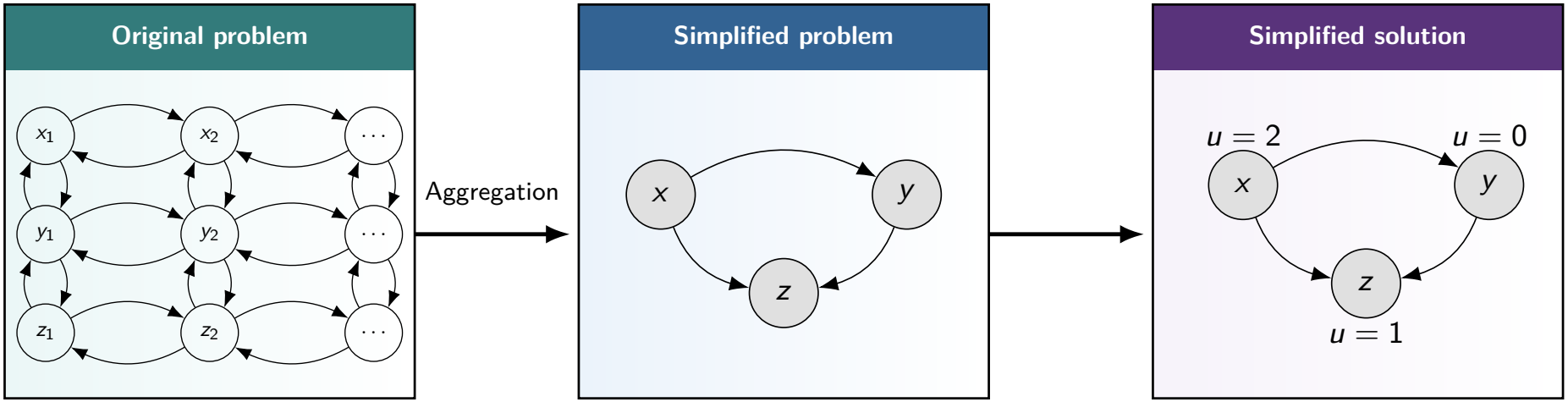
Aggregation Methods for Markov Decision Problems with Perfect and Imperfect State Information

Kim Hammar (khammar1@asu.edu), Yuchao Li (yuchaoli@asu.edu),
and Dimitri P. Bertsekas (dimitrib@mit.edu)

Based on
Section 3.6 of “A Course in Reinforcement Learning: 2nd Edition", by D.P. Bertsekas
as well as the upcoming work by Y. Li, K. Hammar, and D.P. Bertsekas

Aggregation is A Form of Problem Simplification

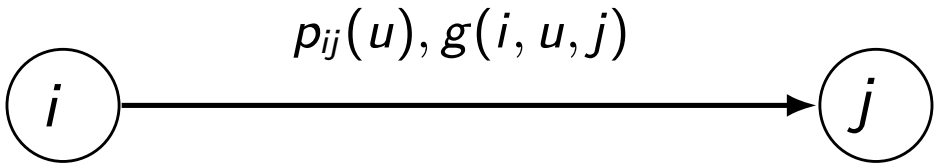


The Aggregation Methodology

1. Combine groups of similar states into aggregate states.
2. Formulate an aggregate dynamic programming problem based on these states.
3. Solve the aggregate problem using some computational method.
4. Use the solution to the aggregate problem to compute a cost function approximation for the original problem.

Outline

Recap of Markov Decision Problems (MDP)



- State space: $X = \{1, \ldots, n\}$, states are denoted by i, j.
- Control constraint set: $U(i)$.
- Probability of transitioning from state i to j given control u: $p_{ij}(u)$.
  - Equivalent formulation: $x_{k+1} = f(x_k, u_k, w_k)$.
- Cost of transitioning from state i to j given control u: $g(i,u,j)$.
- Cost-to-go from state i: $J(i)$.
- Discount factor: α.

Approximation in Value Space

$$\min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha \tilde{J}(j)\big)$$

First Step "Future"

- Optimal policy: μ^* can be computed via

$$\mu^*(i) \in \arg\min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\big(g(i,u,j) + \alpha J^*(j)\big), \quad i = 1, \ldots, n$$

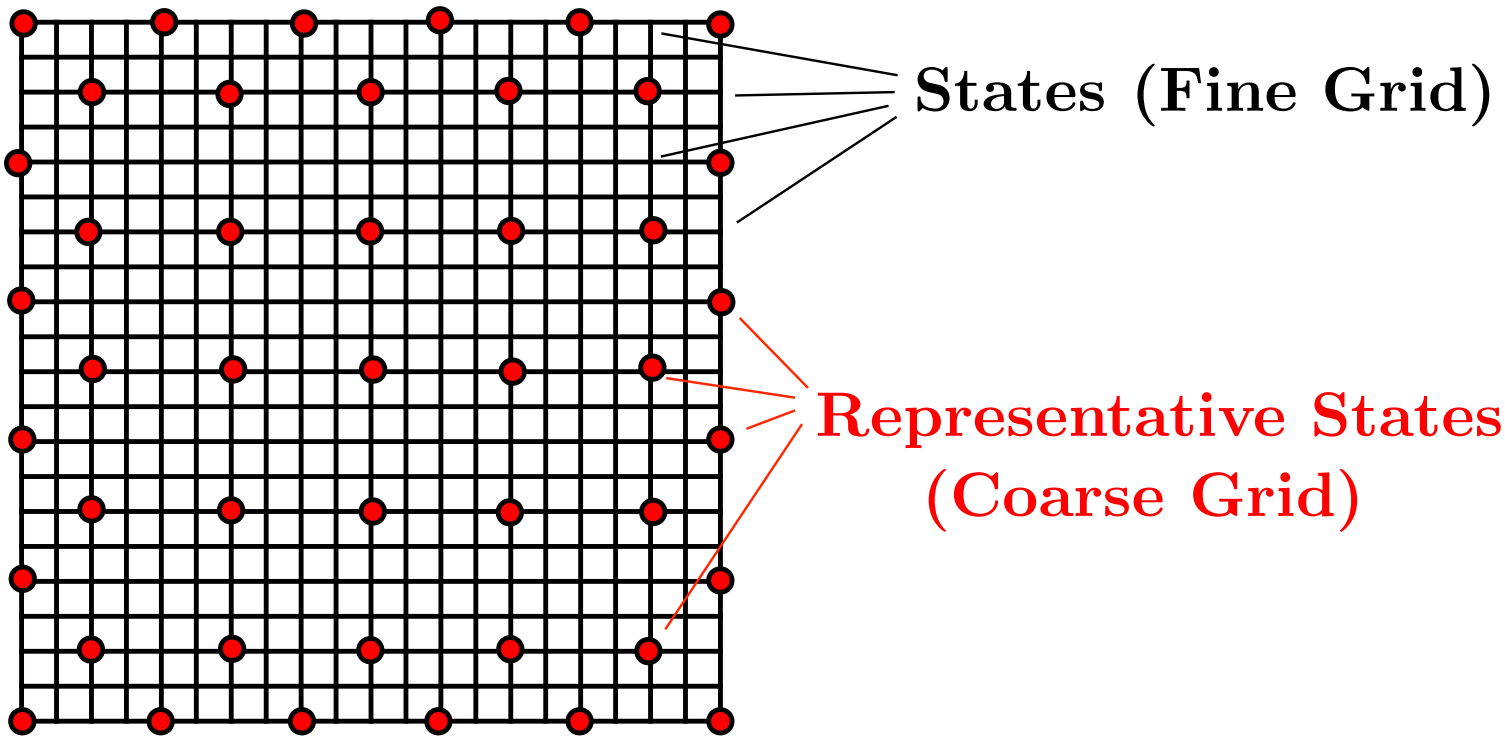
  where J^* is the optimal cost function satisfying Bellman's equation
- When computing J^* is intractable, aggregation computes some $\tilde{J}$ to approximate J^*
- A suboptimal policy $\tilde{\mu}$ can be computed online via

$$\tilde{\mu}(i_k) \in \arg\min_{u \in U(i_k)} \sum_{j=1}^{n} p_{i_k j}(u)\big(g(i_k,u,j) + \alpha \tilde{J}(j)\big)$$

  upon reaching state i_k at stage k

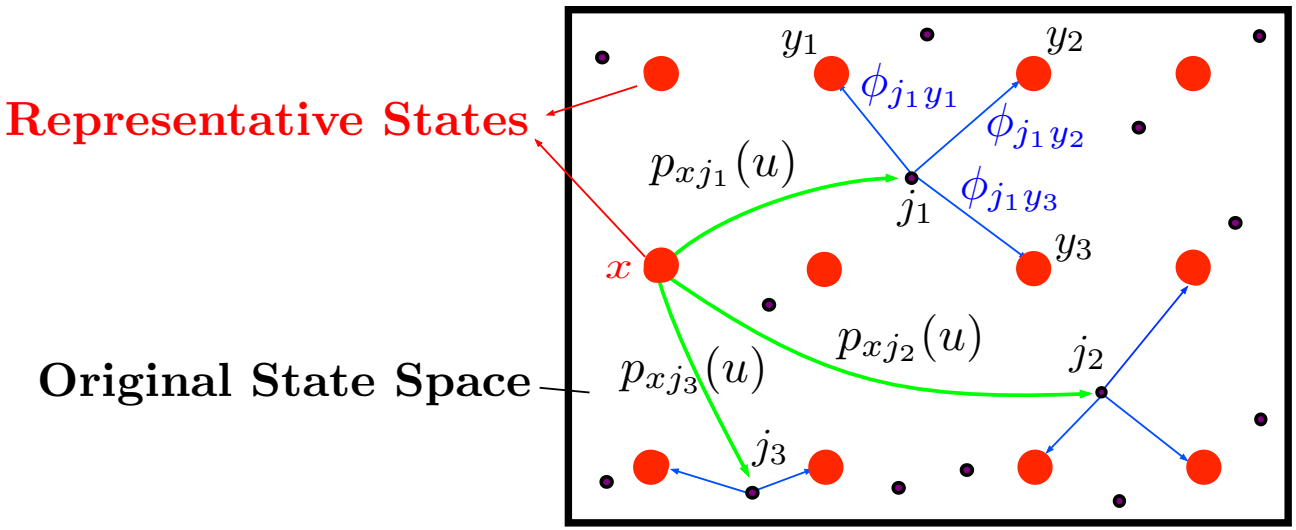
Representative States

- Introduce a **subset** $\mathcal{A}$ of the original states $1, \ldots, n$, called representative states.
- We use i, j to denote original states and x, y to denote representative states.



