Bounded Backward Induction for Max-Min Dynamic Programs

Justin C. Goodson Luca Bertazzi

July 22, 2024

Abstract

Max-min dynamic programs model sequential decision problems where policies are evaluated via the worst-case reward across a set of scenarios representing future uncertainty. In the standard backward induction algorithm for max-min dynamic programs, the effort required to identify an optimal policy grows exponentially with the number of state variables. We develop a *bounded backward induction* (BBI) procedure that uses upper and lower bounds on rewards-to-go to curb this exponential growth by eliminating suboptimal decisions. BBI shifts the problem of dimensionality to the task of identifying strong and tractable bounds. We propose a dual bounding technique that reduces the uncertainty faced by the decision maker. The technique leads to a family of dual bounds that vary in strength and in the computational effort required to obtain them. Policies are recovered by embedding dual bounds in a lookahead procedure, which in turn yields a policy performance guarantee. For budget-style scenario sets, we provide results that ease the optimization required for policy evaluation and dual bound calculation. We demonstrate the utility of BBI via application to a media selection problem with yield uncertainty. Using our general bounding procedures, BBI identifies optimal policies for problem instances orders of magnitude larger than what is tractable with conventional backward induction.

1 Introduction

Dynamic programs (DPs) model decisions in sequence and in the face of uncertainty. Such problems are found in many domains, including business, engineering, the sciences, and health care. When outcomes are not probabilistic, or when distributional information is not readily available, uncertainty is normally modeled as a set of scenarios. This leads to a max-min DP whose objective is to identify a decision policy that maximizes the worst-case reward across the scenario set (Bertsekas, 2017, ch. 1.6). In contrast to stochastic DPs, which enjoy a growing collection of methods to design policies (Bertsekas, 2019a; Sutton and Barto, 2020; Powell, 2022) and dual bounds (Adelman and Mersereau, 2008; Brown et al., 2010; Brown and Smith, 2014; Ye et al., 2018; Balseiro and Brown, 2019), max-min DPs have received relatively little attention (Bertsekas, 2022). Indeed, despite the prevalence of sequential decision problems, solution methods for max-min DPs are limited.

Backward induction is the standard methodology for solving max-min DPs with finite horizons, state spaces, action spaces, and scenario sets (Bertsekas, 2017, ch. 1.6). It suffers from the so-called "curse of dimensionality." The effort required to execute the procedure depends on the size of the state space, which grows exponentially with the number of state variables. In this paper, we develop a *bounded backward induction* (BBI) procedure that uses upper and lower bounds on rewards-to-go to curb exponential growth by eliminating suboptimal decisions. It shifts the problem of dimensionality to the task of developing strong and tractable bounds.

We propose a dual bounding technique that reduces the uncertainty faced by the decision maker. It draws on the notions of partial information relaxations in stochastic DPs (Brown et al., 2010) and refinement chains in multistage stochastic programs (Maggioni and Pflug, 2016). The technique leads to a family of dual bounds that vary in strength and in the computational effort required to obtain them. Policies are recovered by embedding dual bounds in a lookahead procedure. The approach leads to a policy performance guarantee that connects lower bounds, upper bounds, and the optimal policy value across consecutive stages of the DP. A major deterrent to solving max-min DPs is the optimization required to assess the value of a policy. This is in contrast to stochastic DPs, where expected policy values are easily estimated via simulation. For budget-style scenario sets, we partially characterize optimal solutions to the policy evaluation problem. The result reduces the

effort required to evaluate policies and to calculate dual bounds. Together, dual bounds and policy values facilitate BBI.

Our computational experience makes a compelling case for BBI. We use it to identify optimal policies for a media selection problem with yield uncertainty. In this problem, audience exposure is reduced whenever purchased spots are randomly bumped. The objective is to maximize worst-case audience exposure subject to spot limits and a budget constraint. The problem is of significant practical interest, but can be difficult to solve with conventional backward induction. Using our general methods for upper and lower bounds, BBI solves max-min DP instances orders of magnitude larger than what is tractable with conventional backward induction. This is notable because our bounding methods are independent of the problem. Results demonstrate how stronger bounds reduce the dimensionality of BBI, but require more computational effort. We also highlight the benefits of our policy performance guarantees and show how our analysis of budget scenario sets can simplify computation. These results point toward BBI as a useful solution methodology for max-min DPs.

The paper's contributions are broad. BBI is a general framework for solving max-min DPs with finite horizons, state spaces, action spaces, and scenario sets. By itself, however, BBI is not fully operational. It requires the specification of upper and lower bounds on rewards-to-go. The dual bounds, policies, and performance guarantee are valid for any max-min DP, and our analysis of budget-style scenario sets is widely applicable. Indeed, these results are contributions in and of themselves. But using our bounds is not the only way to implement BBI. For example, the field of reinforcement learning offers many techniques to obtain policies. Additionally, problem-specific analyses may lead to specialized insights for dual bound generation. Our hope is that the general methodologies developed in this paper will facilitate more widespread use of max-min DPs to address sequential decision problems.

The paper proceeds as follows. In §2, we review related literature. We formalize a max-min DP model in §3. We introduce BBI in §4. Dual bounds are developed in §5. In §6, we discuss lookahead policies. In §7, we examine budget scenarios. In §8, we illustrate BBI via application to a media selection problem. We conclude the paper in §9.

2 Related Literature

Max-min DPs and related concepts cut across several areas. Bertsekas (2019b) and Bertsekas (2021) represent the state of the art in max-min DP methodology. They develop a policy iteration procedure to solve robust shortest path problems, where the horizon is potentially infinite. While these papers establish vital theory, their methods face the familiar "curse of dimensionality." For max-min DPs with a finite horizon, BBI offers a more tractable alternative.

Max-min DPs arise in adverserial games, such as chess. Here, the popular alpha-beta pruning technique uses notions of bounds to improve the minimax algorithm. As part of a depth-first search, the procedure tracks the best value obtained so far by each player. These values can be used to reduce the size of the search tree by up to half, in the best case (Pearl, 1980, 1982). In contrast, BBI does not require depth-first search, our theory works with general bounds, and we observe substantially larger reductions in dimensionality.

DPs also model zero-sum games with two players and a dynamic system. Here, the problem of optimal decision making may be modeled as a max-min DP for one player and a min-max DP for the other. The typical aim is to leverage circumstances where the value of the max-min problem is equal to the value of the min-max problem. In general, these values are usually different (Bertsekas, 2019a). Further, BBI does not require that they be equal.

The idea of optimizing across a set of scenarios is also fundamental to robust optimization (Delage and Iancu, 2015). Though much of the literature in this area centers on static decision policies, adjustable robust optimization seeks dynamic decisions in response to uncertainty. Although Shapiro (2011) connects adjustable robust optimization with max-min DPs, the field's focus on tractability typically restricts the form of decision rules, e.g., to affine or piecewise constant functions (Bertsimas and Brown, 2011). BBI does not require that decision rules conform to a particular shape, and thus BBI may identify better policies. Indeed, the current limitations of adjustable robust optimization are a motivation for the research in this paper.

Decision making via a max-min criterion arises in two additional contexts. Stochastic games are similar to max-min DPs, except the environment changes probabilistically at each stage (Haugh and Wang, 2015; Bertsekas, 2021). In robust DPs, policies are assessed as the worst-case expected value across uncertain payoff parameters and transition probabilities (Iyengar, 2005; Nilim and

El Ghaoui, 2005; Xu and Mannor, 2006; Delage and Mannor, 2010; Goyal and Grand-Clément, 2023). In both cases, the decision maker has access to distributional information, even if that information is ambiguous. In this paper, we treat the situation where uncertainty is described only as a set of scenarios (Bertsekas, 2017, ch. 1.6).

3 Model

We consider max-min DPs within the framework of Bertsekas (2017, ch. 1.6). Decisions are made across a finite horizon at epochs $t = 1, \ldots, T$. Epoch t marks the beginning of period t at which time the system occupies state s_t and the decision maker chooses an action x_t from the finite set of actions $X_t(s_t)$ available in state s_t. Following action selection, outcome w_t in uncertainty set $W_t(s_t, x_t)$ is observed. The decision maker faces uncertainty in the form of a finite scenario set $\boldsymbol{W}$. Each scenario $\boldsymbol{w} = (w_1, \ldots, w_T)$ in $\boldsymbol{W}$ is a trajectory of possible outcomes across all epochs. In state s_t, the set of possible scenarios is $\boldsymbol{W}_t(s_t)$. Uncertainty set $W_t(s_t, x_t)$ consists of period-t outcomes in $\boldsymbol{W}_t(s_t)$ that may result from selecting action x_t in state s_t. The reward $r_t(s_t, x_t, w_t)$ earned in period t is a function of the state, selected action, and observed outcome. A transition from state s_t in epoch t to state $s_{t+1} = S(s_t, x_t, w_t)$ in epoch $t + 1$ depends on the same. Reward $r_{T+1}(s_{T+1})$ is accrued in terminal state s_{T+1}. A policy π is a sequence of decision rules $(\mu_1^\pi, \ldots, \mu_T^\pi)$ where each rule $\mu_t^\pi(s_t) : s_t \to X_t(s_t)$ is a function that maps the current state to an action. A policy is evaluated from state s_t as the worst-case reward across scenarios in $\boldsymbol{W}_t(s_t)$:

$$J_t^\pi(s_t) = \min_{\boldsymbol{w} \in \boldsymbol{W}_t(s_t)} \left\{ r_{T+1}(s_{T+1}) + \sum_{t'=t}^{T} r_{t'}\left(s_{t'}, \mu_{t'}^\pi(s_{t'}), w_{t'}\right) \right\}. \tag{1}$$

Denote by Π the set of all policies. The value of an optimal policy from state s_t maximizes the worst-case reward:

$$J_t(s_t) = \max_{\pi \in \Pi} \left\{ J_t^\pi(s_t) \right\}. \tag{2}$$

We seek an optimal policy from initial state s_1 with value $J_1(s_1)$.

