Risk-Sensitive Planning in Partially Observable Environments

Janusz Marecki
IBM T.J. Watson Research Center
1101 Kitchawan Road, Route 134
Yorktown Heights, NY 10598
marecki@us.ibm.com

Pradeep Varakantham
Singapore Management University
Singapore, 207855
pradeepv@smu.edu.sg

ABSTRACT

Partially Observable Markov Decision Process (POMDP) is a popular framework for planning under uncertainty in partially observable domains. Yet, the POMDP model is risk-neutral in that it assumes that the agent is maximizing the expected reward of its actions. In contrast, in domains like financial planning, it is often required that the agent decisions are risk-sensitive (maximize the *utility* of agent actions, for non-linear utility functions). Unfortunately, existing POMDP solvers cannot solve such planning problems exactly. By considering piecewise linear approximations of utility functions, this paper addresses this shortcoming in three contributions: (i) It defines the Risk-Sensitive POMDP model; (ii) It derives the fundamental properties of the underlying value functions and provides a functional value iteration technique to compute them exactly and (c) It proposes an efficient procedure to determine the dominated value functions, to speed up the algorithm. Our experiments show that the proposed approach is feasible and applicable to realistic financial planning domains.

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence

General Terms

Algorithms, Economics

Keywords

POMDPs, Risk Sensitivity, Utility Theory

1. INTRODUCTION

Recent years have seen an unprecedented rise of interest in autonomous agents deployed in high-risk domains ranging from planetary exploration [4] to autonomous trading [9]. Simultaneously, research in devising optimal planning policies for these agents has progressed significantly. Partially Observable Markov Decision Processes (POMDPs) [16] in particular have received a lot of attention, due to their ability to handle sequential decision making, the uncertainty of the outcomes of agent actions and the uncertainty of the agent observations of the environment.

Cite as: Risk-Sensitive Planning in Partially Observable Environments, Janusz Marecki and Pradeep Varakantham, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. XXX-XXX.

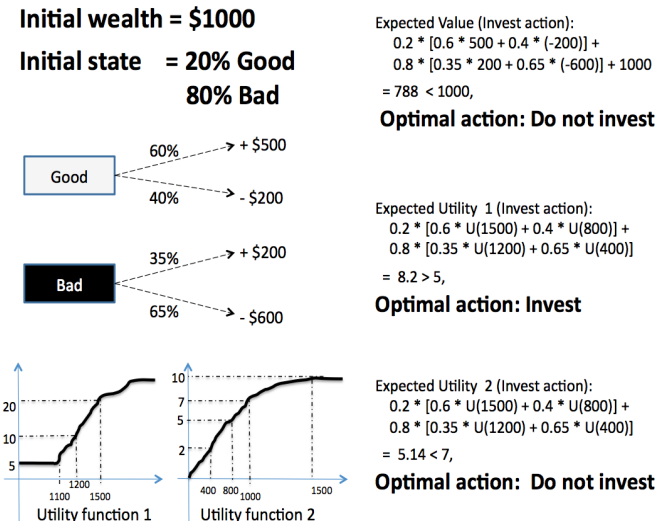

Figure 1: Expected Utility maximization vs Expected Value maximization.

Yet, POMDP solvers [7, 8, 12, 14, 15, 17, 18] typically maximize the expected *reward* of agent actions. In contrast, in high-stake domains such as financial planning, agents are often required to maximize the expected *utility* of their actions, for non-linear utility functions that characterize the agent attitude towards risk [19, 2]. To date, only [10, 11] have demonstrated how to solve planning problems where risk-sensitivity is expressed via utility functions, yet, only for problems characterized by fully observable environments.

In this paper we address these shortcomings by first defining Risk-Sensitive POMDPs. Next, by considering piecewise linear approximations of utility functions, we provide a functional value iteration method to compute the value functions exactly, by exploiting their piecewise bilinear properties. Finally, to speed up the algorithm, we show how to find and prune the dominated value functions using efficient approximations to the underlying non-convex bilinear programs.

2. MOTIVATION

In this section, we provide an illustrative example from a typical financial planning scenario to motivate risk sensitive planning in partially observable domains. In this example, we need to take a decision on whether to invest the current

wealth of $1000. This decision has to be made considering that the state of the market is uncertain, 20% good and 80% bad and the return on investment is also uncertain. Figure 1 provides further details on this example setting. In this example, for purposes of exposition, we focus on a single decision. However, in general and in our experimental section, we consider problems where a sequential set of decisions need to be made.

There are two ways to make decisions in such settings: (a) Expected value maximization: This is the risk neutral way to make decisions, i.e., by not considering that people have various attitudes towards risk. Thus, there is always the same decision made by this method. As shown in Figure 1, maximizing expected value provides a decision to not invest; (b) Expected utility maximization: This mechanism is sensitive to the risk attitude of the person and as shown in the figure, depending on whether the person is risk seeking (utility function 1) or risk averse (utility function 2), the decision appropriately changes.

Such reasoning is required in many high stakes domains such as disaster rescue, mars exploration, gambling where the utility functions are non-linear and the decisions made should be sensitive to the risk attitude. In fact, in our experimental results section, we experimented with extensions to the illustrative problem above.

3. RISK-SENSITIVE POMDPS

Utility theory [19] defines utility functions as transforming the current wealth of the agent (its initial wealth plus the sum of the immediate rewards it received so far) into a utility value. The theory postulates that the shape of the utility function can be used to define the agent attitude towards risk. To compute optimal policies for such risk-sensitive agents, acting in partially observable environments, we are interested in solving finite horizon POMDPs that maximize the expected total utility (as opposed to expected total reward) of agent actions. On account of being sensitive to risk attitudes, we refer to these planning problems as Risk-Sensitive POMDPs and formalize them as follows: S is a finite set of discrete states of the process and A is a finite set of agent actions. The process starts in some state $s_0 \in S$ and runs for N consecutive decision epochs. In particular, if the process is in state $s \in S$ in decision epoch $0 \leq n < N$, the agent controlling it chooses an action $a \in A$ to be executed next. The agent then receives the immediate reward $R(s,a)$ while the process transitions with probability $P(s'|s,a)$ to state $s' \in S$ at decision epoch $n+1$. Otherwise, in decision epoch $n = N$, the process terminates. The utility of the actions that the agent has executed is then a scalar $U(w_0 + \sum_{n=0}^{N-1} r_n)$ where w_0 is the initial wealth of the agent, U is the agent utility function and r_n is the immediate reward that the agent received in decision epoch n. The goal of the agent is to come up with a policy π that maximizes its total *expected* utility $\mathbb{E}[U(w_0 + \sum_{n=0}^{N-1} r_n)|\pi]$.

What further complicates the agent's search for π is that the process is only partially observable to the agent. That is, the agent receives noisy information about the current state $s \in S$ of the process and can therefore only maintain the current probability distribution $b(s)$ over states $s \in S$ (referred to as the agent *belief state*). When the agent executes some action $a \in A$ and the process transitions to state s', the agent receives with probability $O(z|a,s')$ an observation z from a finite set of observations Z. The agent then uses z to update its current belief state b, as shown later. In the following, $\mathcal{B}$ denotes an infinite set of all possible agent belief states and $b_0 \in \mathcal{B}$ is the agents' *starting belief state* (unknown at the planning phase). Also, $\mathcal{W} := \cup_{0 \leq n \leq N} \mathcal{W}^n$ is the set of all possible agent wealth levels where $\mathcal{W}^n$ denotes the set of all possible agent wealth levels in decision epoch n. For the initial range of agent wealth levels $\mathcal{W}^0 := [\underline{w}^0, \overline{w}^0]$ we determine $\mathcal{W}^n = [\underline{w}^n, \overline{w}^n]$ where $\underline{w}^n = \underline{w}^{n-1} + \min_{s \in S, a \in A} R(s,a)$ and $\overline{w}^n = \overline{w}^{n-1} + \max_{s \in S, a \in A} R(s,a)$, for $n = 1, \ldots, N$. (Notice, that $\mathcal{W}^0 \subset \mathcal{W}^1 \subset \ldots \subset \mathcal{W}^N$.) A policy π of the agent therefore indicates which action $\pi(n,b,w) \in A$ the agent should execute in decision epoch n, belief state b, with wealth level w, for all $0 \leq n < N$, $b \in \mathcal{B}$, $w \in \mathcal{W}^n$.

4. SOLVING RISK-SENSITIVE POMDPS

We first show how value iteration [3] can be used to find the optimal policy π^*. However, due to the continuous nature of belief and wealth space, it is not practical to solve Risk-Sensitive POMDPs with value iteration techniques (employed to solve POMDPs). To address this issue, we introduce one of the key contributions of this paper: a functional value iteration technique to solve Risk-Sensitive POMDPs.

4.1 Value Iteration

Since the value function has to be sensitive to the risk attitudes (dependent on the current wealth), while also accounting for sequential decision making under uncertainty (both observational and transitional), we cannot employ the same value function definition as expected value POMDPs. We denote by $V_U^n(b,w)$ the maximum expected utility for the agent if its starts acting in decision epoch n in belief state b with wealth level w. The agent maximizes $V_U^n(b,w)$ by executing an action $\pi^*(n,b,w)$ that is computed by:

$$\arg\max_{a \in A} \big\{ \sum_{z \in Z} P(z|b,a) V_U^{n+1}\big(T(b,a,z), w + R(b,a)\big)\big\} \quad (1)$$

where $P(z|b,a) = \sum_{s' \in S} O(z|a,s') \sum_{s \in S} P(s'|s,a) b(s)$ is the probability of observing z after executing action a from belief state b, $R(b,a) := \sum_{s \in S} b(s) R(s,a)$ is the expected immediate reward that the agent will receive for executing action a in belief state b and $T(b,a,z)$ is the new belief state of the agent after executing action a from belief state b and observed z. Formally, for each $s' \in S$ it holds that: [16] $T(b,a,z)(s') = [O(z|a,s')/P(z|b,a)] \sum_{s \in S} P(s'|s,a) b(s)$.

