Beyond Exponential Utility Functions: A Variance-Adjusted Approach for Risk-Averse Reinforcement Learning*** TYPOS CORRECTED

Abhijit A. Gosavi*, Sajal K. Das†, Susan L. Murray‡
*Department of Engineering Management and Systems Engineering; Email: gosavia@mst.edu
†Department of Computer Science; Email: sdas@mst.edu
‡Department of Engineering Management and Systems Engineering; Email: murray@mst.edu
Missouri University of Science and Technology, Rolla, MO 65409

***Abstract*—Utility theory has served as a bedrock for modeling risk in economics. Where risk is involved in decision-making, for solving Markov decision processes (MDPs) via utility theory, the *exponential* utility (EU) function has been used in the literature as an objective function for capturing risk-averse behavior. The EU function framework uses a so-called risk-averseness coefficient (RAC) that seeks to quantify the risk appetite of the decision-maker. Unfortunately, as we show in this paper, the EU framework suffers from computational deficiencies that prevent it from being useful in practice for solution methods based on reinforcement learning (RL). In particular, the value function becomes very large and typically the computer overflows. We provide a simple example to demonstrate this. Further, we show empirically how a variance-adjusted (VA) approach, which approximates the EU function objective for reasonable values of the RAC, can be used in the RL algorithm. The VA framework in a sense has two objectives: maximize expected returns and minimize variance. We conduct empirical studies on a VA-based RL algorithm on the semi-MDP (SMDP), which is a more general version of the MDP. We conclude with a mathematical proof of the boundedness of the iterates in our algorithm.**

I. Introduction

The Markov decision process (MDP) has been studied for maximizing expected gains (rewards) extensively. The MDP framework has been used in many problems where the decision-maker/agent is a human being — a manager. Most managers, however, are risk-averse. There is literature to support the fact that managers with a moderate degree of risk averseness tend to be more successful [8]. There is also literature to indicate that the risk-averseness of managers can be measured [31]. Yet most applications of the MDP framework have been restricted to the classical framework, which is risk-neutral and thus has a single criterion for optimization. In a risk-neutral framework, one ignores variability in the rewards of solutions (called policies). In other words, the framework recommends policies that have higher returns but also high variability. However, it is the variability in rewards that is closely tied to the concept of "risk," and is frequently an important consideration for managers. For instance, consider the situation in which a manager has to choose one from the following two scenarios:

- Scenario 1: Make \$900 with a probability of 0.9 and \$100 with a probability of 0.1.
- Scenario 2: Make \$900 with a probability of 0.8 and \$500 with a probability of 0.2.

Note that both scenarios result in the same *expected* returns of \$820. A risk-*prone* person will choose Scenario 1, since the probability of obtaining the highest reward is higher with it. Most managers are risk-averse, however, and will choose Scenario 2, because Scenario 2 appears to be less "risky." There are numerous ways to measure risk here: the lowest amount obtainable, which is higher with Scenario 2 ($\$500 > \$100$), can also be used to measure the risk; such a criterion for risk analysis would lead to selecting Scenario 2. This criterion is closely related to the idea of "worst-case" risk.

The literature on risk is quite rich now, and we will present a review relevant to this paper later. We begin by noting that the so-called *risk-adjusted* or risk-penalized framework can be adapted for use in solving MDPs. It employs the following objective function:

$$\text{Maximize } \mathsf{E}[Returns] - \theta \mathsf{Risk}[Returns], \qquad (1)$$

where E denotes the expectation while Risk denotes the risk in the returns, and θ is small positive scalar. The scalar θ is called the risk-averseness coefficient (RAC). Equation (1) presents a general format of the objective function employed in a risk-penalized framework. Via Equation (1), the goal is to attack a multi-objective problem: maximize returns *and* minimize risks.

Another risk criterion called *downside risk* sets a psychological threshold for the rewards, and the probability of the reward falling below that threshold, or the downside probability, is measured. With this criterion, the risk equals the downside probability. If the psychological threshold for the manager here is set at \$100 for the reward (returns), X, then the probability $P(X < 100)$ is 0.1 for Scenario 1 and 0 for Scenario 2. Hence, when used in the above, i.e., Equation (1), for reasonable values of θ, e.g., 0.2, Scenario 2 is returned as optimal. Of course, this approach can be somewhat subjective, since changing the threshold can change the solution. For instance, a threshold of \$600 will reverse the choice, although clearly Scenario 2 appears to be less risky. We now turn to the most popular risk metric in practice.

Variance was first proposed as a metric in Markowitz [22] to measure risk in returns from financial portfolios. Since

then, it has become a very popular risk metric in the world of finance, where its square root (standard deviation) is also called volatility. Variance for Scenario 1 is 57600 $\$^2$ while the same for Scenario 2 is 37120 $\$^2$. For small values of θ, e.g., 0.2, here, Scenario 2 will turn out to be optimal under Equation (1) when variance is used as a metric for risk. As stated above, while the risk-averse human is instinctively likely to prefer Scenario 2, for an artificial intelligence algorithm, it is necessary to develop an objective function that *automatically* captures this risk-revenue tradeoff with minimal user inputs. We must emphasize here that the discussion above, although presented in the form a financial (portfolio allocation) problem, applies in general to any managerial problem, e.g., preventive maintenance, where variability in rewards is unattractive.

Surprisingly, the classical MDP framework is risk-neutral; in other words, when faced between two policies with the same expected returns, it does *not* know how to distinguish between the two. In an MDP, the decision-maker (manager) must choose an action in each state visited by the system. Visits from one state to the next are governed by Markov chains. An immediate reward is earned in visits from one state to another. The goal is to select the action in each state in such a manner that some function of the immediate rewards is optimized. The semi-MDP (SMDP) framework [3] seeks to capture the effects of time and is a more general version of the MDP. In an SMDP, the time spent in transitioning from one state to another is a random variable, while in an MDP the time is fixed and the same for all transitions. Thus, an MDP is only a special case of the SMDP. Problems with the real world more often tend to be SMDPs. In this paper, we will present algorithms from the perspective of the SMDP — in an attempt to provide a more general perspective. Further, in this paper, we are interested in a Reinforcement Learning (RL) approach ([4], [28], [12]). The RL approach seeks to solve the MDP/SMDP in a simulator without generating the transition probabilities of the underlying Markov chains.

