Multigrid Q-Learning

Charles W. Anderson
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
anderson@cs.colostate.edu

Stewart G. Crawford-Hines
Department of Computer Science
Colorado State University
Fort Collins, CO 80523
sgcraw@cs.colostate.edu

Category: Control, Navigation, and Planning

Keywords: Q-Learning, reinforcement learning, multigrid

Preference: oral

Abstract

Reinforcement learning scales poorly when reinforcements are delayed. The problem of propagating information from delayed reinforcements to the states and actions that have an effect the reinforcement is similar to the problem of propagating information in a discretized boundary value problem. Multigrid methods have been shown to decrease the number of updates required to solve boundary value problems. Here we extend Q-Learning by casting it as a multigrid method and show a reduction in updates required to reach a given error level in the Q-function for a simple, 1-d Markov decision task.

1 INTRODUCTION

Current reinforcement learning algorithms scale poorly to large problems for a number of reasons, such as the difficulty of searching high-dimensional state spaces, the temporal credit assignment problem due delayed reinforcement, and the structural credit assignment problem that results from parameter interactions in a function approximator. Here we focus on the temporal credit assignment problem that is present when many state transitions occur between state-action pairs and the external reinforcement that they affect. Many steps are required of iterative reinforcement-learning algorithms to propagate the influence of delayed reinforcement to all states and actions that have an effect on that reinforcement.

This situation is also present in the iterative algorithms for solving boundary value problems, such as determining the steady-state temperature distribution in a long uniform rod (Briggs, 1987). Many steps are needed to propagate the effect of the boundary conditions to interior points of the domain over which the problem is defined. Multigrid methods have been developed for the solution of boundary value problems as a way to decrease the number of iterations needed in a relaxation approach (McCormick, 1992; Rüde, 1993). The problem is transformed to equivalent problems defined over the domain discretized at different resolutions.

Here we apply the multigrid approach to the Q-Learning algorithm (Watkins, 1989). In the remaining sections, we recast Q-Learning as a multigrid method and describe results of applying the combined approach on a simple Markov decision task. The results show that the multigrid approach reduces by half the number of updates required to reach a particular error level.

2 MULTIGRID-Q

The primary step in extending Q-Learning to multiple levels of resolution is to define how to transform the Q-Learning problem from one level to another. Let us start with an expression of Watkin's one-step Q-Learning algorithm:

$$Q_{k+1} = Q_k + \alpha \left(R(x_k, a_k) + \gamma \max_{a' \in A} Q_k(y_k, a') - Q_k(x_k, a_k) \right),$$

where $Q_k(x_k, a_k)$ is the value of the Q function at the k^{th} iteration for action a_k taken in state x_k, α is a constant, $R(x_k, a_k)$ is the external reinforcement received for action a_k taken in state x_k, γ is a constant discount factor, and y_k is the state that results from taking action a_k in state x_k. There are a finite set of states and actions, S and A, respectively, i.e., $x_k \in S$, $y_k \in S$, and $a_k \in A$.

To define this algorithm for different levels of resolution, we must redefine R, γ, and the set of actions A, and define a procedure for modifying the Q values converged on at one level to Q values for another level. Let δ represent the current level, typically δ is the spacing of the grid, relative to the finest resolution. Using superscripts to indicate the current level, we can rewrite the Q-Learning algorithm as:

$$Q^\delta_{k+1} = Q^\delta_k + \alpha \left(R^\delta(x_k, a_k) + \gamma^\delta \max_{a' \in A^\delta} Q^\delta_k(y_k, a') - Q^\delta_k(x_k, a_k) \right),$$

where $a_k \in A^\delta$ and $x_k, y_k \in S^\delta$.