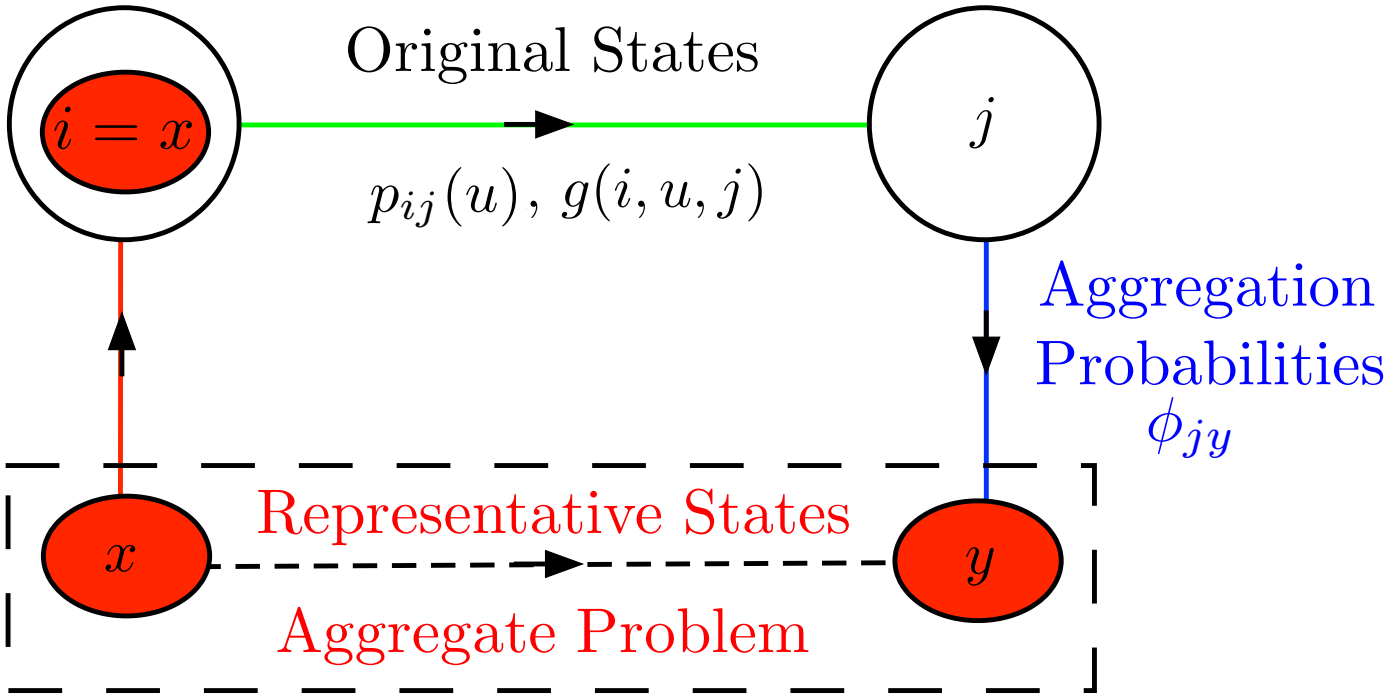
Aggregation Probabilities

- For each state i we define aggregation probabilities $\{\phi_{ix} \mid x \in \mathcal{A}\}$.
- Intuitively, ϕ_{ix} expresses similarity between states i and x, where $\phi_{xx} = 1$.



Aggregation Probabilities
ϕ_{jy}
Relate
Original States to
Representative States

Dynamics of the Aggregate System

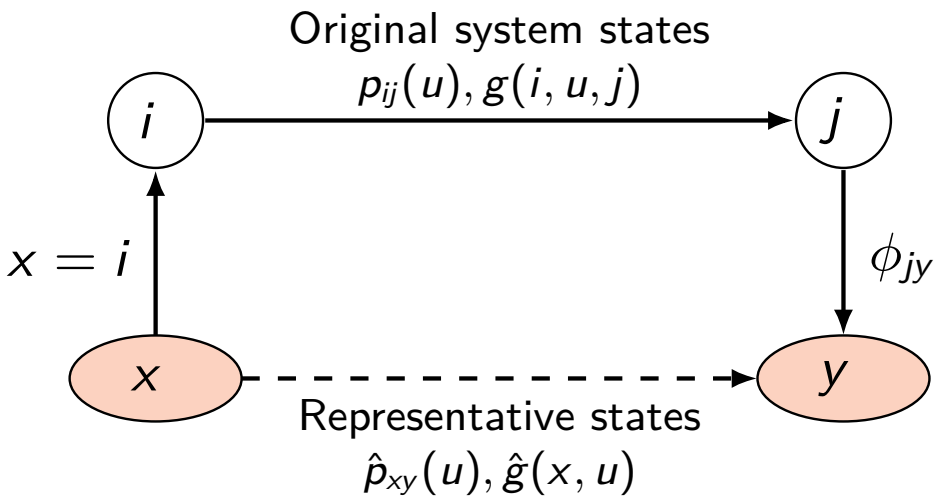


Formulating the Aggregate Dynamic Programming Problem

- State space: $\mathcal{A}$ (the set of representative states).
- Control constraint set: $U(i)$ (the original control constraint set).
- Transition probabilities and costs

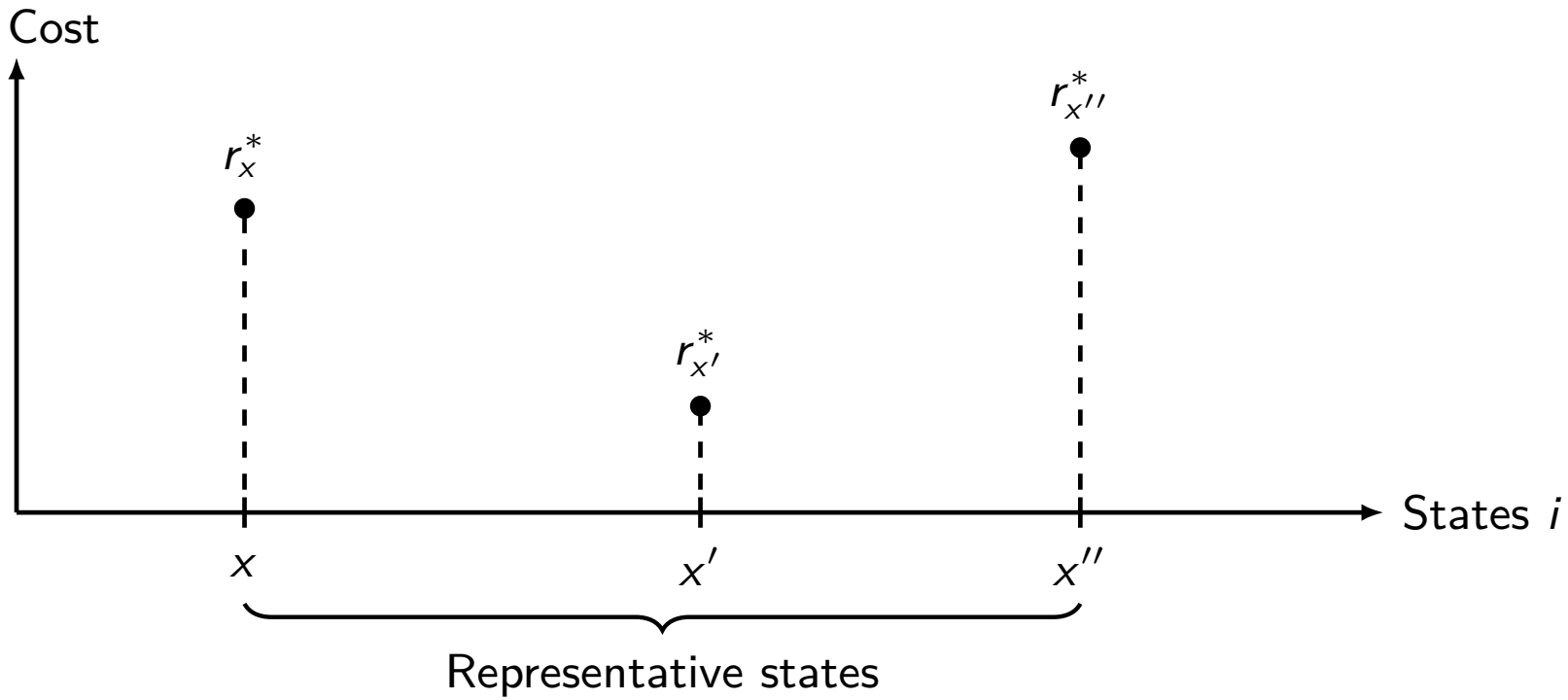
$$\hat{p}_{xy}(u) = \sum_{i=1}^{n} p_{xi}(u)\phi_{iy}, \qquad \text{for all representative states } (x, y) \text{ and controls } u,$$

$$\hat{g}(x, u) = \sum_{i=1}^{n} p_{xi}(u)g(x, u, i), \qquad \text{for all representative states } x \text{ and controls } u.$$



