A Hierarchical Structure For Finite Horizon Dynamic Programming Problems

Chang Zhang, John S. Baras *

December 14, 2000

Abstract

In dynamic programming (Markov decision) problems, hierarchical structure (aggregation) is usually used to simplify computation. Most research on aggregation of Markov decision problems is limited to the infinite horizon case, which has good tracking ability. However, in real life, finite horizon stochastic shortest path problems are often encountered. In this paper, we propose a hierarchical structure to solve finite horizon stochastic shortest path problems in parallel. In general, the approach reduces the time complexity of the original problem to a logarithm level, which has significant practical meaning.

1 Introduction

In Markov decision problems, hierarchical structure (aggregation) is usually used to simplify computation. Most research is limited in infinite horizon Markov decision problems, which have good tracking ability [2]-[8]. However, in real life, all problems are finite horizon and we often have to solve finite horizon stochastic shortest path problems. Although there exist some research about aggregation of finite horizon case, they are limited to computing acyclic deterministic shortest path problems either with approximation [1] or accurately[10]. In this paper, we propose a hierarchical structure to solve general finite horizon stochastic shortest path problem. The contributions of the paper are twofold. First, it introduces a hierarchical structure to solve the acyclic stochastic shortest path problem in parallel. Secondly, it extends to the general finite horizon stochastic shortest path problems by the help of reachable sets. The hierarchical structure can speed up computation significantly. Another advantage of hierarchical structure lies in the scalability of the system. If we have known the result of K stages and want to calculate the result for more stages, then the hierarchical structure will utilize the original information without computing all the stages again. Therefore, this paper has important practical meanings.

Consider a discrete-time finite state system S. Let state 0 being the destination (termination state). Assume that there is a finite decision space U, and for each state, the state transition probabilities are given and independent of the stage. Furthermore, given a policy $\pi = \{\mu_0, \mu_1, ..., \mu_K\}$, the sequence of states becomes a Markov chain with transition probabilities

$$P(x_{k+1} = j | x_k = i, ..., x_0 = i_0, \mu_k(i), ..., \mu_0(i_0)) = P_{i,j}(\mu_k(i)) \quad (1)$$

*This work was supported by the Center for Satellite and Hybrid Communication Networks, under NASA cooperative agreement NCC3-528

Given an initial state $i \in S$ and a bounded positive integer K, the problem is to find an optimal policy, such that

$$J_K^*(i) = \min_{\pi} E[\sum_{k=0}^{K} \alpha^k g(i_k, \mu_k(i_k), i_{k+1}) | i_0 = i] \tag{2}$$

where α is the discount factor ($0 < \alpha \leq 1$) and $g(i_k, \mu_k(i_k), i_{k+1})$ is the transition cost. If in stage $k < K$, the sequence reaches the destination, then it will stay there with 0 cost. If at stage K, the sequence does not reach the termination state, it will stop at the current stage too.

The paper is arranged as follows. In section 2, we first look at the simple case of acyclic stochastic shortest path problems and introduce the idea of hierarchical structure. In section 3, we extend it into general finite horizon Markov decision processes. The hierarchical structure can also be used to do approximate computation when exact computation is impractical, which is given in section 4. Finally, conclusions are given in section 5.

2 Acyclic Finite Horizon Stochastic Shortest Path Problems

Before we turn to the general finite horizon stochastic shortest path problems in equation (2), let us look at a simpler problem: find an expected shortest path for an acyclic stochastic shortest path problem.

In practice, we usually have known that some policies are better than other policies. Therefore, in application of dynamic programming, we want to test these policies first. Assume that a policy $\pi = \{\mu_0, \mu_1, ..., \mu_K\}$ is given, we want to calculate the expected cost. Dynamic programming will calculate the expected cost for a state i_k in stage k in serial:

$$\begin{aligned} J^{\pi}(i_k) &= \sum_{i_{k+1}} P_{i_k, i_{k+1}} [g(i_k, i_{k+1}) + \alpha J^{\pi}(i_{k+1})] \\ &= G(i_k) + \alpha \sum_{i_{k+1}} P_{i_k, i_{k+1}} J^{\pi}(i_{k+1}) \end{aligned} \tag{3}$$

