Reinforcement Learning and Optimal Control

ASU, CSE 691, Winter 2019

Dimitri P. Bertsekas
dimitrib@mit.edu

Lecture 11

Outline

Aggregation within the Approximation in Value Space Framework

Approximate minimization

$$\min_{u\in U(i)} \sum_{j=1}^{n} p_{ij}(u)\Big(\overbrace{g(i,u,j)}^{\text{First Step}} + \overbrace{\alpha\tilde{J}(j)}^{\text{“Future”}}\Big)$$

Approximations:

Replace $E\{\cdot\}$ with nominal values (certainty equivalence)

Adaptive simulation

Monte Carlo tree search

Computation of $\tilde{J}$:

Problem approximation

Rollout

Approximate PI

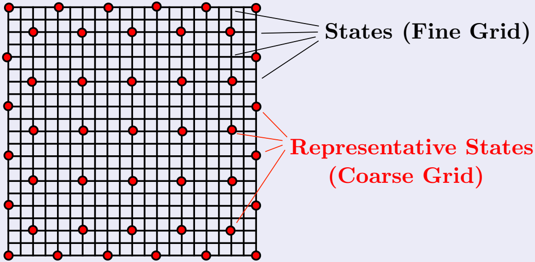
Parametric approximation

Aggregation

- Aggregation is a form of problem approximation. We approximate our DP problem with a “smaller/easier" version, which we solve optimally to obtain $\tilde{J}$.
- Is related to feature-based parametric approximation (e.g., when $\tilde{J}$ is piecewise constant, the features are 0-1 membership functions).
- Can be combined with (global) parametric approximation (like a neural net) in two ways. Either use the neural net to provide features, or add a local parametric correction to a $\tilde{J}$ obtained by a neural net.
- Several versions: multistep lookahead, finite horizon, etc ...

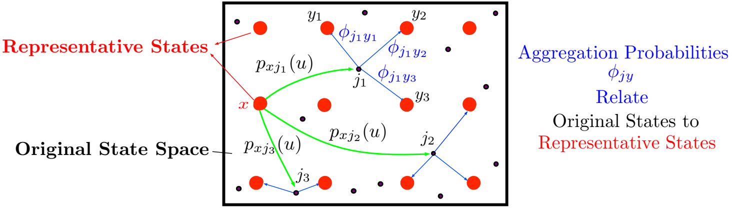
Illustration: A Simple Classical Example of Approximation

Approximate the state space with a coarse grid of states



- Introduce a “small" set of “representative" states to form a coarse grid.
- Approximate the original DP problem with a coarse-grid DP problem, called aggregate problem (need transition probs. and cost from rep. states to rep. states).
- Solve the aggregate problem by exact DP.
- “Extend" the optimal cost function of the aggregate problem to an approximately optimal cost function for the original fine-grid DP problem.
- For example extend the solution by a nearest neighbor/piecewise constant scheme (a fine grid state takes the cost value of the “nearest" coarse grid state).

Approximate the Problem by “Projecting" it onto Representative States



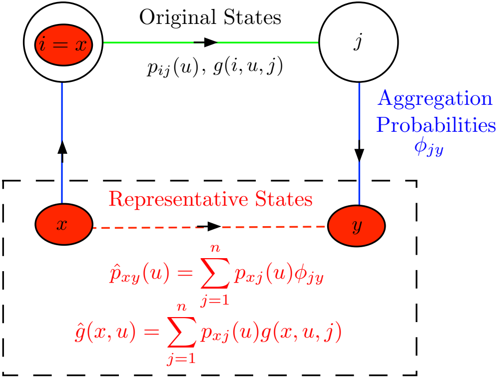
- Introduce a finite subset of “representative states" $\mathcal{A} \subset \{1, \ldots, n\}$. We denote them by x and y.
- Original system states j are related to rep. states $y \in \mathcal{A}$ with aggregation probabilities ϕ_{jy} (“weights" satisfying $\phi_{jy} \geq 0$, $\sum_{y \in \mathcal{A}} \phi_{jy} = 1$).
- Aggregation probabilities express “similarity" or “proximity" of original to rep. states.
- Aggregate dynamics: Transition probabilities between rep. states x, y