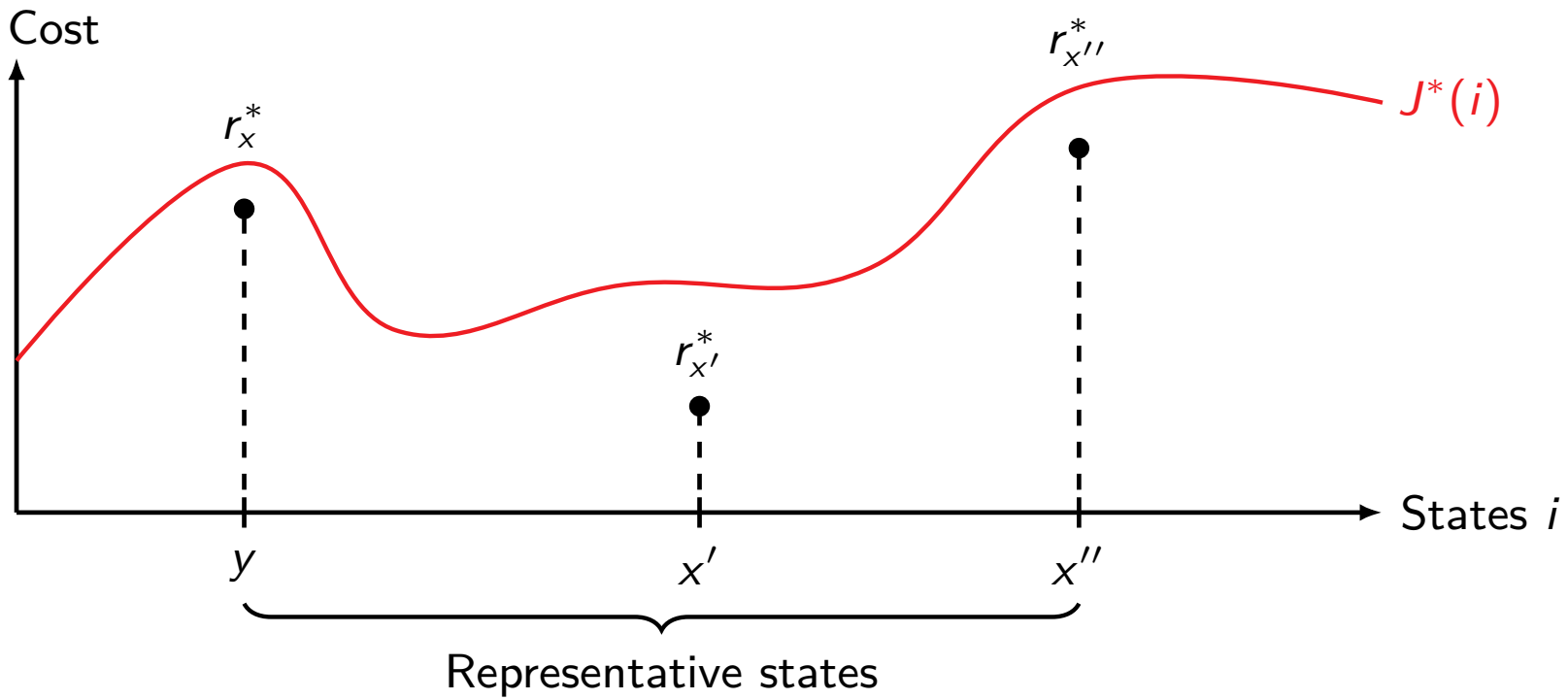
Solving the Aggregate Dynamic Programming Problem

- The aggregate problem can be solved "exactly" using dynamic programming/simulation; see [Ber19, Section 6.3]
- The optimal cost from a representative state x in this problem is denoted by r_x^*.



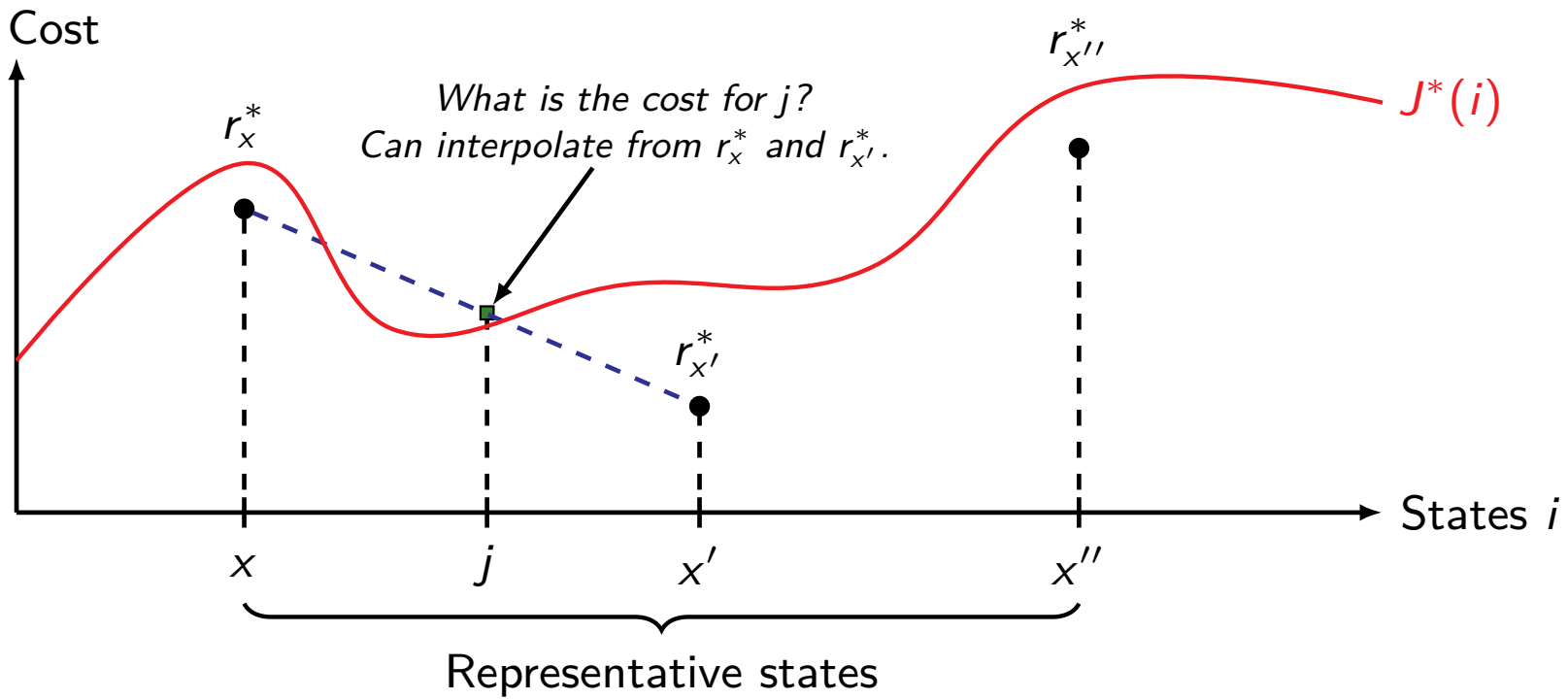
Cost Difference Between the Aggregate and Original Problems

- The aggregate cost function r_x^* is only defined for representative states $x \in \mathcal{A}$.
- The optimal cost function $J^*(i)$ is defined for the entire state space $i = 1, \ldots, n$.
- For a representative state x, we generally have $r_x^* \neq J^*(x)$.



Cost Difference Between the Aggregate and Original Problems

- The aggregate cost function r_x^* is only defined for representative states $x \in \mathcal{A}$.
- The optimal cost function $J^*(i)$ is defined for the entire state space $i = 1, \ldots, n$.
- For a representative state x, we generally have $r_x^* \neq J^*(x)$.



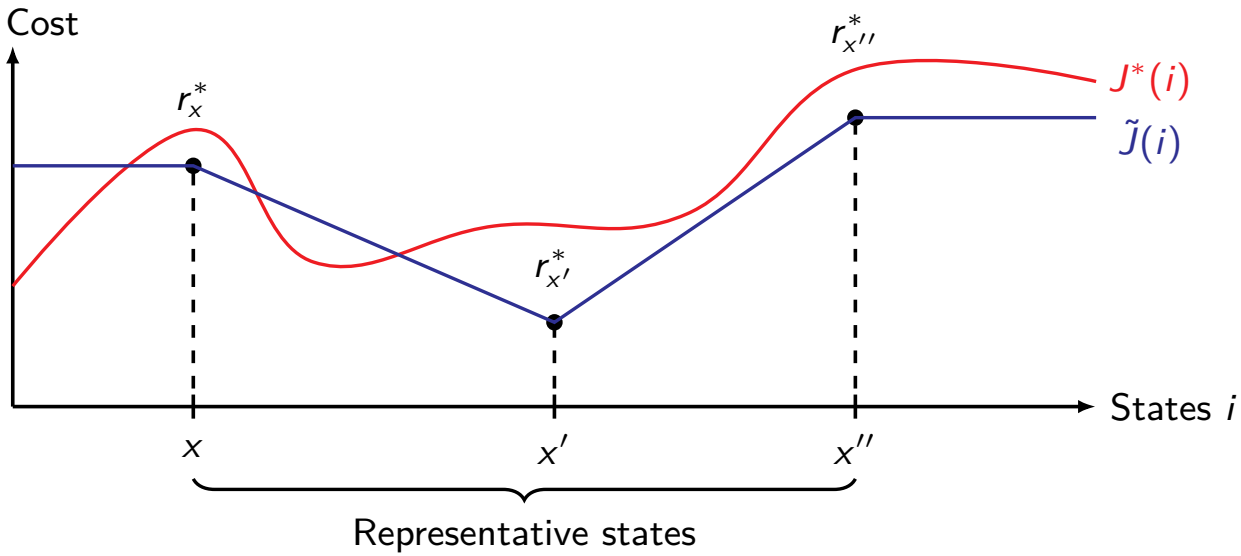
Using the Aggregate Solution to Approximate the Original Problem

- We obtain an approximate cost function $\tilde{J}$ for the original problem via **interpolation**:

$$\tilde{J}(i) = \sum_{x \in \mathcal{A}} \phi_{ix} r_x^*, \qquad i = 1, \ldots, n.$$

- Using this cost function, we can obtain a one-step lookahead policy:

$$\tilde{\mu}(i) \in \arg\min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u) \left(g(i,u,j) + \alpha \tilde{J}(j) \right), \qquad i = 1, 2, \ldots, n$$