(Note that in equation (3), $P_{i_k, i_{k+1}}$ and $g(i_k, i_{k+1})$ depend on the particular policy)

By expanding to the next step, equation (3) is changed into:

$$\begin{aligned} J^{\pi}(i_k) &= G(i_k) + \alpha \sum_{i_{k+1}} P_{i_k, i_{k+1}} \sum_{i_{k+2}} P_{i_{k+1}, i_{k+2}} (g(i_{k+1}, i_{k+2}) + \alpha J^{\pi}(i_{k+2})) \\ &= G(i_k) + \alpha \sum_{i_{k+1}} P_{i_k, i_{k+1}} \sum_{i_{k+2}} P_{i_{k+1}, i_{k+2}} g(i_{k+1}, i_{k+2}) \\ &\quad + \alpha^2 \sum_{i_{k+1}} P_{i_k, i_{k+1}} \sum_{i_{k+2}} P_{i_{k+1}, i_{k+2}} J^{\pi}(i_{k+2}) \\ &= G^{(1)}(i_k) + \sum_{i_{k+2}} P^{(1)}_{i_k, i_{k+2}} J^{\pi}(i_{k+2}) \end{aligned} \tag{4}$$

where

$$\begin{aligned} G^{(1)}(i_k) &= G(i_k) + \alpha \sum_{i_{k+1}} P_{i_k, i_{k+1}} \sum_{i_{k+2}} P_{i_{k+1}, i_{k+2}} g(i_{k+1}, i_{k+2}) \\ &= G(i_k) + \alpha \sum_{i_{k+1}} P_{i_k, i_{k+1}} G(i_{k+1}) \end{aligned} \tag{5}$$

and

$$P^{(1)}_{i_k, i_{k+2}} = \alpha^2 \sum_{i_{k+1}} P_{i_k, i_{k+1}} P_{i_{k+1}, i_{k+2}} \tag{6}$$

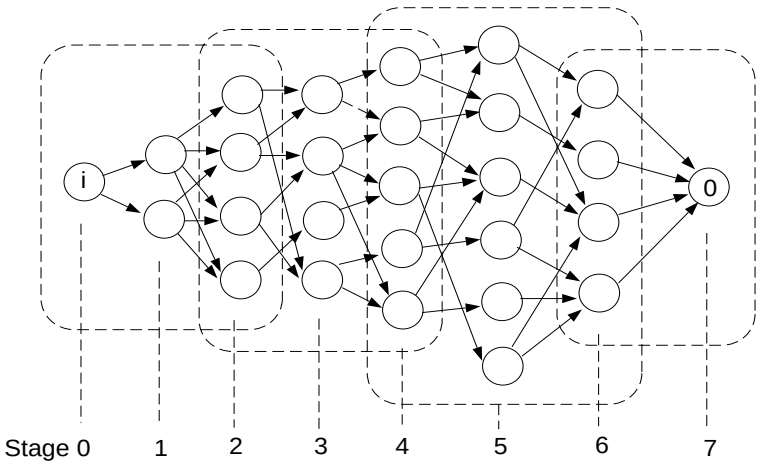

Figure 1: An acyclic stochastic Markov decision process

Similarly, we can express $J^\pi(i_k)$ to higher levels for $j = 2, ...$ and $i_k + 2^j < K$

$$J^\pi(i_k) = G^{(j)}(i_k) + \sum_{i_{k+2^j}} P^{(j)}_{i_k, i_{k+2^j}} J^\pi(i_{k+2^j}) \tag{7}$$

where

$$G^{(j)}(i_k) = G^{(j-1)}(i_k) + \alpha \sum_{i_{k+2^{j-1}}} P^{(j-1)}_{i_k, i_{k+2^{j-1}}} G(i_{k+2^{j-1}}) \tag{8}$$

and

$$P^{(j)}_{i_k, i_{k+2^j}} = \alpha^2 \sum_{i_{k+2^{j-1}}} P^{(j-1)}_{i_k, i_{k+2^{j-1}}} P^{(j-1)}_{i_{k+2^{j-1}}, i_{k+2^j}} \tag{9}$$