Two main frameworks that use risk in this context are: (i) the risk-sensitive framework of Howard and Matheson [21] and (ii) the variance-penalized framework of Filar *et al*. [10]. The risk-sensitive framework has some very useful features that enable it to model extreme risk — seen in e.g., bankruptcy [6]. Unfortunately, as we will demonstrate below, computationally, the risk-sensitive framework breaks down on problems where there is high variability. This is because it requires computation of exponential of the immediate reward, i.e., e^X where if X is large, the computer overflows. The use of the exponential term stems from the exponential utility function employed in this framework. Hence, the risk-sensitive framework is also called the *exponential utility* (EU) framework. It is important to emphasize that problems where risk-averseness becomes critical acquire significance *only* in cases where there is significant variability in the rewards, i.e., where some values of X are small while others are large. Hence, a viable solution method must work on numerical problem instances where some values of X are large. Some other difficulties with the EU framework have been identified before in the literature; see [23]. The variance-penalized metric is called variance-adjusted (VA) metric in stock markets where this metric was used originally. Hence, in this paper, we will refer to this approach as the VA approach.

Contributions of this paper: The main contributions of this paper are fourfold: First, we demonstrate, via a simple numerical example, how the EU framework may break down in practice on problems with significant variability in rewards and where risk-averseness becomes an issue as a consequence. We will show that the breakdown occurs because some elements of the value function become too large in practice. Our second contribution is that of developing a novel Bellman equation that incorporates variance and a contraction factor that allows the value function to remain bounded. Our third contribution is to show how the VA approach can perform well in practice to produce risk-averse solutions on problem instances where the EU approach breaks down; in particular, we propose a new RL algorithm that is a modified version of the one presented in [16]. We demonstrate, via simple experiments on small problems, how our VA approach can be used on the SMDP, which is a more general version of the MDP. Last but not least, we provide a mathematical proof of the boundedness of the iterates in our new algorithm.

The rest of this paper is organized as follows. Section II provides a review of the literature and the connection between the EU and the VA framework. The new RL algorithm based on variance adjustment is presented in Section III along with computational results; here we will also show how the EU framework breaks down via a counterexample. Section IV provides a mathematical proof of boundedness of the iterates in our new algorithm. Section V ends this paper with some concluding remarks and scope for future work.

II. Related Work and Preliminaries

We first present a review of literature relevant to this topic in order to motivate the need for our work and the gap in the literature. Thereafter, we show how the VA approach can be linked to the EU approach.

A. Literature review

Other than [21], some other works that consider the EU framework include [24], [32]. Although the EU-based Bellman equation has some nice mathematical properties that make it tractable for mathematical analysis, it has three important drawbacks:

Unstable iterates: As stated above, the main difficulty with the EU framework is that the iterates in the RL algorithm can become so large, especially in problems with large variance, that the computer can overflow — rendering these algorithms useless in practice. What is even more important is that the EU framework actually is aimed at problems with large variability!
Infinite objective function: A main difficulty with the EU framework is that the objective function's value equals infinity ([23], [24]); i.e., it cannot be measured in finite terms, which makes it unsuitable for quantitative comparisons (or calibrations) of two different policies. Real world managers prefer quantifiable and explainable metrics, which makes the EU function very unsuitable for managerial problems.
Unrealistic solutions: EU functions may also generate unrealistic solutions in practice [30] and "stochastic policies."

Recently, there has been some interest in the literature on management science to reduce variability in costs ([9], [1]). Other than these, there are some other papers that study

risk: Filar *et al*. [10] develop the VA framework in Equation (1), but do not propose a Bellman equation framework that can be used in RL; see also [27] for variance within MDPs. Mihatsch and Neuneier [23] develop a scaling parameter which is used to transform the reward function. Geibel [11] study problems in which some states are declared to be risky and are avoided; worst-case risk (discussed above) problems are studied in Heger [20]. Prior work in VA-based Bellman equation algorithms includes [26], [15], [18], and [16]. The work in Sato and Kobayashi [26] provides a Bellman equation but present a so-called policy-gradient algorithm rather than a direct value iteration approach. The algorithm in [15] uses one-step variance rather than long-run variance studied here, but provides for the first time a dynamic programming and Bellman equation for risk-adjusted Bellman equations. The algorithm in [18] is for long-run variance but is based on a *relative* value iteration approach, which is different than the one we will consider in this paper, while the work in [16] is based on an algorithm in which the average reward and variance are estimated on a second time scale like in the R-SMART algorithm [13]. Our work in this paper is closely tied to that in [16]; however, for the algorithm in [16], there are no guarantees that the iterates will remain bounded. Therefore, in our current work, we will use a modified version which will employ an artificial contracting factor — in order to keep the iterates bounded. We note that [25] also studied risk-averse Bellman equations but in the context of finite time horizon and the discounted infinite time horizon, neither of which is under consideration here. When the VA objective function is used in the MDP framework, a (deterministic) *stationary* optimal policy exists; see [10], and this implies that the optimal policy from a VA framework is also "time consistent" (time consistency is defined in [5] and [25]). In other words, time consistency of the solutions generated from using a VA objective does not provide any difficulties to us, unlike the VaR and CVaR measures popular in literature and analyzed at length in [5].

B. EU-VA connection

We now show how the EU and the VA approach are related. We begin with some notation. Let $\mathcal{S}$ denote the finite set of states, $\mathcal{A}(i)$ the finite set of actions permitted in state i, and $\mu(i)$ the action chosen in state i when policy μ is pursued, where $\cup_{i\in\mathcal{S}}\mathcal{A}(i) = \mathcal{A}$. Let $r(i,a,j)$, $p(i,a,j)$, and $t(i,a,j)$ denote the associated reward, transition probability, and transition time respectively of transiting from state i to state j under the influence of a. Then the *expected* immediate reward earned in i when a is chosen in it can be expressed as: $\bar{r}(i,a) = \sum_{j=1}^{|\mathcal{S}|} p(i,a,j)r(i,a,j)$. Similarly, $\bar{t}(i,a) = \sum_{j=1}^{|\mathcal{S}|} p(i,a,j)t(i,a,j)$. As stated above, the "risk-sensitivity" metric alluded to above uses exponential utility function. The so-called multiplicative form of the Bellman equation [21] needed in the EU framework seeks to maximize this function. It is defined as:

$$\Lambda_\mu(i) \equiv \liminf_{k\to\infty} \frac{\ln \mathsf{E}_\mu \left[\exp\left(\sum_{s=1}^{k} \Theta \bar{r}(x_s,\mu(x_s))\right)|x_1 = i\right]}{\Theta k}, \tag{2}$$

where x_s denotes the state of the Markov chain before the sth transition, $\mu(i)$ is the action in state i when policy μ is used, Θ is the RAC, and E_μ is the expectation operator under μ. In the above formulation, $\Theta > 0$. Rewards in all states encountered are added together, and then a logarithm is computed of the infinite sum, leading to an infinite-valued objective function. However, using the associated Bellman equation, mathematically, one can obtain a policy that penalizes risk. Unfortunately, as stated above, the exponential terms lead to numerical difficulties.

