Multigrid Methods for Policy Evaluation and Reinforcement Learning

Omer Ziv and Nahum Shimkin, *Senior Member, IEEE*

Abstract – **We introduce a new class of multigrid temporal-difference learning algorithms for speeding up the estimation of the value function related to a stationary policy, within the context of discounted cost Markov Decision Processes with linear functional approximation. The proposed scheme builds on the multigrid framework which is used in numerical analysis to enhance the iterative solution of linear equations. We first apply the multigrid approach to policy evaluation in the known model case. We then extend this approach to the learning case, and propose a scheme in which the basic TD(λ) learning algorithm is applied at various resolution scales. The efficacy of the proposed algorithms is demonstrated through simulation experiments.**

1. INTRODUCTION

Reinforcement Learning (RL) [6,13] encompasses a set of methods and algorithms for learning an optimal control policy in a stochastic dynamic environment, modeled as a Markov Decision Process (MDP). One of the basic algorithms in RL is TD(λ). This family of algorithms is intended for learning the value function of a fixed stationary policy, and may be considered an on-line version of the value iteration method of dynamic programming (DP). For large state spaces, the value function typically needs to be represented by some parametric function approximator in order to provide size tractability and learning generalization. Convergence of TD(λ) with linear function approximation was proven in [15]. Recently, two new algorithms of a least-squares nature were introduced, namely LSTD(λ) [3,16] and λ-LSPE [9,4]. They offer considerably faster convergence (in terms of data samples) at the expense of an increase from $O(K)$ to $O(K^2)$ in computational complexity per sample and in memory resources, where K is the number of variables to be learned. Consequently, for complicated applications, K may be too large for these algorithms to be feasible. It is therefore of interest to study intermediate algorithms with $O(K)$

A common approach to speed up convergence of dynamic programming algorithms is state aggregation (see [1,5,10] and complexity that enhance the converge rate of TD(λ).references within), while various hierarchical approaches, often heuristic, have been used in the RL context (see [2] for a recent survey). In this paper we apply the multigrid framework to the policy evaluation problem. Multigrid is a family of multiscale numerical methods for solving large sparse system of linear equations [14]. Geometric multigrid relies on a hierarchy of discretizations of a continuous problem on regular meshes. Algebraic multigrid (AMG) is a more recent variant that starts with the algebraic equation $\mathbf{Ax} = \mathbf{b}$ and constructs all coarser levels fully automatically during a setup phase, based only on algebraic information contained in $\mathbf{A}$ [11,7]. This makes AMG attractive as a "black-box" solver of numerical problems, and a promising technique for constructing hierarchies in dynamic programming problems even when the state space does not possess an obvious geometric structure. In this paper we apply the multigrid approach to speed up iterative and on-line policy evaluation. We focus on AMG, although the basic scheme is applicable irrespec-

Submitted to the 2005 International Symposium on Intelligent Control; January 9, 2005.

O. Ziv is with the Department of Electrical Engineering, Technion, Haifa 32000, Israel.

N. Shimkin is with the Department of Electrical Engineering, Technion, Haifa 32000, Israel (phone: 972-4-8294734; fax: 972-4-8295757; e-mail: shimkin@ee.technion.ac.il)

tively of the origin of the multigrid structure. This results in a Multigrid-TD(λ) algorithm of comparable computational complexity to TD(λ), with faster convergence rate.

The necessary background on multigrid is given in the next section, while section 3 presents the TD(λ) algorithm. The straightforward application of multigrid to value iteration in the know model case is discussed in section 4, leading to the multigrid TD(λ) algorithm in section 5. Section 6 presents some experimental results, followed by concluding remarks.

2. ALGEBRAIC MULTIGRID

The objective of multigrid is to efficiently solve a sparse system of linear equations, namely $\mathbf{Ax} = \mathbf{b}$. The basic principle is to use two complementary procedures: one nullifies fast-to-converge, oscillatory ("high frequency") error components using standard iteration such as Richardson (value iteration) or Gauss-Seidel; the other eliminates smooth ("low frequency") error components that are slow-to-converge under iteration, by applying an additive correction. The correction is calculated by mapping the smooth error at the fine level to a coarser level, solving the mapped problem at the coarse level, and interpolating the error correction back to the fine level. Applying this two-level scheme recursively to solve the mapped problem results in a multilevel scheme, where at the coarsest level the number of the variables is sufficiently small so that the problem can be solved directly (see Figure 2).