Therefore, we can aggregate the states by the stages and compute the expected cost in each cluster separately. The aggregation is as follows: In level 1, the states in stage 0, 1 and 2 are aggregated together as cluster 0, the states in stage 2, 3 and 4 are aggregated together as cluster 1, the states in stage 4, 5 and 6 are aggregated together as cluster 3, ..., until the final stage is reached. In the last cluster, there are 2 or 3 stages which depends on the stage number of the problem. In each cluster, we compute $G^{(1)}(i_k)$ and $P^{(1)}_{i_k, i_{k+2}}$ in parallel. In the last cluster, $J^\pi(i_{K-2})$ is computed if the number of stages in the cluster is 3; otherwise, $J^\pi(i_{K-1})$ is computed. In level 2, the clusters 0, 1 and 2 are aggregated, the clusters 2, 3 and 4 are aggregated, ..., until the last cluster in level 1 is reached. We calculate $G^{(2)}(i_k)$ and $P^{(2)}_{i_k, i_{k+4}}$ in parallel according to equation (8-9) and the result from level 1. The aggregation in higher level is made similarly, until finally there is only one cluster left. In the last cluster at each level, the number of stages can be 2 or 3. The final expected cost for the original problem is obtained in the top level.

The procedure can be shown by the following example in Figure 1. Suppose for a given policy, the possible transitions are shown as in Figure 1. Then at level 1, the aggregation is shown in Figure 2 (a). Level 2 aggregation is shown in Figure 2 (b). The procedure can also be shown by Figure 3, where the circles in each level stand for the clusters. In level 0, there are 4 clusters, in level 1, there are 2 clusters and there is one cluster in level 2.

Assume that the time complexity of the original problem to be O(K), with O(1) per stage. Then the time complexity of the new problem will be $O(log(K))$.

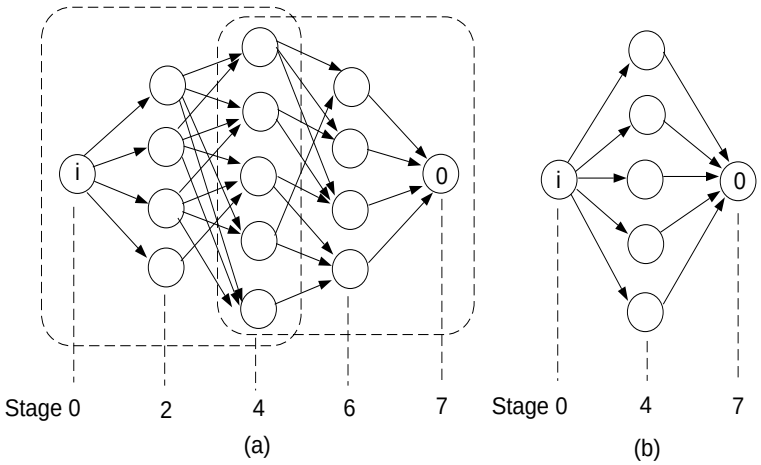

Figure 2: (a) Aggregated system for the stochastic process in level 1. (b) Aggregated system for the stochastic process in level 2.

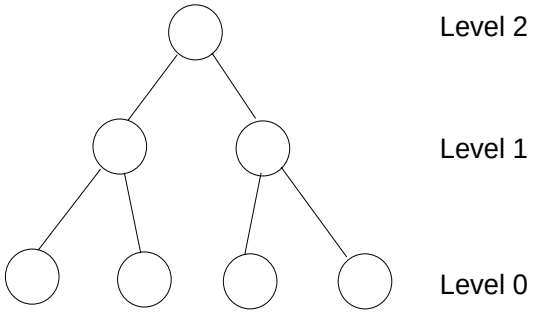

Figure 3: A hierarchical structure for the example

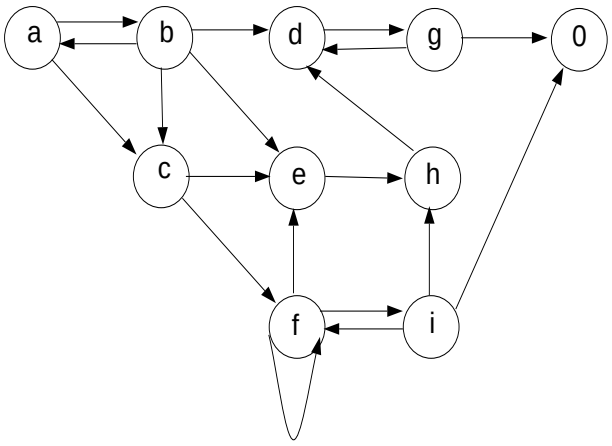

Figure 4: A general stochastic path example

3 General Finite Horizon Stochastic Shortest Path Problems

Now let us turn to the general finite horizon stochastic shortest path problems (equation 2). First let us define the concept of *reachable set* S_k^π of a finite horizon Markov decision problem under a policy π. It is defined as:

