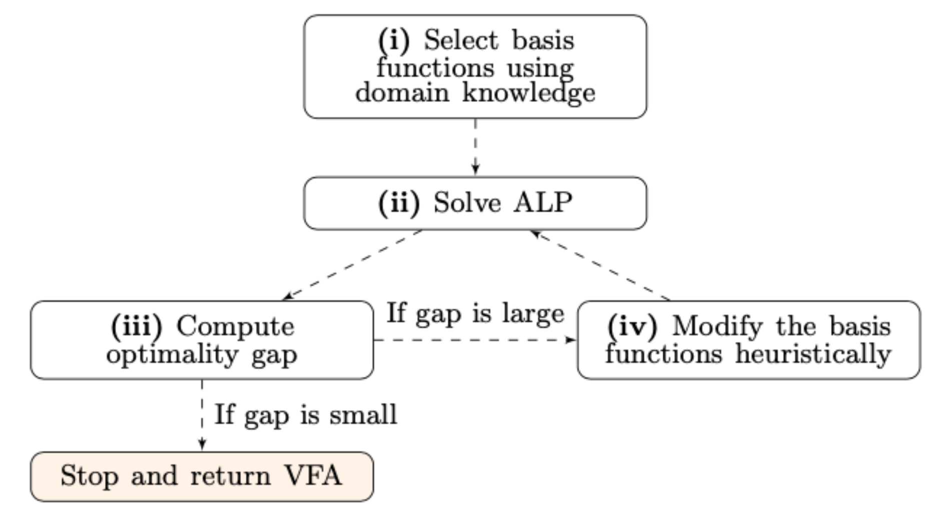
- State-wise improvement of the VFA sequence
- Monotonic improvement of ALP policies worst-case cost

Feature-based Approximate Linear Program (FALP)

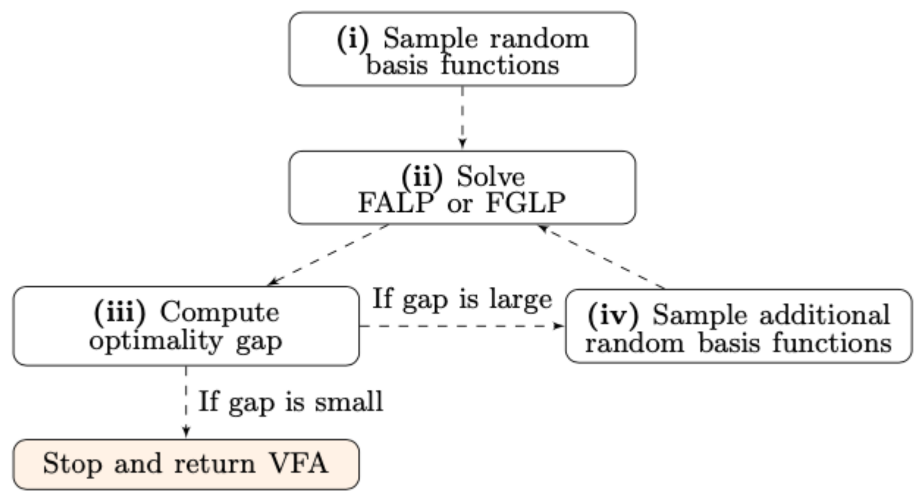
$$\max_{\boldsymbol{\beta}} \ \beta_0 + \sum_{i=1}\beta_i\mathbb{E}_\nu\left[\varphi(s;\theta_i)\right]$$
$$\text{s.t.} \quad (1-\gamma)\beta_0 + \sum_{i=1}^{N}\beta_i\left(\varphi(s;\theta_i) - \gamma\mathbb{E}\left[\varphi(s';\theta_i)\mid s,a\right]\right) \le c(s,a), \quad \forall (s,a)$$
$$\beta_0 + \sum_{i=1}^{N}\beta_i\varphi(s;\theta_i) \ge V\left(s;\boldsymbol{\beta}^{\text{FG}}_{\text{N-B}}\right), \quad \forall s$$

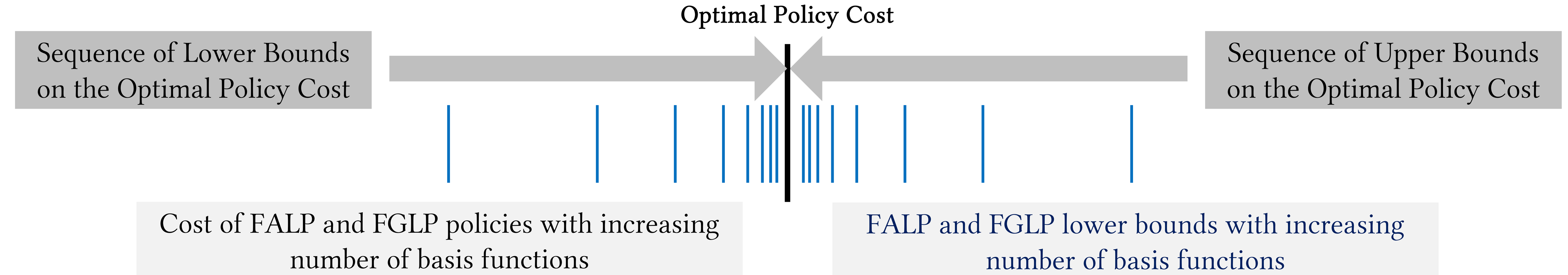
Self-guided ALPs in Practice

Standard Implementation



Proposal Implementation





Theoretical Guarantees

What Do FALP and FGLP Guarantee?

VFA Finite Sampling Bound: For a finite number of samples, the sequence of VFAs from both FALP and FGLP models converges to the true value function with a high probability

Convergence of Optimality Gap Sequence: For a finite number of samples, the sequence of upper bounds and lower bounds in our framework converge to a neighborhood of the optimal policy cost with a high probability.

What Do Self-guiding Constraints Add?

Monotonic Improvement of Lower Bounds: The sequence of lower bounds from FGLP is non-decreasing

Monotonic Improvement of Worst-case Policy Cost: A worst-case cost of greedy policies from FGLP VFAs monotonically decreases.

Numerical Assessments

Value in FALP and FGLP compared to Benchmarks

FGLP Near-optimal Policies: On our instances of perishable inventory control, the FGLP policy *optimality gap* is at most **5%** across these instances and *improves* by up to **8%** of the previously known gaps

Upper and Lower Bound Improvement: The cost of *FALP and FGLP policies* is up to **14%** *better* than the existing policies in the literature.

Lower bounds from FALP and FGLP *improve* existing lower bounds by up to **7%**.

Value in Self-guiding Constraints

FALP vs FGLP (Policy): The *worst-case performance* of the FGLP policies is *up to* **36%** *better* than the worst-case cost of the FALP policies.

FALP vs FGLP (runtime): Since *FGLP* self-guides its policies, it requires up to **10%** *fewer samples* to converge compared to *FALP,* and it thus has, on average, **7** *minutes shorter runtime*