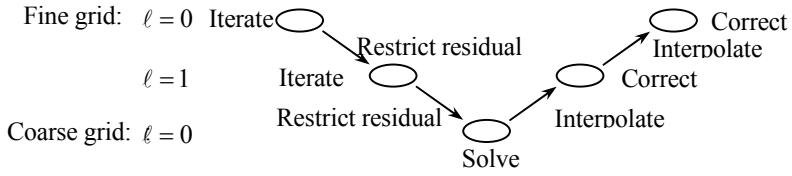

Figure 1 – V-cycle for 3 levels

$\mathbf{x}_\ell$=V-cycle$\left(\mathbf{b}_\ell, \ell\right)$

1. Unless on the finest grid ($\ell = 0$), set $\mathbf{x}_\ell := \mathbf{0}$ as the initial correction for level $\ell - 1$.
2. Pre-iterate $\mathbf{x}_\ell := \mathbf{x}_\ell + \mathbf{Q}_\ell^{-1}\left(\mathbf{b}_\ell - \mathbf{A}_\ell \mathbf{x}_\ell\right)$, where $\mathbf{Q}$ determines the iteration method.
   (Common iteration methods are synchronous value iteration (Richardson) for which $\mathbf{Q}_\ell = \mathbf{I}$, and Gauss-Seidel for which $\mathbf{Q}_\ell$ is the lower triangular part of $\mathbf{A}_\ell$ including the diagonal [14]).
3. Coarse grid correction step:
   3.1. Let $\mathbf{res}_\ell = \mathbf{b}_\ell - \mathbf{A}_\ell \mathbf{x}_\ell$. Solve the approximated residual equation $\mathbf{A}_{\ell+1}\mathbf{e}_{\ell+1} = \mathbf{I}_\ell^{\ell+1}\mathbf{res}_\ell$, either directly if on the coarsest grid, or via recursion $\mathbf{e}_{\ell+1}$:=V-cycle$\left(\mathbf{I}_\ell^{\ell+1}\mathbf{res}_\ell, \ell+1\right)$.
   3.2. Correct using the interpolated error $\mathbf{x}_\ell := \mathbf{x}_\ell + \mathbf{I}_{\ell+1}^{\ell}\mathbf{e}_{\ell+1}$.
4. Post-iterate $\mathbf{x}_\ell := \mathbf{x}_\ell + \mathbf{Q}_\ell^{-1}\left(\mathbf{b}_\ell - \mathbf{A}_\ell \mathbf{x}_\ell\right)$ to reduce interpolation errors.

Figure 2 – Solve phase at level ℓ

A common implementation of the multigrid scheme described above uses the V-cycle function presented in Figure 2. It is initiated by calling $\mathbf{x}$:=V-cycle$\left(\mathbf{b}, \ell = 0\right)$. Resolution levels are indexed by $\ell \in 0,1,\ldots,\ell_{\max}$, with $\ell = 0$ being the finest level and $\ell_{\max}$ the coarsest. V-cycle at level ℓ solves $\mathbf{A}_\ell \mathbf{x}_\ell = \mathbf{b}_\ell$, where $\mathbf{x}_\ell, \mathbf{b}_\ell$ are $N_\ell \times 1$ vectors, and $\mathbf{A}_\ell$ is a predefined matrix. In addition, multigrid uses two predefined inter-level operators: an $N_{\ell+1} \times N_\ell$ restrictor matrix, denoted by $\mathbf{I}_\ell^{\ell+1}$ that maps solution or error vectors from level ℓ to level $\ell+1$; and an $N_\ell \times N_{\ell+1}$ interpolator matrix that maps from level $\ell+1$ to ℓ.