$$\hat{p}_{xy}(u) = \sum_{j=1}^{n} p_{xj}(u)\phi_{jy}$$

- Expected cost at rep. state x under control u:

$$\hat{g}(x, u) = \sum_{j=1}^{n} p_{xj}(u) g(x, u, j)$$

The Aggregate Problem



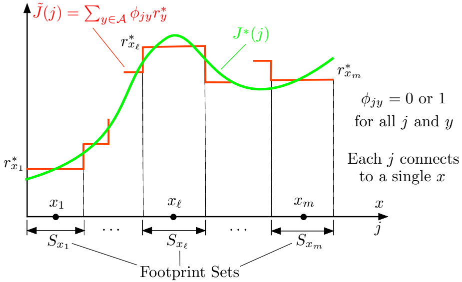
- If r_x^*, $x \in \mathcal{A}$, are the optimal costs of the aggregate problem, approximate the optimal cost function of the original problem by

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

- If $\phi_{jy} = 0$ or 1 for all j and y, $\tilde{J}(j)$ is piecewise constant. It is constant on each set

$$S_y = \{j \mid \phi_{jy} = 1\}, \quad y \in \mathcal{A}, \qquad \text{(called the footprint of } y\text{)}$$

The Piecewise Constant Case ($\phi_{jy} = 0$ or 1 for all j, y)



The approximate cost function $\tilde J = \sum_{y\in\mathcal{A}} \phi_{jy} r_y^*$ is constant within $S_y = \{j \mid \phi_{jy} = 1\}$.

Approximation error for the piecewise constant case ($\phi_{jy} = 0$ or 1 for all j, y)

Consider the footprint sets

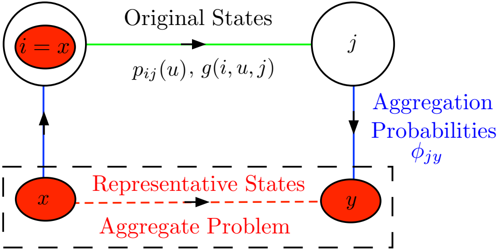
$$S_y = \{j \mid \phi_{jy} = 1\}, \qquad y \in \mathcal{A}$$

The $(J^* - \tilde J)$ error is small if J^* varies little within each S_y. In particular,

$$\left|J^*(j) - \tilde J(j)\right| \le \frac{\epsilon}{1-\alpha}, \qquad j \in S_y,\ y \in \mathcal{A},$$

where $\epsilon = \max_{y\in\mathcal{A}} \max_{i,j\in S_y} \left|J^*(i) - J^*(j)\right|$ is the max variation of J^* within the S_y.

Solution of the Aggregate Problem



Data of aggregate problem (it is stochastic even if the original is deterministic)

$$\hat{p}_{xy}(u) = \sum_{j=1}^{n} p_{xj}(u)\phi_{jy}, \quad \hat{g}(x,u) = \sum_{j=1}^{n} p_{xj}(u) g(x,u,j), \qquad \tilde{J}(j) = \sum_{y \in \mathcal{A}} \phi_{jy} r_y^*$$

Exact methods

Once the aggregate model is computed (i.e., its transition probs. and cost per stage), any exact DP method can be used: VI, PI, optimistic PI, or linear programming.

Model-free simulation methods - Needed for large n, even if model is available

Given a simulator for the original problem, we can obtain a simulator for the aggregate problem. Then use an (exact) model-free method to solve the aggregate problem.

Extension: Continuous State Space - POMDP Discretization

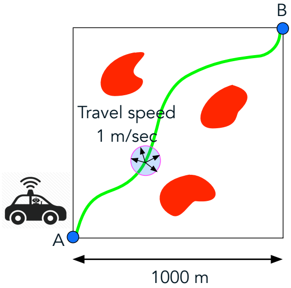
Continuous state space

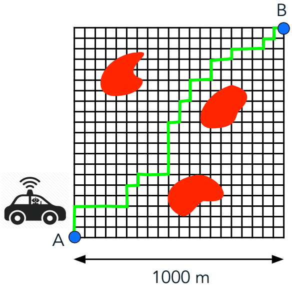
- The rep. states approach applies with no modification to continuous spaces discounted problems.
- The number of rep. states should be finite.
- The cost per stage should be bounded for the “good"/contraction mapping-based theory to apply to the original DP problem.
- A simulation/model-free approach may still be used for the aggregate problem.
- We thus obtain a general discretization method for continuous-spaces discounted problems.