It can be shown, see e.g., [23], that the objective function above can be approximated, via a Taylor series expansion, as follows:

$$\frac{1}{\Theta}\ln \mathsf{E}\left[\exp(\Theta X)\right] = \mathsf{E}[X] + \frac{\Theta}{2}\mathsf{Var}[X] + O(\Theta^2).$$

If we ignore terms of the order of Θ^2, we obtain the following objective function to be maximized:

$$\mathsf{E}[X] + \frac{\Theta}{2}\mathsf{Var}[X].$$

Now, if we use the above, however, we must reverse the sign of Θ and ensure that $\Theta < 0$; otherwise, variance will not be penalized. In other words, if $\Theta > 0$, then our objective function in the approximation ought to be:

$$\mathsf{E}[X] - \frac{\Theta}{2}\mathsf{Var}[X].$$

Hence, if we set $\theta = \Theta/2$, where $\theta > 0$, then our objective function, which is to be maximized, becomes: $\mathsf{E}[X] - \theta\mathsf{Var}[X]$, which is the VA objective function proposed in [10]. Fortunately, the VA objective function is finite and it does not have the exponential terms which are difficult to compute when the power of the exponential is large, e.g., 20. A Bellman equation approach for this objective has been presented in [18]. The algorithm presented in [18] however uses a relative value iteration approach. In this paper, we use a two-time-scale approach that is described below.

III. RL ALGORITHM

In this section, we will first present the RL algorithm for the VA objective function. Thereafter, we will present a numerical example to demonstrate how the EU framework breaks down. Thereafter, we will present numerical studies on small SMDPs. We will conclude with a comparison of our approach to existing approaches from the literature.

We begin with some definitions from [19] and the Bellman equation based on [16]: We first define three quantities:

$$\varrho_\mu(i) \equiv \lim_{k\to\infty} \mathsf{E}_\mu \left[\sum_{s=1}^{k} \bar{r}(x_s,\mu(x_s))|x_1 = i\right] \Big/ k,$$

which is the first moment of the immediate reward,

$$\tau_\mu(i) \equiv \lim_{k\to\infty} \mathsf{E}_\mu \left[\sum_{s=1}^{k} \bar{t}(x_s,\mu(x_s))|x_1 = i\right] \Big/ k,$$

which is the first moment of the time in one transition, and

$$\sigma_\mu(i) \equiv \lim_{k\to\infty} \mathsf{E}_\mu \left[\sum_{s=1}^{k} \bar{r}^2(x_s,\mu(x_s))|x_1 = i\right] \Big/ k,$$

which is the second moment of the immediate reward. Then for irreducible and recurrent Markov chains, the long-run average reward of a policy μ in an SMDP, starting at state i, is

$$\rho_\mu(i) = \frac{\varrho_\mu(i)}{\tau_\mu(i)}.$$

Via Theorem 1 of [15], the long-run variance of rewards of the policy μ in an SMDP, starting at state i, is defined as:

$$\psi_\mu(i) = \frac{\sigma_{\mu(i)}}{\tau_\mu(i)} - \frac{(\varrho_\mu(i))^2}{\tau_\mu(i)}.$$

We can show that for irreducible and recurrent Markov chains, ϱ, τ, σ, ρ and ψ do not depend on the starting state i. Then, we have that for any i, i.e., for the entire system, the objective function via Equation (1), for a given policy μ, is:

$$\phi_\mu = \rho_\mu - \theta\psi_\mu. \tag{3}$$

Expressions for ϕ, ρ, and ψ can be obtained in terms of the steady-state probabilities of the Markov chain of the policy μ as shown in [19]. Let $\Pi_\mu(i)$ denote the steady-state probability of state i under policy μ. These probabilities can be determined by solving the following system of linear equations: For $j = 1, 2, \ldots, |\mathcal{S}|$,

$$\sum_{i\in\mathcal{S}} \Pi_\mu(i)p(i,\mu(i),j) = \Pi_\mu(j); \sum_{j\in\mathcal{S}} \Pi_\mu(j) = 1.$$

Then, the average reward of policy μ can be expressed as:

$$\rho_\mu = \frac{\sum_{i\in\mathcal{S}} \Pi_\mu(i)\bar{r}(i,\mu(i))}{\sum_{i\in\mathcal{S}} \Pi_\mu(i)\bar{t}(i,\mu(i))}.$$

The variance of policy μ, using the formulation developed in [19], can be expressed as

$$\psi_\mu = \frac{\sum_{i\in\mathcal{S}} \Pi_\mu(i)\left[\bar{r}(i,\mu(i))\right]^2}{\sum_{i\in\mathcal{S}} \Pi_\mu(i)\bar{t}(i,\mu(i))} - \frac{\left[\sum_{i\in\mathcal{S}} \Pi_\mu(i)\bar{r}(i,\mu(i))\right]^2}{\sum_{i\in\mathcal{S}} \Pi_\mu(i)\bar{t}(i,\mu(i))}.$$

Then, the expressions above for ρ_μ and ψ_μ can be used in Equation (3) to obtain the value of the objective function for any given policy μ. This allows us to perform an exhaustive evaluation of all policies; however, as is well known, that approach is computationally very burdensome for large problems and a Bellman equation approach is preferred for larger problems. The variance-adjusted Bellman equation for the SMDP proposed in [16] is as follows: For all $(i,a) \in \mathcal{S}\times\mathcal{A}(i)$,

$$Q(i,a) = \sum_{j\in\mathcal{S}} p(i,a,j)[r(i,a,j) - \theta(r(i,a,j) - \varrho^*)^2 - \phi^* t(i,a,j) + \max_{b\in\mathcal{A}(j)} Q(j,b)], \tag{4}$$

where ϱ^* denotes the optimal average reward on a *per transition* basis (not on unit time basis) and ϕ^* denotes the optimal score (score of the optimal policy) on a unit time basis. In other words, if μ^* denotes the policy that optimizes the VA objective function, then $\varrho^* = \varrho_{\mu^*}$ and $\phi^* = \phi_{\mu^*}$.