In AMG, the matrices $\mathbf{A}_\ell$, $\mathbf{I}_{\ell+1}^{\ell}$, $\mathbf{I}_{\ell}^{\ell+1}$ for all levels are automatically built during a setup phase using only the matrix $\mathbf{A}$. The interpolator $\mathbf{I}_{\ell+1}^{\ell}$ is built based on $\mathbf{A}_\ell$ and theoretically motivated guidelines that balance accuracy of smooth error representation and computational complexity. The basic idea is to interpolate a variable *i* on the fine grid only from coarse variables *j* that are strongly connected, i.e. if $|\mathbf{A}_{ij}|$ is relatively large. For brevity we omit the details here, and refer the reader to [11] and [17] for a full description. The interpolator also determines the reduction in size between consecutive levels, which is typically moderate (4 being a typical ratio). The restrictor $\mathbf{I}_{\ell}^{\ell+1}$ calculates coarse grid variables as linear combinations of fine grid variables, and is often taken as the transpose of $\mathbf{I}_{\ell+1}^{\ell}$. $\mathbf{A}_{\ell+1}$ is defined as the *Galerkin* operator $\mathbf{A}_{\ell+1}=\mathbf{I}_{\ell}^{\ell+1}\mathbf{A}_\ell\mathbf{I}_{\ell+1}^{\ell}$, and $\mathbf{b}_{\ell+1}$ is calculated as a restriction of the residual $\mathbf{res}_\ell=\mathbf{b}_\ell-\mathbf{A}_\ell\mathbf{x}_\ell$, namely $\mathbf{b}_{\ell+1}=\mathbf{I}_{\ell}^{\ell+1}(\mathbf{b}_{\ell+1}-\mathbf{A}_\ell\mathbf{x}_\ell)$.

A particular case to which we refer as *strict state aggregation* is obtained when the interpolator is defined by dividing the set of variable into disjoint groups denoted $\{\mathrm{G}_k\}_{k=1}^{K}$, and setting

$$\left(\mathbf{I}_{\ell+1}^{\ell}\right)_{ik}=1 \text{ if } i\in\mathrm{G}_k, \text{ and } 0 \text{ otherwise} \tag{1}$$

More general inter-level operators use fractional weights with overlapping ranges, such as those constructed by the Ruge-Stüben scheme [11]. Borrowing the terminology of [12], we refer to this general case as soft state aggregation.

Asymptotic convergence of AMG is guaranteed for *any* choice of inter-level operators if **A** is a symmetric M-matrix with a strictly dominant diagonal [14]. However, the rate of convergence highly depends on the construction of appropriate inter-level operators. Furthermore, its efficiency in terms of computational operations to reach a certain level of accuracy depends on **A** being sparse, and a balance between the computational cost of inter-level operators and their approximation properties of smooth errors. While convergence for a non-symmetric **A** is not guaranteed, ample empirical results indicate the effectiveness of AMG in this case as well.

3. THE MDP MODEL AND TD(λ)

An MDP is defined by a 4-tuple $\{\mathrm{S},\mathrm{A},P,g\}$. S and A are the state and action spaces respectively. We assume here a finite or countably infinite state space (but note that the algorithms in this paper are essentially applicable to general state spaces). P defines the *Markovian* dynamics via the transition probabilities $p(s_{t+1}\,|\,s_t,a_t)$ from state s_t to state s_{t+1} given that the action taken is a_t. The reward received for such a transition is $g_t=g(s_t,s_{t+1},a_t)$. For a finite MDP we denote the number of states by *N*.

A *stationary policy* π is a mapping $\pi:\mathrm{S}\times\mathrm{A}\to[0,1]$, where $\pi(s,a)$ is the probability of taking action a at state s. Applying a stationary policy to an MDP induces a Markov chain, with the transition probabilities $p(s'\,|\,s)=\sum_{a\in\mathrm{A}(s)}\pi(s,a)\,p(s'\,|\,s,a)$, and rewards $g(s)=\sum_{a\in\mathrm{A}(s)}\pi(s,a)\sum_{s'\in S}p(s'\,|\,s,a)\,g(s,s',a)$. We henceforth fix the control policy, and assume that the induced Markov chain is *irreducible*, *aperiodic*, with a unique stationary distribution $q(s)$ that satisfies $q(s')=\sum_{s\in\mathrm{S}}p(s'\,|\,s)q(s)$. For future use we denote the transition matrix **P** with $\mathbf{P}_{s,s'}=p(s'\,|\,s)$, the reward vector **g** with elements $\mathbf{g}_s=g(s)$, and the diagonal matrix **D** with $\mathbf{D}_{ss}=q(s)$. We consider the discounted cost functional with a discount factor $\gamma\in[0,1)$, namely $v(s)=E(\sum_{t=0}^{\infty}\gamma^t g_t\,|\,s_0=s)$. It is well known that the value function of a stationary policy π is the unique solution of the Bellman equation