4 Bounded Backward Induction

Bertsekas (2017, ch. 1.6) formalizes value functions as a recursive method to identify optimal policies for max-min DPs. Bertsekas shows that $J_t(s_t)$ may be calculated as

$$V_t(s_t) = \max_{x_t \in X_t(s_t)} \left\{ \min_{w_t \in W_t(s_t, x_t)} \{ r_t(s_t, x_t, w_t) + V_{t+1}(s_{t+1}) \} \right\}, \tag{3}$$

for $t = 1, \ldots, T$, with $V_{T+1} = r_{T+1}(s_{T+1})$. Calculating $V_t(s_t)$ is accomplished in two steps. First, a decision tree is constructed that contains every trajectory of states that might be observed beginning from state s_t and ending at some final state s_{T+1}. This is done in forward fashion by enumerating feasible actions and outcomes to identify possible future states in each subsequent period. Second, the value function recursion is executed on the decision tree in backward fashion: $V_{T+1}(s_{T+1})$ is calculated for each state s_{T+1} in period $T+1$, $V_T(s_T)$ is calculated for each state s_T in period T, and so on until $V_t(s_t)$ is calculated for state s_t in period t. This two-step procedure is the conventional backward induction algorithm for max-min DPs.

As is common in dynamic programming, the size of the decision tree in the first step can be prohibitively large. Consequently, solving a max-min DP via backward induction may be intractable. We seek to overcome this "curse of dimensionality" by using lower and upper bounds on the rewards-to-go to identify suboptimal actions and thus significantly reduce the number of states in the decision tree. We refer to this method of decision tree construction followed by backward solution of the value functions as *bounded backward induction* (BBI).

BBI leverages two results. Let $\underline{V_t}(s_t) \leqslant V_t(s_t)$ be a lower bound on the reward-to-go from state s_t and let $\overline{V}_t(s_t) \geqslant V_t(s_t)$ be an upper bound. The first result is the straightforward observation that if $\underline{V_t}(s_t) = \overline{V}_t(s_t)$, then the value function is sandwiched by the bounds. Because $\underline{V_t}(s_t) = V_t(s_t) = \overline{V}_t(s_t)$, the reward-to-go is known and it is not necessary to extend the decision tree in the direction of state s_t. Second, if $\underline{V_t}(s_t)$ is less than $\overline{V}_t(s_t)$, then Theorem 1 provides a check on actions. Let $x_t^\star$ be an action in $X_t(s_t)$ that achieves $V_t(s_t)$ and let $\overline{V}_{t+1}(s_{t+1}) \geqslant V_{t+1}(s_{t+1})$ be an upper bound on the reward-to-go from state s_{t+1}. Theorem 1 asserts that for all outcomes in $W_t(s_t, x_t^\star)$, the lower bound must be less than or equal to the sum of the period-t reward plus the upper bound that ensues from action $x_t^\star$. The result follows from the definition of the value functions and from the existence of lower and upper bounds. It implies that in state s_t, for a given action x_t in $X_t(s_t)$ and any outcome w_t in $W_t(s_t, x_t)$, if $\underline{V_t}(s_t) > r_t(s_t, x_t, w_t) + \overline{V}_{t+1}(s_{t+1})$, then

x_t does not belong to an optimal policy. Thus, any trajectory of states that results from taking action x_t in state s_t may be excluded from the decision tree.

Theorem 1 (Bounds and Optimal Actions)**.** *For all w_t in $W_t(s_t, x_t^\star)$, $\underline{V_t}(s_t) \leqslant r_t(s_t, x_t^\star, w_t) + \overline{V}_{t+1}(s_{t+1})$.*

Proof.

$$\underline{V_t}(s_t) \leqslant V_t(s_t) \tag{4}$$

$$= \max_{x_t \in X_t(s_t)} \left\{ \min_{w_t \in W_t(s_t, x_t)} \{r_t(s_t, x_t, w_t) + V_{t+1}(s_{t+1})\} \right\} \tag{5}$$

$$= \min_{w_t \in W_t(s_t, x_t^\star)} \{r_t(s_t, x_t^\star, w_t) + V_{t+1}(s_{t+1})\} \tag{6}$$

$$\leqslant r_t(s_t, x_t^\star, w_t) + V_{t+1}(s_{t+1}), w_t \in W_t(s_t, x_t^\star) \tag{7}$$

$$\leqslant r_t(s_t, x_t^\star, w_t) + \overline{V}_{t+1}(s_{t+1}), w_t \in W_t(s_t, x_t^\star). \tag{8}$$

Equation (4) holds by assumption of a lower bound, Equation (5) holds by definition of the value function, Equation (6) holds by the optimality of $x_t^\star$, Equation (7) holds by minimization, and Equation (8) holds by assumption of an upper bound. □

Corollary 1 shows how Theorem 1 simplifies the optimization required to identify optimal policies. Let $X_t^\star(s_t) = \{x_t \in X_t(s_t) : \underline{V_t}(s_t) \leqslant r_t(s_t, x_t, w_t) + \overline{V}_{t+1}(s_{t+1}) \ \forall \ w_t \in W_t(s_t, x_t)\}$ be the actions available in state s_t that satisfy the condition of Theorem 1. Require decision rules $\mu_t^\pi(s_t) : s_t \to X_t^\star(s_t)$ to map states to these actions in all periods $t = 1, \ldots, T$. Let $\Pi^\star \subseteq \Pi$ be the resulting subset of policies. Corollary 1 demonstrates that $J_t(s_t)$ may be obtained by maximizing over $\Pi^\star$ instead of Π and that $V_t(s_t)$ may be identified by maximizing over $X_t^\star(s_t)$ instead of $X_t(s_t)$.

Corollary 1 (Policy and Action Selection)**.** *The value of an optimal policy beginning in state s_t may be calculated as*

$$J_t(s_t) = \max_{\pi \in \Pi^\star} \{J_t^\pi(s_t)\} \tag{9}$$

and the associated value functions may be calculated as

$$V_t(s_t) = \max_{x_t \in X_t^\star(s_t)} \left\{ \min_{w_t \in W_t(s_t, x_t)} \{r_t(s_t, x_t, w_t) + V_{t+1}(s_{t+1})\} \right\} \tag{10}$$

for $t = 1, \ldots, T$, with $V_{T+1}(s_{T+1}) = r_{T+1}(s_{T+1})$.

Proof. Let $\Pi^{\text{opt}} = \arg\max_{\pi \in \Pi} J_t^\pi(s_t)$ be the set of optimal policies. By the construction of $\Pi^\star$, Theorem 1 guarantees that $\Pi^\star \cap \Pi^{\text{opt}}$ is nonempty. It follows that $\max_{\pi \in \Pi^\star}\{J_t^\pi(s_t)\} = \max_{\pi \in \Pi}\{J_t^\pi(s_t)\}$, which establishes Equation (9). Equation (10) may be derived as in Bertsekas (2017, ch. 1.6) by employing Theorem 1. □

Algorithm 1 is a procedure to construct a decision tree to facilitate calculation of $V_t(s_t)$. It leverages Equation (10) and checks bounds at each state. Let $S_{t'}$ be a set of states in period t'. Line 1 initializes S_t to the current state and the set of states at all subsequent periods to the empty set. Line 2 loops across periods, Line 3 loops through states, Line 4 checks lower and upper bounds on the reward-to-go from state $s_{t'}$, Line 5 loops over actions in $X_{t'}^\star(s_{t'})$, Line 6 loops across outcomes in $W_{t'}(s_{t'}, x_{t'})$, and Line 7 records states. The resulting decision tree may consist of both complete and partial trajectories. A partial trajectory terminates in a state $s_{t'}$ such that $\underline{V_{t'}}(s_{t'}) = \overline{V}_{t'}(s_{t'})$. Induction across these trajectories begins at state $s_{t'}$ with $V_{t'}(s_{t'})$ set by the bounds. This method of decision tree construction followed by backward solution of the value functions yields the value of an optimal policy from state s_t onward. If lower and upper bounds are strong, then Algorithm 1 can lead to considerable reductions in the size of the decision tree.

Algorithm 1 Decision tree from state s_t

1: $S_t \leftarrow \{s_t\}$, $S_{t'} \leftarrow \varnothing$ for $t' = t+1, \ldots, T+1$
2: **for** $t' = t$ **to** T **do**
3: **for** $s_{t'} \in S_{t'}$ **do**
4: **if** $\underline{V_{t'}}(s_{t'}) < \overline{V}_{t'}(s_{t'})$ **then**
5: **for** $x_{t'} \in X_{t'}^\star(s_{t'})$ **do**
6: **for** $w_{t'} \in W_{t'}(s_{t'}, x_{t'})$ **do**
7: $S_{t'+1} \leftarrow S_{t'+1} \cup \{S(s_{t'}, x_{t'}, w_{t'})\}$

If the value of a policy π is required, for example to calculate a lower bound, we can identify $J_t^\pi(s_t)$ via classical dynamic programming: states and transitions are the same as that of the max-min DP, actions select outcomes from uncertainty sets, and the objective is to minimize the total reward collected by the policy. The value functions for the policy evaluation DP are obtained from $V_t(s_t)$ by fixing actions to correspond to those chosen by a policy π:

$$V_t^\pi(s_t) = \min_{w_t \in W_t(s_t, \mu_t^\pi(s_t))} \left\{ r_t\left(s_t, \mu_t^\pi(s_t), w_t\right) + V_{t+1}^\pi(s_{t+1}) \right\}, \tag{11}$$

for $t = 1, \ldots, T$, with $V^{\pi}_{T+1} = r_{T+1}(s_{T+1})$. By Puterman (1994, ch. 4.3), $J^{\pi}_t(s_t) = V^{\pi}_t(s_t)$.