Hence, to find the optimal policy, π^*, value iteration must calculate values $V_U^n(b,w)$ for all $0 \leq n \leq N$, $b \in \mathcal{B}$, $w \in \mathcal{W}^n$. Value iteration calculates these values for $n = N, N-1, \ldots, 0$. Specifically, for $n = N$ the process terminates and thus

$$V_U^N(b,w) = U(w) \quad (2)$$

for all $w \in \mathcal{W}^N$, $b \in \mathcal{B}$. Otherwise, for all $0 \leq n < N$,

$$V_U^n(b,w) = \max_{a \in A} \big\{ \sum_{z \in Z} P(z|b,a) V_U^{n+1}\big(T(b,a,z), w + R(b,a)\big)\big\}. \quad (3)$$

for all $b \in \mathcal{B}$ and $w \in \mathcal{W}^n$. In the following, we group values $V_U^n(b,w)$ over all $(b,w) \in \mathcal{B} \times \mathcal{W}$ into *value functions* $V_U^n : \mathcal{B} \times \mathcal{W} \rightarrow \mathbb{R}$, for each $0 \leq n \leq N$. Note, that computing value functions V_U^n from value functions V_U^{n+1} exactly may be hard because $\mathcal{B}$ and $\mathcal{W}$ are infinite. In addition, POMDP

solution techniques—that already handle an infinite $\mathcal{B}$—are not applicable for solving Risk-Sesitive POMDPs as they do not handle an infinite $\mathcal{W}$.

4.2 Functional Value Iteration

We now provide a functional value iteration technique for solving Risk-Sensitive POMDPs exactly. This technique backs up utility **functions** (unlike just reward values in value iteration) defined on the wealth over the entire time horizon. The key insight of this technique is to iteratively construct the finite partitioning of the $\mathcal{B} \times \mathcal{W}$ search space into regions where the value functions can be represented with point based policies. To this end, denote by Z^n a set of agent observation histories of length less than n. Also, for each decision epoch $0 \leq n \leq N$, we define a *point based policy* $\dot{\pi}^n$ as a function

$$\dot{\pi}^n : Z^{N-n} \rightarrow A \tag{4}$$

and the *expected utility to go* of $\dot{\pi}^n$ at some belief state and wealth level pair $(b, w) \in \mathcal{B} \times \mathcal{W}^n$ as a value:[1]

$$\upsilon\langle\dot{\pi}^n\rangle(b, w) := \mathbb{E}[U(w + \sum_{n'=n}^{N-1} r_{n'})|\dot{\pi}^n, b_0 = b]). \tag{5}$$

Let $\{\dot{\pi}_i^n\}_{i \in I(n)}$ be a collection of point-based policies such defined, for a decision epoch n. It is then obvious that any policy π can be represented as some (possibly infinite) collection of point-based policies. For example, to represent π in decision epoch n we may maintain a different point-based policy $\dot{\pi}_i^n$ for each $(b, w) \in \mathcal{B} \times \mathcal{W}^n$. In particular, to represent π^* in decision epoch n we may need to maintain a different point-based policy $\arg\max_{\dot{\pi}_i^n} \upsilon\langle\dot{\pi}_i^n\rangle(b, w)$ for each $(b, w) \in \mathcal{B} \times \mathcal{W}^n$. Fortunately, (as we show below) a *finite* collection $\{\dot{\pi}_i^n\}_{i \in I(n)}$ is sufficient to represent π^*, for each $0 \leq n < N$. That is, there exists a finite partitioning $\{\mathcal{Y}_i^n\}_{i \in I(n)}$ of $\mathcal{B} \times \mathcal{W}^n$ and a finite collection $\{\dot{\pi}_i^n\}_{i \in I(n)}$ such that $\upsilon\langle\dot{\pi}_i^n\rangle(b, w) = V_U^n(b, w)$ for all $(b, w) \in \mathcal{Y}_i^n$.

We now show how to find such finite collections $\{\dot{\pi}_i^n\}_{i \in I(n)}$ for $0 \leq n < N$ that represent π^*. Our technique assumes that the utility function $U(w)$ is piecewise linear over $w \in \mathcal{W}^N$ (or, that it has already been approximated with a piecewise linear function with a desired accuracy). Specifically, we assume that there exist wealth levels $\underline{w}^N = w_1 < \ldots < w_K = \overline{w}^N$ and pairs of constants $(C_1, D_1), \ldots, (C_K, D_K)$ such that $U(w) = C_k w + D_k$ for all $w \in [w_k, w_{k+1})$ over all $1 \leq k \leq K$.

For such U we now claim that, for all $0 \leq n \leq N$:

1. The value function V_U^n can be represented by a finite set of functions $\{\upsilon\langle\dot{\pi}_i^n\rangle\}_{i \in I(n)}$. That is, there exists a partitioning $\{\mathcal{Y}_i^n\}_{i \in I(n)}$ of $\mathcal{B} \times \mathcal{W}^n$ and a set of point-based policies $\{\dot{\pi}_i^n\}_{i \in I(n)}$ such that for all $(b, w) \in \mathcal{B} \times \mathcal{W}^n$ there exists $i \in I(n)$ such that $(b, w) \in \mathcal{Y}_i^n$ and $V_U^n(b, w) = \upsilon\langle\dot{\pi}_i^n\rangle(b, w) = \max_{i' \in I(n)} \upsilon\langle\dot{\pi}_{i'}^n\rangle(b, w)$.

2. For all $i \in I(n)$, $\upsilon\langle\dot{\pi}_i^n\rangle$ is piecewise bilinear. That is, there exists a finite partitioning $\{\mathcal{B} \times \mathcal{W}_{i,k}^n\}_{k \in I(n,i)}$ of $\mathcal{B} \times \mathcal{W}^n$ such that $\mathcal{W}_{i,k}^n$ is a convex set and for all $(b, w) \in \mathcal{B} \times \mathcal{W}_{i,k}^n$, $\upsilon\langle\dot{\pi}_i^n\rangle(b, w) = \sum_{s \in S} b(s)(c_{i,k,s}^n w + d_{i,k,s}^n)$, for all $k \in I(n, i)$;

3. For all $i \in I(n)$, $\upsilon\langle\dot{\pi}_i^n\rangle$ can be derived from the set of functions $\{\upsilon\langle\dot{\pi}_{i'}^{n+1}\rangle\}_{i' \in I(n+1)}$.

We prove claims 1,2,3 by induction on $n = N, \ldots, 0$. (The reader only interested in implementing our algorithm may wish to only implement the operations given by Equations (19), (9), (21), (24)—in that given order.)

Induction base: Assume $n = N$. Let $\mathcal{Y}_0^N := \mathcal{B} \times \mathcal{W}^N$, $I(N) := \{0\}$ and $\dot{\pi}_0^N$ be an arbitrary policy. Because at decision epoch N the process terminates, it holds for all $(b, w) \in \mathcal{Y}_0^N$ that (from Equations (2) and (5)) $V_U^N(b, w) = U(w) = \mathbb{E}[U(w)] = \mathbb{E}[U(w + \sum_{n=N}^{N-1} r_n)|\dot{\pi}_0^N, b_0 = b]) = \upsilon\langle\dot{\pi}_0^N\rangle(b, w) = \max_{i \in I(N)} \upsilon\langle\dot{\pi}_i^N\rangle(b, w)$, which proves claim 1. Furthermore, to prove that $\upsilon\langle\dot{\pi}_0^N\rangle$ is piecewise bilinear, let $I(N, 0) := \{1, \ldots, K\}$ and $\mathcal{W}_{0,k}^N := [w_k, w_{k+1})$, $k \in I(N, 0)$. Clearly, $\{\mathcal{B} \times \mathcal{W}_{0,k}^N\}_{k \in I(N,0)}$ is a finite partitioning of $\mathcal{B} \times \mathcal{W}^n$ and sets $\mathcal{W}_{0,k}^N, k \in I(N, 0)$ are convex. In addition, $\upsilon\langle\dot{\pi}_0^N\rangle(b, w) = \sum_s b(s)(C_k w + D_k) = C_k w + D_k$ for all $(b, w) \in \mathcal{B} \times \mathcal{W}_{0,k}^N$, $k \in I(N, 0)$ and hence, $\upsilon\langle\dot{\pi}_0^N\rangle(b, w)$ is linear—thus also piecewise bilinear—over $(b, w) \in \mathcal{B} \times \mathcal{W}^N$, which proves claim 2. Finally, claim 3 holds because we constructed $\upsilon\langle\dot{\pi}_0^N\rangle$ without even considering the set of functions $\{\upsilon\langle\dot{\pi}_{i'}^{N+1}\rangle\}_{i' \in I(N+1)}$ and our choice of $\dot{\pi}_0^N$ was arbitrary. The induction thus holds for $n = N$.

Induction step: Assume now that the induction holds for $n + 1$. Our goal is to prove that it also holds for n. To this end, recall from Equation (3) that $V_U^n(b, w)$ is calculated by

$$\max_{a \in A} \{\sum_{z \in Z} P(z|b, a) V_U^{n+1}\big(T(b, a, z), w + R(b, a)\big)\}.$$

We break this calculation into five **stages**. First, we calculate $V_{U,a,z}^n(b, w) := V_U^{n+1}\big(T(b, a, z), w\big)$ where V_U^{n+1} is represented by $\{\upsilon\langle\dot{\pi}_i^{n+1}\rangle\}_{i \in I(n+1)}$ from the induction assumption. Next, we derive $\overline{V}_{U,a,z}^n(b, w) := P(z|b, a) V_{U,a,z}^n(b, w)$ and then $V_{U,a}^n(b, w) := \sum_{z \in Z} \overline{V}_{U,a,z}^n(b, w)$. Finally, we derive $\overline{V}_{U,a}^n(b, w) := V_{U,a}^n(b, w + R(b, a))$ and conclude the proof of the induction step by deriving $V_U^n(b, w) := \max_{a \in A} \overline{V}_{U,a}^n(b, w)$ where V_U^n is represented by $\{\upsilon\langle\dot{\pi}_i^n\rangle\}_{i \in I(n)}$.