$$S_k^\pi = \{i_k \in S | \Pi_{l=1}^k P_{i_{l-1},i_l}^\pi > 0\}, k = 1, 2, ..., K \tag{10}$$

The general finite horizon stochastic shortest path problems can be solved with the help of reachable sets. Because from stage k to stage k+1, the reachable set changes from S_k^π to S_{k+1}^π, we can divide the stages according to the reachable sets. Therefore, even the stochastic shortest problem is acyclic, we can still apply the hierarchical structure based on the reachable sets.

One problem arises when the termination state can be reached before stage K. We introduce the dummy state of the termination state with transition cost 0 to itself. Then we can expand the termination state to stage K. If the termination state can not be reached, we simply stop at the reachable set S_K. During parallel computing, if the cluster including the initial state reaches the termination state with expected minimum cost, then the clusters behind it can be discarded and the problem is solved.

Example. Change a general stochastic path problem into an acyclic graph. Given a policy, the possible stochastic path is shown in Figure 4, where 0 is the termination state. Suppose we want to calculate the expected cost for K=6. Then the corresponding acyclic graph based on the reachable sets is shown in Figure 5. Therefore, the hierarchical structure can be applied. First level aggregation is shown in this figure.

Again, suppose the time complexity of the original stochastic shortest problem to be K, then the time complexity of the hierarchical structure will be $O(log(K))$.

4 Approximate Computation with the Hierarchical Structure

In previous discussion, we assume that some knowledge of the optimal policies are known so that we can do policy evaluation for a limited number of policies. If we don't have any such information, then theoretically, we still can apply the hierarchical structure. But this time we have to calculate the expected cost for each policy. If the dimension of the state space is huge and there are multiple choices of actions in each state,

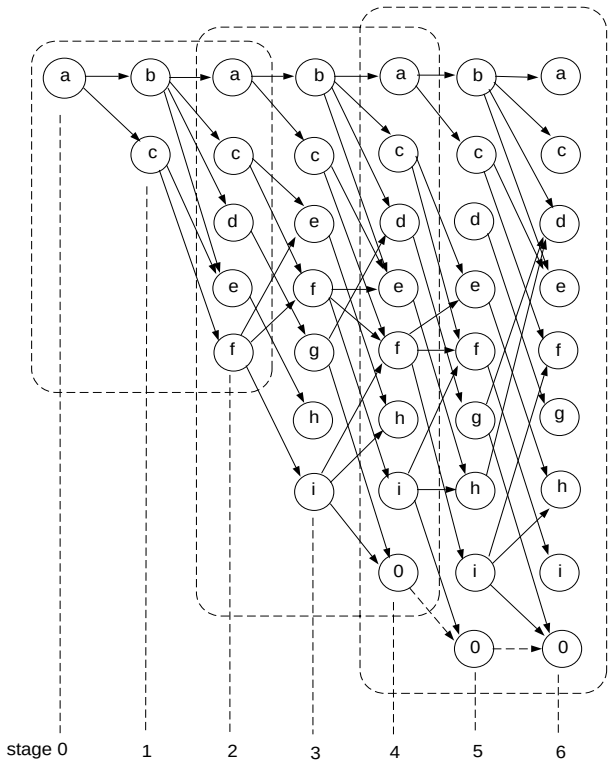

Figure 5: The corresponding graph of reachable sets for the example

it will require large number of memories and become impractical. In this case, we have to employ the hierarchical structure to find an approximate solution.