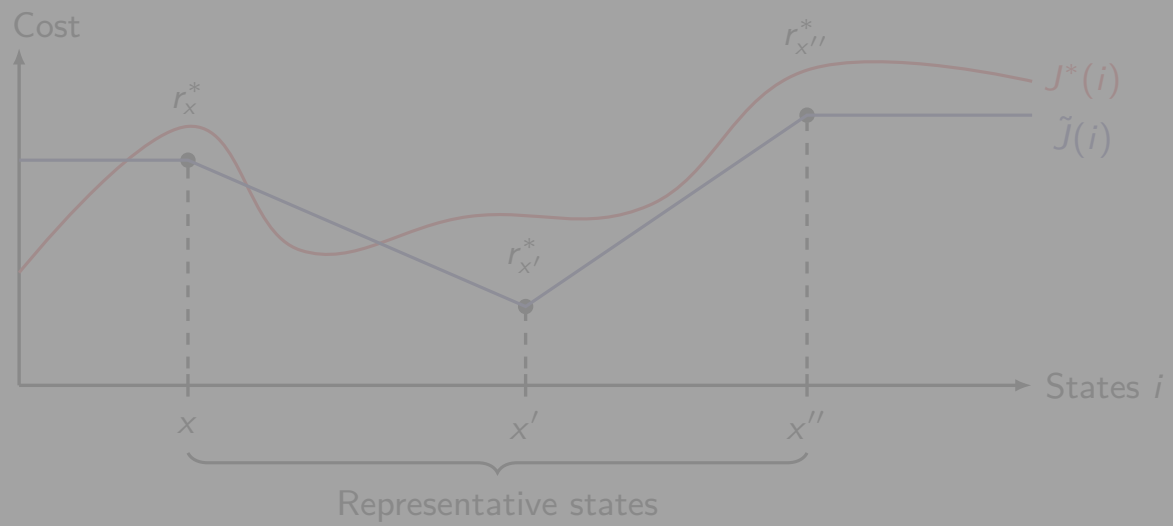
Approximating the Original Problem

- We obtain an approximate cost function $\tilde{J}$ for the original problem via **interpolation**:

$$\tilde{J}(j) = \sum_{y \in \mathcal{A}} \phi_{jy} r_y^*, \qquad j = 1, \ldots, n.$$

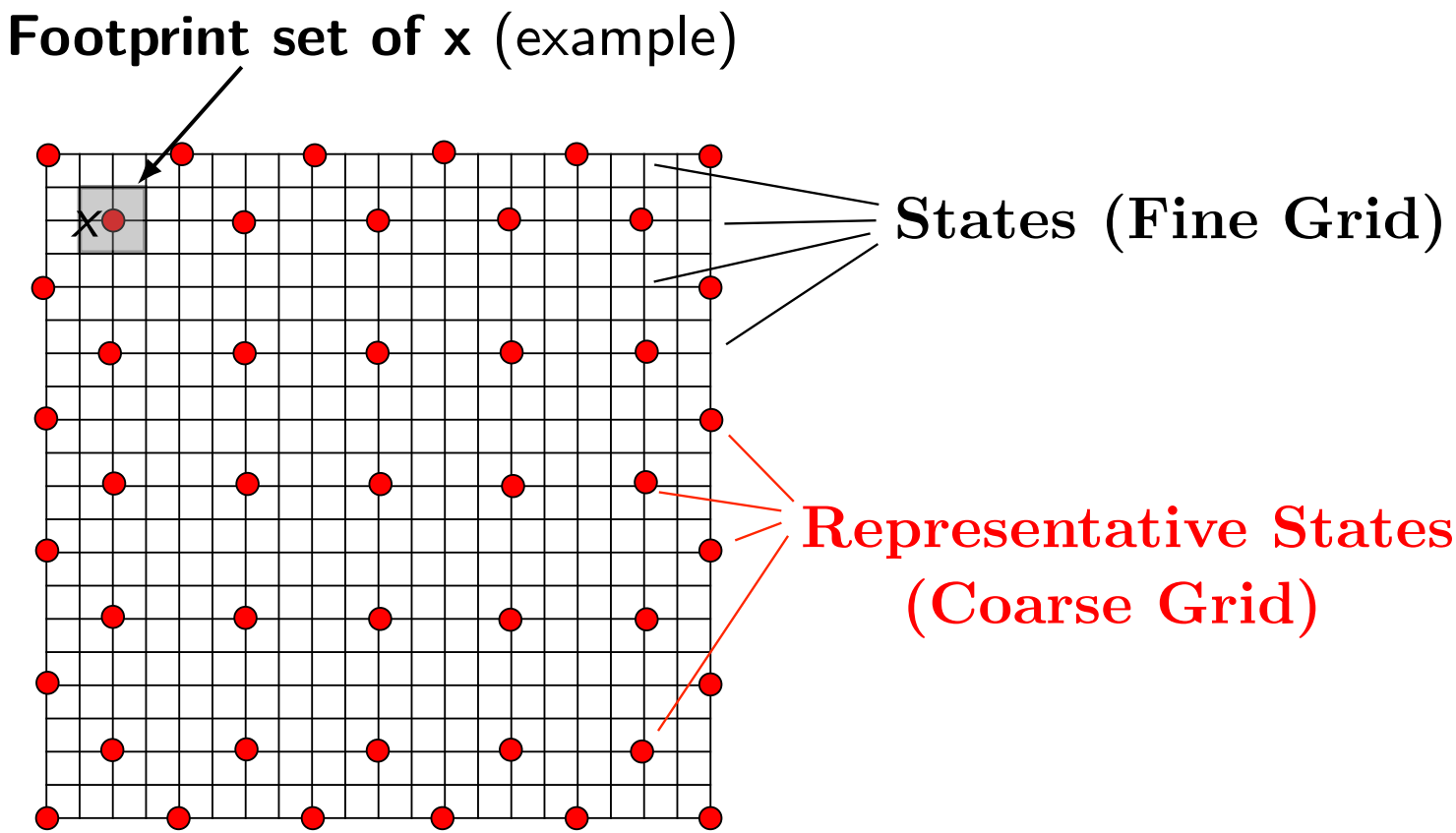
- Using this cost function, we can obtain a one-step lookahead policy:

What is the difference between the approximation $\tilde{J}$ and the optimal cost function J^*?



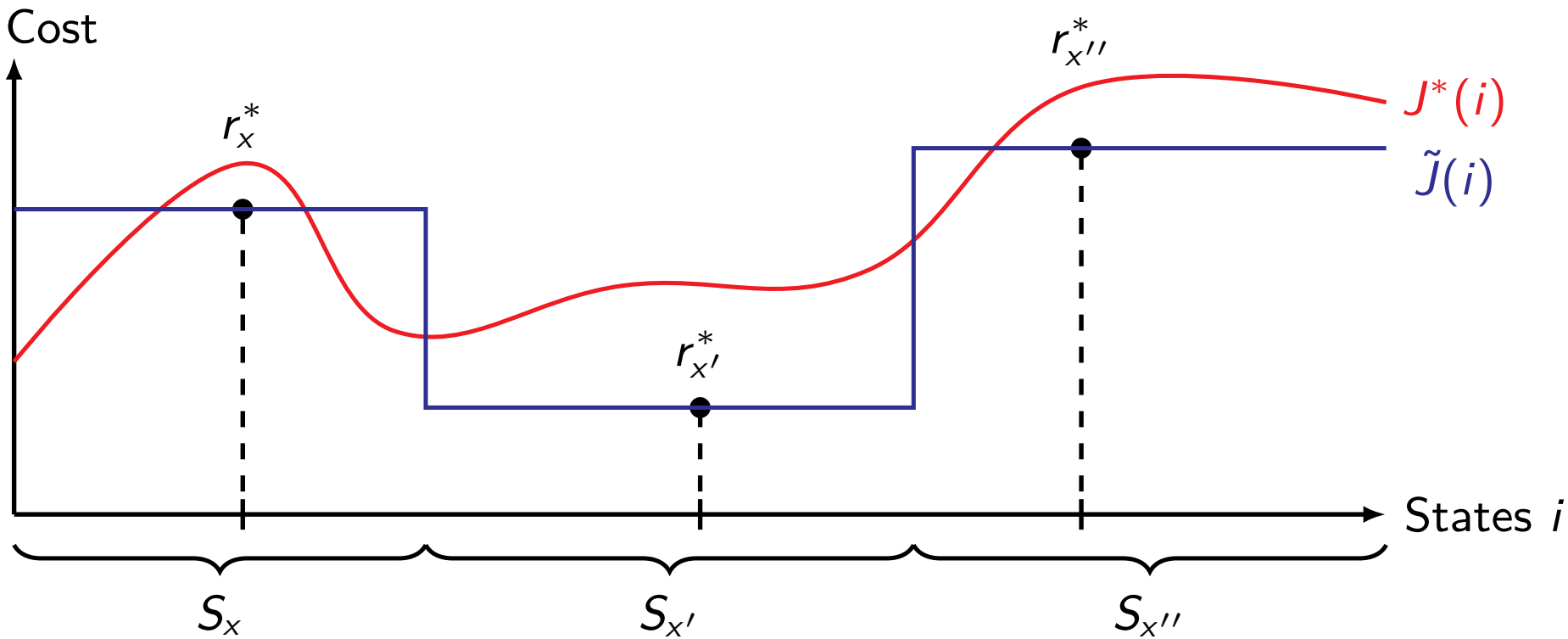
Hard Aggregation

- Consider the case where $\phi_{jx} = 0$ for all representative states x except one.
- Let S_x denote the set of states that aggregate to the representative state x.
  - i.e., the *footprint* of x, where $\{1, \ldots, n\} = \bigcup_{x \in \mathcal{A}} S_x$.



Structure of the Cost Function Approximation

- In the case of hard aggregation, $\tilde{J}(i) = \sum_{x \in \mathcal{A}} \phi_{ix} r_x^* = r_y^*$ for all $i \in S_y$.
- Hence, $\tilde{J}$ is piecewise constant.