Stage 1: Calculate $V_{U,a,z}^n(b, w) := V_U^{n+1}\big(T(b, a, z), w\big)$.

From the induction assumption, V_U^{n+1} is represented by a finite set of functions $\{\upsilon\langle\dot{\pi}_i^{n+1}\rangle\}_{i \in I(n+1)}$, corresponding to point-based policies $\dot{\pi}_i, i \in I(n + 1)$, and each $\upsilon\langle\dot{\pi}_i^{n+1}\rangle$ is piecewise bilinear. We now prove that $V_{U,a,z}^n(b, w) := V_U^{n+1}\big(T(b, a, z), w\big)$ can be represented by a finite set of functions $\mathcal{V}_{a,z}^n = \{\upsilon_{a,z,i}^n\}_{i \in I(n+1)}$ derived from a collection of functions $\{\upsilon\langle\dot{\pi}_i^{n+1}\rangle\}_{i \in I(n+1)}$ and that each function $\upsilon_{a,z,i}^n$ is piecewise bilinear. To this end, define a finite partitioning $\{\mathcal{Y}_{a,z,i}^n\}_{i \in I(n+1)}$ of $\mathcal{B} \times \mathcal{W}^{n+1}$ where

$$\mathcal{Y}_{a,z,i}^n := \big\{(b, w) \in \mathcal{B} \times \mathcal{W}^{n+1} \mid \upsilon\langle\dot{\pi}_i^{n+1}\rangle(T(b, a, z), w) = \max_{i' \in I(n+1)} \upsilon\langle\dot{\pi}_{i'}^{n+1}\rangle(T(b, a, z), w)\big\} \tag{6}$$

and a finite set of functions $\mathcal{V}_{a,z}^n = \{\upsilon_{a,z,i}^n\}_{i \in I(n+1)}$ where

$$\upsilon_{a,z,i}^n(b, w) := \upsilon\langle\dot{\pi}_i^{n+1}\rangle(T(b, a, z), w) \tag{7}$$

for all $(b, w) \in \mathcal{B} \times \mathcal{W}^{n+1}$. It is then true that for all $(b, w) \in \mathcal{B} \times \mathcal{W}^{n+1}$ there exists $i \in I(n + 1)$ such that $(b, w) \in \mathcal{Y}_{a,z,i}^n$ and $\upsilon_{a,z,i}^n(b, w) := \upsilon\langle\dot{\pi}_i^{n+1}\rangle(T(b, a, z), w) =$

[1]Note, that $\upsilon\langle\dot{\pi}_i^n\rangle(b, w)$ is a function over $\mathcal{B} \times \mathcal{W}^n$.

$\max_{i'} v\langle\dot{\pi}_{i'}^{n+1}\rangle(T(b,a,z),w) = V_U^{n+1}(T(b,a,z),w) = V_{U,a,z}^n(b,w)$. Thus, $V_{U,a,z}^n(b,w)$ can be represented by a finite set of functions $\mathcal{V}_{a,z}^n = \{v_{a,z,i}^n\}_{i\in I(n+1)}$ derived from $\{v\langle\dot{\pi}_i^{n+1}\rangle\}_{i\in I(n+1)}$. In addition, each $v_{a,z,i}^n$ is piecewise bilinear as proven by Lemma 1 in the Appendix.

Finally, notice that if function $v_{a,z,i}^n \in \mathcal{V}_{a,z}^n$ is dominated by other functions $v_{a,z,i'}^n \in \mathcal{V}_{a,z}^n$, i.e., if for any $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ there exists $i' \in I(n+1), i' \neq i$ such that $v_{a,z,i}^n(b,w) < v_{a,z,i'}^n(b,w)$ then (from definition (6)) $\mathcal{Y}_{a,z,i}^n = \emptyset$. In such case (to speed up the algorithm) $v_{a,z,i}^n$ can be pruned from $\mathcal{V}_{a,z}^n$ and $\mathcal{Y}_{a,z,i}^n$ be removed from $\{\mathcal{Y}_{a,z,i}^n\}_{i\in I(n+1)}$ as that will not affect the representation of $V_{U,a,z}^n$. (How to determine if a function $v_{a,z,i}^n$ is dominated is explained later.) The value functions $V_{U,a,z}^n(b,w)$ can thus be represented by a finite sets of piecewise bilinear functions $\mathcal{V}_{a,z}^n = \{v_{a,z,i}^n\}_{i\in I(n,a,z)}$ where $I(n,a,z) \subset I(n+1)$.

Stage 2: Calculate $\overline{V}_{U,a,z}^n(b,w) := P(z|b,a)V_{U,a,z}^n(b,w)$. Consider the value functions $V_{U,a,z}^n(b,w)$ represented after stage 1 by finite sets of piecewise bilinear functions $\mathcal{V}_{a,z}^n = \{v_{a,z,i}^n\}_{i\in I(n,a,z)}$. We now demonstrate that the value function $\overline{V}_{U,a,z}^n(b,w) := P(z|b,a)V_{U,a,z}^n(b,w)$ can be represented by a set of piecewise bilinear functions $\overline{\mathcal{V}}_{a,z}^n = \{\overline{v}_{a,z,i}^n\}_{i\in I(n,a,z)}$ where

$$\overline{v}_{a,z,i}^n(b,w) := P(z|b,a)v_{a,z,i}^n(b,w) \tag{8}$$

for all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$. Indeed, since $\{\mathcal{Y}_{a,z,i}^n\}_{i\in I(n,a,z)}$ is a partitioning of $\mathcal{B}\times\mathcal{W}^{n+1}$ (from definition (6)), it holds for all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ that there exists $i \in I(n,a,z)$ such that $(b,w) \in \mathcal{Y}_{a,z,i}^n$ and $\overline{V}_{U,a,z}^n(b,w) := P(z|b,a)V_{U,a,z}^n(b,w) = P(z|b,a)v_{a,z,i}^n(b,w) = \overline{v}_{a,z,i}^n(b,w)$. Furthermore, each function $\overline{v}_{a,z,i}^n$ is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ because for the existing partitioning $\{\mathcal{B}\times\mathcal{W}_{i,k}^{n+1}\}_{k\in I(n+1,i)}$ of $\mathcal{B}\times\mathcal{W}^{n+1}$ it holds that

$$\begin{aligned}\overline{v}_{a,z,i}^n(b,w) &:= P(z|b,a)v_{a,z,i}^n(b,w)\\ &= P(z|b,a)\sum_{s\in S} b(s)(c_{a,z,i}^{n,k,s}w + d_{a,z,i}^{n,k,s})\\ &= \sum_{s\in S} b(s)(\overline{c}_{a,z,i}^{n,k,s}w + \overline{d}_{a,z,i}^{n,k,s})\end{aligned} \tag{9}$$

for all $(b,w) \in \mathcal{B}\times\mathcal{W}_i^{n+1}, k \in I(n+1,i)$ where $\overline{c}_{a,z,i}^{n,k,s} = P(z|b,a)c_{a,z,i}^{n,k,s}$ and $\overline{d}_{a,z,i}^{n,k,s} = P(z|b,a)d_{a,z,i}^{n,k,s}$ are constants.

Stage 3: Calculate $V_{U,a}^n(b,w) := \sum_{z\in Z}\overline{V}_{U,a,z}^n(b,w)$. Consider the value functions $\overline{V}_{U,a,z}^n$ represented after stage 2 by the sets of piecewise bilinear functions $\overline{\mathcal{V}}_{a,z}^n = \{\overline{v}_{a,z,i}^n\}_{i\in I(n,a,z)}$. We now show that $V_{U,a}^n$ can be represented with a finite set of piecewise bilinear functions $\mathcal{V}_a^n = \{v_{a,i}^n\}_{i\in I(n,a)}$ derived from the sets of functions $\overline{\mathcal{V}}_{a,z}^n = \{\overline{v}_{a,z,i}^n\}_{i\in I(n,a,z)}, z \in Z$. To this end, let $\mathbf{i} := [\mathbf{i}(z)]_{z\in Z} \in I(n,a)$ denote a vector where $\mathbf{i}(z) \in I(n,a,z), z \in Z$. For each such vector $\mathbf{i} \in I(n,a)$ define a set

$$\mathcal{Y}_{a,\mathbf{i}}^n := \bigcap_{z\in Z} \mathcal{Y}_{a,z,\mathbf{i}(z)}^n \tag{10}$$

and a function

$$v_{a,\mathbf{i}}^n(b,w) := \sum_{z\in Z} \overline{v}_{a,z,\mathbf{i}(z)}^n(b,w) \tag{11}$$

for all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$. To show that $V_{U,a}^n$ can be represented with a set of functions $\mathcal{V}_a^n = \{v_{a,\mathbf{i}}^n\}_{\mathbf{i}\in I(n,a)}$ we first prove that $\{\mathcal{Y}_{a,\mathbf{i}}^n\}_{\mathbf{i}\in I(n,a)}$ is a finite partitioning of $\mathcal{B}\times\mathcal{W}^{n+1}$. To this end, first observe that $\mathcal{Y}_{a,\mathbf{i}}^n \cap \mathcal{Y}_{a,\mathbf{i}'}^n = \emptyset$ for all $\mathbf{i},\mathbf{i}' \in I(n,a), \mathbf{i} \neq \mathbf{i}'$. Indeed, if $\mathbf{i} \neq \mathbf{i}'$ then $\mathbf{i}(z) \neq \mathbf{i}'(z)$ for some $z \in Z$. Thus, if $(b,w) \in \mathcal{Y}_{a,\mathbf{i}}^n \cap \mathcal{Y}_{a,\mathbf{i}'}^n$ then in particular $(b,w) \in \mathcal{Y}_{a,z,\mathbf{i}(z)}^n \cap \mathcal{Y}_{a,z,\mathbf{i}'(z)}^n$ which is impossible because $\mathcal{Y}_{a,z,\mathbf{i}(z)}^n \cap \mathcal{Y}_{a,z,\mathbf{i}'(z)}^n = \emptyset$ for $\mathbf{i}(z) \neq \mathbf{i}'(z)$ (from definition (6)). Also, if $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ then for all $z \in Z$ there exists some $\mathbf{i}(z) \in I(n,a,z)$ such that $(b,w) \in \mathcal{Y}_{a,z,\mathbf{i}(z)}^n$ (from definition (6)). Hence, for the vector $\mathbf{i} := [\mathbf{i}(z)]_{z\in Z} \in I(n,a)$ it must hold that $(b,w) \in \bigcap_{z\in Z}\mathcal{Y}_{a,z,\mathbf{i}(z)}^n = \mathcal{Y}_{a,\mathbf{i}}^n$.