5 Dual Bounds

Upper bounds on the reward-to-go may be obtained by reducing the set of scenarios across which policies are evaluated. These dual bounds, combined with the lookahead policies in the next section, are one way to facilitate BBI. For clarity, in this section we augment our notation to indicate the set of scenarios employed in the optimization. Let $\boldsymbol{W}_t \subseteq \boldsymbol{W}_t(s_t)$ be a subset of possible scenarios in state s_t. Then, $J^{\pi}_t(s_t, \boldsymbol{W}_t)$ is the value of a policy π from state s_t when the scenario set is $\boldsymbol{W}_t$ and $J_t(s_t, \boldsymbol{W}_t)$ is the value of an optimal policy.

Theorem 2 asserts that for any subset $\boldsymbol{W}'_t$ of $\boldsymbol{W}_t$, the value $J^{\pi}_t(s_t, \boldsymbol{W}'_t)$ of a policy π across $\boldsymbol{W}'_t$ is an upper bound on the value $J^{\pi}_t(s_t, \boldsymbol{W}_t)$ of the same policy across $\boldsymbol{W}_t$. Further, the value $J_t(s_t, \boldsymbol{W}'_t)$ of an optimal policy across $\boldsymbol{W}'_t$ is an upper bound on the value $J_t(s_t, \boldsymbol{W}_t)$ of an optimal policy across $\boldsymbol{W}_t$. The result recognizes that a subset of scenarios shrinks the feasible region of the policy evaluation problem, thereby weakly increasing the objective value.

Theorem 2 (Dual Bounds)**.** *If $\boldsymbol{W}'_t \subseteq \boldsymbol{W}_t$, then $J^{\pi}_t(s_t, \boldsymbol{W}_t) \leqslant J^{\pi}_t(s_t, \boldsymbol{W}'_t)$ for any policy π and $J_t(s_t, \boldsymbol{W}_t) \leqslant J_t(s_t, \boldsymbol{W}'_t)$.*

Proof. If $\boldsymbol{W}'_t \subseteq \boldsymbol{W}_t$, then because the feasible region is smaller, $J^{\pi}_t(s_t, \boldsymbol{W}_t) \leqslant J^{\pi}_t(s_t, \boldsymbol{W}'_t)$ for any policy π. Let $\pi^{\star} = \arg\max_{\pi \in \Pi} J_t(s_t, \boldsymbol{W}_t)$. Then,

$$
\begin{aligned}
J_t\left(s_t, \boldsymbol{W}_t\right) &= J^{\pi^{\star}}_t\left(s_t, \boldsymbol{W}_t\right) && (12)\\
&\leqslant J^{\pi^{\star}}_t\left(s_t, \boldsymbol{W}'_t\right) && (13)\\
&\leqslant J_t\left(s_t, \boldsymbol{W}'_t\right). && (14)
\end{aligned}
$$

Equation (12) follows from the optimality of $\pi^{\star}$. Equation (13) follows from $\boldsymbol{W}'_t \subseteq \boldsymbol{W}_t$. Equation (14) holds by maximization across policies. □

The first part of Theorem 2 can reduce the computation necessary to calculate the value of a policy π via dynamic programming. Consider the well-known reaching algorithm (Denardo, 2003). Moving forward through the state-space graph from current state s_t toward all terminal states s_{T+1}, the reaching algorithm calculates the reward-so-far for each state. At stage $t < t' < T + 1$ of the

graph, suppose states $s_{t'}$ and $\tilde{s}_{t'}$ are identical and the associated scenario sets satisfy $\tilde{\boldsymbol{W}}_{t'}(\tilde{s}_{t'}) \subseteq \boldsymbol{W}_{t'}(s_{t'})$. By Theorem 2, $J^{\pi}_{t'}(s_{t'}, \boldsymbol{W}_{t'}(s_{t'})) \leqslant J^{\pi}_{t'}(\tilde{s}_{t'}, \tilde{\boldsymbol{W}}_{t'}(\tilde{s}_{t'}))$. Denote the rewards-so-far by $\gamma(s_{t'})$ and $\gamma(\tilde{s}_{t'})$. If $\gamma(s_{t'}) \leqslant \gamma(\tilde{s}_{t'})$, then $\gamma(s_{t'}) + J^{\pi}_{t'}(s_{t'}, \boldsymbol{W}_{t'}(s_{t'})) \leqslant \gamma(\tilde{s}_{t'}) + J^{\pi}_{t'}(\tilde{s}_{t'}, \tilde{\boldsymbol{W}}_{t'}(\tilde{s}_{t'}))$. Thus, the reward collected by the trajectory through state $s_{t'}$ is no larger than the reward collected by the trajectory through state $\tilde{s}_{t'}$. Consequently, any trajectories through state $\tilde{s}_{t'}$ may be ignored.

The second part of Theorem 2 can reduce the computation required to calculate the value of an optimal policy. There are many ways to construct scenario subsets. Ideally, $\boldsymbol{W}'_t$ is chosen to yield a dual bound that is as tight as possible. Because $\boldsymbol{W}_t$ is a subset of itself, strong duality exists in principle, though this amounts to solving the original problem and is unhelpful in practice. Below, we propose a family of dual bounds that navigates the trade-off between computational effort and quality of the bound.

Let $\boldsymbol{W}^n_t = \boldsymbol{W}_t \times \cdots \times \boldsymbol{W}_t$ be the Cartesian product of $\boldsymbol{W}_t$ with itself n times. Each element of $\boldsymbol{W}^n_t$ is a tuple of n scenarios belonging to $\boldsymbol{W}_t$. Let

$$J^n_t(s_t, \boldsymbol{W}_t) = \min_{\boldsymbol{W}'_t \in \boldsymbol{W}^n_t} \{J_t(s_t, \boldsymbol{W}'_t)\} \tag{15}$$

be the smallest dual bound across all n-tuples that compose $\boldsymbol{W}^n_t$. We call $J^n_t(s_t, \boldsymbol{W}_t)$ the *n-tuple dual bound*. As n increases, the complexity of the optimization required to obtain the bound also increases. When n is 1, the 1-tuple dual bound is equivalent to the *perfect information dual bound* where a policy is chosen in response to each scenario:

$$\begin{aligned} J^1_t(s_t, \boldsymbol{W}_t) &= \min_{\boldsymbol{w} \in \boldsymbol{W}_t} \Big\{ J_t\big(s_t, \{\boldsymbol{w}\}\big) \Big\} \\ &= \min_{\boldsymbol{w} \in \boldsymbol{W}_t} \left\{ \max_{\pi \in \Pi} \left\{ r_{T+1}(s_{T+1}) + \sum_{t'=t}^{T} r_{t'}\left(s_{t'}, \mu^{\pi}_{t'}(s_{t'}), w_t\right) \right\} \right\}. \end{aligned} \tag{16}$$

At the other extreme, when n is the cardinality of the underlying set of trajectories, $\boldsymbol{W}^{|\boldsymbol{W}_t|}_t = \{\boldsymbol{W}_t\}$ and $J^{|\boldsymbol{W}_t|}_t(s_t, \boldsymbol{W}_t) = J_t(s_t, \boldsymbol{W}_t)$ reduces to the original problem. Theorem 3 asserts that the quality of the bound increases weakly with n. The result follows from the structure of the n-ary Cartesian power. Given any $n' < n$, for each element of $\boldsymbol{W}^{n'}_t$, there exists at least one element of $\boldsymbol{W}^n_t$ that contains the same scenarios as a subset. Consequently, increasing n cannot degrade the dual bound.

Theorem 3 (n-Tuple Dual Bounds)**.** *$J^n_t(s_t, \boldsymbol{W}_t) \leqslant J^{n'}_t(s_t, \boldsymbol{W}_t)$ for $n' < n$.*

Proof. Let $\boldsymbol{W}_t^\dagger = \arg\min_{\boldsymbol{W}'_t \in \boldsymbol{W}_t^{n'}} \{J_t(s_t, \boldsymbol{W}'_t)\}$. By construction, there exists a $\boldsymbol{W}''_t$ in $\boldsymbol{W}_t^n$ such that $\boldsymbol{W}_t^\dagger \subseteq \boldsymbol{W}''_t$. Then,

$$J_t^{n'}(s_t, \boldsymbol{W}_t) = J_t(s_t, \boldsymbol{W}_t^\dagger) \tag{17}$$

$$\geqslant J_t(s_t, \boldsymbol{W}''_t) \tag{18}$$

$$\geqslant J_t^n(s_t, \boldsymbol{W}_t). \tag{19}$$

Equation (17) follows from the optimality of $\boldsymbol{W}_t^\dagger$. Equation (18) holds by Theorem 2. Equation (19) follows from minimization across $\boldsymbol{W}_t^n$. □

6 Optimistic Lookahead Policies

In addition to serving as upper bounds, dual bounds yield *optimistic lookahead policies* with performance guarantees. A one-step optimistic lookahead policy selects actions via an overestimate $\overline{V}_{t+1}(s_{t+1})$ of the optimal policy value from state s_{t+1}. The decision rule in period t is

$$\mu_t^{\overline{\pi}}(s_t) = \arg\max_{x_t \in X_t(s_t)} \left\{ \min_{w_t \in W_t(s_t, x_t)} \left\{ r_t(s_t, x_t, w_t) + \overline{V}_{t+1}(s_{t+1}) \right\} \right\}, \tag{20}$$

and the one-step lookahead policy $\overline{\pi}$ is the sequence of decision rules $(\mu_1^{\overline{\pi}}, \ldots, \mu_T^{\overline{\pi}})$. The value $J_t^{\overline{\pi}}(s_t)$ of the policy is a lower bound on the reward-to-go from state s_t. Thus, dual bounds can facilitate both the upper and lower bounds required for BBI.