An early form of the multigrid algorithm followed a coarse-to-fine schedule of relaxation at the various levels. To shift to the next finer level, the coarse solution is interpolated to obtain values at grid points halfway between points whose values were modified at the coarser level. The coarse level solution and its interpolation is taken as a starting point for the solution of the problem at the finer level. This

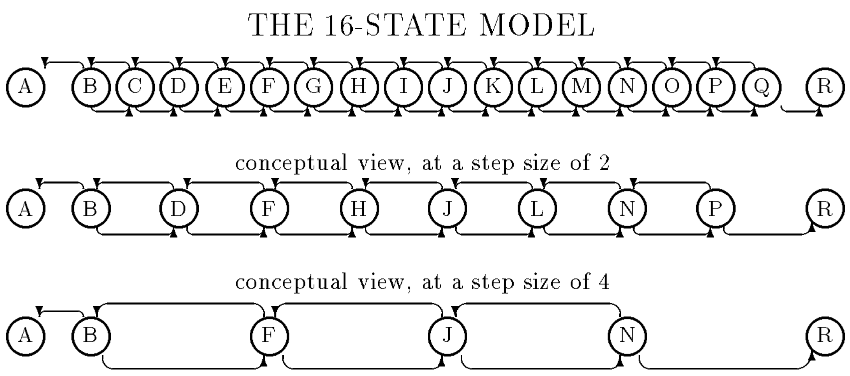

Figure 1: The State Space at different resolutions.

solution-interpolation process continues until the problem is solved at the finest level of resolution. Other schedules for moving among the levels have been tried. Another variation, called *adaptive* multigrid, is to develop different resolutions for different parts of the grid.

The notion of multilevels of resolution is not new to the reinforcement learning literature. Dayan and Hinton (1993) developed a hierarchical approach to reinforcement learning and Moore (1991) adaptively increases the resolution of the state space near experience trajectories. However, the only prior explicit contact with the multigrid literature is the work of Chow and Tsitsiklis (1988) who show that a multigrid implementation of the successive approximation algorithm for dynamic programming results in a near optimal computational complexity.

3 METHOD

To study empirically the viability of these multigrid methods for Q-learning, we ran a series of progressively more detailed simulations for a 1-dimensional random walk. The temporal difference (TD) methods of learning the Q-function converge to a stable set of values over a series of random walks. With this known stable set in hand, we studied the speed of convergence to this stable state from a variety of different initial conditions, expressed as known error function added to that final, stable state.

The linear state space of these simulations has 16 non-absorbing states, and end states off to both the left and right. The state space is diagrammed in Figure 1. If the random walk ends on the right-side in state R, there is a reward of 1; if it ends on the left-side in state A, there is no reward.

The Q function to be learned represents a discounted value of the best possible future reward for a given move. We used a discount factor of .9 throughout these simulations. Thus, for example, in state P, by moving *right* the best possible future is to move to state R in one more step; this implies $Q(state-P, move-right)$ is

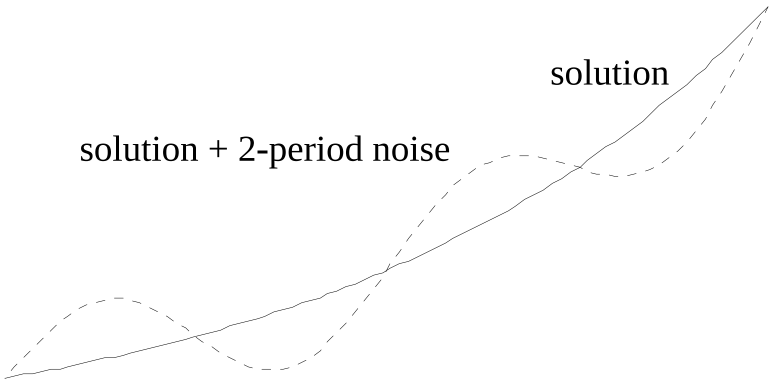

Figure 2: Creating a *2-period noisy* initial distribution

.9, which is the reward of 1 discounted by .9 for the one further *right* required to reach that reward. It should be clear that the final Q values for *right* moves form a series of powers of .9: $Q(\textit{state-Q}, \textit{move-right}) = 1$, $Q(\textit{state-P}, \textit{move-right}) = 0.9$, $Q(\textit{state-O}, \textit{move-right}) = 0.81$, etc.