We then show that $V_{U,a}^n$ can be represented with a set of functions $\mathcal{V}_a^n = \{v_{a,\mathbf{i}}^n\}_{\mathbf{i}\in I(n,a)}$ as follows: Since $\{\mathcal{Y}_{a,\mathbf{i}}^n\}_{\mathbf{i}\in I(n,a)}$ is a partitioning of $\mathcal{B}\times\mathcal{W}^{n+1}$, for each $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ there exists $\mathbf{i} = [\mathbf{i}(z)]_{z\in Z} \in I(n,a)$ such that $(b,w) \in \mathcal{Y}_{a,\mathbf{i}}^n$ and $V_{U,a}^n(b,w) := \sum_{z\in Z}\overline{V}_{U,a,z}^n(b,w) = \sum_{z\in Z}\overline{v}_{a,z,\mathbf{i}(z)}^n(b,w) = v_{a,\mathbf{i}}^n(b,w)$. In addition, each function $v_{a,\mathbf{i}}^n(b,w)$ is piecewise bilinear as proven by Lemma 2 in the Appendix.

Finally, notice that if function $v_{a,\mathbf{i}}^n \in \mathcal{V}_a^n$ is dominated by other functions $v_{a,\mathbf{i}'}^n \in \mathcal{V}_a^n$ then $\mathcal{Y}_{a,\mathbf{i}}^n = \emptyset$. Precisely, for any $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$, if there exists some other function $v_{a,\mathbf{i}'}^n \in \mathcal{V}_a^n$ such that $v_{a,\mathbf{i}}^n(b,w) < v_{a,\mathbf{i}'}^n(b,w)$ then (from definition 11) $\overline{v}_{a,z,\mathbf{i}(z)}^n(b,w) < \overline{v}_{a,z,\mathbf{i}'(z)}^n(b,w)$ for some $z \in Z$ and obviously (from definition (9)) $v_{a,z,\mathbf{i}(z)}^n(b,w) < v_{a,z,\mathbf{i}'(z)}^n(b,w)$ which implies that (from definition (6)) $(b,w) \notin \mathcal{Y}_{a,z,\mathbf{i}(z)}^n$ and obviously (from definition (10)), $(b,w) \notin \mathcal{Y}_{a,\mathbf{i}}^n$. Therefore (to speed up the algorithm), if function $v_{a,\mathbf{i}}^n \in \mathcal{V}_a^n$ is dominated by other functions $v_{a,\mathbf{i}'}^n \in \mathcal{V}_a^n$ then $v_{a,\mathbf{i}}^n$ can be pruned from $\mathcal{V}_a^n$ and set $\mathcal{Y}_{a,\mathbf{i}}^n$ be removed from $\{\mathcal{Y}_{a,\mathbf{i}}^n\}_{\mathbf{i}\in I(n,a)}$ as that will not affect the representation of $V_{U,a}^n$. (How to determine if a function $v_{a,\mathbf{i}}^n$ is dominated is explained later.)

Stage 4:
Calculate $\overline{V}_{U,a}^n(b,w) := V_{U,a}^n(b,w + R(b,a))$.

For notational convenience in this stage (but without the loss of precision), we denote vectors $\mathbf{i},\mathbf{k}$ defined in stage 3, as i,k. Recall that $\mathcal{W}^n$ is the set of all possible wealth levels at decision epoch n and that $\mathcal{W}^{n-1} = [\underline{w}^{n-1},\overline{w}^{n-1}] \subset [\underline{w}^n,\overline{w}^n] = \mathcal{W}^n$ where $\underline{w}^n = \underline{w}^{n-1} + \min_{s\in S,a\in A} R(s,a)$ and $\overline{w}^n = \overline{w}^{n-1} + \max_{s\in S,a\in A} R(s,a)$, for all $1 \leq n \leq N$. Hence, we only have to calculate the values $\overline{V}_{U,a}^n(b,w)$, $(b,w) \in \mathcal{B}\times\mathcal{W}^n$, from the values $V_{U,a}^n(b,w+R(b,a))$, $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$. To this end, we show how to represent $\overline{V}_{U,a}^n(b,w)$, $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ with a finite set of piecewise bilinear functions $\overline{\mathcal{V}}_a^n = \{\overline{v}_{a,i}^n : \mathcal{B}\times\mathcal{W}^n \to \mathbb{R}\}_{i\in I(n,a)}$ derived from the set of piecewise bilinear functions $\mathcal{V}_a^n = \{v_{a,i}^n : \mathcal{B}\times\mathcal{W}^{n+1} \to \mathbb{R}\}_{i\in I(n,a)}$ from stage 3. Formally, for each $i \in I(n,a)$ define a set

$$\begin{aligned}\overline{\mathcal{Y}}_{a,i}^n := \big\{&(b,w) \in \mathcal{B}\times\mathcal{W}^n \text{ such that}\\ &(b,w + R(b,a)) \in \mathcal{Y}_{a,i}^n\big\}\end{aligned} \tag{12}$$

and a function

$$\overline{v}_{a,i}^n(b,w) := v_{a,i}^n(b,w + R(b,a)). \tag{13}$$

To show that $\overline{V}_{U,a}^n$ can be represented by $\{\overline{\mathcal{V}}_a^n = \{\overline{v}_{a,i}^n\}_{i\in I(n,a)}$ we first need to prove that $\{\overline{\mathcal{Y}}_{a,i}^n\}_{i\in I(n,a)}$ is a finite partitioning of $\mathcal{B}\times\mathcal{W}^n$. Indeed, if $(b,w) \in \overline{\mathcal{Y}}_{a,i}^n \cap \overline{\mathcal{Y}}_{a,j}^n$ for some $i,j \in I(n,a)$ then $(b,w+R(b,a)) \in \mathcal{Y}_{a,i}^n \cap \mathcal{Y}_{a,j}^n$ and thus $i = j$ because $\{\mathcal{Y}_{a,i}^n\}_{i\in I(n,a)}$ is a partitioning of $\mathcal{B}\times\mathcal{W}^{n+1}$ (from stage 3). In addition, for any $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ we have that $(b,w + R(b,a)) \in \mathcal{B}\times\mathcal{W}^{n+1}$ (because $\min_{s\in S,a\in A} R(s,a) \leq R(b,a) \leq \max_{s\in S,a\in A} R(s,a)$ and thus, $(b,w + R(b,a)) \in$

$\mathcal{Y}^n_{a,i}$ for some $i \in I(n,a)$, which implies (from definition (12)) that $(b,w) \in \overline{\mathcal{Y}}^n_{a,i}$.

We then show that $\overline{V}^n_{U,a}(b,w)$ can be represented for all $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ with the set of functions $\overline{\mathcal{V}}^n_a = \{\overline{v}^n_{a,i}\}_{i\in I(n,a)}$ as follows: Since $\{\overline{\mathcal{Y}}^n_{a,i}\}_{i\in I(n,a)}$ is a finite partitioning of $\mathcal{B}\times\mathcal{W}^n$, for all $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ there exists $i \in I(n,a)$ such that $(b,w) \in \overline{\mathcal{B}}^n_{a,i}\times\overline{\mathcal{W}}^n_{a,i}$ and $\overline{V}^n_{U,a}(b,w) := V^n_{U,a}(b,w+R(b,a)) = v^n_{a,i}(b,w+R(b,a)) = \overline{v}^n_{a,i}(b,w)$. In addition, each function $\overline{v}^n_{a,i}(b,w) \in \overline{\mathcal{V}}^n_a$ is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ and can be derived from $v^n_{a,i} \in \mathcal{V}^n_a$, as shown in Lemma (3) in the Appendix.

Stage 5:
Calculate $V^n_U(b,w) := \max_{a\in A}\overline{V}^n_{U,a}(b,w)$.

Consider the value functions $\overline{V}^n_{U,a}$ represented after stage 4 by the set of piecewise bilinear functions $\overline{\mathcal{V}}^n_a = \{\overline{v}^n_{a,i}\}_{i\in I(n,a)}$. To conclude the proof of the induction step, we show how to represent V^n_U with a finite set of piecewise bilinear functions $\mathcal{V}^n = \{v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle\}_{(a,\mathbf{i})\in I(n)}$ derived from functions from sets $\overline{\mathcal{V}}^n_a, a \in A$. To this end, let $I(n) := \{(a,i)|a \in A, \mathbf{i} = [\mathbf{i}(z)]_{z\in Z} \in I(n,a)\}$. For each pair $(a,\mathbf{i}) \in I(n)$ then define a set

$$\begin{aligned}\mathcal{Y}^n_{(a,\mathbf{i})} := \{(b,w) \in \mathcal{B}\times\mathcal{W}^n \mid \overline{v}^n_{a,\mathbf{i}}(b,w) \\ = \max_{(a',\mathbf{i}')\in I(n)} \overline{v}^n_{a',\mathbf{i}'}(b,w)\}\end{aligned} \tag{14}$$

and a point based policy $\dot{\pi}^n_{(a,\mathbf{i})}$ according to which the agent first executes action $a \in A$ and then, depending on the observation $z \in Z$ received, follows the policy $\dot{\pi}^{n+1}_{\mathbf{i}(z)}$ given by the induction assumption.