When rewards are non-negative and $J_t(s_t)$ is strictly positive, the lookahead policy enjoys two performance guarantees. A straightforward guarantee is $J_t^{\overline{\pi}}(s_t)/J_t(s_t) \geqslant J_t^{\overline{\pi}}(s_t)/\overline{V}_t(s_t)$. From state s_t, the ratio of the lookahead policy value to the optimal policy value is at least as large as the ratio of the lookahead policy value to the upper bound. A second guarantee connects lower and upper bounds across stages t and $t+1$. Let $S_{t+1}(s_t, \mu_t^{\overline{\pi}}(s_t)) = \{s_{t+1} = S(s_t, \mu_t^{\overline{\pi}}(s_t), w_t) : w_t \in W_t(s_t, \mu_t^{\overline{\pi}}(s_t))\}$ be the set of states at epoch $t+1$ that results from taking lookahead action $\mu_t^{\overline{\pi}}(s_t)$ in state s_t. Theorem 4 asserts that if the ratio of the lookahead policy value $J_{t+1}^{\overline{\pi}}(s_{t+1})$ to the upper bound $\overline{V}_{t+1}(s_{t+1})$ is greater than or equal to some $\epsilon \geqslant 0$ for all states s_{t+1} in $S_{t+1}(s_t, \mu_t^{\overline{\pi}}(s_t))$, then the ratio of the lookahead policy value $J_t^{\overline{\pi}}(s_t)$ to the optimal policy value $J_t(s_t)$ is also greater than or equal to ϵ. The result follows from overestimation of the reward-to-go in the lookahead decision rule. BBI facilitates the calculation of both performance guarantees. When $J_t(s_t) = 0$, all policies have value zero.

Theorem 4 (Performance Guarantee)**.** *When rewards are non-negative and* $J_t(s_t) > 0$*, if* $J^{\overline{\pi}}_{t+1}(s_{t+1})/\overline{V}_{t+1}(s_{t+1}) \geqslant \epsilon$ *for all states* s_{t+1} *in* $S_{t+1}(s_t, \mu^{\overline{\pi}}_t(s_t))$*, then* $J^{\overline{\pi}}_t(s_t)/J_t(s_t) \geqslant \epsilon$*.*

Proof. For clarity, we denote states $s_{t+1} = S(\cdot)$ in period $t+1$ via the transition function. Let $x^\star_t$ be an action in $X_t(s_t)$ that achieves $V_t(s_t)$ and let $\overline{x}_t = \mu^{\overline{\pi}}_t(s_t)$. Then,

$$\frac{J^{\overline{\pi}}_t(s_t)}{J_t(s_t)} = \frac{V^{\overline{\pi}}_t(s_t)}{V_t(s_t)} \tag{21}$$

$$= \frac{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + V^{\overline{\pi}}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}}{\min_{w_t \in W_t(s_t, x^\star_t)} \left\{ r_t\left(s_t, x^\star_t, w_t\right) + V_{t+1}\left(S\left(s_t, x^\star_t, w_t\right)\right)\right\}} \tag{22}$$

$$\geqslant \frac{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + V^{\overline{\pi}}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}}{\min_{w_t \in W_t(s_t, x^\star_t)} \left\{ r_t\left(s_t, x^\star_t, w_t\right) + \overline{V}_{t+1}\left(S\left(s_t, x^\star_t, w_t\right)\right)\right\}} \tag{23}$$

$$\geqslant \frac{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + V^{\overline{\pi}}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}}{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + \overline{V}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}} \tag{24}$$

$$\geqslant \frac{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + \epsilon\overline{V}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}}{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + \overline{V}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}} \tag{25}$$

$$\geqslant \epsilon \frac{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + \overline{V}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}}{\min_{w_t \in W_t(s_t, \overline{x}_t)} \left\{ r_t\left(s_t, \overline{x}_t, w_t\right) + \overline{V}_{t+1}\left(S\left(s_t, \overline{x}_t, w_t\right)\right)\right\}} \tag{26}$$

$$= \epsilon. \tag{27}$$

Equation (21) follows from Puterman (1994, ch. 4.3) and Bertsekas (2017, ch. 1.6). Equation (22) holds by definition. Equation (23) holds because $V_{t+1}(s_{t+1}) \leqslant \overline{V}_{t+1}(s_{t+1})$ for any state s_{t+1}. Equation (24) follows from decision rule (20), which associates a value with action $\overline{x}_t$ at least as large as the value tied to action $x^\star_t$. Equation (25) follows from Puterman (1994, ch. 4.3) and from the assumption that $J^{\overline{\pi}}_{t+1}(s_{t+1})/\overline{V}_{t+1}(s_{t+1}) \geqslant \epsilon$ for all s_{t+1} in $S_{t+1}(s_t, \mu^{\overline{\pi}}_t(s_t))$. Equation (26) holds because ϵ is constant and less than or equal to one because $J^{\overline{\pi}}_{t+1}(s_{t+1}) \leqslant J_{t+1}(s_{t+1}) \leqslant \overline{V}_{t+1}(s_{t+1})$ for any s_{t+1}. Equation (27) cancels terms. □

Theorem 4 underscores the importance of good dual bounds. Smaller upper bounds at stage $t+1$ ensure a better performance guarantee from state s_t. This becomes evident when we take ϵ to be the minimum ratio $J^{\overline{\pi}}_{t+1}(s_{t+1})/\overline{V}_{t+1}(s_{t+1})$ across all states s_{t+1} in $S_{t+1}(s_t, \mu^{\overline{\pi}}_t(s_t))$. Equivalently, Theorem 4 guarantees the optimality gap $J_t(s_t) - J^{\overline{\pi}}_t(s_t)$ is at most $(1/\epsilon - 1)J^{\overline{\pi}}_t(s_t)$. As dual bounds and lookahead policy values approach the reward-to-go from above and below, ϵ approaches one and the gap vanishes.

7 Budget Scenario Sets

Budget scenario sets lead to refined methods for policy evaluation and dual bound generation, and thus to a more tractable BBI. Budget scenario sets draw outcomes from a base set $\overline{W}$ subject to a constraint on total uncertainty. Given a non-negative integer $\overline{w}$, we consider base sets of the form $\{-\overline{w}, \ldots, \overline{w}\}$ and $\{0, \ldots, \overline{w}\}$. The first form is used in the context of robust optimization to represent variation across a range (Betsimas and Thiele, 2006, ch. 3.2.3). If uncertainty is modeled to lie in $[\underline{y}, \overline{y}]$ with midpoint $\tilde{y}$, then the range may be discretized into $2\overline{w} + 1$ equidistant points as $\{\tilde{y} + w_t\hat{y} : w_t \in \{-\overline{w}, \ldots, \overline{w}\}\}$, where $\hat{y} = (\overline{y} - \underline{y})/(2\overline{w})$ is the distance between consecutive points. The second form is useful to represent uncertainty that unfolds in one direction. For example, if $\overline{w} = 1$, then $\{0, 1\}$ may be used to model binary uncertainty that either does or does not occur. In both cases, larger outcome magnitudes represent higher levels of uncertainty.

Scenarios are constructed with respect to budget of uncertainty Γ chosen from the non-negative integers. Each scenario in scenario set $\boldsymbol{W}(\Gamma) = \{(w_1, \ldots, w_T) \in \overline{W}^T : \sum_{t=1}^{T} |w_t| \leqslant \Gamma\}$ is a trajectory of outcomes such that each element belongs to base set $\overline{W}$ and the sum of outcome magnitudes across all periods does not exceed Γ. Thus, larger values of Γ increase the uncertainty in the model up to $\overline{w}T$, at which point Γ becomes redundant. At each period, the magnitude of the observed outcome depletes the budget resulting in a remaining budget of uncertainty $\Gamma_{t+1} = \Gamma_t - |w_t|$, where $\Gamma_1 = \Gamma$. Consequently, by maintaining Γ_t in the state variable, the state-s_t scenario set $\boldsymbol{W}_t(\Gamma_t) = \{\boldsymbol{w} \in \boldsymbol{W}(\Gamma) : \sum_{t'=t}^{T} |w_{t'}| \leqslant \Gamma_t\}$ consists of all scenarios in $\boldsymbol{W}(\Gamma)$ such that the sum of outcome magnitudes from period t forward does not exceed Γ_t. The state-s_t uncertainty set $W_t(\Gamma_t) = \{w_t \in \overline{W} : |w_t| \leqslant \Gamma_t\}$ contains all period-t outcomes in $\overline{W}$ with magnitude at most Γ_t.

Under the condition of non-increasing rewards, we can simplify the optimization required to evaluate policies and to identify the value of an optimal policy. For $t = 1, \ldots, T-1$ we require $r_t(s_t, x_t, w_t)$ to be non-increasing in the magnitude $|w_t|$ of outcomes. We also require $r_T(s_T, x_T, w_T) + r_{T+1}(s_{T+1})$ to be non-increasing in $|w_T|$. This models a circumstance that is natural in many contexts, that worst-case outcomes do not lead to larger rewards. Theorem 5 partially characterizes scenarios that solve the policy evaluation problem. It asserts that there exists a scenario $\boldsymbol{w}^\star$ in $\boldsymbol{W}_t(\Gamma_t)$ achieving $V_t^\pi(s_t)$ such that the sum $\sum_{t'=t}^{T} |w_{t'}^\star|$ of outcome magnitudes from period t forward is $\omega(\Gamma_t) = \min\{\Gamma_t, \overline{w}(T - t + 1)\}$. The quantity $\omega(\Gamma_t)$ is the largest magnitude

of total uncertainty that may be realized from period t forward, either the remaining budget Γ_t or $\overline{w}$ multiplied by the number of remaining periods $T - t + 1$, whichever is smaller.