We will now modify the original Bellman equation, shown above, in order to introduce a contraction. Our modified Bellman equation for SMDPs is as follows:

$$Q(i,a) = \sum_{j\in\mathcal{S}} p(i,a,j)[r(i,a,j) - \theta(r(i,a,j) - \varrho)^2 - \phi t(i,a,j) + \eta \max_{b\in\mathcal{A}(j)} Q(j,b)], \tag{5}$$

where η is an artificially introduced constant (scaling factor) taking values in the interval $(0,1)$. For any given values of the scalars ϱ and ϕ, the above equation is guaranteed to have a unique solution. Use of scaling factors such as $\eta \in (0,1)$ is common in the literature on average reward (see [29] where an eligibility trace is used and [2] in policy gradients to force a unique solution to the Bellman equation) in order to facilitate convergence. In practice, η will be set to a value very close to 1. We will refer to our new equation as the contractive variance-adjusted Bellman equation (CV-ABE) and the corresponding solution framework as the CV-ABE framework. Our proposed CV-ABE framework thus combines the classical risk-neutral Bellman equation with (i) variance and (ii) the contraction factor. The novelty of the new framework, depicted via Figure 1, stems from the fact that the algorithm behaves gracefully in practice, unlike the large values exhibited by the iterates of the EU framework. In order to use the CV-ABE framework, we will make the following assumption about η.

Assumption A: There exists a value for $\bar{\eta}$ in the interval $(0,1)$ such that for all $\eta \in (\bar{\eta}, 1)$, the unique solution, $Q(.,.)$, of Equation (5) with $\phi \equiv \phi^*$ and $\varrho = \varrho^*$ produces a policy d defined as follows

$$d(i) \in \arg\max_{a\in\mathcal{S}} \sum_{j\in\mathcal{S}} p(i,a,j)[r(i,a,j) - \theta(r(i,a,j) - \varrho^*)^2 - \phi^* t(i,a,j) + \eta \max_{b\in\mathcal{A}(j)} Q(j,b)]$$

such that ϕ_d equals ϕ^* and ϱ_d equals ϱ^*. Under this assumption, if we use Equation (5) instead of (4), we should still obtain the optimal solution. In practice, the assumption is usually satisfied when η is very close to 1. Note that when $\eta = 1$, the two equations are identical. The numerical advantage of Equation (5) is that the iterates remain bounded in practice. We present details of the algorithm below, while a flowchart is provided in Figure 2.

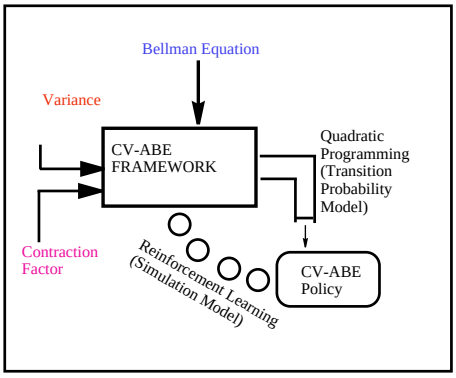

Fig. 1. CV-ABE: The new proposed Bellman equation framework for the SMDP that combines variance and the contraction factor, η within the Bellman optimality equation. The QP (Quadratic Programming) approach needs the transition probability model, while the RL model bypasses it and works within a simulator.

Steps in Algorithm:

Step 1. Set k, the number of state changes or the number of iterations, to 0. Set for all (l,u), where $l \in \mathcal{S}$ and $u \in \mathcal{A}(l)$,

$Q^k(l,u) \leftarrow 0$. Set ϱ^k, the estimate of the long-run reward per state change in the kth iteration, and σ^k, the estimate of the long-run squared reward per state change in the kth iteration, to 0. Let τ^k denote the estimate of the time spent in each transition in the kth iteration, which should be initialized to a very small positive quantity. Set η to a value close to 1, e.g., 0.99. Let α^k and β^k be step-sizes that are decayed according to standard RL rules. Set $k_{\max}$, the number of iterations for which the algorithm is run, to a sufficiently large number. Start system simulation at any arbitrary state.

Step 2. Let the current state be i. An action u will be considered greedy if $u = \arg\max_{b \in \mathcal{A}(i)} Q^k(i,b)$. Select action a such that all actions are selected with equal probability in the first iteration, but gradually the probability of selecting the non-greedy action is reduced.

Step 3. Simulate action a using an ϵ-greedy strategy [28]. Let the next state be j. Let $r(i,a,j)$ be the immediate reward earned in the transition to j from i under action a.

Step 4. Compute $\psi^k = \frac{\sigma^k - (\varrho^k)^2}{\tau^k}$ and $\phi^k = \left[\frac{\varrho^k}{\tau^k} - \theta\psi^k\right]$. Update $Q(i,a)$ as follows:

$$Q^{k+1}(i,a) \leftarrow (1-\alpha^k)Q^k(i,a) + \alpha^k[r(i,a,j) - \theta\left(r(i,a,j) - \varrho^k\right)^2 - \phi^k t(i,a,j) + \eta \max_{b \in \mathcal{A}(j)} Q^k(j,b)].$$

Step 5. If a is greedy, update ϱ, σ, and τ using the following:

$$\varrho^{k+1} \leftarrow (1-\beta^k)\varrho^k + \beta^k \frac{\left[r(i,a,j) + \varrho^k k\right]}{k+1};$$

$$\sigma^{k+1} \leftarrow (1-\beta^k)\sigma^k + \beta^k \frac{\left[(r(i,a,j))^2 + \sigma^k k\right]}{k+1};$$

$$\tau^{k+1} \leftarrow (1-\beta^k)\tau^k + \beta^k \frac{\left[t(i,a,j) + \tau^k k\right]}{k+1}.$$

Step 6. Increment k by 1. If $k < k_{\max}$, set $i \leftarrow j$ and then go to Step 2. Otherwise, go to Step 7.