Clearly, $\{\mathcal{Y}^n_{(a,\mathbf{i})}\}_{(a,\mathbf{i})\in I(n)}$ is a finite partitioning of $\mathcal{B}\times\mathcal{W}^n$. Thus, for all $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ there exists some $(a,\mathbf{i}) \in I(n)$ such that $(b,w) \in \mathcal{Y}^n_{(a,\mathbf{i})}$ and $V^n_U(b,w) := \max_{a'\in A}\overline{V}^n_{U,a'}(b,w) = \max_{(a',\mathbf{i}')\in I(n)}\overline{v}^n_{a',\mathbf{i}}(b,w) = \overline{v}^n_{a,\mathbf{i}}(b,w) = v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle(b,w)$ (the last equality follows directly from definitions (13) (11) (8) (7)). Therefore, V^n_U can indeed be represented by a finite set of piecewise bilinear functions $\mathcal{V}^n = \{v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle\}_{(a,\mathbf{i})\in I(n)} = \{\overline{v}^n_{a,i}\}_{(a,i)\in I(n)}$ derived (through stages 1,2,3,4,5) from functions $\{v\langle\dot{\pi}^{n+1}_{i'}\rangle\}_{i'\in I(n+1)}$, which proves claims 1, 2 and 3 of the induction step and the whole proof by induction.

Finally, notice that if a function $v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle \in \mathcal{V}^n$ is dominated by other functions $v\langle\dot{\pi}^n_{(a',\mathbf{i}')}\rangle \in \mathcal{V}^n$, i.e., if for all $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ there exists some $v\langle\dot{\pi}^n_{(a',\mathbf{i}')}\rangle \in \mathcal{V}^n$ such that $v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle(b,w) < v\langle\dot{\pi}^n_{(a',\mathbf{i}')}\rangle(b,w)$ then $\mathcal{Y}^n_{(a,\mathbf{i})} = \emptyset$. In such case, (to speed up the algorithm) $v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle$ can be pruned from $\mathcal{V}^n$ and $\mathcal{Y}^n_{(a,i)}$ be removed from $\{\mathcal{Y}^n_{(a,i)}\}_{(a,i)\in I(n)}$ as that will not affect the representation of V^n_U. (How to determine if function $v\langle\dot{\pi}^n_{(a,\mathbf{i})}\rangle$ is dominated is explained in the next section.)

5. PRUNING THE DOMINATED BILINEAR FUNCTIONS

In stages 1,3,5 of the proof by induction we indicated the possibility to speed up the algorithm by pruning from a set of piecewise bilinear functions these functions that are jointly dominated by other functions. The goal of this section is then to show how to quickly and accurately identify if a function is dominated or not. Formally, for a set of piecewise bilinear functions $\mathcal{V} = \{v_i : \mathcal{B}\times\mathcal{W} \to \mathbb{R}\}_{i\in I}$ we now show how to determine if some $v_j \in \mathcal{V}$ is dominated, i.e., if for all $(b,w) \in \mathcal{B}\times\mathcal{W}$ there exists $v_i \in \mathcal{V}$, $i \neq j$ such that $v_i(b,w) > v_j(b,w)$.

Let $v_i \in \mathcal{V}$ be piecewise bilinear over $\mathcal{B}\times\mathcal{W}$, i.e., there is a partitioning $\{\mathcal{B}\times\mathcal{W}_{i,k}\}_{1\le k\le K(i)}$ of $\mathcal{B}\times\mathcal{W}$ such that set $\mathcal{W}_{i,k}$ is convex and $v_i(b,w) = \sum_{s\in S} c^s_{i,k}w + d^s_{i,k}$ for all $(b,w) \in \mathcal{B}\times W_{i,k}$, $1 \le k \le K(i)$. Thus, there must exist wealth levels $\underline{w} = w_{i,0} < \ldots < w_{i,k} < \ldots < w_{i,K(i)} = \overline{w}$ such that $\mathcal{W}_{i,k} = [w_{i,k-1}, w_{i,k}]$ for all $1 \le k \le K(i)$. In determining whether $v_j \in \mathcal{V}$ is dominated we first split functions of $\mathcal{V}$ into functions defined over common wealth intervals. Precisely, let $W = \{w_k\}_{0\le k\le K} := \bigcup_{i\in I}\{w_{i,k}\}_{1\le k\le K(i)}$ be a set of common wealth levels where $\underline{w} = w_0 < \ldots < w_k < \ldots < w_K = \overline{w}$. For all $(b,w) \in \mathcal{B}\times[w_{k-1}, w_k]$, $1 \le k \le K$ we then represent $v_i(b,w)$ with $v_{i,k}(b,w) := \sum_{s\in S}\overline{c}^s_{i,k}w + \overline{d}^s_{i,k}$ where $\overline{c}^s_{i,k} := \overline{c}^s_{i,k'}$, $\overline{d}^s_{i,k} := \overline{d}^s_{i,k'}$ for k' such that $w \in [w_{i,k'-1}, w_{i,k'}]$, for all $i \in I$.

$v_j \in \mathcal{V}$ is then *not* dominated if there exists $1 \le k \le K$ and $(b,w) \in \mathcal{B}\times[w_{k-1}, w_k]$ such that for all $v_i \in \mathcal{V}$, $i \neq j$ it holds that $v_{i,k}(b,w) < v_{j,k}(b,w)$. That is, if for some $1 \le k \le K$ there exists a feasible solution $(\mathbf{b},\mathbf{w})$ to Program

$$\max 0 \left| \begin{array}{l} v_{j,k}(\mathbf{b},\mathbf{w}) - v_{i,k}(\mathbf{b},\mathbf{w}) > 0 \quad \forall v_i \in \mathcal{V} \\ w_{k-1} \le \mathbf{w} \le w_k \\ \sum_{s\in S}\mathbf{b}(s) = 1 \end{array} \right. \tag{15}$$

also written as

$$\max 0 \left| \begin{array}{l} \sum_{s\in S}\mathbf{b}(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) > 0 \quad \forall v_i \in \mathcal{V} \\ w_{k-1} \le \mathbf{w} \le w_k \\ \sum_{s\in S}\mathbf{b}(s) = 1 \end{array} \right. \tag{16}$$

where $\mathbf{b} = [\mathbf{b}(s)]_{s\in S}$ is a vector, $c^s_{i,j,k} := \overline{c}^s_{j,k} - \overline{c}^s_{i,k}$ and $d^s_{i,j,k} := \overline{d}^s_{j,k} - \overline{d}^s_{i,k}$. Unfortunately, Program (16) is hard to solve exactly, because of non-linear, non-convex constraints $\sum_{s\in S}\mathbf{b}(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) > 0$, $v_i \in \mathcal{V}$.

5.1 Error-free Pruning

Observe that, by relaxing the constraints of Program (16), we can only increase the chance of finding a feasible solution $(\mathbf{b},\mathbf{w})$ i.e, only decrease the chance of pruning v_j from $\mathcal{V}$. Therefore such a relaxation does not violate the optimality of the algorithm but rather, may result in keeping in $\mathcal{V}$ some of the dominated functions, which may slow down the algorithm. A relaxation that we propose approximates Program (16) with a linear program

$$\max 0 \left| \begin{array}{l} \sum_{s\in S}\mathbf{x}(s)c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} > 0 \quad \forall v_i \in \mathcal{V} \\ \mathbf{b}'(s)w_{k-1} \le \mathbf{x}(s) \le \mathbf{b}'(s)w_k \quad \forall s \in S \\ \sum_{s\in S}\mathbf{b}'(s) = 1 \end{array} \right. \tag{17}$$

where $\mathbf{b}' = [\mathbf{b}'(s)]_{s\in S}$ and $\mathbf{x} = [\mathbf{x}(s)]_{s\in S}$ are vectors. Program (17) relaxes Program (16) because for any feasible solution $(\mathbf{b},\mathbf{w})$ there exists a corresponding feasible solution $(\mathbf{b}' := \mathbf{b}, \mathbf{x} := \mathbf{bw})$. Indeed, if $\sum_{s\in S}\mathbf{b}(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) > 0$ in Program (16) then $\sum_{s\in S}\mathbf{b}(s)\mathbf{w}c^s_{i,j,k} + \mathbf{b}(s)d^s_{i,j,k} > 0$ and thus, $\sum_{s\in S}\mathbf{x}(s)c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} > 0$ in Program (17), for all $v_i \in \mathcal{V}$. Next, if $w_{k-1} \le \mathbf{w} \le w_k$ in Program (16) then for all $s \in S$, $\mathbf{b}(s)w_{k-1} \le \mathbf{b}(s)\mathbf{w} \le \mathbf{b}(s)w_k$ and thus $\mathbf{b}'(s)w_{k-1} \le \mathbf{x}(s) \le \mathbf{b}'(s)w_k$ in Program (17). Finally, if $\sum_{s\in S}\mathbf{b}(s) = 1$ then $\sum_{s\in S}\mathbf{b}'(s) = 1$. Conversely, a feasible solution $(\mathbf{b}',\mathbf{x})$ may not imply a corresponding feasible solution $(\mathbf{b},\mathbf{w})$. Specifically, even though $\sum_{s\in S}\mathbf{x}(s)c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} > 0$ in Program (17) implies that $\sum_{s\in S}\mathbf{b}'(s)([\mathbf{x}(s)/\mathbf{b}'(s)]c^s_{i,j,k} + d^s_{i,j,k}) > 0$, all the ratios

$[\mathbf{x}(s)/\mathbf{b}'(s)], s \in S$ would need to be equal to some unique $w_{k-1} \leq \mathbf{w} \leq w_k$ for $\sum_{s\in S} \mathbf{b}'(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) > 0$ to hold.