$$(\mathbf{I}-\gamma\mathbf{P})\mathbf{v}=\mathbf{g}, \tag{2}$$

where **I** denotes the identity matrix and **v** is a vector of state values, i.e. $\mathbf{v}_s = v(s)$. The value function is approximated as a linear combination of *K* basis functions

$$v(s) \approx \phi(s)^T \theta \tag{3}$$

where $\phi(s) \in \Re^K$ is a vector valued function and $\theta \in \Re^K$ is a weight vector to be learned. The k th element of $\phi(s)$ is known as a feature or basis function and denoted by $\phi_k(s)$. The TD(λ) algorithm [15] iteratively applies the following update rule

$$\theta_t = \theta_{t-1} + \alpha_t \mathbf{z}_t \left(g_t - \left(\phi(s_t) - \gamma \phi(s_{t+1}) \right)^T \theta_{t-1} \right) \quad ; \quad \mathbf{z}_t = \lambda \gamma \mathbf{z}_{t-1} + \phi(s_t) \tag{4}$$

where $\mathbf{z}_t$ is a vector of *eligibility traces*, initialized by $\mathbf{z}_{-1} = \mathbf{0}_{K\times 1}$. $\lambda \in [0,1]$ is an algorithm parameter, and α_t is a non-increasing positive step size sequence. Under certain technical assumptions [15], TD(λ) converges with probability 1 to a unique fixed point θ^* that satisfies

$$\mathbf{A}\theta^* = \mathbf{b}\,, \tag{5}$$

where **A** is a $K\times K$ matrix and **b** a $K\times 1$ vector defined as follows

$$\mathbf{A} = \mathbf{\Phi}^T \left(\mathbf{I} - \gamma\lambda \mathbf{P} \right)^{-1} \mathbf{D} \left(\mathbf{I} - \gamma \mathbf{P} \right) \mathbf{\Phi} \quad ; \quad \mathbf{b} = \mathbf{\Phi}^T \mathbf{D} \left(\mathbf{I} - \gamma\lambda \mathbf{P} \right)^{-1} \mathbf{g} \tag{6}$$

and $\mathbf{\Phi}$ is an $N\times K$ matrix $\mathbf{\Phi}_{sk} = \phi_k(s)$, with basis functions as its columns.

4. AMG FOR VALUE ITERATION

In this section we assume the MDP model is fully known. In this case, the application of AMG as a "black box" solver to solve (2) is straightforward, by defining $\mathbf{A} = \mathbf{I} - \gamma \mathbf{P}$ and $\mathbf{b} = \mathbf{g}$. Similarly, we can apply AMG to solve (5) where **A** and **b** are defined in (6), which is the basis for the multigrid TD algorithm in the next section.

We remind that in order to guarantee convergence of the basic multigrid algorithm, **A** should be a symmetric M-matrix with a strictly dominant diagonal. It may be shown that the last two properties follow from **P** being a probability matrix and $\gamma \in [0,1)$. However, since **P** is in general not symmetric, neither is **A.** Thus, the general theory of AMG convergence applies here only in the special case of a symmetric transition matrix. It should however be noted that convergence can always be enforced, simply by "turning off" the coarse grid corrections at some point, or by monitoring the error reduction as done in other hierarchical schemes [510]. For the problems we tested, convergence was always observed for the unmodified algorithm. We note that since **P** is often sparse in practical problems so is **A**, which is important for AMG efficiency.

An interesting question is whether the Markov chain structure is preserved at the coarse levels. Recall that AMG uses the Galerkin operator $\mathbf{A}_{\ell+1} = \mathbf{I}_{\ell}^{\ell+1} \mathbf{A}_{\ell} \mathbf{I}_{\ell+1}^{\ell}$ to define coarse grid equations. Suppose we choose $\mathbf{I}_{\ell}^{\ell+1} = (\mathbf{I}_{\ell+1}^{\ell})^{\#}$, the Moore-Penrose pseudo inverse of $\mathbf{I}_{\ell+1}^{\ell}$ (this choice is equivalent to the standard choice of $\mathbf{I}_{\ell}^{\ell+1} = (\mathbf{I}_{\ell+1}^{\ell})^T$ mentioned above, as it may be readily seen that multiplying $\mathbf{I}_{\ell}^{\ell+1}$ from the left by a nonsingular matrix merely scales the algorithm). With $\mathbf{A}_{\ell} = \mathbf{I} - \gamma \mathbf{P}_{\ell}$, this results in $\mathbf{A}_{\ell+1} = \mathbf{I} - \gamma \mathbf{P}_{\ell+1}$. For the special case of strict state aggregation defined in Section 2, $\mathbf{P}_{\ell+1}$ turns out to be a probability matrix (see Lemma 1 in [5]). This enables to interpret the coarse grid equations as resulting from a reduced state Markov chains. General inter-level operators such as Ruge-Stüben, do not generally posses this property.