Step 7. For each $l \in \mathcal{S}$, select $d(l) \in \arg\max_{b \in \mathcal{A}(l)} Q^k(l,b)$. The policy returned is d. Stop.

In the above, the exploration is gradually reduced. The algorithm can be used for an MDP by setting $t(.,.,.) = 1$ for all transition times and $\tau^k = 1$ for all k, i.e., τ^k is not updated.

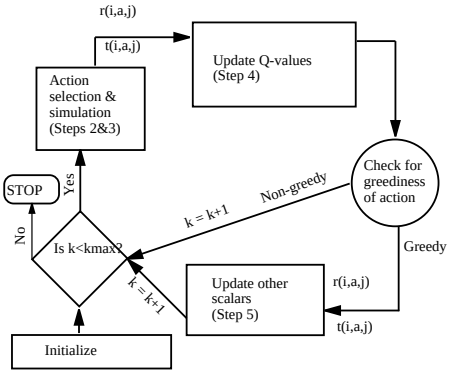

Fig. 2. Working mechanism of the RL algorithm: A flowchart

A. Counterexample: How the EU algorithm breaks down

We will now illustrate how a Q-Learning algorithm breaks down numerically on problems where the immediate reward can take on high values. The Q-Learning algorithm based on the EU function will follow a format similar to the RL algorithm presented above with the following differences:

- Any state-action pair, e.g., (1,1) is selected to be a distinguished state-action pair, (i^*, a^*) in Step 1. All the Q-values are initialized to 1 in Step 1.
- Exploration is never decayed.
- In Step 4, for an MDP version [6] of the problem, the update of the Q-value will be as follows:

$$Q^{k+1}(i,a) = (1-\alpha)Q^k(i,a) + \alpha\left[\frac{\exp(\Theta r(i,a,j))}{Q^k(i^*,a^*)} \max_{b \in \mathcal{A}(j)} Q^k(j,b)\right].$$

- The scalars, ρ, ψ, ϕ, ϱ, σ, and τ, are not needed.

We now show via an illustrative example how the EU algorithm breaks down via a counterexample.

Example A: We choose a 2-state MDP from [16] as an example. $\mathbf{P_a}$ and $\mathbf{R_a}$ denote the transition probability and reward matrices for action a respectively; $\mathbf{P_a}(i,j) = p(i,a,j)$ and $\mathbf{R_a}(i,j) = r(i,a,j)$.

$$\mathbf{P}_1 = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}; \mathbf{P}_2 = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}; \quad (6)$$

$$\mathbf{R}_1 = \begin{bmatrix} 6.0 & -5 \\ 7.0 & 12 \end{bmatrix}; \mathbf{R}_2 = \begin{bmatrix} 5.0 & 68 \\ -2 & 12 \end{bmatrix}. \quad (7)$$

The optimal policy is computed via an exhaustive evaluation of all policies via determining the steady-state probabilities of the Markov chain for each policy and then determining the values of ρ and ψ as shown at the start of this section. In what follows, we will use the notation (a_1, a_2) to denote the policy where a_1 will denote the action selected by the policy in state 1, while a_2 will denote the same for state 2. Thus, for Example A, there are four policies to be evaluated: $(1,1)$, $(1,2)$, $(2,1)$, and $(2,2)$. The optimal policy using the VA objective function with $\theta = 0.15$ and the risk-neutral solution for Example A are shown in Table 1. The EU function algorithm computes

TABLE I. RESULTS OF EXHAUSTIVE EVALUATION FOR EXAMPLE A: THE OPTIMAL POLICY'S METRICS ARE IN BOLD

	Example A ($\theta = 0.15$)			Example A ($\theta = 0$)
Policy	ρ	ψ	ϕ	ρ
(1,1)	5.828571	30.142041	1.307265	5.828571
(1, 2)	8.625000	31.284375	**3.932344**	8.625000
(2,1)	11.04000	287.23840	-32.04576	**11.04000**
(2,2)	10.95000	187.54750	-17.182125	10.95000

$\exp(2\theta \cdot 68) = \exp(20.4) = 723781420.9$; unfortunately, such a large number has to be further multiplied in the multiplicative form of the algorithm; the multiplicative form stems from the multiplicative form of the underlying Bellman equation [21], and within a few iterations, the computer overflows. We note that in problem instances for which risk becomes an issue, some values of the immediate reward must be high and some must be small. This was kept in mind while designing this test

instance. The VA algorithm uses the standard additive form of the Bellman equation (as opposed to the multiplicative form required with the EU function) and is hence numerically more stable.

Our VA-based RL algorithm uses the following parameters: $\eta = 0.99$, $\alpha^k = 1500/(3000 + k)$, and $\beta^k = 150/(300 + k)$. The probability of selecting either action is 0.5 in the first iteration, but the probability of selecting the non-greedy action (exploration) is set to $0.5(0.999)^{k-1}$. The VA-based RL algorithm was run in a discrete-event simulator of the Markov chains and produced the following Q-values: $Q(1,1) = 230.7135$; $Q(1,2) = 106.3651$; $Q(2,1) = 155.5511$; $Q(2,2) = 247.2558$. Then, $\arg\max\{Q(1,1), Q(1,2)\} = \arg\max\{230.7135, 106.3651\} = 1$, and thus the optimal action for state 1 is 1. Similarly, for state 2: $\arg\max\{Q(2,1), Q(2,2)\} = \arg\max\{155.5511, 247.2558\} = 2$, which means that the optimal action for state 2 is 2. This implies that the policy returned by the algorithm is $(1,2)$, which coincides with the optimal policy shown in Table 1. Figures 3 and 4 show the evolution of the learning of the Q-values for actions 1 and 2 respectively. It is clear that the optimal action is learned within a few iterations, although the Q-values take some time to converge. The simulator was coded in MATLAB and the learning took about 8 seconds on an Intel Pentium Processor with a speed of 2.66 GHz on a 64-bit operating system.