Approximation Error Bound in the Case of Hard Aggregation [TR96]

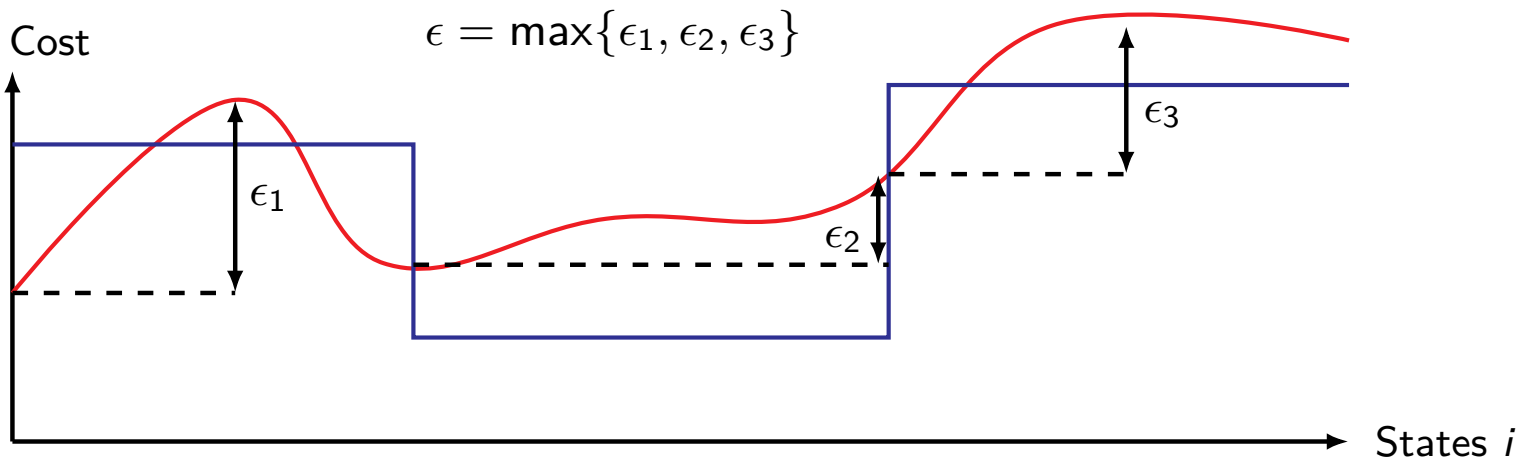
- Let ϵ be the maximum variation of J^* within a footprint set S_x, i.e.,

$$\epsilon = \max_{x \in \mathcal{A}} \max_{i,j \in S_x} |J^*(i) - J^*(j)|.$$

- We refer to the difference $|J^*(i) - \tilde{J}(i)|$ as the *approximation error*.
- This error is bounded as

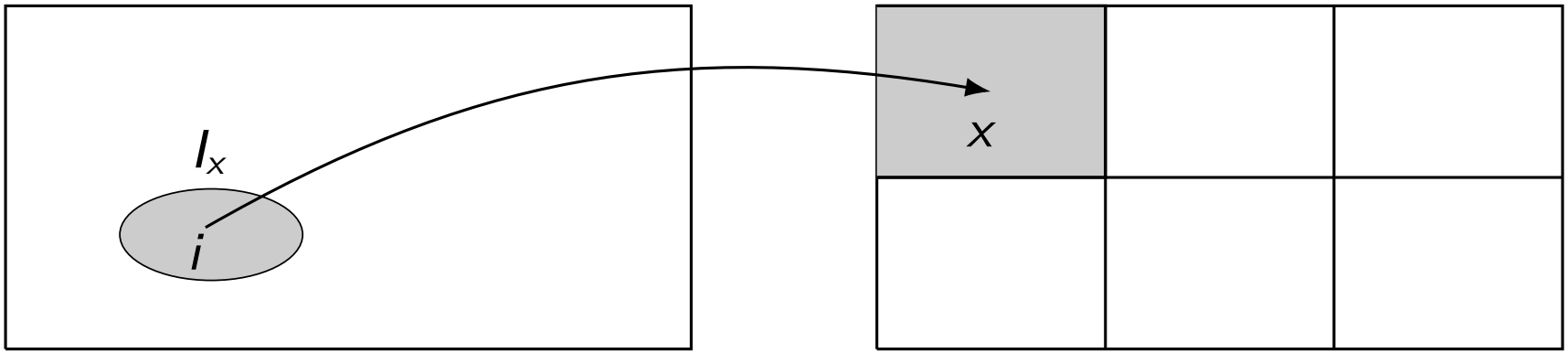
$$|J^*(i) - \tilde{J}(i)| \leq \frac{\epsilon}{1-\alpha} \qquad i = 1, \ldots, n.$$

- **Takeway**: choose the footprint sets so that ϵ is small.



General Aggregation and Approximation Error

- Introduce a finite set of aggregate states $\mathcal{A}$.
- Each aggregate state $x \in \mathcal{A}$ is associated with a disjoint subset $I_x \subset \{1, \ldots, n\}$.
- An aggregation problem can be defined similarly; see Section 3.6.4 of the course book [Ber25] for details



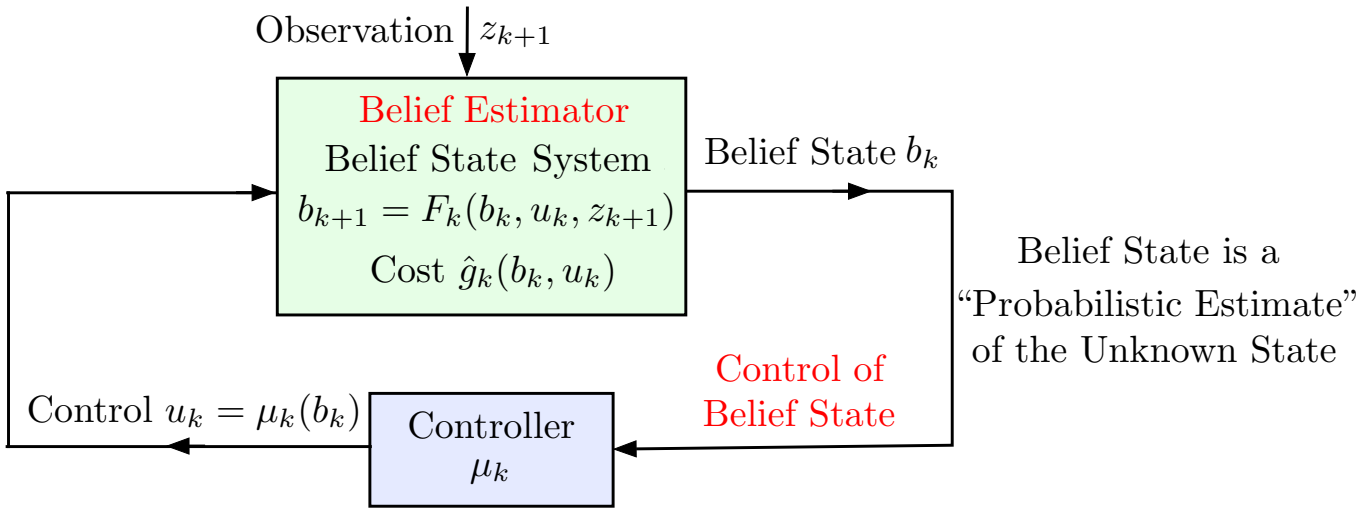
State space $\{1, \ldots, n\}$ Aggregate states $\mathcal{A}$

Upcoming Work

We show that similar error bound also hold for general aggreation with soft aggregation probabilities; i.e., $\phi_{jx} \neq 0$ for several $x \in \mathcal{A}$ [LHB25]

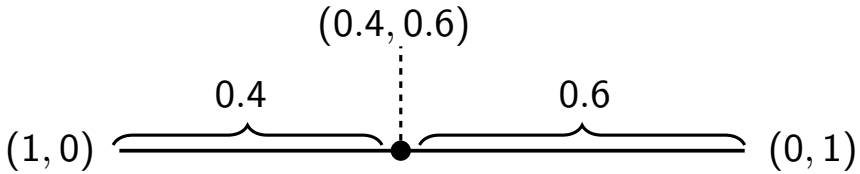
Partially Observed Markov Decision Problems (POMDPs)

- State space $X = \{1, \ldots, n\}$, observation space Z, and control constraint set $U(i)$.
- Each state transition (i, j) generates a cost $g(i, u, j)$;
- and an observation z with probability $p(z \mid j, u)$.
- Let $b(i)$ denote the conditional probability that the state is i, given the history.
- The belief state is defined as $b = \big(b(1), b(2), \ldots, b(n)\big)$.
- The belief b is updated using a **belief estimator** $F(b, u, z)$.
- Goal: Find a policy as a function of b that minimizes the cost.

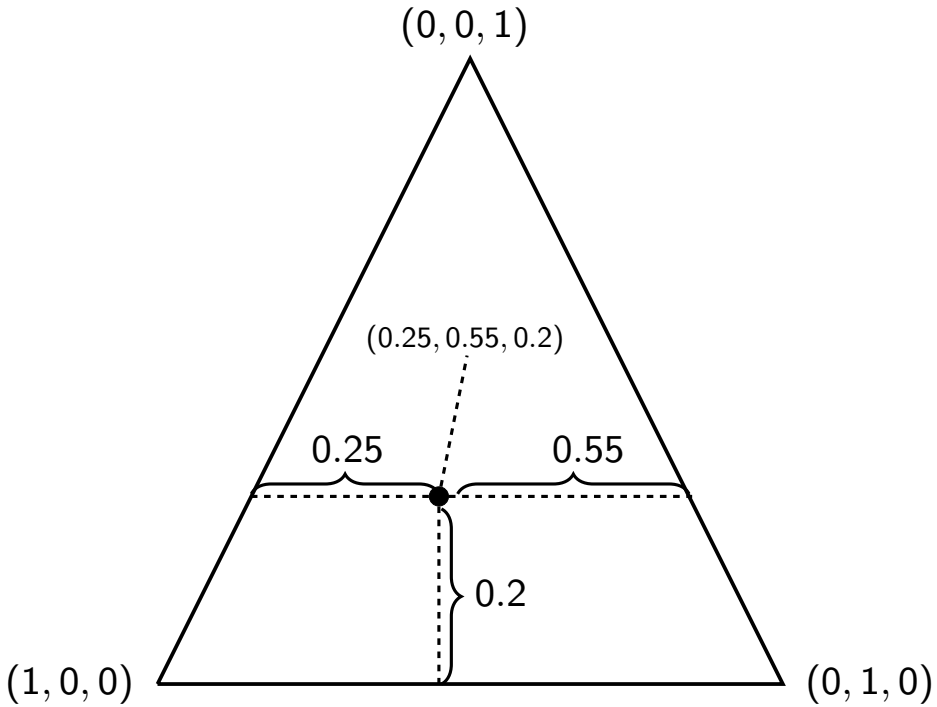


The Belief Space

- The belief b resides in the belief space B, i.e., the $n-1$ dimensional unit simplex.
- For example, if the states are $\{0, 1\}$, then $b \in [0, 1]$.