5. A MULTIGRID TD ALGORITHM

To motivate the proposed multigrid approach, we consider the convergence of the mean of the parameter vector $\mathrm{E}\{\theta_t\}$, denoted by $\bar{\theta}_t$. The stochastic dynamics of the TD(λ) algorithm are expressed in [4] leads to the following approximate mean dynamics

$$\bar{\theta}_{t+1} = \bar{\theta}_t + \alpha_t \left(\mathbf{b} - \mathbf{A}\bar{\theta}_t \right). \tag{7}$$

where $\mathbf{A}$ and $\mathbf{b}$ were defined in (6). Subtracting both sides of (7) from the convergence point θ^* and denoting the mean error as $\bar{\mathbf{e}}_t = \theta^* - \bar{\theta}_t$, we obtain $\bar{\mathbf{e}}_t = \bar{\mathbf{e}}_{t-1} - \alpha_t \mathbf{A}\bar{\mathbf{e}}_{t-1}$. It is easy to realize that convergence is slow if $\bar{\mathbf{e}}_\tau$ is an eigenvector of $\mathbf{A}$ with an eigenvalue close to zero. More important is that convergence rates of error components are not uniform, so the error eventually aligns with slow-to-converge eigenvectors, which limit the convergence rate. TD(λ) therefore may be interpreted as a stochastic smoother of the error, and its performance may be enhanced by the multigrid approach.

Based on this observation we propose a multigrid TD algorithm in Algorithm 1 that follows similar steps to the V-cycle in Figure 2. We assume that the interpolators $\mathbf{I}_{\ell+1}^{\ell}$ are available beforehand. Starting with $\phi_0(s) = \phi(s)$, we recursively define feature vectors for all levels by

$$\phi_{\ell+1}(s)^T = \phi_\ell(s)^T \mathbf{I}_{\ell+1}^{\ell}. \tag{8}$$

The algorithm requires some criterion to determine when to switch between resolution levels. In the experiments section, we use a simple criterion, switching levels after a fixed number of iterations. The algorithm is initiated by iteratively applying $\theta_0 :=$MG-TD$(\theta_0, \ell = 0)$, where θ_0 is an initial guess. The approximated value function at the end of a complete cycle is given by $v(s) = \phi_0(s)^T \theta_0$. The value function for intermediate times before a cycle completes may be computed as $v(s) = \sum_{m=0}^{\ell-1} \phi_m(s)^T \theta_m^- + \phi_\ell(s)^T \theta_\ell$.

Algorithm 1: Multigrid-TD algorithm at level ℓ

MG-TD$\left(\vartheta_\ell, \ell, \theta_0^-, \theta_1^-, \ldots, \theta_{\ell-1}^-\right)$

1. **Initialize level correction** $\theta_\ell := \vartheta_\ell$, $\mathbf{z}_\ell := \mathbf{0}$
2. **Pre-iterate at level ℓ with residual rewards**:
   - 2.1. Observe the transition $s_t \to s_{t+1}$ and the reward g_t at time t.
   - 2.2. Update the eligibility traces $\mathbf{z}_\ell := \lambda\gamma\mathbf{z}_\ell + \phi_\ell(s_t)$
   - 2.3. Sample the residual $r_\ell := g_t - \sum_{m=0}^{\ell-1} \left(\phi_m(s_t) - \gamma\phi_m(s_{t+1})\right)^T \theta_m^-$
   - 2.4. Calculate the temporal difference $d_\ell = r_\ell - \left(\phi_\ell(s_t) - \gamma\phi_\ell(s_{t+1})\right)^T \theta_\ell$
   - 2.5. Update $\theta_\ell := \theta_\ell + \alpha_{\ell,t}\mathbf{z}_\ell d_\ell$
   - 2.6. If the switching criterion is met then continue, otherwise repeat from 2.1.
3. **Apply coarse grid correction**:
   - 3.1 Set $\theta_\ell^- := \theta_\ell$
   - 3.2. Recursive call $\vartheta_{\ell+1} :=$ MG-TD$\left(\vartheta_{\ell+1} = \mathbf{0}, \ell+1, \theta_0^-, \theta_1^-, \ldots, \theta_\ell^-\right)$
   - 3.3. Correction using the interpolated error $\theta_\ell := \theta_\ell^- + \mathbf{I}_{\ell+1}^{\ell}\vartheta_{\ell+1}$
4. **Post-iterate**: repeat step 2 until meeting the switching criterion.