Because Program (17) relaxes Program (16), its decision to *not* prune υ_j from $\mathcal{V}$—a result of finding a feasible solution $(\mathbf{b}', \mathbf{x})$—may be too conservative. However, as we now show, the smaller the wealth interval $[w_{k-1}, w_k]$, the more accurate Program (17) becomes, that is, the greater the chance that a feasible solution $(\mathbf{b}', \mathbf{x})$ implies a feasible solution $(\mathbf{b}, \mathbf{w})$. To see it, for a given feasible solution $(\mathbf{b}, \mathbf{x})$, let $(\mathbf{b} := \mathbf{b}', \mathbf{w} := w_{k-1})$ be a *candidate* solution to Program (16). Clearly $\sum_{s\in S} \mathbf{b}(s) = 1$ and $w_{k-1} \leq \mathbf{w} \leq w_k$. In addition, for all $\upsilon_i \in \mathcal{V}$ it holds for $C^{max}_i := \max_{s\in S} |c^s_{i,j,k}|$ that

$$
\begin{aligned}
&(w_k - w_{k-1})C^{max}_i + \sum_{s\in S} \mathbf{b}(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) \\
&= \sum_{s\in S} \mathbf{b}'(s)(w_k - w_{k-1})C^{max}_i + \sum_{s\in S} \mathbf{b}'(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) \\
&\geq \sum_{s\in S} (\mathbf{x}(s) - \mathbf{b}'(s)w_{k-1})c^s_{i,j,k} + \sum_{s\in S} \mathbf{b}'(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) \\
&= \sum_{s\in S} \mathbf{x}(s)c^s_{i,j,k} - \mathbf{b}'(s)w_{k-1}c^s_{i,j,k} + \mathbf{b}'(s)w_{k-1}c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} \\
&= \sum_{s\in S} \mathbf{x}(s)c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} > 0
\end{aligned}
$$

and thus, $\lim_{w_k - w_{k-1}\to 0} \mathbf{Pr}\big[\sum_{s\in S} \mathbf{b}(s)(c^s_{i,j,k}\mathbf{w} + d^s_{i,j,k}) > 0\big] = 1$. Consequently, as $w_k - w_{k-1} \to 0$, the probability that a feasible solution $(\mathbf{b}', \mathbf{x})$ implies a *feasible* solution $(\mathbf{b}, \mathbf{w})$ approaches 1 and the error of approximating Program (16) with Program (17) approaches 0.

5.2 Error-bounded Pruning

We conclude this section by noting that, to speed up the algorithm, we can tighten the constraint $\sum_{s\in S} \mathbf{x}(s)c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} > 0$ of Program (17) by some $\epsilon > 0$. Specifically, we are less likely to find a feasible solution to Program

$$
\max 0 \left| \begin{array}{ll} \sum_{s\in S} \mathbf{x}(s)c^s_{i,j,k} + \mathbf{b}'(s)d^s_{i,j,k} > \epsilon & \forall \upsilon_i \in \mathcal{V} \\ \mathbf{b}'(s)w_{k-1} \leq \mathbf{x}(s) \leq \mathbf{b}'(s)w_k & \forall s \in S \\ \sum_{s\in S} \mathbf{b}'(s) = 1 \end{array} \right. \quad (18)
$$

than to Program (17) and thus, more likely to prune more functions from $\mathcal{V}$, which can speed up the algorithm. However, Program (18) may classify some of the non-dominated functions as dominated ones and hence, the pruning procedure will no longer be error-free. The total error of the algorithm, however, can trivially be bounded by $\epsilon \cdot 3 \cdot N$, where a tunable parameter ϵ of Program (18) is the error of the pruning procedure, 3 is the number of stages (of the proof by induction) that call the pruning procedure and N is the planning horizon.

6. EXPERIMENTS

We illustrate that our algorithm easily scales to larger extensions on the illustrative example problem provided in Section 2. We considered a bigger domain, where there are 100 different states of the market (primarily considering markets of different countries), considering 5 different actions to invest in markets of different countries. With respect to the algorithm, we tested with different values (0.5,1,1.5,2,2.5) of our approximation parameter ϵ (used in Program (18)). Also, the planning horizon was fixed at $N = 10$ and we run the algorithm for each utility function $(A), (B), (C), (D), (E)$ in Figure 2. We present our results in Figure 2, where ϵ is plotted on the $x-$axis whereas the runtime (in seconds on the logarithmic scale) and the solution quality are plotted on the $y-$axes. As can be seen, irrespective of the utility function considered, the algorithm runtime decreases drastically (with only small increases in ϵ) while the solution quality remains almost constant. For example, for the utility function (C), a change of ϵ from 0.5 to 1.5 caused the reduction of the algorithm runtime by over one order of magnitude (from $149s$ to only $12s$) and only 18% (from 9.08 to 7.38) decrease of the solution quality.

7. CONCLUSIONS AND RELATED WORK

Motivated by high-risk domains such as financial planning, in this paper we proposed Risk-Sensitive POMDPs, an extension of POMDPs that allows the agents to maximize the expected utility of their actions, and an exact algorithm for solving Risk-Sensitive POMDPs, for piecewise linear utility functions. The key idea of the algorithm is to represent the underlying value functions with sets of piecewise bilinear functions—computed exactly using functional value iteration—and to prune the dominated bilinear functions using efficient linear programming approximations of the underlying non-convex bilinear programs.

In terms of related work, POMDP solvers [7, 8, 12, 14, 15, 17, 18] are risk-invariant in that they maximize the expected *reward* of agent actions. One could argue that if $\mathcal{W}$ is finite then, rather than solving a Risk-Sensitive POMDP, the underlying risk-sensitive planning problems could be cast as POMDPs with augmented state-spaces $S' := S \times \mathcal{W}$ and transition, observation and reward functions modified accordingly. However, such an approach suffers from two problems: (a) State space increases considerably even for small number of wealth samples and so does the complexity of solving POMDPs; and (b) It is difficult to know before hand the number of wealth samples that would provide good quality solutions.

To remedy that, one could view $\mathcal{W}$ as a continuous state and use existing continuous POMDP solvers [5, 13] to find the underlying value functions. However, not only are these continuous POMDP solvers locally optimal, but in addition, it is unclear if they are directly applicable to risk-sensitive planning as the Bellman update operator they employ differs significantly from the operator given by Equation (3). In fact, only [10, 11] have provided the algorithms for solving planning problems where risk-sensitivity is addressed head-on (via one-switch or piecewise linear utility functions), yet, only for MDPs. Finally, beyond utility theory [19] there are other methods to express the agent's sensitivity towards risk [1]. However, as illustrated in [6], these methods typically cannot be handled by Bellman agents, even when the environment is fully observable.

8. ACKNOWLEDGMENTS

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defense and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defense or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and

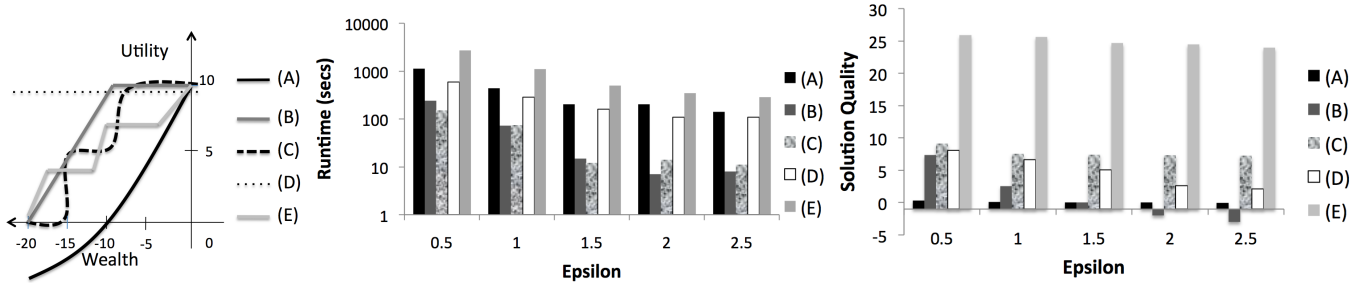

Figure 2: Runtime and solution quality comparison for different utility functions and approximation parameters ϵ.

distribute reprints for Government purposes notwithstanding any copyright notation hereon.

9. REFERENCES

[1] P. Artzner, F. Delbaen, J.-M. Eber, D. Heath, E. O. T. Hochschule, Z. Urich, and A. M. S-t. Coherent measures of risk, 1998.

[2] D. E. Bell. One-switch utility functions and a measure of risk. *Management Sciences*, 34(12):1416–1424, 1988.

[3] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, 1957.

[4] J. Bresina, R. Dearden, N. Meuleau, D. Smith, and R. Washington. Planning under continuous time and resource uncertainty: A challenge for AI. In *UAI*, pages 77–84, 2002.

[5] E. Brunskill, L. Kaelbling, T. Lozano-perez, and N. Roy. Continuous-state POMDPs with hybrid dynamics. In *ISAIM*, 2008.

[6] B. Defourny, D. Ernst, and L. Wehenkel. Risk-aware decision making and dynamic programming. In *NIPS 2008 Workshop on Model Uncertainty and Risk in RL*, 2008.

[7] Z. Feng and S. Zilberstein. Region-based incremental pruning for POMDPs. In *UAI*, pages 146–15, 2004.