Discounted POMDP with a belief state formulation

- Discounted POMDP models with belief states, fit neatly into the continuous state discounted aggregation framework.
- The aggregate/rep. states POMDP problem is a finite-state MDP that can be solved for r^* with any (exact) model-based or model-free method (VI, PI, etc).
- The optimal aggregate cost r^* yields an approximate cost function $\tilde{J}(j) = \sum_{y \in \mathcal{A}} \phi_{jy} r_y^*$, which defines a one-step or multistep lookahead suboptimal control scheme for the original POMDP.

A Challenge Question - Think for Five Mins

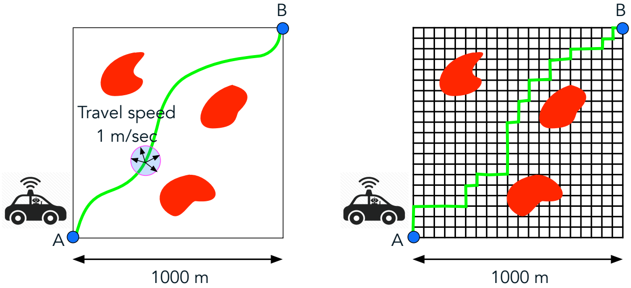




Discretizing Continuous Motion

- A self-driving car wants to drive from A to B through obstacles. Find the fastest route.
- Car speed is 1 m/sec in any direction.
- We discretize the space with a fine square grid; restrict directions of motion to horizontal and vertical.
- We take the discretized shortest path solution as an approximation to the continuous shortest path solution.
- Is this a good approximation?

Answer to the Challenge Question



Discretizing Continuous Motion

- The discretization is FLAWED.
- Example: Assume all motion costs 1 per meter, and no obstacles.
- The continuous optimal solution (the straight A-to-B line) has length $\sqrt{2}$ kilometers.
- The discrete optimal solution has length 2 kilometers regardless of how fine the discretization is.
- Here the state space is discretized finely but the control space is not.
- This is not an issue in POMDP (the control space is finite).

From Representative States to Representative Features

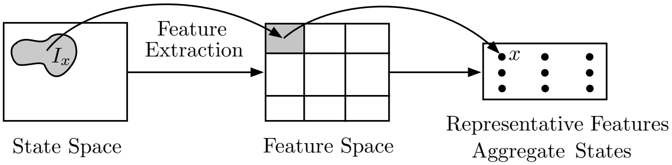
The main difficulty with rep. states/discretization schemes:

- It may not be easy to find a set of rep. states and corresponding piecewise constant or linear functions that approximate well J^*.
- Too many rep. states may be required for good approximate costs $\tilde{J}(j)$.

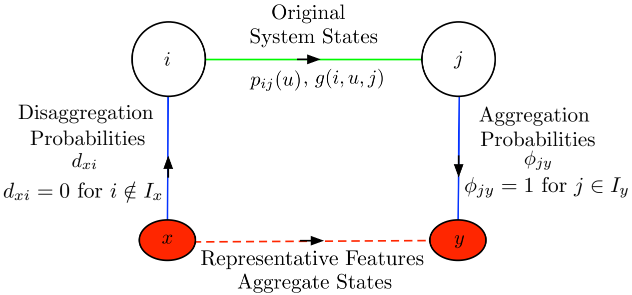
Suppose we have a good feature vector $F(i)$: We discretize the feature space

- We introduce representative features that span adequately the feature space
$$\mathcal{F} = \{F(i) \mid i = 1, \ldots, n\}$$
- We aim for an aggregate problem whose states are the rep. features.
- We associate each rep. feature x with a subset of states I_x that nearly map onto feature x, i.e.,
$$F(i) \approx x, \qquad \text{for all } i \in I_x$$
- This is done with the help of weights d_{xi} (called disaggregation probabilities) that are 0 outside of I_x.
- As before, we associate each state j with rep. features y using aggregation probabilities ϕ_{jy}.
- We construct an aggregate problem using d_{xi}, ϕ_{jy}, and the original problem data.