The hierarchical structure is obtained in the same approach as previous chapters with the help of reachable sets. The difference lies in that, in each level, we calculate the stochastic shortest path directly for each cluster. In level 1, we find all the expected shortest paths from the states in each beginning stage to the ending stage in each cluster and compute corresponding $G^{(1)}(i_k)$ and $P^{(1)}_{i_k,i_{k+2}}$ for each optimal sub-policies. In each cluster, the number of optimal sub-policies is equal to the number of states in the beginning stage. In level 2, we find the expected shortest paths from the states in each beginning stage to the ending stage in each cluster using the result from level 1, and compute corresponding $G^{(2)}(i_k)$ and $P^{(2)}_{i_k,i_{k+4}}$ for each optimal sub-policies. The procedure is repeated until the top level is reached.

This approach can be thought as a greedy algorithm starting from the initial state. Instead of looking one step ahead, the algorithm looks two-step ahead and choose the best strategy it can achieve after moving two steps. In the best case, this approach can achieve the optimal expected shortest path problem. In the worst case, this approach can give expected path better than or equal to the expected path of the single-step greedy algorithm.

When the shortest path problem is deterministic, it is easy to show that the result obtained by the hierarchical structure is optimal. When the graph is acyclic, our hierarchical structure is the same as the one in [10]. However, our result also applies to cyclic graphs with the help of reachable sets.

5 Conclusions

To summarize, we introduce a hierarchical structure for general finite horizon stochastic shortest path problems, which have not been analyzed before. It is solved with the help of reachable sets. For fixed policy, it can calculate the expected cost fast and accurately. It can also be used to approximate the original problem and simplify computation. The structure can speed up computation significantly since each cluster calculates its own stochastic shortest path in parallel. Therefore, the approach in this paper has significant practical meaning.

In fact, the hierarchical structure can be thought as a stage aggregation method. In each level, we aggregate the states in the adjacent 3 stages together and change them into 2 stages. We can also aggregate more stages and therefore more states in each cluster instead of 3 stages each time. However, the fastest way is to aggregate 3 stages because we can employ the hierarchical structure to the maximum extent.

When the dimension of the state space is extremely large and direct computation of the shortest path problem in each cluster is impossible, state aggregation technique in the same stage or value function approximation technique can be combined with our hierarchical structure.

References

[1] J. C. Bean, J. R. Birge and R. L. Smith, "Aggregation in dynamic programming", Operations Research, vol. 35, pp. 215-220, 1987.

[2] R. G. Phillips and P. V. Kokotovic, "A singular perturbation approach to modeling and control of Markov chains", IEEE Trans. Automatic Control, vol. AC-26, no. 5, pp. 1087-1094, 1981.

[3] J. N. Tsitsiklis and B. V. Roy, "Feature-based methods for large scale dynamic programming", Machine Learning, vol. 22, pp. 59-94, 1996.

[4] C. Chow, J. N. Tsitsiklis, "An optimal one-way multigrid algorithm for discrete-time stochastic control", IEEE Trans. Automatic Control, vol. 36, no. 8, pp. 898-914, 1991.

[5] D. P. Bertsekas and D. A. Castanon, "Adaptive aggregation methods for infinite horizon dynamic programming", IEEE Trans. Automatic Control, vol. 34, no. 6, pp. 589-598, 1989.

[6] R. W. Aldhaheri and H. K. Khalil, "Aggregation of the policy iteration method for nearly completely decomposable Markov chains", IEEE Trans. Automatic Control, vol. 36, no. 2, pp. 178-187, 1991.

[7] F. Delebecque and J. Quadrat, "Optimal control of Markov chains admitting strong and weak interaction", Automatica, vol. 17, no. 2, pp. 281-296, 1981.

[8] R. Mdndelssohn, "An iterative aggregation procedure for Markov decision processes", Operations Research, vol. 30, no. 1, pp. 62-73, 1981.

[9] S. Axsater, "State aggregation in dynamic programming - An application to scheduling of independent jobs on parallel processes", Operations Research Letters, vol. 2, no. 4, pp. 171-176, 1983.

[10] W. K. Tsai, G. M. Huang and J. K. Antonio, "Fast parallel hierarchical aggregation/disaggregation algorithm for multistage optimization problems and shortest path problems", Journal of Parallel and Distributed Computing, pp. 1789-1794, 1991

"The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government."