[8] M. Hauskrecht. Value-function approximations for POMDPs. *JAIR*, 13:33–94, 2000.

[9] M. He and N. R. Jennings. Southamptontac: An adaptive autonomous trading agent. *ACM Trans. Internet Technol.*, 3(3):218–235, 2003.

[10] Y. Liu and S. Koenig. Functional value iteration for decision-theoretic planning with general utility functions. In *AAAI*, pages 1186–1193, 2006.

[11] Y. Liu and S. Koenig. An exact algorithm for solving MDPs under risk-sensitive planning objectives with one-switch utility functions. In *AAMAS*, pages 453–460, 2008.

[12] J. Pineau, G. Gordon, and S. Thrun. PBVI: An anytime algorithm for POMDPs. In *IJCAI*, pages 335–344, 2003.

[13] J. M. Porta, N. Vlassis, M. T. Spaan, and P. Poupart. Point-based value iteration for continuous POMDPs. *J. Mach. Learn. Res.*, 7:2329–2367, 2006.

[14] G. Roy, N.; Gordon. Exponential family PCA for belief compression in POMDPs. In *NIPS*, pages 1043–1049, 2002.

[15] T. Smith and R. Simmons. Point-based pomdp algorithms: Improved analysis and implementation. In *UAI*, 2005.

[16] E. J. Sondik. The optimal control of partially observable Markov processes. In *Ph.D Thesis*. Stanford University, 1971.

[17] M. T. J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *JAIR*, 24:195–220, 2005.

[18] P. Varakantham, R. Maheswaran, G. T., and M. Tambe. Towards efficient computation of error bounded solutions in POMDPs: Expected value approximation and dynamic disjunctive beliefs. In *IJCAI*, 2007.

[19] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

APPENDIX

LEMMA 1. *Function* $v^n_{a,z,i} := v\langle\dot{\pi}_i^{n+1}\rangle(T(b,a,z),w)$ *is piecewise bilinear over* $(b,w) \in \mathcal{B} \times \mathcal{W}^{n+1}$.

PROOF. From induction assumption, $v\langle\dot{\pi}_i^{n+1}\rangle(b,w)$ is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$, i.e., there exists a finite partitioning $\{\mathcal{B} \times \mathcal{W}^{n+1}_{i,k}\}_{k\in I(n+1,i)}$ of $\mathcal{B} \times \mathcal{W}^{n+1}$ such that $\mathcal{W}^{n+1}_{i,k}$ is a convex set and $v\langle\dot{\pi}_i^{n+1}\rangle(b,w) = \sum_{s\in S} b(s)(c^{n+1}_{i,k,s}w + d^{n+1}_{i,k,s})$ for all $(b,w) \in \mathcal{B} \times \mathcal{W}^{n+1}_{i,k}, k \in I(n+1,i)$. We now prove that $v^n_{a,z,i}(b,w) := v\langle\dot{\pi}_i^{n+1}\rangle(T(b,a,z),w)$ too is piecewise bilinear over $(b,w) \in \mathcal{B} \times \mathcal{W}^{n+1}$ for the partitioning $\{\mathcal{B} \times \mathcal{W}^{n+1}_{i,k}\}_{k\in I(n+1,i)}$ of $\mathcal{B} \times \mathcal{W}^{n+1}$. To this end, for each $s \in S$ distinguish a belief state $b_s \in \mathcal{B}$ such that $b_s(s) = 1$. It then holds for all $(b,w) \in \mathcal{B} \times \mathcal{W}^{n+1}_i, k \in I(n+1,i)$ that

$$
\begin{aligned}
v^n_{a,z,i}(b,w) &:= v\langle\dot{\pi}_i^{n+1}\rangle(T(b,a,z),w) \\
&= \sum_{s'\in S}[T(b,a,z)(s')](c^{n+1}_{i,k,s'}w + d^{n+1}_{i,k,s'}) \\
&= \sum_{s'\in S}\sum_{s\in S} b(s)[T(b_s,a,z)(s')](c^{n+1}_{i,k,s'}w + d^{n+1}_{i,k,s'}) \\
&= \sum_{s\in S} b(s) \sum_{s'\in S} P(s'|s,a)O(z|a,s')(c^{n+1}_{i,k,s'}w + d^{n+1}_{i,k,s'}) \\
&= \sum_{s\in S} b(s)(c^{n,k,s}_{a,z,i}w + d^{n,k,s}_{a,z,i}) \qquad (19)
\end{aligned}
$$

for constants $c^{n,k,s}_{a,z,i} = \sum_{s'\in S} P(s'|s,a)O(z|a,s')c^{n+1}_{i,k,s'}$ and $d^{n,k,s}_{a,z,i} = \sum_{s'\in S} P(s'|s,a)O(z|a,s')d^{n+1}_{i,k,s'}$. Consequently, function $v^n_{a,z,i}(b,w)$ is piecewise bilinear over $(b,w) \in \mathcal{B} \times \mathcal{W}^{n+1}$ which proves the Lemma. □

LEMMA 2. *Function* $v^n_{a,\mathbf{i}}(b,w) := \sum_{z\in Z} \overline{v}^n_{a,z,\mathbf{i}(z)}(b,w)$ *is piecewise bilinear over* $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$.

PROOF. After stage 2 it holds for all $z \in Z$ that $\overline{v}^n_{a,z,\mathbf{i}(z)}(b,w)$ is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$, i.e., there exist a partitioning $\{\mathcal{B}\times\mathcal{W}^{n+1}_{\mathbf{i}(z),k}\}_{k\in I(n+1,\mathbf{i}(z))}$ of $\mathcal{B}\times\mathcal{W}^{n+1}$ such that $\mathcal{W}^{n+1}_{\mathbf{i}(z),k}$ is a convex set and $\overline{v}^n_{a,z,\mathbf{i}(z)}(b,w) = \sum_{s\in S} b(s)(\overline{c}^{n,k,s}_{a,z,\mathbf{i}(z)}w + \overline{d}^{n,k,s}_{a,z,\mathbf{i}(z)})$ for all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}_{\mathbf{i}(z),k}, k \in I(n+1,\mathbf{i}(z))$. To prove that $v^n_{a,\mathbf{i}}(b,w) := \sum_{z\in Z}\overline{v}^n_{a,z,\mathbf{i}(z)}(b,w)$ too is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ we represent $v^n_{a,\mathbf{i}}$ with the set of bilinear functions $\{v^n_{a,\mathbf{i},\mathbf{k}}\}_{\mathbf{k}\in I(n,a,\mathbf{i})}$. Precisely, let $\mathbf{k} := [\mathbf{k}(z)]_{z\in Z} \in I(n,a,\mathbf{i})$ denote a vector where $\mathbf{k}(z) \in I(n+1,\mathbf{i}(z))$. For each vector $\mathbf{k} \in I(n,a,\mathbf{i})$ we define a set

$$\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}} := \bigcap_{z\in Z} \mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)} \tag{20}$$

and a bilinear function

$$v^n_{a,\mathbf{i},\mathbf{k}}(b,w) := \sum_{s\in S} b(s)(c^{n,\mathbf{k},s}_{a,\mathbf{i}}w + d^{n,\mathbf{k},s}_{a,\mathbf{i}}) \tag{21}$$

for all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ and constants $c^{n,\mathbf{k},s}_{a,\mathbf{i}} := \sum_{z\in Z}\overline{c}^{n,\mathbf{k}(z),s}_{a,z,\mathbf{i}(z)}$, $d^{n,\mathbf{k},s}_{a,\mathbf{i}} := \sum_{z\in Z}\overline{d}^{n,\mathbf{k}(z),s}_{a,z,\mathbf{i}(z)}$. To show that $v^n_{a,\mathbf{i}}(b,w)$ can be represented by $\{v^n_{a,\mathbf{i},\mathbf{k}}(b,w)\}_{\mathbf{k}\in I(n,a,\mathbf{i})}$ over all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ we first prove that $\{\mathcal{B}\times\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}\}_{\mathbf{k}\in I(n,a,\mathbf{i})}$ is a finite partitioning of $\mathcal{B}\times\mathcal{W}^{n+1}$. To this end, first observe that $\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}} \cap \mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}'} = \emptyset$ for any $\mathbf{k},\mathbf{k}' \in I(n,a,\mathbf{i}), \mathbf{k} \neq \mathbf{k}'$. Indeed, if $\mathbf{k} \neq \mathbf{k}'$ then $\mathbf{k}(z) \neq \mathbf{k}'(z)$ for some $z \in Z$. Hence, if $w \in \mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}} \cap \mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}'}$ then in particular $w \in \mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)} \cap \mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}'(z)}$ which is cannot be true as $\mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)} \cap \mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}'(z)} = \emptyset$ for $\mathbf{k}(z) \neq \mathbf{k}'(z)$ (from claim 2 of the induction assumption). Also, observe that for any $w \in \mathcal{W}^{n+1}$ there must exist $\mathbf{k} \in I(n,a,\mathbf{i})$ such that $w \in \mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}$, because for all $z \in Z$, there exists $\mathbf{k}(z) \in I(n+1,\mathbf{i}(z))$ such that $w \in \mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)}$ (since $\{\mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)}\}_{\mathbf{k}(z)\in(n+1,\mathbf{i}(z))}$ is a partitioning of $\mathcal{W}^{n+1}$, from claim 2 of the induction assumption). Thus, vector $\mathbf{k} := [\mathbf{k}(z)]_{z\in Z} \in I(n,a,\mathbf{i})$ such that $w \in \bigcap_{z\in Z}\mathcal{W}^{n+1}_{a,\mathbf{i}(z),\mathbf{k}(z)} = \mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}$ truly exists. Consequently, $\{\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}\}_{\mathbf{k}\in I(n,a,\mathbf{i})}$ is a finite partitioning of $\mathcal{W}^{n+1}$ and $\{\mathcal{B}\times\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}\}_{\mathbf{k}\in I(n,a,\mathbf{i})}$ a finite partitioning of $\mathcal{B}\times\mathcal{W}^{n+1}$.