Return θ_ℓ

Similarly to the V-cycle function, the MG-TD procedure at level ℓ is used to solve equations of the form $\mathbf{A}_\ell\theta_\ell = \mathbf{b}_\ell$, with $\mathbf{A}_0 = \mathbf{A}$, $\mathbf{b}_0 = \mathbf{b}$ at level $\ell = 0$ defined in (6). At level ℓ, TD(λ) is applied to smooth the error (step 2). However, its convergence rate deteriorates as the error becomes smooth. Based on the multigrid approach we would like to apply a correction of the form $\theta_\ell := \theta_\ell + \mathbf{I}_{\ell+1}^{\ell}\vartheta_{\ell+1}$, where $\vartheta_{\ell+1}$ is the solution of the residual equation

$$\left(\mathbf{I}_\ell^{\ell+1}\mathbf{A}_\ell\mathbf{I}_{\ell+1}^{\ell}\right)\vartheta_{\ell+1} = \mathbf{I}_\ell^{\ell+1}\left(\mathbf{b}_\ell - \mathbf{A}_\ell\theta_\ell\right). \tag{9}$$

In the learning context, this equation cannot be solved directly nor represented explicitly since the model of the Markov chain is not known, hence $\mathbf{A}_\ell$ and $\mathbf{b}_\ell$ are unavailable. We circumvent this obstacle by using a TD(λ) variant that converges to the solution of (9). Proposition 1 below states that TD(λ) applied at level $\ell+1$, with the fea-

ture vector $\phi_{\ell+1}(s)$ and the sampled residual r_ℓ replacing the one step reward converges to the solution of (9). Keeping θ_ℓ frozen (step 3.1) the error in θ_ℓ is approximated via a TD(λ) variant applied at level $\ell+1$ (step 3.2). The error is interpolated and used to correct θ_ℓ (step 3.3). Finally, TD(λ) is applied again to smooth interpolation errors (step 4).

The validity of each level in this algorithm, when operated in isolation, is verified as follows.

Proposition 1 *Consider a Markov chain with either a finite or countable infinite state space, with discounted cost. Let the assumption set in [15] be satisfied at level* $\ell = 0$. *If MG-TD is kept indefinitely at level* $\ell+1$, *then* $\vartheta_{\ell+1} = \theta_{\ell+1}$ *converges to the solution of the coarse grid equations (9).*

We omit the proof which may be found in [17].

6. SIMULATION RESULTS

We present here preliminary simulation results for two test-bed problems: one is a 1-D random walk problem described in Figure 3 on the left. The *Markov* chain has N states, ordered on a 1-D line. Transition probabilities and rewards from inner states and edge states are defined in Figure 3, and the discount factor is $\gamma = \sqrt[N-1]{0.5}$. This problem is similar to the hop-world problem in [16]. The second test-bed is the Mountain car problem. The objective is to bring an underpowered car positioned at the bottom of a valley to the top of the mountain at zero speed (see Figure 3, right side). Since its engine is too weak, it cannot drive straight up the mountain, but has to back up to gain momentum. We descretized the continuous problems on a 100x100 grid resulting in a 10,000 state MDP. Further details of the problem and the discretization scheme are found in [8].