Temporal difference methods provide a set of approximations to the Q function as random walks are made through the state space. A *trajectory* is often used in the same sense as *random walk* in our model: it implies starting at some random initial state, and taking a series of left and right moves until the walk ends at either final state A or R. The probabilities of moving left or right from a given state are determined by exponentially normalizing the Q values for the corresponding left or right move. A *run* is a series of walks or trajectories which starts with some initially predefined set of Q values, and continues until the Q function iterations converge to to within some specified error tolerance. For this 16-state model, a run comprised of 50 trajectories will visit approximately 1000 states. Each visit to a new state implies one TD update to the approximate Q function. In all the following simulations, we measured runs by the number of updates involved, since this is both proportional to the computational complexity of the approximation method, and representative of amount of "experience" acquired through the series of trajectories through the state space.

In drawing analogies to the previous multigrid work, some early motivating examples demonstrated how iterated approximate solutions converged at different speeds depending on the frequency distribution of the noise superimposed on the stable solution. To study this effect, we created various initial distributions, each with a characteristic error frequency added to the stable solution. Figure 2 illustrates a sinusoidal noise added onto a geometric stable solution, the solution which is characteristic of the Q functions for these 1-D random walks. Each of the peaks is equally distant from the solution curve when measured vertically, though this may not be visually obvious due to the increasing slope of the solution curve. We denote this particular example as *2-period* noise. Through the course of this work, we studied sinusoidal and square-wave noise of *1-period*, *2-periods*, *4-periods*, and *8-periods*, as well as a *constant-bias* added to all the states. All the noise functions

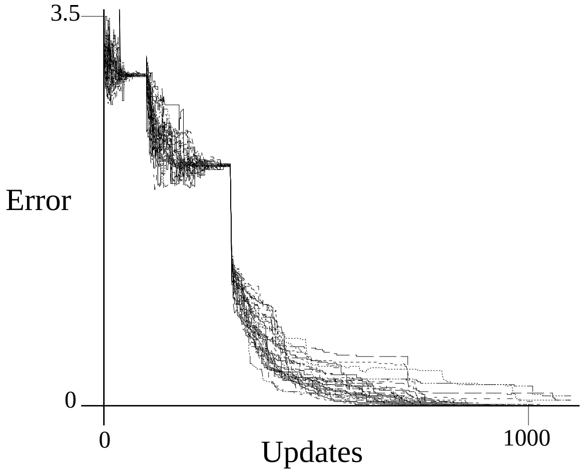

Figure 3: *Errors in Q over 33 multigrid approximation runs*

were calculated such that the sums of their absolute error over all 16 states were equal.

In implementing a multigrid perspective on this random walk, conceptually we just took larger steps on the walk, as illustrated in Figure 1. A step-size coarseness of 4 in our 16-state model is the equivalent of walking in a 4-state model consisting of only states **BFJN**. To converge to the appropriate Q values in this reduced state model, the effective discount factor was raised to the fourth power, since each large step right is equivalent to four individually discounted single steps right. Similarly, a step-size coarseness of 2 is equivalent to walking only the states **BDFHJLNP** and squaring the discount factor. This approach to restructuring the state space is in direct analogy to the multigrid approach to solving differential equations at different mesh sizes across the desired solution space.

Our multigrid approach to approximating Q in this 16-state space proceeds in this manner:

1. randomly walk with a step size of 4 until the Q values for the 4 states **BFJN** have roughly converged;
2. linearly interpolate the middle states **DHLP** from **BFJN**;
3. randomly walk with a step size of 2 until the Q values for the 8 states **BDFHJLNP** have roughly converged;
4. linearly interpolate the middle states **CEGIKMOQ**;
5. continue walking at a step size of 1 until convergence criteria is met.