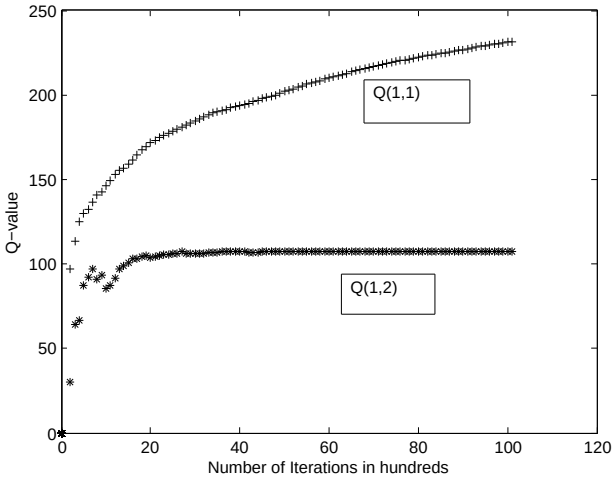

Fig. 3. Run-time performance of the algorithm: $Q(1,.)$ during the simulation

B. Numerical simulation results with the SMDP

We now present numerical results with our algorithm on four small SMDPs. We have named these four SMDPs: Examples B1 through B4. We present the relevant data for each example below. For each example, the algorithm was run for 10,000 iterations using $\eta = 0.99$ with the following rules for the step sizes: $\alpha^k = log(k+1)/(k+1)$ and $\beta = 150/(300+k)$. The probability of selecting the non-greedy action was set at $0.5(0.999)^{k-1}$. For each example, the optimal solution was also determined via exhaustive enumeration. In each case, our algorithm converged on the optimal solution. In each case, the

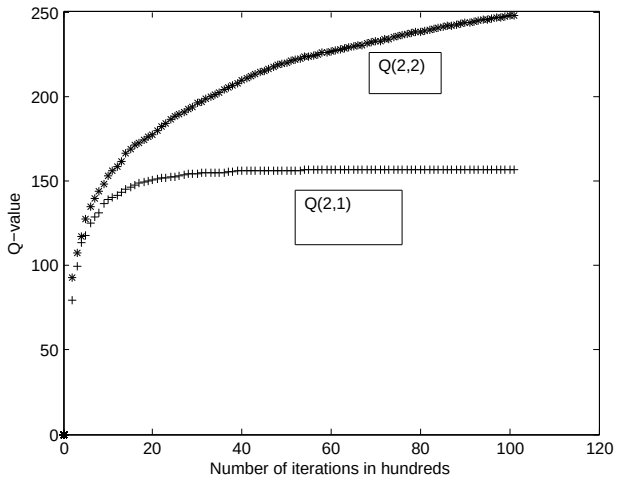

Fig. 4. Run-time performance of the algorithm: $Q(2,.)$ during the simulation

simulation ended in about 8 seconds on the same computer used for the experiment with Example A.

Example B1: The transition probability matrices were identical to those defined in (6), except $p(2,2,1) = 0.2$ and $p(2,2,2) = 0.8$. The transition reward matrices were also identical to those given in (7). The transition times were fixed and are given as:

$$\mathbf{T}_1 = \begin{bmatrix} 10 & 5 \\ 20 & 60 \end{bmatrix}; \mathbf{T}_2 = \begin{bmatrix} 50 & 75 \\ 7 & 20 \end{bmatrix}; \quad (8)$$

Also, $\theta = 0.15$. The optimal policy is (1,1) and $\phi^* = -0.0195$ with $\rho = 0.1427$ and $\psi = 1.0807$.

The resulting Q-values are: $Q(1,1) = 70.9$; $Q(1,2) = 7.96$; $Q(2,1) = 85$ and $Q(2,2) = 51.57$, which indicates that the algorithm identifies the optimal policy.

Example B2: The input parameters are identical to those of Example B1 with the following exception: $r(2,1,2) = 120$. The optimal policy is (1,2) whose $\phi^* = -0.1598$ with $\rho = 0.4769$ and $\psi = 4.2442$. The resulting Q-values are: $Q(1,1) = 203.76$; $Q(1,2) = 113.6$; $Q(2,1) = -740.8$ and $Q(2,2) = 246.27$, indicating that the algorithm identifies the optimal policy.

Example B3: The input parameters are identical to those of Example B1 with the following exception: $r(2,1,2) = 7$. The optimal policy is (1,1) whose $\phi^* = -0.0102$ with $\rho = 0.1112$ and $\psi = 0.8092$. The resulting Q-values are: $Q(1,1) = 65.01$; $Q(1,2) = 4.82$; $Q(2,1) = 76.72$ and $Q(2,2) = 49.41$; hence the algorithm identifies the optimal policy.

Example B4: The input parameters are identical to those of Example B1 with the following exception: $\theta = 0.35$. The optimal policy is (1,1) whose $\phi^* = -0.2356$ with $\rho = 0.1427$ and $\psi = 1.0807$. The resulting Q-values are: $Q(1,1) = 110.048$; $Q(1,2) = -31.75$; $Q(2,1) = 146.0471$ and $Q(2,2) = 71.1518$; thus, the algorithm is able to identify the optimal policy.

C. A comparison

We now compare our approach described in the CV-ABE framework to existing approaches in the literature. Essentially, there are three approaches to solving the risk-penalized problem: (i) the EU framework, (ii) the VA-framework using a

quadratic programming (QP) procedure described in Filar et al. [10], and (iii) the CV-ABE framework described in this paper. The QP approach of [10], which has been adapted for the SMDP in [19], does not yield a dynamic programming algorithm and hence does not yield a simulation-based RL algorithm either. The QP approach in itself is stable, but works only when the transition probabilities are available. The EU framework is rooted in a Bellman equation amenable to dynamic programming and hence to a simulation-based RL algorithm, but as stated above, the simulation-based algorithm is numerically unstable. In light of this, our new framework not only provides a simulation-based algorithm but also one that is numerically stable. It is important to emphasize the need for a simulation-based algorithm: a simulation-based (RL) algorithm allows one to bypass the transition probabilities, simulate the system, and generate a solution for the problem. For large-scale and complex problems, transition probabilities are often hard to find and thus a solution procedure that works without them is very advantageous. The discussion is summarized in Table 2.