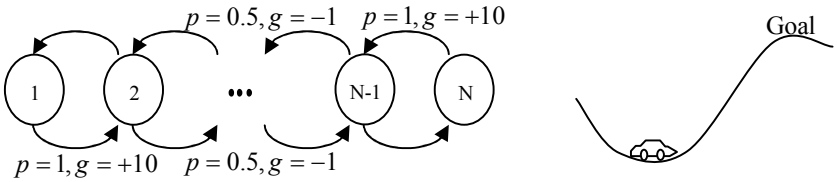

Figure 3 – Left: 1-D random walk problem. Right: Mountain car task

In the setup phase of AMG we used two interpolation methods: the well-known Ruge-Stüben method [11] and the strict state aggregation (1), with state groups formed as in [11].In Figure 4, we show results for policy evaluation for the problems above. One computational unit equals the number of mathematical operations in a single value iteration. In both problems, AMG methods show superior convergence rates to standard iteration methods.

In Figure 5 we applied TD learning algorithms to the 1-D random walk Markov chain. For AMG algorithms, we switched levels every 5000 samples and used 6 resolution levels. Based on Algorithm 1, we include a "one way" coarse to fine variant that starts at level 5 and interpolates its result up to level 0, and remains in this level thereafter. Its convergence rate is fast at first, but deteriorates due to smooth error caused by interpolation. As expected, its asymptotic convergence rate is similar to that of TD(λ). However, the Multigrid-TD(0) algorithm converges 10 times faster than TD(0).

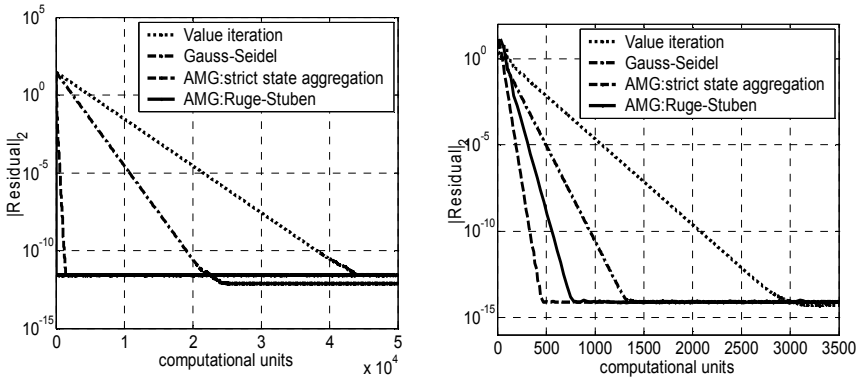

Figure 4 – Convergence curves of policy evaluation. Left: for the 1-D random walk with N=1000 states. Right: for the mountain car task. In AMG methods, 9 resolution levels are used.

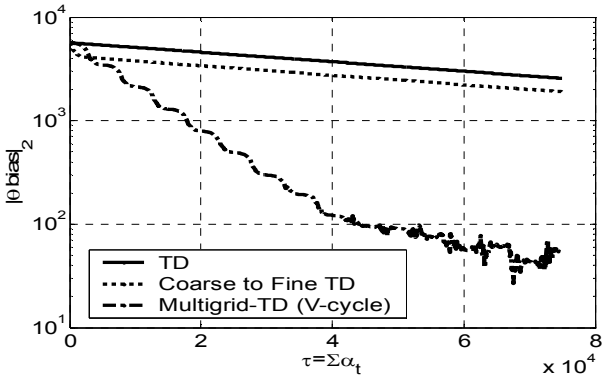

Figure 5 – Convergence curves of TD algorithms for the 1-D random walk problem with N=256 states. In AMG methods, 6 resolution levels are used. Each curve is an average of 5 Monte-Carlo runs. A constant learning step α_t=0.1 was used.

7. CONCLUSION

In this paper, the AMG approach is used to speed up policy evaluation for the known model case and temporal difference learning. A new temporal difference learning algorithm with a complexity per step similar to TD(λ), called Multigrid-TD, is proposed. Separate convergence of each level is shown, providing rational for the scheme. Preliminary experimental results on two test-bed problems show that AMG for policy evaluation speeds up convergence. Similarly, Multigrid-TD shows a considerable speed up relative to TD(λ). During our experiments, we observed that increasing number of states in this problem makes Multigrid-TD more superior to TD, suggesting that multigrid-TD is less susceptible to scalability. This property is well known in multigrid literature. Experimenting with Multigrid-TD on more complex problems is currently an undergoing work.

