Scaling Up: Solving POMDPs through Value Based Clustering

Yan Virin, Guy Shani, Solomon E. Shimony, Ronen I. Brafman
{virin,shanigu,shimony,brafman}@cs.bgu.ac.il

Abstract

We present here a point-based value iteration algorithm for solving POMDPs, that orders belief state backups smartly based on a clustering of the underlying MDP states. We show our SCVI algorithm to converge faster than state of the art point-based algorithms.

Introduction

A recent promising breakthrough in finding value function approximations for POMDPs is the point-based approach (Pineau, Gordon, & Thrun 2003), where a value function is computed only over a small set of reachable belief-points.

In point-based value iteration for POMDPs, the order by which belief points are updated through the backup operation can dramatically influence the speed of convergence.

In this paper we cluster the underlying MDP states by their optimal MDP value. We then project this clustering onto the POMDP belief space, creating a soft clustering of the belief space. We sort the clusters by their average value and iterate over them in decreasing order selecting belief states to backup from the current cluster only.

Our algorithm, Soft Clustering Value Iteration (SCVI) is demonstrated to rapidly compute good backup orders. Although convergence can be achieved by fewer backups SCVI provides a good tradeoff between the number of backups and the computation time of the backup order.

Background and Related Work

Point Based Algorithms

The family of point-based algorithms for POMDPs uses a finite set B of belief points and compute a value function V represented using α-vectors for belief points in B only, hoping that it would generalize well to other, unobserved belief points. New α-vectors are computed using a point-based backup operation.

$$g^{\alpha}_{a,o}(s) = \sum_{s'} O(a,s',o)tr(s,a,s')\alpha^{i}(s') \quad (1)$$

$$g^{b}_{a} = r_a + \gamma \sum_{o} \text{argmax}_{g^{\alpha}_{a,o}:\alpha\in V}\, b\cdot g^{\alpha}_{a,o} \quad (2)$$

$$backup(b) = \text{argmax}_{g^{b}_{a}:a\in A}\, b\cdot g^{b}_{a} \quad (3)$$

Algorithms of this family differ in two key properties — the selection of the belief set B and the order by which backups are executed over the belief states in B.

For belief state gathering, Spaan and Vlassis (Spaan & Vlassis 2005) suggest to explore the world randomly, adding all observed belief points to B. Shani et al. (Shani, Brafman, & Shimony) take a similar approach, using instead of a random walk a heuristic search attempting to push the agent during the walk towards the rewards of the environment.

When selecting a belief state to backup, Perseus algorithm takes again a random approach, selecting the next belief state at random from B and then removing from B all belief states that were improved using the new α-vector. Once B has emptied new interation of the algorithm is started.

Shani et al. show in their Prioritized Value Iteration (PVI) algorithm that the number of backups until the algorithm converges can be greatly reduced if we choose the next belief state to backup using the Bellman error.

$$e(s) = (\max_{a} R(s,a) + \gamma \sum_{s'\in S} tr(s,a,s')V(s')) - V(s) \quad (4)$$

While the resulting sequence of backups selected by PVI is much shorter than the sequence selected by Perseus, the advantage does not fully manifest into runtime because the computation of the Bellman error is time consuming too.

SCVI - Soft Clustering Value Iteration

Our SCVI algorithm attempts to create backup sequences that are both relatively short and fast to compute, finding a good balance between backup computation time and the time required to compute which belief state to update next.

Our algorithm has the following structure:

1. Use the pre-computed value function for the underlying MDP for a hard clustering of the MDP state space using K-means algorithm over Q-values. The clusters contains states with similar Q-values.

2. Sample a set B of belief states using a random walk over the belief space.

3. Define a soft clustering for the belief states in B using $pr(b \in c) = \sum_{s \in c} b(s)$.

4. Sort the belief states in every cluster c by decreasing probability of belonging to c.

5. Iterate over the clusters in decreasing cluster value $V(c) = \frac{\sum_{s \in c} max_a Q*(s,a)}{|c|}$. For each cluster c we iterate over the belief states in B in decreasing order of $pr(b \in c)$ and update their value using a point-based backup.