TABLE II. A COMPARISON OF CV-ABE TO THE EU AND THE QP SOLUTION TECHNIQUES: NA DENOTES NOT APPLICABLE WHILE TBD DENOTES TO BE DETERMINED

Characteristic	*EU*	*QP*	*CV-ABE*
Dynamic programming algorithm	Yes	No	Yes
Simulation-based algorithm	Yes	No	Yes
Numerical stability of simulation-based algorithm	No	NA	Yes
Convergence proof	Yes	Yes	TBD

IV. BOUNDEDNESS OF ITERATES

We will now show that the iterates in our proposed RL algorithm will remain bounded. Our main result is as follows.

Theorem IV.1. *The sequence* $\{Q^k, \varrho^k, \phi^k\}_{k=1}^{\infty}$ *remains bounded.*

Proof: We will first prove that the iterates ϱ^k, σ^k, and τ^k remain bounded. It is clear from Step 4 that if these iterates remain bounded, ϕ^k will remain bounded.

Lemma IV.2. *The sequence* $\{\varrho^k\}_{k=1}^{\infty}$ *remains bounded.*

Proof: Define $\bar{\varrho} = \max\{\varrho^1, R\}$ where $R = \max|r(.,.,.)|$, i.e., R denotes the maximum of the absolute value of the immediate reward $r(.,.,.)$. We will use induction to show that $|\varrho^k| \leq \bar{\varrho}$ for all k. From Step 5 of the RL algorithm, for $k = 1$,

$$\begin{aligned} |\varrho^2| &\leq (1-\beta^1)|\varrho^1| + \beta^1 \frac{R}{1+1} + \beta^1 \frac{1|\varrho^1|}{1+1} \\ &\leq (1-\beta^1)\bar{\varrho} + \beta^1 \frac{\bar{\varrho}}{2} + \beta^1 \frac{\bar{\varrho}}{2} = \bar{\varrho}. \end{aligned}$$

Assuming the result is true for $k = m$, i.e., $|\varrho^m| \leq \bar{\varrho}$, we have from Step 5,

$$\begin{aligned} |\varrho^{m+1}| &\leq (1-\beta^m)|\varrho^m| + \beta^m \frac{R}{m+1} + \beta^m \frac{m|\varrho^m|}{m+1} \\ &\leq (1-\beta^m)\bar{\varrho} + \beta^m \frac{\bar{\varrho}}{m+1} + \beta^m \frac{m\bar{\varrho}}{m+1} = \bar{\varrho}. \end{aligned}$$

■

In a manner similar to the lemma above, one can show boundedness of σ^k and τ^k. We will next show boundedness of the Q-factor using arguments along the lines of [14].

We first claim that for every state-action pair (i, a):

$$|Q^k(i,a)| \leq M(1 + \eta + \eta^2 + \cdots + \eta^k), \tag{9}$$

where M is a positive finite number defined as follows:

$$M = \max\left\{w_{\max}, \max_{i\in\mathcal{S}, a\in\mathcal{A}(i)} Q^1(i,a)\right\}, \tag{10}$$

$$\text{where } w_{\max} = \max_{i,j\in\mathcal{S}, a\in\mathcal{A}(i)} |w(i,a,j)| \text{ and} \tag{11}$$

$$w(i,a,j) = r(i,a,j) - \theta\left(r(i,a,j) - \varrho^k\right)^2 - \phi^k t(i,a,j)$$

Since ϱ, ϕ, $r(.,.,)$, and $t(.,.,)$ are bounded, $w_{\max}$ must be bounded. Since we start with finite values for the Q-factors, then M too must be bounded. Then, from the above claim (9), boundedness follows since if $k \to \infty$,

$$\limsup_{k\to\infty} |Q^k(i,a)| \leq M\frac{1}{1-\eta}$$

for all $i \in \mathcal{S}$ and $a \in \mathcal{A}(i)$, since $0 \leq \eta < 1$. We now prove our claim in (9) via induction.

In asynchronous updating, the Q-factor of only one state-action pair is updated in a given iteration, while the other Q-factors remain un-updated. Hence, in the kth iteration of the asynchronous algorithm, the update for $Q^k(i,a)$ is either according to Case 1 or Case 2.

Case 1: The state-action pair is updated in the kth iteration:

$$Q^{k+1}(i,a) = (1-\alpha)Q^k(i,a) + \alpha\left[w(i,a,j) + \eta \max_{b\in\mathcal{A}(j)} Q^k(j,b)\right].$$

Case 2: The state-action pair is not updated in the kth iteration:

$$Q^{k+1}(i,a) = Q^k(i,a).$$

Now, if the update is carried out as in Case 1:

$$\begin{aligned} |Q^2(i,a)| &\leq (1-\alpha)|Q^1(i,a)| + \alpha|w(i,a,j) + \\ &\quad \eta \max_{b\in\mathcal{A}(j)} Q^1(j,b)| \\ &\leq (1-\alpha)M + \alpha M + \alpha\eta M \text{ (from (11) and (10))} \\ &\leq (1-\alpha)M + \alpha M + \eta M \text{ (since } \alpha \leq 1) \\ &= M(1+\eta) \end{aligned}$$

Now, if the update is carried out as in Case 2:

$$\begin{aligned} |Q^2(i,a)| &= |Q^1(i,a)| \\ &\leq M \leq M(1+\eta). \end{aligned}$$

From the above, our claim in (9) is true for $k = 1$. Now assuming that the claim is true when $k = m$, we have that for all $(i,a) \in (\mathcal{S} \times \mathcal{A}(i))$.

$$|Q^m(i,a)| \leq M(1 + \eta + \eta^2 + \cdots + \eta^m). \tag{12}$$

Now, if the update is carried out as in Case 1:

$$\begin{aligned}
|Q^{m+1}(i,a)| &\leq (1-\alpha)|Q^m(i,a)| + \alpha|w(i,a,j) + \\
&\quad \eta \max_{j\in\mathcal{A}(j)} Q^m(j,b)| \\
&\leq (1-\alpha)M(1+\eta+\eta^2+\cdots+\eta^m) \\
&\quad +\alpha M + \alpha\eta M(1+\eta+\eta^2+\cdots+\eta^m) \\
&\quad \text{(from (12)} \\
&= M(1+\eta+\eta^2+\cdots+\eta^m) \\
&\quad -\alpha M(1+\eta+\eta^2+\cdots+\eta^m) \\
&\quad +\alpha M + \alpha\eta M(1+\eta+\eta^2+\cdots+\eta^m) \\
&= M(1+\eta+\eta^2+\cdots+\eta^m) + \alpha M\eta^{m+1} \\
&\leq M(1+\eta+\eta^2+\cdots+\eta^m) + M\eta^{m+1} \\
&= M(1+\eta+\eta^2+\cdots+\eta^m+\eta^{m+1})
\end{aligned}$$