We can therefore prove that functions $\{v^n_{a,\mathbf{i},\mathbf{k}}\}_{\mathbf{k}\in I(n,a,\mathbf{i})}$ represent $v^n_{a,\mathbf{i}}(b,w)$ over all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ as follows: For each $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$ there exists $\mathbf{k} \in I(n,a,\mathbf{i})$ such that $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}$. Hence, (from definition (20)) $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)}$ and thus, (from definition (9)) $\overline{v}^n_{a,z,\mathbf{i}(z)}(b,w) = \sum_{s\in S} b(s)(\overline{c}^{n,\mathbf{k}(z),s}_{a,z,\mathbf{i}(z)}w + \overline{d}^{n,\mathbf{k}(z),s}_{a,z,\mathbf{i}(z)})$. We can then easily prove that $v^n_{a,\mathbf{i}}(b,w) := \sum_{z\in Z}\overline{v}^n_{a,z,\mathbf{i}(z)}(b,w) = \sum_{z\in Z}\sum_{s\in S} b(s)(\overline{c}^{n,\mathbf{k}(z),s}_{a,z,\mathbf{i}(z)}w + \overline{d}^{n,\mathbf{k}(z),s}_{a,z,\mathbf{i}(z)}) = \sum_{s\in S}(c^{n,\mathbf{k},s}_{a,\mathbf{i}}w + d^{n,\mathbf{k},s}_{a,\mathbf{i}}) = v^n_{a,\mathbf{i},\mathbf{k}}(b,w)$. Finally, each set $\mathcal{W}^{n+1}_{a,\mathbf{i},\mathbf{k}}$ is convex because (from definition (20)) it is an intersection of convex sets $\mathcal{W}^{n+1}_{\mathbf{i}(z),\mathbf{k}(z)}$, $z \in Z$. □

LEMMA 3. *Function* $\overline{v}^n_{a,i}(b,w) := v^n_{a,i}(b,w + R(b,a))$ *is piecewise bilinear over* $(b,w) \in \mathcal{B}\times\mathcal{W}^n$.

PROOF. After stage 3 it is true for all $i \in I(n,a)$ that $v^n_{a,i}(b,w)$ is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}$, i.e., there exist a partitioning $\{\mathcal{B}\times\mathcal{W}^{n+1}_{a,i,k}\}_{k\in I(n,a,i)}$ of $\mathcal{B}\times\mathcal{W}^{n+1}$ such that $\mathcal{W}^{n+1}_{a,i,k}$ is convex and $v^n_{a,i}(b,w) = v^n_{a,i,k}(b,w) = \sum_{s\in S} b(s)(c^{n,k,s}_{a,i}w + d^{n,k,s}_{a,i})$ for all $(b,w) \in \mathcal{B}\times\mathcal{W}^{n+1}_{a,i,k}$, for all $k \in I(n,a,i)$. To prove that $\overline{v}^n_{a,i}(b,w) := v^n_{a,i}(b,w + R(b,a))$ is piecewise bilinear over $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ we represent $\overline{v}^n_{a,i}$ with a set of bilinear functions $\{\overline{v}^n_{a,i,\mathbf{k}}\}_{\mathbf{k}\in\overline{I}(n,a,i)}$. To this end, first, for each $k \in I(n,a,i)$, $s \in S$ define a set

$$\overline{\mathcal{W}}^{n,s}_{a,i,k} := \{w \in \mathcal{W}^n \mid w + R(s,a) \in \mathcal{W}^{n+1}_{a,i,k}\}. \tag{22}$$

Now, let $\mathbf{k} := [\mathbf{k}(s)]_{s\in S}$ denote a vector where $\mathbf{k}(s) \in I(n,a,i)$. $\overline{I}(n,a,i)$ is a set of all such vectors $\mathbf{k}$. For each vector $\mathbf{k} \in \overline{I}(n,a,i)$ then define a set

$$\overline{\mathcal{W}}^n_{a,i,\mathbf{k}} := \bigcap_{s\in S} \overline{\mathcal{W}}^{n,s}_{a,i,\mathbf{k}(s)} \tag{23}$$

and a bilinear function

$$\overline{v}^n_{a,i,\mathbf{k}}(b,w) := \sum_{s\in S} b(s)(\overline{c}^{n,\mathbf{k}(s),s}_{a,i}w + \overline{d}^{n,\mathbf{k}(s),s}_{a,i}) \tag{24}$$

for all $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ where $\overline{c}^{n,\mathbf{k}(s),s}_{a,i} := c^{n,\mathbf{k}(s),s}_{a,i}$ and $\overline{d}^{n,\mathbf{k}(s),s}_{a,i} := d^{n,\mathbf{k}(s),s}_{a,i} + c^{n,\mathbf{k}(s),s}_{a,i}R(s,a)$ are constants. To show that $\overline{v}^n_{a,i}$ can be represented by $\{\overline{v}^n_{a,i,\mathbf{k}}\}_{\mathbf{k}\in\overline{I}(n,a,i)}$ we first prove that $\{\overline{\mathcal{W}}^n_{a,i,\mathbf{k}}\}_{\mathbf{k}\in\overline{I}(n,a,i)}$ is a finite partitioning of $\mathcal{W}^n$. Indeed, for any $\mathbf{k},\mathbf{k}' \in \overline{I}(n,a,i)$ if $w \in \overline{\mathcal{W}}^n_{a,i,\mathbf{k}} \cap \overline{\mathcal{W}}^n_{a,i,\mathbf{k}'}$ then (from definition (23)) for all $s \in S$, $w \in \overline{\mathcal{W}}^{n,s}_{a,i,\mathbf{k}(s)} \cap \overline{\mathcal{W}}^{n,s}_{a,i,\mathbf{k}'(s)}$ and thus (from definition (22)) $w + R(s,a) \in W^{n+1}_{a,i,\mathbf{k}(s)} \cap W^{n+1}_{a,i,\mathbf{k}'(s)}$ for all $s \in S$, which can only hold if $\mathbf{k} = \mathbf{k}'$ (because $\{W^{n+1}_{a,i,\mathbf{k}(s)}\}_{\mathbf{k}(s)\in I(n,a,i)}$ is a partitioning of $\mathcal{W}^{n+1}$). In addition, for any $w \in \mathcal{W}^n$, $s \in S$ it holds that $w + R(s,a) \in \mathcal{W}^{n+1}$ and thus, there must exists some $\mathbf{k}(s) \in I(n,a,i)$ such that $w + R(s,a) \in \mathcal{W}^{n+1}_{a,i,\mathbf{k}(s)}$. Therefore (from definition (22)) $w \in \overline{\mathcal{W}}^{n,s}_{a,i,\mathbf{k}(s)}$ for all $s \in S$ and thus (from definition (23)) $w \in \overline{\mathcal{W}}^n_{a,i,\mathbf{k}}$. We have therefore proven that $\{\overline{\mathcal{W}}^n_{a,i,\mathbf{k}}\}_{\mathbf{k}\in\overline{I}(n,a,i)}$ is a finite partitioning of $\mathcal{W}^n$ and that $\{\mathcal{B}\times\overline{\mathcal{W}}^n_{a,i,\mathbf{k}}\}_{\mathbf{k}\in\overline{I}(n,a,i)}$ is a finite partitioning of $\mathcal{B}\times\mathcal{W}^n$.

We then show that functions $\{\overline{v}^n_{a,i,\mathbf{k}}\}_{\mathbf{k}\in\overline{I}(n,a,i)}$ represent $\overline{v}^n_{a,i}(b,w)$ over all $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ as follows: For each $(b,w) \in \mathcal{B}\times\mathcal{W}^n$ there must exist $\mathbf{k} \in \overline{I}(n,a,i)$ such that $(b,w) \in \mathcal{B}\times\overline{\mathcal{W}}^n_{a,i,\mathbf{k}}$ and $(b,w + R(s,a)) \in \mathcal{B}\times\overline{\mathcal{W}}^{n+1}_{a,i,\mathbf{k}(s)}$ $\forall s \in S$, for which it holds that[2]

$$\begin{aligned}
\overline{v}^n_{a,i}(b,w) &:= v^n_{a,i}(b,w + R(b,a))\\
&= \sum_{s\in S} b(s) v^n_{a,i}(b_s, w + R(b_s,a))\\
&= \sum_{s\in S} b(s) \sum_{s'\in S} b_s(s')(c^{n,\mathbf{k}(s),s'}_{a,i}(w + R(s,a)) + d^{n,\mathbf{k}(s),s'}_{a,i})\\
&= \sum_{s\in S} b(s)(c^{n,\mathbf{k}(s),s}_{a,i}w + c^{n,\mathbf{k}(s),s}_{a,i}R(s,a) + d^{n,\mathbf{k}(s),s}_{a,i})\\
&= \sum_{s\in S} b(s)(\overline{c}^{n,\mathbf{k}(s),s}_{a,i}w + \overline{d}^{n,\mathbf{k}(s),s}_{a,i}) = \overline{v}^n_{a,i,\mathbf{k}}(b,w)
\end{aligned} \tag{25}$$

Finally, each set $\overline{\mathcal{W}}^n_{a,i,\mathbf{k}}$ is convex because it is an intersection of convex sets $\overline{\mathcal{W}}^{n,s}_{a,i,\mathbf{k}(s)}$, $s \in S$ (translation of a convex set $\mathcal{W}^{n+1}_{a,i,\mathbf{k}(s)}$ by a vector $R(s,a)$ results in a convex set). □

[2]Recall that for each $s \in S$ we distinguish $b_s \in \mathcal{B}$ such that $b_s(s) = 1$.