(a) *1-dimensional unit simplex.*



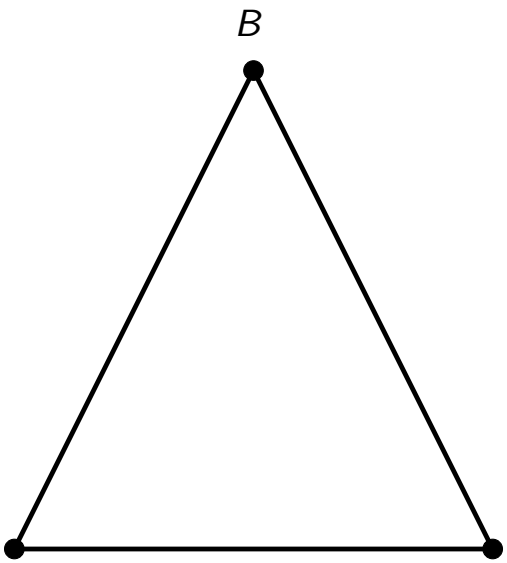
(b) *2-dimensional unit simplex.*

Aggregation of the Belief Space into Representative Beliefs

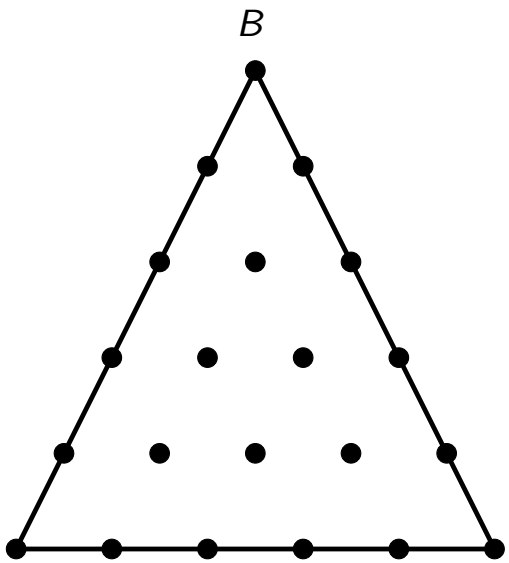
- We can obtain representative beliefs via uniform discretization of the belief space:

$$\mathcal{A} = \left\{ b \,\middle|\, b \in B, b(i) = \delta_i/\rho, \sum_{i=1}^{n} \delta_i = \rho, \delta_i \in \{0, \ldots, \rho\} \right\},$$

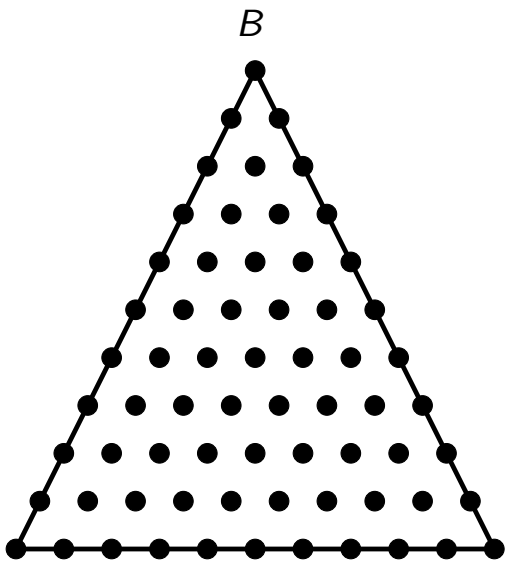
where ρ serves as the discretization resolution.



Discretization resolution $\rho = 1$



Discretization resolution $\rho = 5$

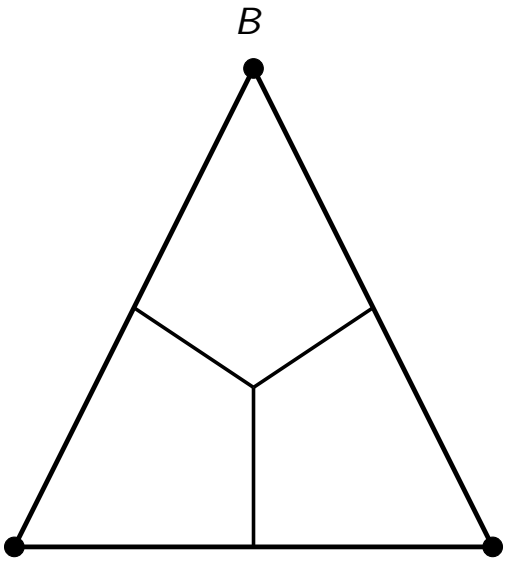


Discretization resolution $\rho = 10$

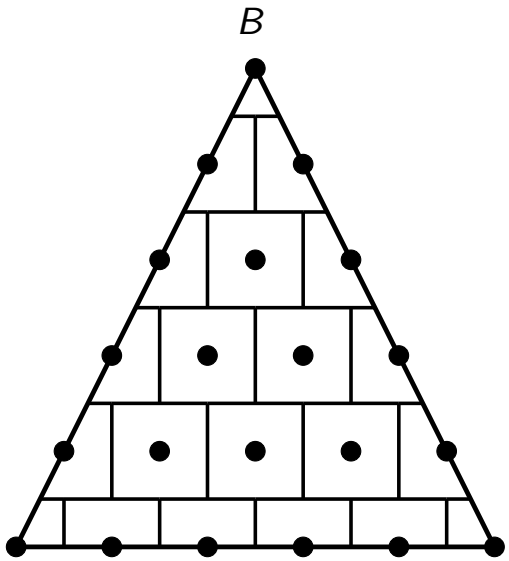
Hard Aggregation of the Belief Space

- We can implement hard aggregation via the nearest neighbor mapping:

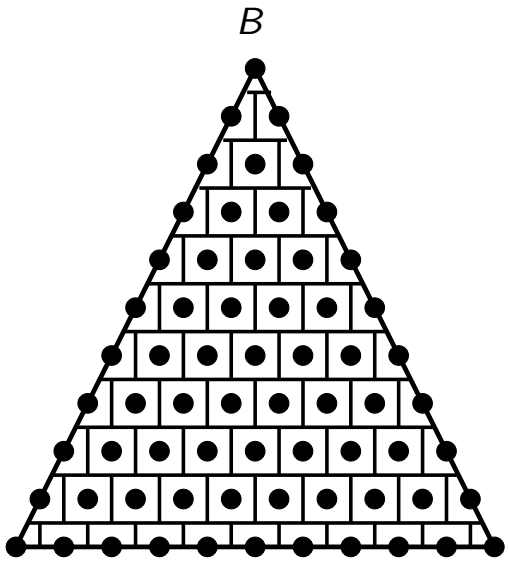
  $\phi_{by} = 1$ if and only if y is the nearest neighbor of b, where $b \in B$ and $y \in \mathcal{A}$.



Discretization resolution $\rho = 1$



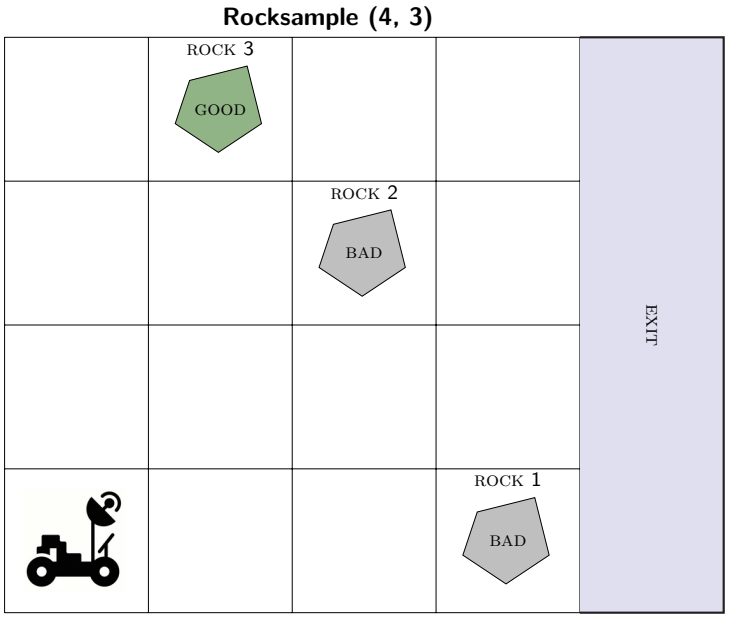
Discretization resolution $\rho = 5$



Discretization resolution $\rho = 10$

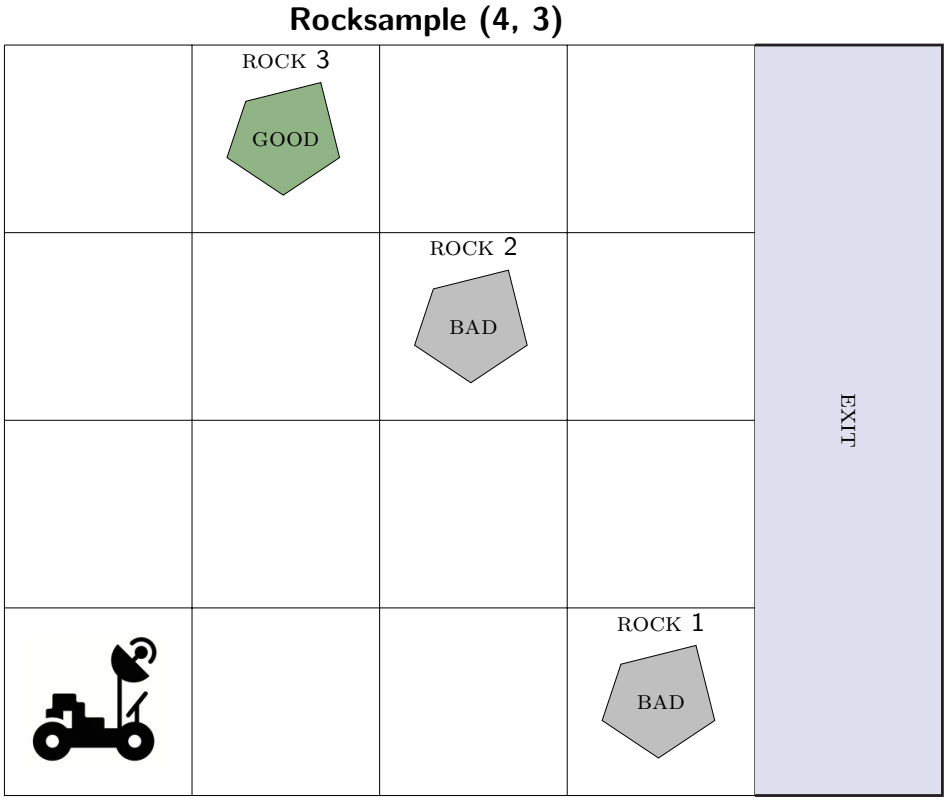
Example POMDP: **Rocksample (4,3)**

- Problem: rover exploration on Mars to find "good" rocks with high scientific value.
- There are 3 rocks on a 4×4 grid. The rover does not know which rocks are good.
- The controls (north, south, east, west) moves the rover (at cost 0.1).
- The control "sampling" determines the rock quality at the rover position (cost 10 for sampling a bad rock and cost -10 for sampling a good rock).
- Control "check-l" applies a **sensor to check the quality of rock l** (at cost 1).
- Accuracy of the sensor decreases exponentially with Euclidean distance to the rock.
- The rover stops the mission by moving to the right, yielding an exit-cost of -10.