Clustering the Belief Points

The Q-values of MDP states typically "trickle down" from states with high rewards to their neighbors. The final value of a state usually depends on its successors that have higher values. Therefore, in MDP context, if optimal Q-values were known prior to value function computation, a good state update order would be by decreasing Q-value. To obtain Q-values one must first compute the value function, so this approach is clearly not feasible for MDPs. However, the idea of sorting MDP states according to their Q-values can be used for accelerating POMDP solvers.

We create a hard clustering of the MDP state space using a distance metric based on the optimal Q-values of the MDP. We first solve the MDP and then execute some clustering algorithm, such as the well known K-means using the difference between their Q-values as the distance metric. We can use the MDP optimal solution as the solution to the MDP is always easier to compute than a solution to the POMDP.

Given any hard clustering of the MDP states we can define a soft clustering of the POMDP belief states. As a belief state b is a vector of state probabilities, we define the probability of b belonging to cluster c to be: $pr(b \in c) = \sum_{s \in c} b(s)$.

We collect a set B of belief states using a random walk in belief space (as done by Perseus (Spaan & Vlassis 2005)), and define the soft clustering over these belief states only.

Cluster and Belief Point Iteration

We iterate over the clusters in decreasing cluster value. Using the MDP Q-values we can define the value of a cluster to be the average of the Q-values of the states that belong to that cluster.

Within the cluster we need to iterate over the inner belief states and update their values. As we use soft clustering, each belief state belongs to all clusters with some probability. Alternatively, each cluster defines a distribution over belief space. We therefore iterate over the belief states in B in decreasing order of probability of belonging to the current cluster, and update them, until some probability threshold has been reached.

Empirical Evaluations

To validate our approach we present here experiments comparing the convergence speed of our SCVI approach to other, state of the art point-based algorithms.

As Table 1 shows SCVI executes more backups than PVI, but much less backups than Perseus. However, as SCVI computes the sequence of backups very rapidly, its overall runtime is less than PVI's.

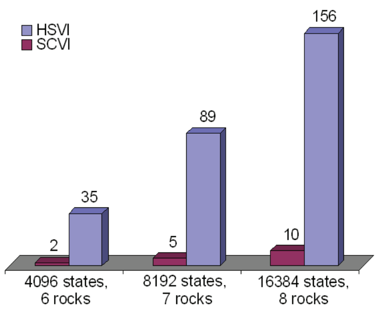

Figure 1: HSVI(left) vs. SCVI(right) convergence time ratio over the RockSample domain with increasing complexity

Method	ADR	$\|V\|$	Time (secs)	# Backups
Rock Sample 5x5				
PVI	19.2	381	36	400
Perseus	19.2	396	303	15352
SCVI (6)	19.2	341	15	750
TagAvoid				
PVI	-6.3	233	137	500
Perseus	-6.3	458	390	22417
SCVI (8)	-6.3	359	127	3662

Table 1: Performance measurements. The numbers in the brackets indicate the number of clusters used by SCVI.

As HSVI (Smith & Simmons 2004), a point-based algorithm, has shown good results in solving POMDPs, we compare SCVI to HSVI on a number of RockSample problems. Figure 1 shows how SCVI scales compared to HSVI over increasing size problems.

Conclusions and Future Work

Our SCVI algorithm balances between the need to select the order of belief state backups and the required computation effort for computing this order. We show it to produce good backup sequences rapidly, thus outperforming three state of the art point-based algorithms — Perseus, PVI and HSVI.

Our clustering mechanism can be viewed as an abstraction method, and future research should look into creating an abstract POMDP model through a clustering over the underlying MDP. Our prioritization technique for both belief states and clusters is static. Considering a dynamic way to update these priorities may result in reduced number of backups and thus, more rapid convergence.

References

Pineau, J.; Gordon, G.; and Thrun, S. 2003. Point-based value iteration: An anytime algorithm for POMDPs. In *IJCAI*.

Shani, G.; Brafman, R. I.; and Shimony, S. E. Prioritizing point-based pomdp solvers. In *ECML*.

Smith, T., and Simmons, R. 2004. Heuristic search value iteration for pomdps. In *UAI 2004*.

Spaan, M. T. J., and Vlassis, N. 2005. Perseus: Randomized point-based value iteration for POMDPs. In *AIR 24*.