Theorem 5 (Optimal Scenarios)**.** *There exists a scenario $\boldsymbol{w}^\star$ in $\boldsymbol{W}_t(\Gamma_t)$ achieving $V_t^\pi(s_t)$ such that $\sum_{t'=t}^{T} |w_{t'}^\star| = \omega(\Gamma_t)$.*

Proof. Consider the case that $\overline{W} = \{-\overline{w}, \ldots, \overline{w}\}$. The proof is by induction. In period T, because $r_T(s_T, \mu_T^\pi(s_T), w_T) + r_{T+1}(s_{T+1})$ is non-increasing in $|w_T|$, $V_T^\pi(s_T)$ is achieved by setting $w_T^\star$ to Γ_T or $-\Gamma_T$ if $\Gamma_T \leqslant \overline{w}$, and to $\overline{w}$ or $-\overline{w}$ if $\Gamma_T > \overline{w}$. Thus, $|w_T^\star| = \min\{\Gamma_T, \overline{w}\}$. Assume the result holds in periods $t+1, \ldots, T-1$. In period t, $|w_t| + \sum_{t'=t+1}^{T} |w_{t'}^\star| = |w_t| + \min\{\Gamma_t - |w_t|, \overline{w}(T-t)\}$. When $\Gamma_t - |w_t| < \overline{w}(T-t)$, because $|w_t| \leqslant \overline{w}$, it follows that $|w_t| + \min\{\Gamma_t - |w_t|, \overline{w}(T-t)\} = \min\{\Gamma_t, \overline{w}(T-t) + |w_t|\} = \min\{\Gamma_t, \overline{w}(T-t+1)\}$. When $\Gamma_t - |w_t| \geqslant \overline{w}(T-t)$, because $|w_t| \geqslant 0$ and $T - t \geqslant 1$, we have $\Gamma_t \geqslant \Gamma_t - |w_t| \geqslant \overline{w}(T-t) \geqslant \overline{w}$. Thus, $\Gamma_t \geqslant \overline{w} \geqslant |w_t|$. Because the reward in each period is non-increasing in $|w_t|$, it follows that setting $w_{t'}^\star$ to $\overline{w}$ or $-\overline{w}$ for $t' = t, \ldots, T$ achieves $V_t^\pi(s_t)$. Then, $|w_t^\star| + \min\{\Gamma_t - |w_t^\star|, \overline{w}(T-t)\} = \overline{w} + \min\{\Gamma_t - \overline{w}, \overline{w}(T-t)\} = \min\{\Gamma_t, \overline{w}(T-t+1)\}$. When $\overline{W} = \{0, \ldots, \overline{w}\}$, the proof is modified as follows: In period T, $w_T^\star = \min\{\Gamma_T, \overline{w}\}$. In period t, when $\Gamma_t - |w_t| \geqslant \overline{w}(T-t)$, $w_t^\star = \cdots = w_T^\star = \overline{w}$. □

Theorem 5 simplifies the optimization required to evaluate policies. From state s_t, instead of optimizing over scenarios whose magnitudes sum to Γ_t or less, we may restrict attention to scenarios whose magnitudes sum to exactly $\omega(\Gamma_t)$. Denote this subset of scenarios by $\boldsymbol{W}_t^\star(\Gamma_t) = \{\boldsymbol{w} \in \boldsymbol{W}(\Gamma) : \sum_{t'=t}^{T} |w_{t'}| = \omega(\Gamma_t)\}$, with $\boldsymbol{W}^\star(\Gamma) = \boldsymbol{W}_1^\star(\Gamma_1)$ representing the subset of scenarios in the initial state. The corresponding uncertainty set is $W_t^\star(\Gamma_t) = \{w_t \in \overline{W} : \omega(\Gamma_t) - \overline{w}(T-t) \leqslant |w_t| \leqslant \Gamma_t\}$. As in $W_t(\Gamma_t)$, outcome magnitudes cannot exceed Γ_t. The lower threshold activates when $\omega(\Gamma_t)$ exceeds $\overline{w}(T-t)$. It ensures that $|w_t|$ is large enough to yield a sum of outcome magnitudes equal to $\omega(\Gamma_t)$. Corollary 2 confirms that $J_t^\pi(s_t)$ may be obtained by minimizing over $\boldsymbol{W}_t^\star(\Gamma_t)$ instead of $\boldsymbol{W}_t(\Gamma_t)$ and that $V_t^\pi(s_t)$ may be achieved by minimizing over $W_t^\star(\Gamma_t)$ instead of $W_t(\Gamma_t)$. Further, Corollaries 1 and 2 may be used in tandem. It is straightforward to show that in Equation (9), $J_t^\pi(s_t)$ may be calculated as in Equation (28), and that in Equation (10), the inner minimization may be conducted across $W_t^\star(\Gamma_t)$. Consequently, Line 6 of Algorithm 1 may operate on $W_{t'}^\star(s_{t'}, x_{t'})$.

Corollary 2 (Policy Evaluation)**.** *The value of a policy π from state s_t may be calculated as*

$$J_t^\pi(s_t) = \min_{\boldsymbol{w} \in \boldsymbol{W}_t^\star(\Gamma_t)} \left\{ r_{T+1}(s_{T+1}) + \sum_{t'=t}^{T} r\left(s_{t'}, \mu_{t'}^\pi(s_{t'}), w_{t'}\right) \right\} \tag{28}$$

and the associated value functions may be calculated as

$$V_t^\pi(s_t) = \min_{w_t \in W_t^\star(\Gamma_t)} \left\{ r_t(s_t, \mu_t^\pi(s_t), w_t) + V_{t+1}^\pi(s_{t+1}) \right\}, \tag{29}$$

for $t = 1, \ldots, T$, with $V_{T+1}^\pi(s_{T+1}) = r_{T+1}(s_{T+1})$.

Proof. We first prove Equation (29). By construction, if $w_{t'}$ belongs to $W_{t'}^\star(\Gamma_{t'})$ for $t' = t, \ldots, T$, then $\sum_{t'=t}^{T} |w_{t'}| = \omega(\Gamma_t)$. By optimizing over all such trajectories via the recursion of Equation (29), then by Theorem 5, the resulting trajectory has value $V_t^\pi(s_t)$. Using this result, we prove Equation (28) by induction. In period T,

$$\begin{aligned} J_T^\pi(s_T) &= V_T^\pi(s_T) && (30)\\ &= \min_{w_T \in W_T^\star(\Gamma_T)} \left\{ r_T\left(s_T, \mu_T^\pi(s_T), w_T\right) + r_{T+1}(s_{T+1}) \right\} && (31)\\ &= \min_{\boldsymbol{w} \in \boldsymbol{W}_T^\star(\Gamma_T)} \left\{ r_T\left(s_T, \mu_T^\pi(s_T), w_T\right) + r_{T+1}(s_{T+1}) \right\}. && (32) \end{aligned}$$

Equation (30) follows from Puterman (1994, ch. 4.3). Equation (31) holds by Equation (29). By construction, if $\boldsymbol{w}$ belongs to $\boldsymbol{W}_T^\star(\Gamma_T)$, then w_T belongs to $W_T^\star(\Gamma_T)$. This establishes Equation (32). Assume the result holds in periods $t+1, \ldots, T-1$. In period t,

$$\begin{aligned} J_t^\pi(s_t) &= V_t^\pi(s_t) && (33)\\ &= \min_{w_t \in W_t^\star(\Gamma_t)} \left\{ r_t\left(s_t, \mu_t^\pi(s_t), w_t\right) + J_{t+1}^\pi(s_{t+1}) \right\} && (34)\\ &= \min_{w_t \in W_t^\star(\Gamma_t)} \left\{ r_t\left(s_t, \mu_t^\pi(s_t), w_t\right) \right. \\ &\qquad \left. + \min_{\boldsymbol{w} \in \boldsymbol{W}_{t+1}^\star(\Gamma_{t+1})} \left\{ r_{T+1}(s_{T+1}) + \sum_{t'=t+1}^{T} r_{t'}\left(s_{t'}, \mu_{t'}^\pi\left(s_{t'}\right), w_{t'}\right) \right\} \right\} && (35)\\ &= \min_{\boldsymbol{w} \in \boldsymbol{W}_t^\star(\Gamma_t)} \left\{ r_{T+1}(s_{T+1}) + \sum_{t'=t+1}^{T} r_{t'}\left(s_{t'}, \mu_{t'}^\pi\left(s_{t'}\right), w_{t'}\right) \right\}. && (36) \end{aligned}$$

Equations (33) and (34) follow from Puterman (1994, ch. 4.3) and from Equation (29). Equation (35) holds by the induction hypothesis. By construction, if $\boldsymbol{w}$ belongs to $\boldsymbol{W}_{t+1}^\star(\Gamma_{t+1} =$

$\Gamma_t - |w_t|)$, then $\boldsymbol{w}$ also belongs to $\boldsymbol{W}_t^\star(\Gamma_t)$, and thus w_t belongs to $W_t^\star(\Gamma_t)$. This establishes Equation (36). □

Budget scenario sets also simplify the task of identifying scenario subsets for dual bound generation. Theorem 6 asserts that for any scenario subset of $\boldsymbol{W}_t(\Gamma_t)$, there exists a scenario subset of $\boldsymbol{W}_t^\star(\Gamma_t)$ with equal cardinality resulting in a dual bound that is no larger. The result follows from the requirement that rewards be non-increasing in the magnitude of outcomes. Thus, rather than construct scenario subsets from $\boldsymbol{W}_t(\Gamma_t)$, we may limit attention to subsets of $\boldsymbol{W}_t^\star(\Gamma_t)$.