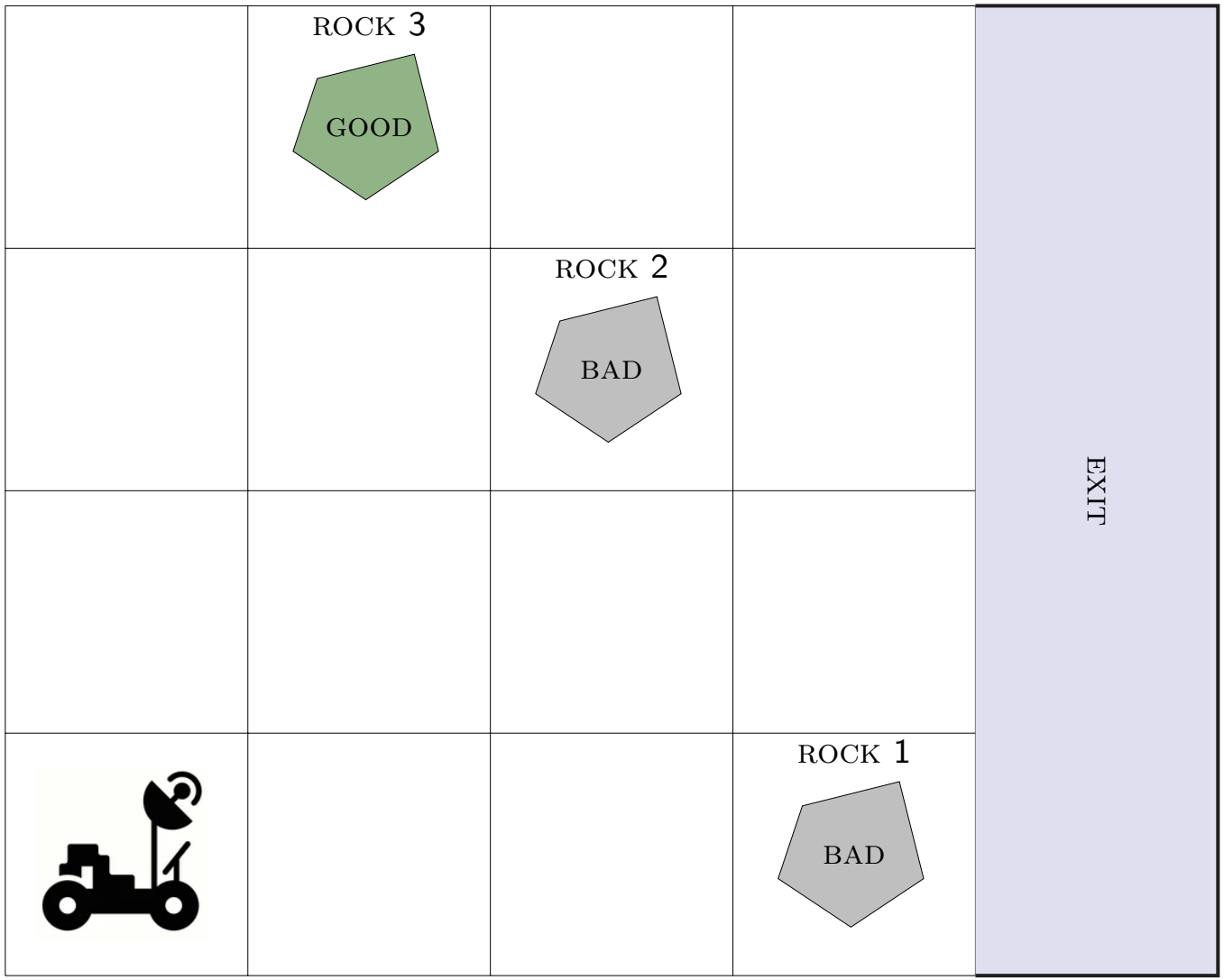
Approximating Rocksample (4,3) via Representative Aggregation

- The Rocksample (4,3) POMDP has a 127-dimensional belief space.
- We **discretize the belief space** with three different resolutions:
  - $\rho = 1$ leads to an aggregate problem with 128 representative beliefs.
  - $\rho = 2$ leads to an aggregate problem with 8256 representative beliefs.
  - $\rho = 3$ leads to an aggregate problem with 357760 representative beliefs.



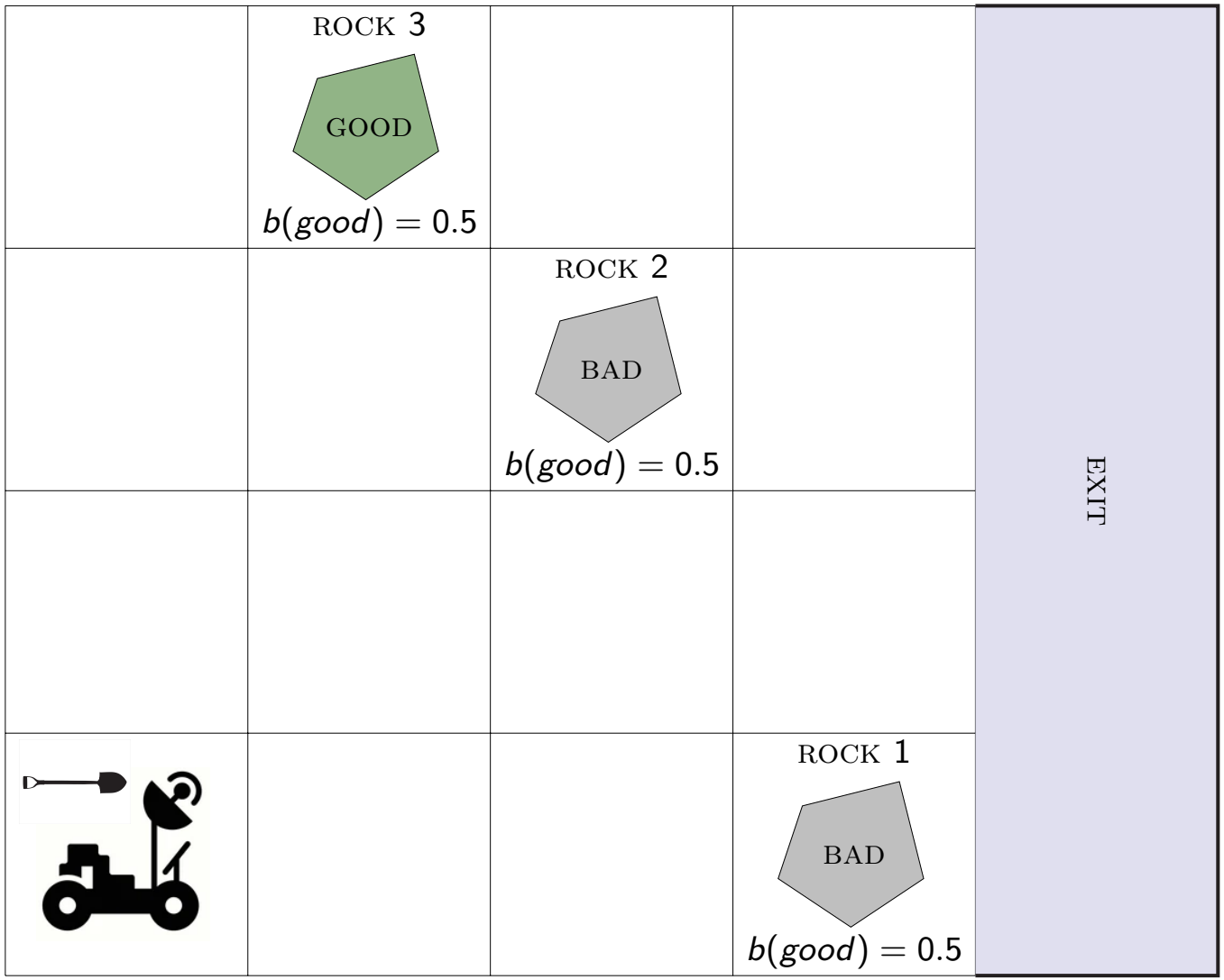
Animation Setup

Rocksample (4, 3)