This would be called a *4/2/1 schedule* for the multigrid simulation. Figure 3 displays in composite the total absolute error over all Q values for 33 runs of a 4/2/1 schedule. The simulations ran for 100 updates at a step size of 4, then for 200 updates at a

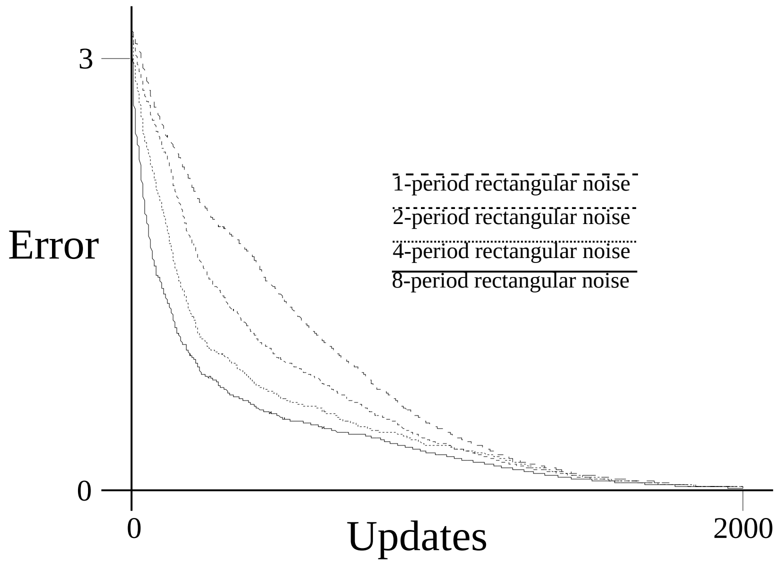

Figure 4: *Error by update for varying Initial Error frequencies*

step size of 2, then until convergence at a step size of 1. The interpolation points at 100 and 300 updates are obvious in the figure.

The standard single-step TD approximation method can be improved by finding the optimum learning rate α for the basic TD method, and by implementing the TD(λ) approximation method. On top of both of these, the multigrid approach can be tuned by selection of the parameters for how many updates are performed at a given step size. As an example of visually tuning the multigrid strategy in this way, looking back on Figure 3 it is apparent that there is minimal improvement in the latter size 4 steps and size 2 steps, therefore in both cases the transition to the smaller step size should have been made earlier, bringing the overall average curve down faster.

4 RESULTS

Our initial results showed an effect noted in previous multigrid work. Figure 4 shows the decrease in error as the Q function is "learned" more closely over the series of updates for four cases. These four cases represent differing initial distributions, where a *1-period*, *2-period*, *4-period*, and *8-period* square wave error was added to the Q solution. The Q function is learned progressively quicker, evidenced by a steeper decrease in error, as the initial Q values have progressively higher frequency error components to them. This is a key observation motivating multigrid methods, since a coarser view of the state space makes the error frequency components appear higher, and thus "learned" more efficiently.

This simple first result is confounded by other optimizations which can be made to

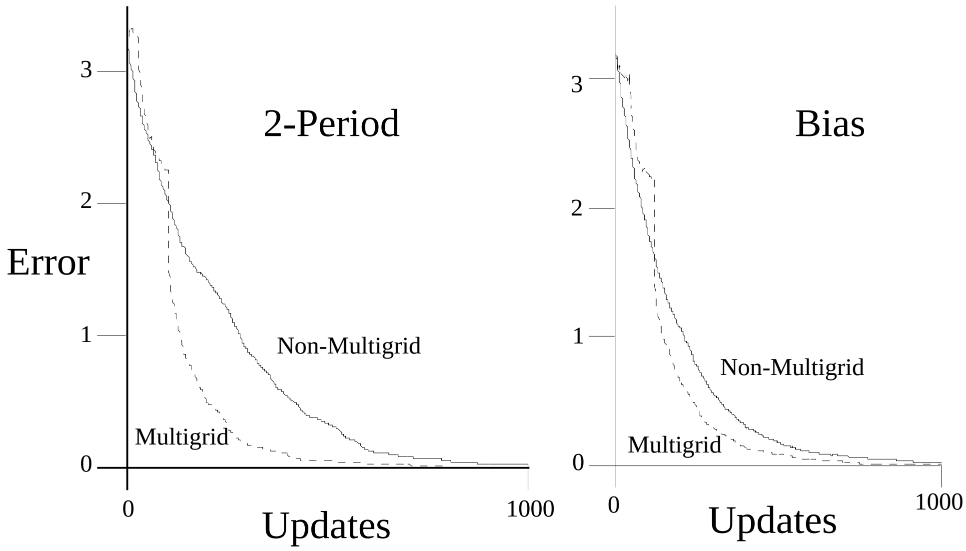

Figure 5: *caption*

the basic single step approach. The standard single-step TD approximation method can be improved by finding the optimum learning rate α for the basic TD method, and by implementing the TD(λ) approximation method. The parameters α and λ will vary depending on the frequency of the error component.