Several aspects of the proposed algorithm need to be examined for possible improvement. For example, in our tests, we used a simple criterion to switch between levels. An improved criterion may estimate the residual convergence rate, and switch between levels when it deteriorates. More importantly, the presented multigrid algorithm is sequential in nature, with each grid level is updated in turn. In the on-line leaning context of TD(λ), an alternative scheme may be developed which updates all levels simultaneously, thus making more efficient use of data samples. Full details of this scheme and its convergence analysis will be reported elsewhere.

A notable feature of AMG is the automatic creation of the multigrid structure at the setup phase. In the known model case, the implementation of a setup phase of AMG is straightforward, with many methods available in AMG literature 14]. In the learning case, the required data on the relevant system matrix may not be available beforehand. One way to obtain the required information is to compute an estimate of the matrix **A** by

$\mathbf{A}_t = \sum_{k=0}^{t} \phi(s_t)\left(\phi(s_t) - \gamma\phi(s_{t+1})\right)^T$, as done in the LSTD algorithm [9]. As the setup phase is not sensitive to inaccuracies **A**, this holds the potential of using rough or even qualitative estimators to evaluate only the significant elements of **A.** More broadly, the considerable theoretical and practical insight of AMG research may hopefully be applied to address the open question of how to form aggregation hierarchies automatically in complex learning problems. .

ACKNOWLEGMENT

We thank Irad Yavne for his invaluable guidance and advice regarding Multigrid methods.

REFERENCES

[1] M. Akian and J. P. Chancelier. Dynamic Programming Complexity and Application. *Proceedings of the 27th Converence on Decision and Control*, pp. 1551-1558, 1988.

[2] A. G. Barto and S. Mahadevan. Recent Advances in Hierarchical Reinforcement Learning. *Discrete Event Dynamic Systems: Theory and Applications*, Vol. 13, pp. 41-77, 2003.

[3] J. A. Boyan. Technical Update: Least Squares Temporal Difference Learning. *Machine Learning*, 49, pp. 233-246, Kluwer Academic, 2002.

[4] D. P. Bertsekas, V. S. Borkar, and A. Nedić. *Improved Temporal Difference Methods with Linear Function Approximation.* Report LIDS-2573, December 2003.

[5] D. P. Bertsekas and D. A. Castañon. Adaptive Aggregation Methods for Infinite Horizon Dynamic Programming. *IEEE Transactions on Automatic Control*, Vol. 34, No. 6, 1989.

[6] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dyanmic Programming.* Athena Scientific, 1995.

[7] S. F. McCormick. *Multigrid Methods, Frontiers in Applied Mathematics.* SIAM, 2000. See. Chapter 4, J.W. Ruge and K. Stüben. Algebraic Multigrid.

[8] R. Munos and A. Moore. Variable Resolution Discretization in Optimal Control. *Machine Learning*, Vol. 49, pp. 291-323, Kluwer Academic, 2002. Parameters for the Mountain Car problem can be found in http://www-2.cs.cmu.edu/~munos/variable/

[9] A. Nedić and D. P. Bertsekas. Least Squares Policy Evaluation Algorithms with Linear Function Approximation. *Discrete Evenet Dynamic Systems: Theory and Applications*, Vol. 13, pp. 79-110, 2003.

[10] P. J. Shweitzer. A Survey of Aggregation-Disaggregation in Large Markov Chains. In: W. Stewart (Ed.), *Numerical Solution of Markov Chains*, Marcel Dekker, New York, 1991

[11] K. Stüben, An Introduction to Algebraic Multigrid. Appendix in [14], 2001.

[12] S. P. Singh, T. Jaakkola, and M. I. Jordan. Reinforcement Learning with Soft State Aggregation. *Advances in Neural Information Processing Systems*, Vol. 7, pp. 361-368, MIT Press, 1995.

[13] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[14] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[15] J. N. Tsitsiklis and B. Van Roy. An Analysis of Temporal-Difference Learning with Function Approximation. *IEEE Transactions on Automatic Control.* Vol. 42, No. 5, pp. 674-690, 1997.

[16] X. Xu, H. He and D. Hu. Efficient Reinforcement Learning Using Recursive Least-Squares Methods. *Journal of Artificial Intelligence Research*, Vol. 16, pp. 259-292, 2002.

[17] O. Ziv, *Algebraic Multigrid for Reinforcement Learning*, Master's Thesis, Technion, January 2005.