Illustration of Feature-Based Aggregation Framework

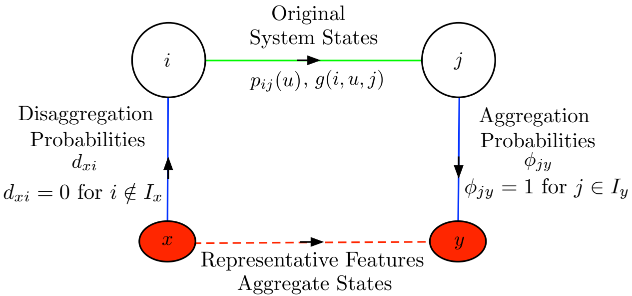


Representative feature formation



Transition diagram for the aggregate problem

Working Break: Feature Formation Methods in Aggregation



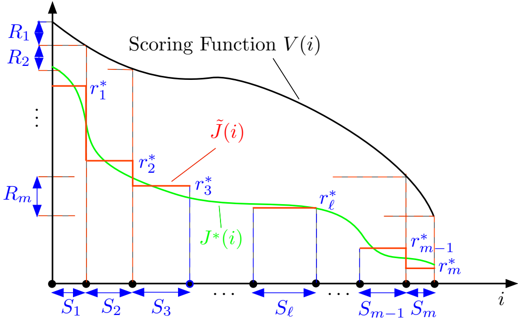
Question 1: Why is the rep. states model a special case of the rep. features model?

Assume the following general principle for feature-based aggregation:

Choose features so that states i with similar features $F(i)$ have similar $J^*(i)$, i.e., $J^*(i)$ changes little within each of the “footprint" sets $I_x = \{i \mid d_{xi} > 0\}$ and $S_y = \{j \mid \phi_{jy} > 0\}$.

Question 2: Can you think of examples of useful features for aggregation schemes?

Feature Formation Using Scoring Functions



Idea: Suppose that we have a scoring function $V(i)$ with $V(i) \approx J^*(i)$. Then group together states with similar score.

- We partition the range of values of V into m disjoint intervals $R_1, \dots, R_m$.
- We define a feature vector $F(i)$ according to

$$F(i) = \ell, \qquad \text{all } i \text{ such that } V(i) \in R_\ell, \quad \ell = 1, \dots, m$$

- Defines a partition of the state space into the footprints $S_\ell = I_\ell = \{i \mid F(i) = \ell\}$.

Examples of Scoring Functions

- Cost functions of heuristics or policies.
- Approximate cost functions produced by neural networks.

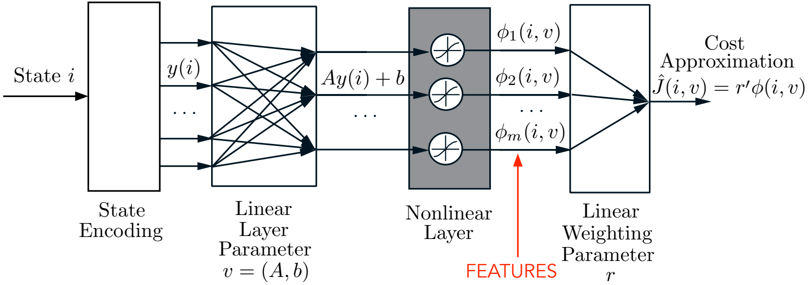
Let the scoring function be the cost function J_μ of a policy μ

Let's compare with rollout:

- Rollout uses as cost approximation $\tilde J = J_\mu$.
- Score-based aggregation uses J_μ as scoring function to form features. The resulting $\tilde J$ is a “nonlinear function of J_μ" that aims to approximate J^*.
- If the scoring function quantization were so fine as to have a single feature value per interval R_ℓ, we would have $\tilde J = J^*$ (much better than rollout).
- Score-based aggregation can be viewed as a more sophisticated form of rollout.
- Score-based aggregation is more computation-intensive, less suitable for on-line implementation.

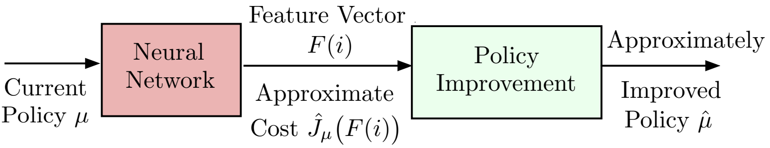
It is possible to use multiple scoring functions to generate more complex feature maps.