After much exploratory work with the various parameters available to us, we averaged a series of 33 runs at the best possible α and λ values for the "constant bias" and the "2-period error" initial distributions. The convergence speed was quantified by estimating the area under the error decay curve. With these best settings, we made a set of multigrid runs for comparison against the best of what the standard TD methods could do. The interpolation points were visually placed as previously discussed. The "2-period error" initial distribution showed a dramatic multigrid improvement of 40%; the "constant bias" initial distribution still showed a multigrid improvement, though of only a more modest 15%. The 33-run average curves are shown in Figure 5.

With these results in mind, we theorized that in a larger state space, which is harder to learn overall, the multigrid methods would have the most flexibility for improved learning. So a confirmatory simulation was made in a 32-state space for a series of runs on an "4-period error" initial distribution and a near-optimal single-step α and λ. The multigrid method with an 8/4/2/1 schedule outperformed the best single-step TD method by 50%.

5 DISCUSSION

The modification of Q-Learning to a multigrid form that followed a coarse-to-fine schedule considerably reduced the number of updates required to reach a given error level. However, the formulation of Multigrid-Q described here is based on a number of assumptions that somewhat limit its practicality.

The abstraction of the state space to multiple levels of resolution assumes knowledge of the topology of the state space. We do not assume knowledge of the state transition probabilities, since the algorithm is based on the stochastic approximation form of Q-Value Iteration Singh (1993). We also assumed we could redefine the action-dependent, state transition probabilities for various levels of resolution. In a real environment, observables are typically sensed at fixed intervals of time. To use this experience at multiple levels of resolution, the observed trajectories could be cached and reduced to the resolution required at each level.

Extensions of the basic Multigrid-Q algorithm presented here include the following. Variations of the coarse-to-fine schedule, such as the V and W-cycles used in solving boundary-value problems (Briggs, 1987), might result in further reductions of updates. An adaptive scheme could be developed whereby the resolution is varied for different states. This adaptive multigrid approach is strongly related to the variable resolution methods studied by Moore (Moore, 1991; Moore and Atkeson, tted).

Acknowledgements

This research was funded by the National Science Foundation through grant IRI-9212191.

References

Briggs, W. L. (1987). *A Multigrid Tutorial.* SIAM, Philadelphia, Pennsylvania.

Chow, C.-S., & Tsitsiklis, J. N. (June 1988). An optimal multigrid algorithm for continuous state discrete time stochastic control. Technical Report OR 181-38, MIT.

Dayan, P., & Hinton, G. E. (1993). Feudal reinforcement learning. In Hanson, S. J., Cowan, J. D., & Giles, C. L., editors, *Advances in Neural Information Processing Systems 5*, pages 271–278. Morgan Kaufmann, San Mateo, CA.

McCormick, S. F. (1992). *Multilevel Projection Methods for Partial Differential Equations.* SIAM, Philadelphia, Pennsylvania.

Moore, A. W., & Atkeson, C. G. (submitted). The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning.*

Moore, A. W. (1991). Variable resolution dynamic programming: Efficiently learning action maps in multivariable real-valued state-spaces. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 333–337, San Mateo, CA. Morgan Kaufmann.

Rüde, U. (1993). *Mathematical and Computational Techniques for Multilevel Adaptive Methods*, volume 13 of *Frontiers in Applied Mathematics.* SIAM, Philadelphia, Pennsylvania.

Singh, S. P. (1993). Learning to solve markovian decision processes. Technical Report CMPSCI 93-77, University of Massachusetts, Amherst, MA.

Watkins, C. (1989). *Learning with Delayed Rewards.* PhD thesis, Cambridge University Psychology Department.