Now, if the update is carried out as in Case 2:

$$\begin{aligned}
|Q^{m+1}(i,a)| &= |Q^m(i,a)| \\
&\leq M(1+\eta+\eta^2+\cdots+\eta^m) \\
&\leq M(1+\eta+\eta^2+\cdots+\eta^m+\eta^{m+1})
\end{aligned}$$

From the above, the claim in (9) is proved for $k = m+1$. ■

V. Conclusions

The goal of this paper was to study RL algorithms for a risk-penalized/risk-adjusted objective function, which is essentially a multi-objective problem. The EU risk-sensitive framework has been widely advocated in the literature for studying the risk-averse MDP. Some deficiencies of the EU framework were identified. In particular, it was shown how the iterates can become very large. The VA Bellman equation framework was first developed in [15] for one-step variance and for one-step target variance in [17]. The problem of long-run variance, studied here, was also studied in [18] under a relative value iteration perspective and is closely tied to the EU framework. Our contribution here is in the same spirit as in the aforementioned papers. Our new algorithm is a modified version of the same in [16], but we show that the iterates will remain bounded. We supplemented our analysis with some encouraging numerical studies with on some small-scale SMDPs. For future work, there are at least two avenues for the algorithm: a convergence analysis (using differential equations) and an application to a large-scale study, e.g., case studies in [28], [12], [7].

References

[1] C. Barz and K. Waldmann. Rsik-sensitive capacity control in revenue management. *Math. Meth. Oper. Res.*, 65:565–579, 2007.

[2] J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence*, 15:319–350, 2001.

[3] D. P. Bertsekas. *Dynamic Programming and Optimal Control: Volume II*. Athena Scientific, Belmont, Massachusetts, fourth edition, 2012.

[4] D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA, 1996.

[5] K. Boda and J.A. Filar. Time consistent dynamic risk measures. *Math. Meth. Oper. Res.*, 63:169–186, 2006.

[6] V. S. Borkar. Q-learning for risk-sensitive control. *Mathematics of Operations Research*, 27(2):294–311, 2002.

[7] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, 2010.

[8] M. Caliendo, F. Fossen, and A. Kritikos. The impact of risk attitudes on entrepreneurial survival. *Journal of Economic Behavior and Organization*, 76:45–63, 2010.

[9] Y. Chen and J. Jin. Cost-variability-sensitive preventive maintenance considering management risk. *IIE Transactions*, 35:1091–1101, 2003.

[10] J. Filar, L. Kallenberg, and H. Lee. Variance-penalized Markov decision processes. *Mathematics of Operations Research*, 14(1):147–161, 1989.

[11] P. Geibel. Reinforcement learning via bounded risk. In *ICML01*, pages 162–169. Morgan Kaufman, 2001.

[12] A. Gosavi. *Simulation-Based Optimization:Parametric Optimization Techniques and Reinforcement Learning*. Kluwer Academic Publishers, Boston, MA, 2003.

[13] A. Gosavi. Reinforcement Learning for long-run average cost. *European Journal of Operational Research*, 155:654–674, 2004.

[14] A. Gosavi. Boundedness of iterates in Q-learning. *Systems and Control Letters*, 55:347–349, 2006.

[15] A. Gosavi. A risk-sensitive approach to total productive maintenance. *Automatica*, 42:1321–1330, 2006.

[16] A. Gosavi. Reinforcement learning for model building and variance-penalized control. In *Proceedings of the Winter Simulation Conference, Austin, TX*. IEEE, 2009.

[17] A. Gosavi. Target-sensitive control of Markov and semi-Markov processes. *International Journal of Control, Automation, and Systems*, 9(5):1–11, 2011.

[18] A. Gosavi. Variance-penalized Markov decision processes: Dynamic programming and reinforcement learning techniques. *International Journal of General Systems*, 43(6):649–669, 2014.

[19] A. Gosavi and M. Purohit. Stochastic policy search for variance-penalized semi-Markov control. In *Proceedings of the 2011 Winter Simulation Conference; S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, eds.*, 2011.

[20] M. Heger. Considerations of risk in reinforcement learning. In *Proceedings of 11th International Conference on Machine Learning*, pages 105–111. Morgan Kaufmann, 1994.

[21] R. Howard and J. Matheson. Risk-sensitive Markov decision processes. *Management Science*, 8:356–369, 1972.

[22] H Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77–91, 1952.

[23] O. Mihatsch and R. Neuneier. Risk-sensitive reinforcement learning. volume 49, pages 267–290, 2002.

[24] M. Rabin. Risk aversion and expected utility theory: A calibration theorem. *Econometrica*, 68:1281–1292, 2000.

[25] A. Ruszczynski. Risk-averse dynamic programming for Markov decision processes. *Math. Program. Ser. B*, 125:235–261, 2010.

[26] Makoto Sato and Shigenobu Kobayashi. Average-reward reinforcement learning for variance penalized markov decision problems. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 473–480, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[27] M. Sobel. Mean-variance tradeoffs in an undiscounted in an undiscounted MDP. *Operations Research*, 42(1):175–183, 1994.

[28] R. Sutton and A. G. Barto. *Reinforcement Learning*. The MIT Press, Cambridge, Massachusetts, 1998.

[29] J. N. Tsitsiklis and B. Van Roy. Average cost temporal-difference learning. *Automatica*, 35(11):1799–1808, 1999.

[30] C. Wang, S. Webster, and N.C. Suresh. Would a risk-averse newsvendor order less at a higher selling price? *European Journal of Operational Research*, 196:544–553, 2009.

[31] E. Weber, A. Blais, and N. Betz. A domain-specific risk-attitude scale: Measuring risk percep- tions and risk behaviors. *Journal of Behavioral Decision Making*, 15:1–28, 2002.

[32] P. Whittle. *Risk-sensitive optimal control*. John Wiley, NY, USA, 1990.