Theorem 6 (Better Scenario Subsets)**.** *For any $\boldsymbol{W}_t \subseteq \boldsymbol{W}_t(\Gamma_t)$, there exists a $\boldsymbol{W}_t^\star \subseteq \boldsymbol{W}_t^\star(\Gamma_t)$ such that $|\boldsymbol{W}_t^\star| = |\boldsymbol{W}_t|$ and $J_t(s_t, \boldsymbol{W}_t^\star) \leqslant J_t(s_t, \boldsymbol{W}_t)$.*

Proof. Let $\boldsymbol{w}$ be a scenario in $\boldsymbol{W}_t(\Gamma_t)$. Denote by $\boldsymbol{W}_t^\star(\Gamma_t, \boldsymbol{w}) = \{\boldsymbol{w}^\star \in \boldsymbol{W}_t^\star(\Gamma_t) : |w_{t'}^\star| \geqslant |w_{t'}|, t' = t, \ldots, T\}$ the set of scenarios in $\boldsymbol{W}_t^\star(\Gamma_t)$ whose outcome magnitudes are no smaller than those of $\boldsymbol{w}$ from period t forward. By construction, $\boldsymbol{W}_t^\star(\Gamma_t, \boldsymbol{w})$ is nonempty. For each $\boldsymbol{w}$ in $\boldsymbol{W}_t$, choose any $\boldsymbol{w}^\star$ in $\boldsymbol{W}_t^\star(\Gamma_t, \boldsymbol{w})$. Call this collection of scenarios $\boldsymbol{W}_t^\star$. If the collection consists of unique scenarios, then $\boldsymbol{W}_t^\star$ is a set. Otherwise, $\boldsymbol{W}_t^\star$ is a multiset. By construction, $|\boldsymbol{W}_t^\star| = |\boldsymbol{W}_t|$. Select any policy π. For each $\boldsymbol{w}$ and the corresponding $\boldsymbol{w}^\star$, because rewards are non-increasing in outcomes, and because $|w_{t'}| \leqslant |w_{t'}^\star|$ for $t' = t, \ldots, T$, it follows that $r_{T+1}(s_{T+1}) + \sum_{t'=t}^{T} r_{t'}(s_{t'}, \mu_{t'}^\pi(s_{t'}), w_{t'}^\star) \leqslant r_{T+1}(s_{T+1}) + \sum_{t'=t}^{T} r_{t'}(s_{t'}, \mu_{t'}^\pi(s_{t'}), w_{t'})$. Because this relationship holds for each $\boldsymbol{w}$ and the corresponding $\boldsymbol{w}^\star$, it follows that $J_t^\pi(s_t, \boldsymbol{W}_t^\star) \leqslant J_t^\pi(s_t, \boldsymbol{W}_t)$. Because this is true for all policies, it must be that $J_t(s_t, \boldsymbol{W}_t^\star) \leqslant J_t(s_t, \boldsymbol{W}_t)$. □

Corollary 3 demonstrates the utility of Theorem 6. It shows that the n-tuple dual bound built on $\boldsymbol{W}_t(\Gamma_t)$ is equivalent to the n-tuple dual bound built on $\boldsymbol{W}_t^\star(\Gamma_t)$. Because the size of $\boldsymbol{W}_t^\star(\Gamma_t)$ is potentially much smaller than the size of $\boldsymbol{W}_t(\Gamma_t)$, the number of scenario subsets that can be constructed from $\boldsymbol{W}_t^\star(\Gamma_t)$ may be significantly less than the number that can be composed from $\boldsymbol{W}_t(\Gamma_t)$. Consequently, the effort required to calculate the n-tuple dual bound is reduced.

Corollary 3 (Simplified n-Tuple Dual Bounds)**.** $J_t^n(s_t, \boldsymbol{W}_t(\Gamma_t)) = J_t^n(s_t, \boldsymbol{W}_t^\star(\Gamma_t))$.

Proof.

$$J_t^n(s_t, \boldsymbol{W}_t(\Gamma_t)) = \min_{\boldsymbol{W}_t \in [\boldsymbol{W}_t(\Gamma_t)]^n} \{J_t(s_t, \boldsymbol{W}_t)\} \tag{37}$$

$$= \min_{\boldsymbol{W}_t \in [\boldsymbol{W}_t^\star(\Gamma_t)]^n \cup ([\boldsymbol{W}_t(\Gamma_t)]^n \setminus [\boldsymbol{W}_t^\star(\Gamma_t)]^n)} \{J_t(s_t, \boldsymbol{W}_t)\} \tag{38}$$

$$= \min_{\boldsymbol{W}_t \in [\boldsymbol{W}_t^\star(\Gamma_t)]^n} \{J_t(s_t, \boldsymbol{W}_t)\} \tag{39}$$

$$= J_t^n(s_t, \boldsymbol{W}_t^\star(\Gamma_t)). \tag{40}$$

Equations (37) and (40) hold by definition. Equation (38) follows from $\boldsymbol{W}_t(\Gamma_t) = \boldsymbol{W}_t^\star(\Gamma_t) \cup (\boldsymbol{W}_t(\Gamma_t) \setminus \boldsymbol{W}_t^\star(\Gamma_t))$ and the distributive property of the Cartesian product. By Theorem 6, for each scenario subset in $[\boldsymbol{W}_t(\Gamma_t)]^n \setminus [\boldsymbol{W}_t^\star(\Gamma_t)]^n$, there exists a scenario subset in $[\boldsymbol{W}_t^\star(\Gamma_t)]^n$ with an objective value that is no larger. This establishes Equation (39). □

8 Application to Media Selection

We illustrate BBI via an application to media selection. The problem is the classical task of allocating limited budget funds to purchase media spots with the aim of maximizing audience exposure, but with an explicit focus on yield uncertainty. At each epoch, the decision maker purchases spots for advertisement during the current period and across future periods. Uncertainty, in the form of bumped media spots, erases any exposure that would have been captured via spots purchased for a given period. The objective is to identify a policy that maximizes worst-case exposure subject to spot limits and a budget constraint. This criterion is especially suitable for infrequent or limited-time media campaigns, where protection against worst-case bump outcomes is advantageous. The problem is of significant practical interest, but can be difficult to solve with conventional backward induction.

The linear program model of media selection that appears in many introductory management science textbooks stems from Wilson (1962). Little and Lodish (1969), Zufryden (1975), and Srinivasan (1976) bring more realism to the domain by incorporating diminishing returns into the exposure metric, by choosing cost functions that allow for quantity discounts, and by considering how the timing of ads might impact exposure. These nonlinear models were intractable at the time, however, and the authors resort to simple heuristic techniques to identify feasible policies. The ensuing literature explores various approaches to the problem of media selection including portfolio theory (De Kluyver, 1980), analytical hierarchy (Kwak et al., 2005), and data envelopment analysis (Saen, 2011). Yet, the extant literature does not consider the issue of sequential

decision making as a means of responding to uncertainty. The example in this section fills this gap. Our dynamic model and solution via BBI improve on the static models and heuristic solution methods that dominate the literature.

Our formulation employs budget scenarios to model the possibility of bumped media spots. The base set is $\overline{W} = \{0, 1\}$, with $\overline{w} = 1$. Outcome $w_t = 1$ represents the event that spots purchased for advertisement in period t are bumped. Outcome $w_t = 0$ represents the event that spots are not bumped. The state at epoch t is $s_t = (b_t, p, \Gamma_t)$, where b_t is the remaining budget, $p = ((p_{t't''})_{t'=1}^{T})_{t''=t'}^{T}$ tracks the number of spots purchased in each period $t' = 1, \ldots, T$ for advertisement in period $t'' = t', \ldots, T$, and Γ_t is the remaining budget of uncertainty. In initial state s_1, the budget is b_1, all elements of p are zero, and $\Gamma_1 = \Gamma$. During period t, the decision maker may purchase spots for the current period and for any future period from a given media outlet. The cost of purchasing spots during period t for advertisement during period $t' \geqslant t$ is $c_{tt'}(\cdot)$. The media outlet places a limit l_t on the total number of spots that may be purchased for advertisement during period t. An action $x_t = (x_{tt'})_{t'=1}^{T}$ is the number of spots purchased in period t for advertisement during the current period and all remaining periods. The set of actions available in state s_t is

$$X_t(s_t) = \Bigg\{ x_t \in \mathbb{Z}_{\geqslant 0}^{T} : \tag{41}$$

$$\sum_{t'=t}^{T} c_{tt'}(x_{tt'}) \leqslant b_t, \tag{42}$$

$$x_{tt'} + \sum_{t''=1}^{t-1} p_{t''t'} \leqslant l_{t'} \text{ for } t' = t, \ldots, T, \tag{43}$$

$$x_{tt'} = 0 \text{ for } t' = 1, \ldots, t-1 \Bigg\}, \tag{44}$$

where Equation (41) restricts actions to the set of non-negative integer vectors with T elements, Equation (42) requires total cost to be no larger than the remaining budget, Equation (43) implements the spot limit, and Equation (44) disallows the purchase of spots for advertisement in the past. Audience exposure for spots that run during period t is $e_t(\cdot)$. The reward in period t is $r_t(s_t, x_t, w_t) = (1 - w_t)e_t(x_{tt} + \sum_{t'=1}^{t-1} p_{t't})$. If $w_t = 0$, then spots air and audience exposure is determined by the exposure function and the number of spots slotted for advertisement in the current period. If $w_t = 1$, then spots are bumped and exposure is zero. The transition to state s_{t+1} adjusts p to reflect purchases made by the selected action: $p_{tt'} = p_{tt'} + x_{tt'}$ for $t' = t, \ldots, T$. Then, the

remaining budget is updated by subtracting the cost of purchases and adding any refund due to a bump: $b_{t+1} = b_t - \sum_{t'=t}^{T} c_{tt'}(x_{tt'}) + w_t \sum_{t'=1}^{t} c_{t't}(p_{t't})$. Finally, the remaining budget of uncertainty is updated as $\Gamma_{t+1} = \Gamma_t - w_t$. A terminal state s_{T+1} is any state that follows action selection and outcome observation in period T. For all terminal states, $r_{T+1}(s_{T+1}) = 0$.