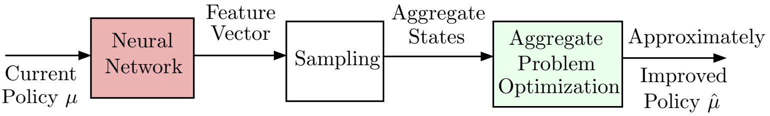
Feature Formation Using Neural Networks



Suppose we have trained a NN that provides an approximation $\hat{J}(i) = r'\phi(i, v)$

- Features from the NN can be used to define rep. features.
- Training of the NN yields lots of state-feature pairs.
- Rep. features and footprint sets of states can be obtained from the NN training set data, perhaps supplemented with additional (state,feature) pair data.
- NN features may be supplemented by handcrafted features.
- Feature-based aggregation yields a nonlinear function $\tilde{J}$ of the features that approximates J^* (not $\hat{J}$).

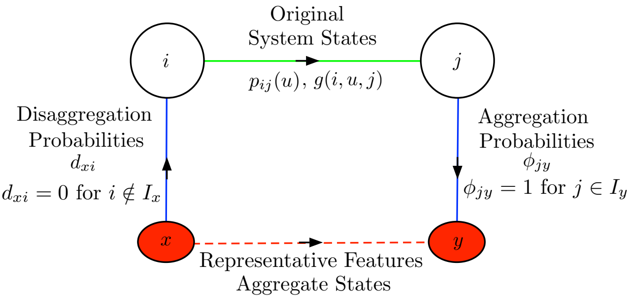




Several options for implementation of mixed NN/aggregation-based PI

- The NN-based feature construction process may be performed multiple times, each time followed by an aggregate problem solution that constructs a new policy.
- Alternatively: The NN training and feature construction may be done only once with some “good" policy.
- After each cycle of NN-based feature formation, we may add problem-specific handcrafted features, and/or features from previous cycles.
- Note: Deep NNs may produce fewer and more sophisticated final features

A Simple Version of the Aggregate Problem



Patterned after the simpler rep. states model.

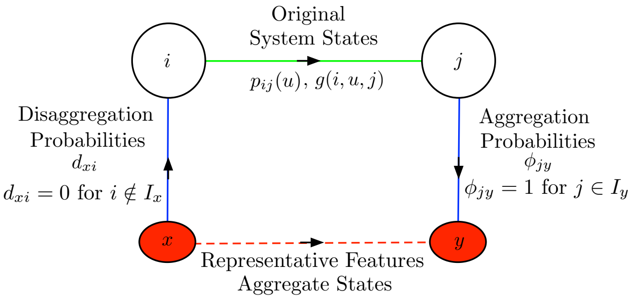
Aggregate dynamics and costs

- Aggregate dynamics: Transition probabilities between rep. features x, y

$$\hat{p}_{xy}(u) = \sum_{i \in I_x} d_{xi} \sum_{j=1}^{n} p_{ij}(u)\phi_{jy}$$

- Expected cost per stage:

$$\hat{g}(x,u) = \sum_{i \in I_x} d_{xi} \sum_{j=1}^{n} p_{xj}(u) g(x,u,j)$$



There is an implicit assumption in the aggregate dynamics and cost formulas

$$\hat{p}_{xy}(u)=\sum_{i\in I_x} d_{xi}\sum_{j=1}^{n} p_{ij}(u)\phi_{jy},\qquad \hat{g}(x,u)=\sum_{i\in I_x} d_{xi}\sum_{j=1}^{n} p_{xj}(u)g(x,u,j)$$

For a given rep. feature x, the same control u is applied at all states i in the footprint I_x.

So the simple aggregate problem is legitimate, but the approximation $\tilde{J}$ of J^* may not be very good. We will address this issue in the next lecture.

About the Next Lecture

We will continue approximation in value space by aggregation. We will cover:

- A more sophisticated aggregate problem formulation.
- Aggregate problem solution methods.
- Variants of aggregation.

CHECK MY WEBSITE FOR READING MATERIAL

PLEASE DOWNLOAD THE LATEST VERSIONS FROM MY WEBSITE