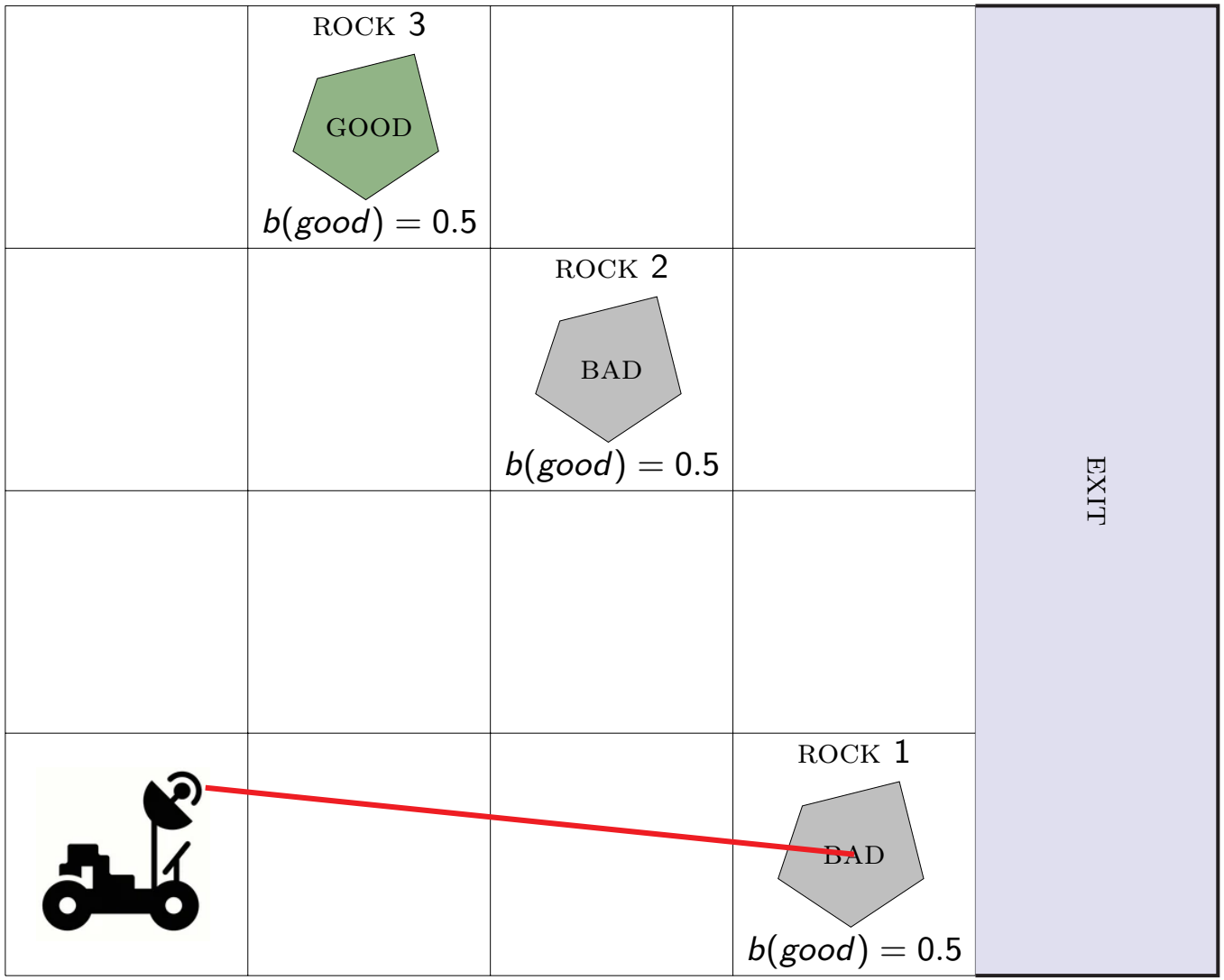


Animation Setup

Rocksample (4, 3)



Animation Setup



Animations of Control Policies Based on Aggregation with $\rho = 1$

Will be presented during the talk.

Animations of Control Policies Based on Aggregation with $\rho = 2$

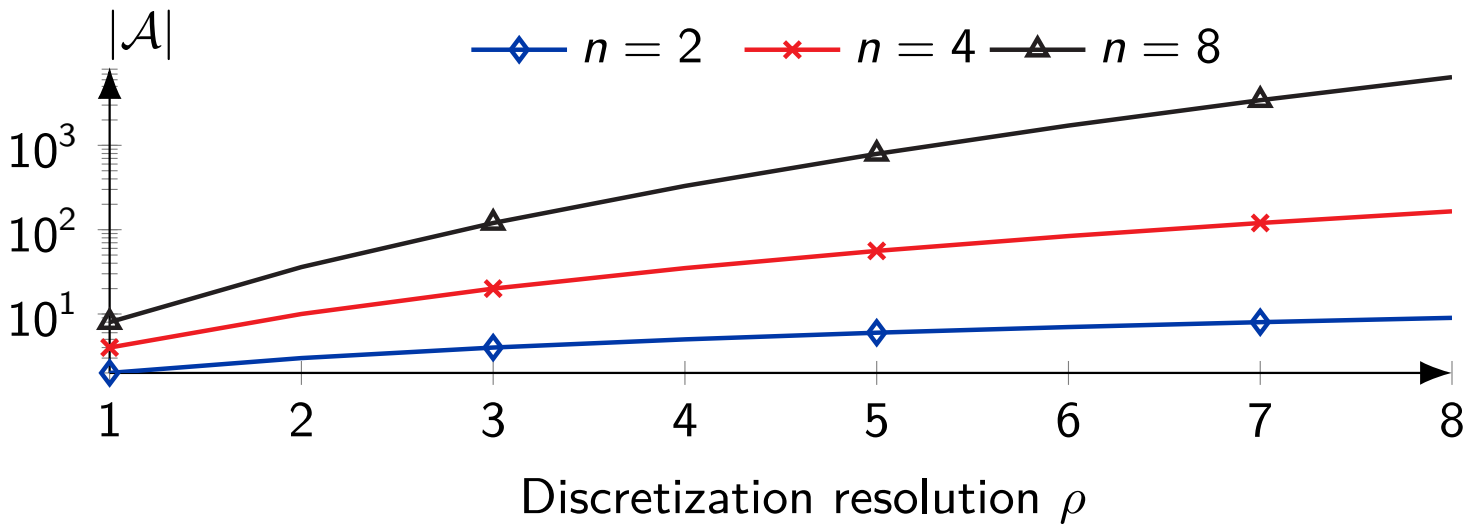
Will be presented during the talk.

Animations of Control Policies Based on Aggregation with $\rho = 3$

Will be presented during the talk.

Computational Trade-Offs

- The animations show that performance improves with the discretization resolution ρ.
- This is not surprising. As ρ increases, ϵ decreases.
- However, the computational complexity increases with the resolution ρ.



Comparison Between Aggregation and Other POMDP Methods

POMDP	States n	Observations $\lvert Z\rvert$	Controls $\lvert U\rvert$	Discount factor α
RS (4,4)	257	2	9	0.95
RS (5,5)	801	2	10	0.95
RS (5,7)	3201	2	12	0.95
RS (7,8)	12545	2	13	0.95
RS (10,10)	102401	2	15	0.95

Table: POMDPs used for the experimental evaluation.

Method	Aggregation	Point-based	Heuristic search	Policy-based	Exact DP
Our method	✓				
IP [CLZ13]					✓
PBVI [PGT06]		✓			
SARSOP [Ong+10]		✓			
POMCP [SV10]			✓		
HSVI [SS12]			✓		
AdaOPS [Wu+21]			✓		
R-DESPOT [Som+13]			✓		
POMCPOW [SK18]			✓		
PPO [Sch+17]				✓	
PPG [Cob+21]				✓	

Table: Methods used for the experimental evaluation; all methods are based on approximation schemes except IP, which uses exact dynamic programming.

Comparison Between Aggregation and Other POMDP Methods

Method \ POMDP	RS (4,4)	RS (5,5)	RS (5,7)	RS (7,8)	RS (10, 10)
Aggregation	-17.15/2.4	-18.12/125.5	-17.51/189.1	-14.71/202	-11.59/500
IP	N/A	N/A	N/A	N/A	N/A
PBVI	-8.24/300	-9.05/300	N/A	N/A	N/A
SARSOP	**-17.92**/10^{-2}	**-19.24**/58.5	N/A	N/A	N/A
POMCP	-8.64/1.6	-8.80/1.6	-9.81/1.6	-9.46/1.6	-8.98/1.6
HSVI	**-17.92**/10^{-2}	**-19.24**/6.2	**-24.69**/721.3	N/A	N/A
PPO	-8.57/300	-8.15/300	-8.76/300	-7.35/300	-4.59/1000
PPG	-8.57/300	-8.24/300	-8.76/300	-7.35/300	-4.41/1000
AdaOPS	-16.95/1.6	-17.39/1.6	-16.14/1.6	**-15.99**/1.6	**-15.29**/1.6
R-DESPOT	-12.07/1.6	-12.09/1.6	-12.00/1.6	-13.14/1.6	-10.41/1.6
POMCPOW	-8.60/1.6	-8.47/1.6	-8.26/1.6	-8.14/1.6	-7.88/1.6

Table: Evaluation results on the benchmark POMDPs; the first number in each cell is the total discounted cost; the second is the compute time in minutes (online methods were given 1 second planning time per control); cells with N/A indicate cases where a result could not be obtained for computational reasons. RS(m,l) stands for an instance of Rocksample with an $m \times m$ grid and l rocks.

Performance can be further enhanced by combining with other RL methods, such as rollout [Ham+25]

Thank you!

References I

[Ber19] Dimitri P. Bertsekas. *Reinforcement Learning and Optimal Control*. Athena Scientific, 2019.

[Ber25] Dimitri P. Bertsekas. *A Course in Reinforcement Learning*. 2nd. Athena Scientific, 2025.

[CLZ13] Anthony R Cassandra, Michael L Littman, and Nevin Lianwen Zhang. "Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes". In: *arXiv preprint arXiv:1302.1525* (2013).

[Cob+21] Karl W Cobbe et al. "Phasic policy gradient". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 2020–2027.

[Ham+25] Kim Hammar et al. "Adaptive Network Security Policies via Belief Aggregation and Rollout". In: *arXiv preprint upcoming* (2025).

[LHB25] Yuchao Li, Kim Hammar, and Dimitri P. Bertsekas. "Feature-Based Belief Aggregation for Partially Observable Markov Decision Problems". In: *arXiv preprint upcoming* (2025).

[Ong+10] Sylvie CW Ong et al. "Planning under uncertainty for robotic tasks with mixed observability". In: *The International Journal of Robotics Research* 29.8 (2010), pp. 1053–1068.

References II

[PGT06] Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. "Anytime point-based approximations for large POMDPs". In: *Journal of Artificial Intelligence Research* 27 (2006), pp. 335–380.

[Sch+17] John Schulman et al. "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347* (2017).

[SK18] Zachary N. Sunberg and Mykel J. Kochenderfer. "Online Algorithms for POMDPs with Continuous State, Action, and Observation Spaces". In: *Proceedings of the 28th International Conference on Automated Planning and Scheduling (ICAPS)*. 2018. URL: https://www.aaai.org/ocs/index.php/ICAPS/ICAPS18/paper/viewFile/17734/16986.

[Som+13] Adhiraj Somani et al. "DESPOT: Online POMDP planning with regularization". In: *Advances in neural information processing systems* 26 (2013).

[SS12] Trey Smith and Reid Simmons. "Heuristic search value iteration for POMDPs". In: *arXiv preprint arXiv:1207.4166* (2012).

[SV10] David Silver and Joel Veness. "Monte-Carlo Planning in Large POMDPs". In: *Advances in Neural Information Processing Systems*. Vol. 23. 2010.

References III

[TR96] John N. Tsitsiklis and Benjamin van Roy. "Feature-based methods for large scale dynamic programming". In: *Machine Learning* 22.1 (Mar. 1996), pp. 59–94.

[Wu+21] Chenyang Wu et al. "Adaptive online packing-guided search for POMDPs". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 28419–28430.