In our numerical experiments, we consider values for horizon T equal to 5, 6, 7, and 8 periods. We examine values for budget of uncertainty Γ equal to 1, 2, and 3. The initial budget is $b_1 =$ \$50,000. Given these parameters, problem instances are generated by randomly setting costs, exposure values, and spot limits. The cost $c_{tt'}(\cdot)$ of purchasing spots is the sum of a base price for each spot adjusted by a quantity discount. The base price varies linearly between minimum and maximum price points as a decreasing function of the number of periods between purchase and advertisement. The minimum and maximum price points are selected randomly as dollar amounts between \$3,000 and \$7,000 and between \$7,001 and \$12,000, respectively. Randomly selected quantity discounts may decrease the price of each additional spot by up to 10 percent. The exposure function $e_t(\cdot)$ diminishes returns via a geometric sequence. The scale factor is 1 and the common ratio is a randomly selected number between 0.95 and 1. The spot limit l_t for each period is a randomly selected integer between 1 and 3. These functional forms and parameter values are drawn from conversations with Ken Allgeyer, Vice President and General Sales Manager at Fox Sports Midwest (personal communication, 9 June 2017). We use n-tuple dual bounds to construct lookahead policies and to facilitate BBI. We explore values for n equal to 1, 2, and 3. Methods are implemented in C++ and executed on a heterogeneous computing cluster.

First, we explore the computational limits of BBI and compare it to conventional backward induction. We randomly generate 10 sets of costs, exposures, and spot limits. We pair these parameters with budget scenario sets $\boldsymbol{W}(\Gamma)$ and attempt to execute conventional backward induction. We also pair these parameters with budget scenario sets $\boldsymbol{W}^\star(\Gamma)$ and attempt to execute BBI with 1-tuple, 2-tuple, and 3-tuple dual bounds and lookahead policies. Table 1 displays the results. For each combination of T and Γ, the table shows the average size of the decision tree for each method along with the average number of CPU seconds required to build and solve the tree.

The figures in Table 1 make a compelling case for BBI. When T is 5, conventional backward induction is tractable and we can make a direct comparison among all methods. As Γ increases from 1 to 3, the additional scenarios result in a conventional decision tree that grows on average

Table 1: Bounded Backward Induction

		Backward Induction		**Bounded Backward Induction**					
				1-tuple		**2-tuple**		**3-tuple**	
T	Γ	**size**	**cpu**	**size**	**cpu**	**size**	**cpu**	**size**	**cpu**
5	1	176,248	1	475	1	5	1	1	1
5	2	505,275	1	74	1	1	2	1	4
5	3	853,331	2	1	1	1	3	1	5
6	1	—	—	3,350	12	1	13	1	19
6	2	—	—	5,491	37	7	126	6	393
6	3	—	—	843	34	1	215	1	782
7	1	—	—	28,092	291	3	410	—	—
7	2	—	—	95,000	1,064	488	12,403	—	—
7	3	—	—	148,114	3,230	548	28,527	—	—
8	1	—	—	194,941	3,624	—	—	—	—
8	2	—	—	869,179	16,302	—	—	—	—
8	3	—	—	1,380,354	28,981	—	—	—	—

nearly five times from just over 176K states to just over 853K states. Sizes vary widely with the number of feasible actions, which depend on the randomly generated cost structure and spot limits. In one instance, the size of the tree is nearly 2.5M. Even at this size, computing time is small and memory usage is manageable. Impressively, BBI reduces decision tree sizes from millions to several hundred, to less than a hundred, to less than ten, and in many cases to only one. As n increases, the reduction is more pronounced and computing time is larger. Although BBI requires more computing time relative to conventional backward induction when T is 5, BBI consumes less memory. Indeed, when T increases to 6, the size of the conventional decision tree grows so large that it is difficult to store in memory and conventional backward induction becomes intractable. In contrast, BBI takes these problem instances in stride. The procedure is tractable with 1-tuple, 2-tuple, and 3-tuple dual bounds and lookahead policies. Though larger values of n yield the smallest decision trees, in most cases they require substantially more computing time. When T increases to 7, BBI with 1-tuple and 2-tuple dual bounds and lookahead policies is tractable. However, the computing time required with 3-tuple dual bounds and policies is prohibitively large. Similarly,

when T increases to 8, the computing time for BBI with 2-tuple dual bounds and lookahead policies becomes excessive. Yet, BBI with 1-tuple dual bounds and lookahead policies still delivers optimal policies. The procedure appears to reach its computational limits here. When Γ is 3, the average decision tree size is just over 1.3M and the average CPU requirement is nearly 30K seconds. For some instances, however, the size is nearly 6.5M and the computing time surpasses 77K seconds, or nearly 22 hours. To go further may require additional computing resources, problem-specific lower and upper bounds, or both. Fortunately, a horizon of T equal to 8 is sufficient for many limited-time media campaigns, e.g., weekly decisions on spot purchases across two months leading up to the release of a new product.

Next, we examine the performance guarantees afforded to lookahead policies. Let $\overline{\pi}$ be the lookahead policy built on the n-tuple dual bound. Let $\epsilon = \min\{J_2^{\overline{\pi}}(s_2)/J_2^n(s_2, \boldsymbol{W}_2^\star(\Gamma_2)) : s_2 \in S_2(s_1, \mu_1^{\overline{\pi}}(s_1))\}$ be the smallest ratio of lookahead policy value to dual bound over all immediate successors of the initial state. By construction, ϵ satisfies the condition of Theorem 4, and thus we know that the ratio $J_1^{\overline{\pi}}(s_1)/J_1(s_1)$ of lookahead policy value to optimal policy value is at least as large as ϵ. We track ϵ for each initial state across all executions of BBI required to assemble the figures in Table 1. We also track the ratio $J_1^{\overline{\pi}}(s_1)/J_1^n(s_1, \boldsymbol{W}^\star(\Gamma))$ of lookahead policy value to dual bound from the initial state. Both performance guarantees for the lookahead policy are byproducts of the decision tree construction procedure outlined in Algorithm 1. Figure 1 displays the results. For all problem instances and policies associated with a particular value of horizon T, the figure shows average values of each performance guarantee individually as well as average values of the maximum of the two. The maximum represents the best performance guarantee that BBI can offer from the initial state.

Across all values of horizon T, Figure 1 shows that, on average, ϵ is 0.740, the ratio $J_1^{\overline{\pi}}(s_1)/J_1(s_1)$ is 0.813, and the maximum of the two is 0.826. In the world of worst-case guarantees, these numbers are very strong. If a full execution of BBI is intractable, then the decision maker may rely on these guarantees as measures of goodness for their policies. Further, the maximum of the two ratios improves on either ratio in isolation. On average, the maximum is 0.086 percentage points higher than ϵ and 0.013 percentage points above $J_1^{\overline{\pi}}(s_1)/J_1(s_1)$. Thus, the value of Theorem 4 is not that it provides a superior performance guarantee across the board. The value is that it connects lower and dual bounds between one stage and the next, and that this connection leads to a better

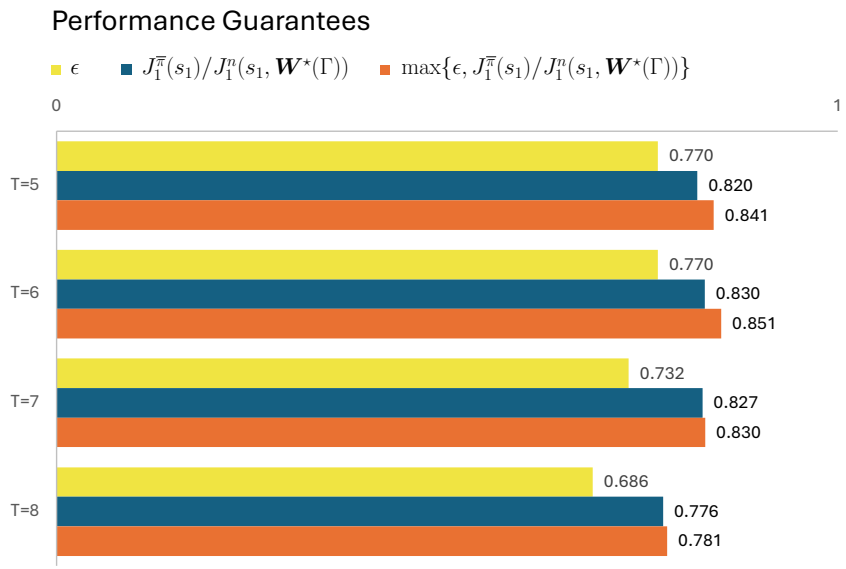

Figure 1: Lookahead Policy Performance Guarantees

performance guarantee in some cases.

