Value Function Approximation in Graph-based Reinforcement Learning

Sephora Madjiheurem and Laura Toni
{sephora.madjiheurem.17, l.toni}@ucl.ac.uk
Department of Electrical and Electronic Engineering
University College London, London, UK

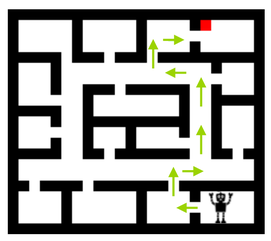
Overall Goals

- To study **value function approximation in reinforcement learning problems with high dimensional state or action spaces**
- To depart from the smoothness assumption for the **state value function**
- To highlight the importance of features learning for an improved value function approximation

Background

Classical Reinforcement Learning (RL)

- An agent takes *actions* in an *environment*
- Sequential *rewards* are observed
- Goal: find a policy that **maximises the cumulative reward** over time



Example: navigating through a maze

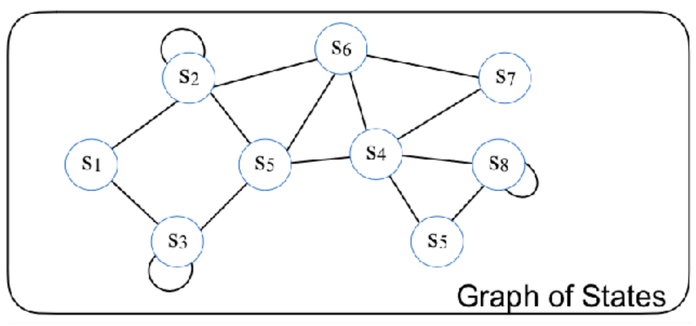
Approximate Dynamic Programming (ADP)

- The *value function V(s)* estimates the expected long term reward for some policy when starting from state *s*
- Challenge: in high-dimensional search space, the problem becomes **intractable** ➜ **approximate V(s)** with a linear combination of *basis functions* $\boldsymbol{\phi}(s)$:

$$\tilde{V}(s|\theta) = \sum_{i=1}^{d} \theta_i \phi_i(s) \approx V(s)$$

- Challenge: identify the right set of basis function ➜ **formulate the RL problem as a graph** of states

ADP in Graph-Based RL



Graph-Based Reinforcement Learning

- Unweighted and undirected **graph of states** (states being the nodes, VF being the signal for each node)
- Transition probabilities and the value function unknown *a priori*
- Graph signal processing tools and/or machine learning to **learn the structure/geometry** of the problem
- Exploit the structure to **efficiently** approximate the state values
- Derive a policy from the approximated value function

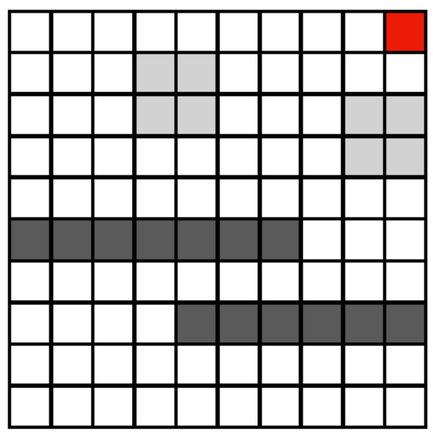
Proto-Value Functions (PVFs)

- **Assumption: the value function is a smooth functio**n
- Basis functions chosen to be t**he smoothest eigenvectors of a diffusion operato**r on the graph (e.g. the graph Laplacian)
- Challenge: the value function is **not necessarily a smooth function**
  - ➜ Need to automatically learn the basis functions from the geometry of the graph

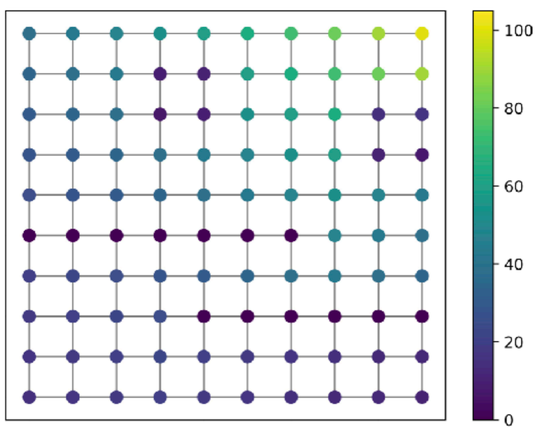
Node2vec

- Graph node embeddings can be used as basis functions
- *Node2vec* **learns continuous feature representation for nodes**
- Based on random walk statistics
- Nodes have similar embedding if they tend to co-occur on short random walks
- The measure of graph proximity is **flexible** and **stochastic**

Case Study



White = accessible room (transition probability *p=1.0*), dark grey = strict walls (*p=0.0*), light grey = difficult access rooms (*p=0.2*), red = goal room (+100 reward and *p=1.0*).

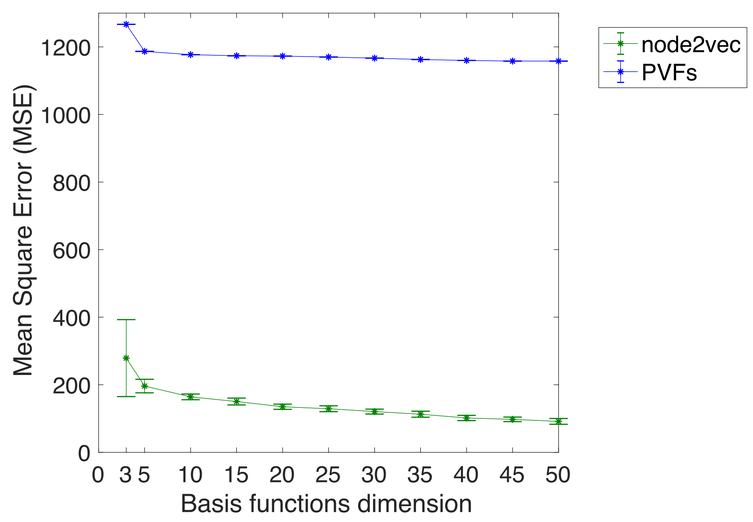


Optimal value function (computed with value iteration). Observe that the value function is not a smooth function on the graph.

1. Build the PVFs (= the *d* smoothest eigenvectors of the graph Laplacian) and learn the *node2vec* embeddings of the graph (representation vectors of size *d*) and used them as basis functions in the linear approximation problem.
2. Approximate the true value function using the different basis functions by minimising the following loss function w.r.t the parameters θ:

$$L(\theta) = \frac{1}{|S|} \sum_{s \in S} \left(V(s) - \sum_{i=1}^{d} \theta_i \phi_i(s) \right)^2$$

Results



Average MSE between the approximated and actual value function is consistently **lower when using the node2vec basis functions** of different dimension

Conclusion

When the value function is not a smooth function across the state space, the *node2vec* representation learning algorithm leads to better value function approximation than using the smooth porto-value function as basis function.

Open Questions

- Other graph-based learning algorithms?
- How do RL techniques (e.g., LSPI) perform?
- Can we learn a weighted graph with partially observed signal?
- How much the VF smoothness affect the performance?

[1] S. Mahadevan, "Learning Representation and Control in Markov Decision Processes: New Frontiers," *Foundations and Trends in Machine Learning*, vol. 1, no. 4, pp. 403–565, 2007.
[2] A. Grover and J. Leskovec, "node2vec," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16, pp. 855–864, 2016.