Next, we study how n-tuple dual bounds affect BBI. We fix the horizon T to 5 periods and randomly generate 10 sets of costs, exposures, and spot limits. We pair these parameters with scenario sets $\boldsymbol{W}(\Gamma)$ and $\boldsymbol{W}^\star(\Gamma)$. For each of the resulting problem instances, we calculate the n-tuple dual bounds. Figure 2 displays the results. Each chart in the top row is a relative frequency histogram. For each n-tuple dual bound $J_1^n(s_1, \boldsymbol{W}(\Gamma))$ and $J_1^n(s_1, \boldsymbol{W}^\star(\Gamma))$, the charts display the proportion of n-tuples $\boldsymbol{W}_1'$ in $[\boldsymbol{W}(\Gamma)]^n$ and in $[\boldsymbol{W}^\star(\Gamma)]^n$ whose dual bound values $J_1(s_1, \boldsymbol{W}_1')$ equal the value of an optimal policy $J_1(s_1)$, the proportion whose dual bound values are between $J_1(s_1)$ and 20 percent above $J_1(s_1)$, the proportion whose dual bound values are between 20 and 40 percent above $J_1(s_1)$, and so on. Each histogram is an average across all problem instances with scenario set $\boldsymbol{W}(\Gamma)$ or scenario set $\boldsymbol{W}^\star(\Gamma)$. The first bar chart in the bottom row displays the average number of n-tuples $|[\boldsymbol{W}(\Gamma)]^n|$ and $|[\boldsymbol{W}^\star(\Gamma)]^n|$ associated with each n-tuple dual bound. The second chart shows the average number of CPU seconds required to calculate each n-tuple dual bound. The third chart depicts the average number of CPU seconds required to calculate the dual bound $J_1(s_1, \boldsymbol{W}_1')$ for each n-tuple $\boldsymbol{W}_1'$ in $[\boldsymbol{W}(\Gamma)]^n$ and $[\boldsymbol{W}^\star(\Gamma)]^n$. The fourth chart reports the average size of the decision trees required to calculate each $J_1(s_1, \boldsymbol{W}_1')$. Across all four charts in the bottom row, each bar represents an average over all problem instances with scenario set

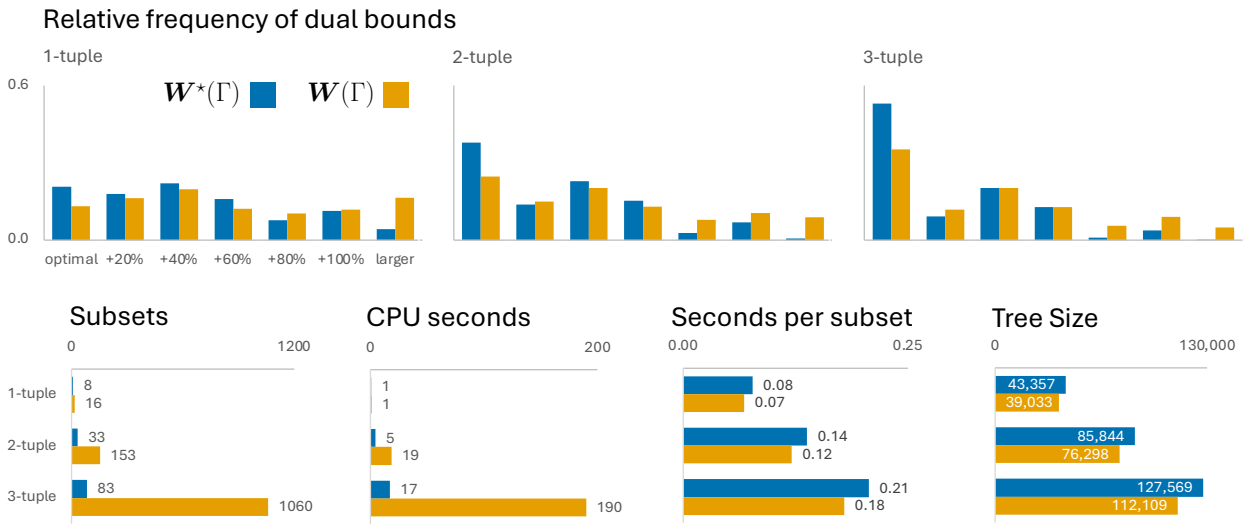

Figure 2: n-Tuple Dual Bounds

$\boldsymbol{W}(\Gamma)$ or scenario set $\boldsymbol{W}^\star(\Gamma)$.

The results portrayed in Figure 2 illustrate the benefit of composing n-tuples from $\boldsymbol{W}^\star(\Gamma)$. They also characterize the trade-off between the quality of n-tuple dual bounds and the computational effort required to calculate them. The histograms in the top row indicate that as n increases, the distributions of dual bounds skew toward the optimal policy value. Then, for a given value of n, the distribution of dual bounds is more skewed toward the optimal policy value for n-tuples in $[\boldsymbol{W}^\star(\Gamma)]^n$ than for those in $[\boldsymbol{W}(\Gamma)]^n$. Thus, the likelihood of identifying a better dual bound is higher when n is larger and when n-tuples are composed of scenarios in $\boldsymbol{W}^\star(\Gamma)$. The first chart in the bottom row shows that the average number of n-tuples in $[\boldsymbol{W}^\star(\Gamma)]^n$ is far fewer than the average number in $[\boldsymbol{W}(\Gamma)]^n$. Accordingly, the second chart indicates a substantial decrease in the average computation time required to calculate $J_1^n(s_1, \boldsymbol{W}^\star(\Gamma))$ versus $J_1^n(s_1, \boldsymbol{W}(\Gamma))$. However, as shown in the third chart, the average number of CPU seconds needed to identify each dual bound $J_1(s_1, \boldsymbol{W}_1')$ is higher for n-tuples $\boldsymbol{W}_1'$ in $[\boldsymbol{W}^\star(\Gamma)]^n$. Recall that, by construction, each scenario in these n-tuples contains outcomes whose magnitudes exhaust as much of the remaining budget of uncertainty as possible. Consequently, the associated decision trees are larger, as seen in the fourth chart, and the backward induction procedure takes longer. Thus, although more time is required to treat each n-tuple in $[\boldsymbol{W}^\star(\Gamma)]^n$, the total time required to calculate $J_1^n(s_1, \boldsymbol{W}^\star(\Gamma))$ is smaller than the total time required to calculate $J_1^n(s_1, \boldsymbol{W}(\Gamma))$. In short, the primary takeaways from Figure 2

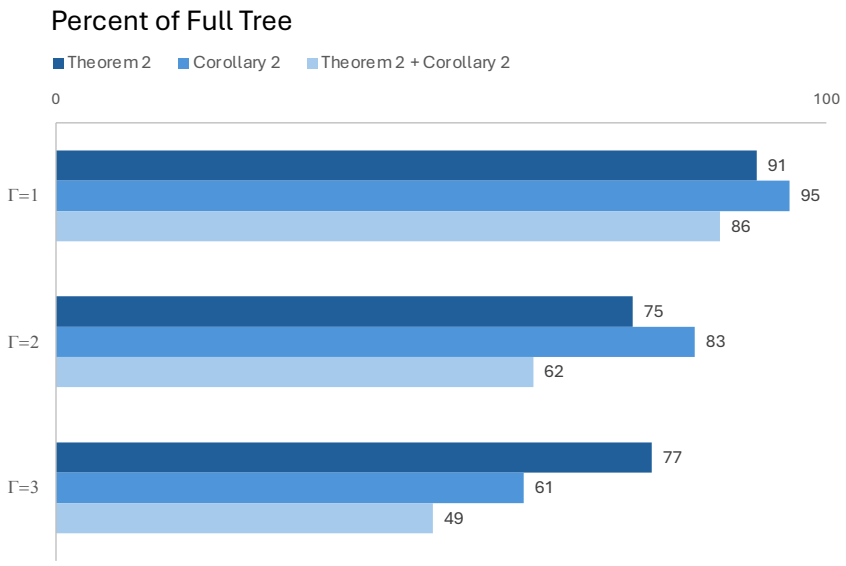

Figure 3: Size of the Policy Evaluation Tree

are that n-tuples should be composed from $\boldsymbol{W}^\star(\Gamma)$ (see Theorem 5, Corollary 2, Theorem 6, and Corollary 3) and that larger values of n yield stronger dual bounds (see Theorem 3), but at the cost of higher computation time.

Finally, we investigate the impact of our theoretical results on policy evaluation. Across the same problem instances with T fixed at 5, we consider lookahead policies $\overline{\pi}$ built on 1-tuple dual bounds. We use the reaching algorithm described in §5 to identify policy values $J_1^{\overline{\pi}}(s_1)$ from initial states. We track the size of the policy evaluation tree across four cases. The first case fully enumerates all states in the tree for problem instances with scenario set $\boldsymbol{W}(\Gamma)$. The second case applies the first part of Theorem 2, as described in §5, to problem instances with scenario set $\boldsymbol{W}(\Gamma)$. Given two sets of budget scenarios $\boldsymbol{W}_t(\Gamma_t)$ and $\boldsymbol{W}_t(\Gamma'_t)$, the first is a subset of the second if $\Gamma_t \leqslant \Gamma'_t$. The third case applies Corollary 2 by reducing the scenario set to $\boldsymbol{W}^\star(\Gamma)$. The fourth case employs Theorem 2 and Corollary 2 in tandem. Figure 3 displays the size of the tree in the latter three cases as a percent of the size in the first case. Each bar represents the average percent across the corresponding problem instances for the displayed value of Γ.

The results shown in Figure 3 indicate that both Theorem 2 and Corollary 2 can in isolation reduce the size of the policy evaluation decision tree. However, the methods are most beneficial when used together. This is particularly true as Γ increases. Indeed, when Γ is 3, the average

size of the decision tree is reduced by more than half, from 56 states to 27 states, on average. Whether the reduction in size is small or large, the computational overhead required to implement these results is negligible. This is notable because BBI builds a decision tree across all policies by examining lower bounds at every candidate state. Because the number of such states can be very large, any reductions to the size of the policy evaluation decision tree can substantially lessen total computation time.

In summary, the application of BBI to media selection illustrates all the theoretical work in the paper. Theorem 1 and Corollary 1 eliminate suboptimal actions. Theorem 2 provides a basis for dual bounds via scenario subsets and Theorem 3 supplies a general methodology for n-tuple dual bounds. Theorem 4 establishes a second performance guarantee for lookahead policies. Theorem 5 and Corollary 2 significantly reduce the number of budget scenarios. Then, Theorem 6 transfers this benefit to dual problems with budget scenarios and Corollary 3 shows an application to n-tuple dual bounds. Collectively, these theoretical results make a practical contribution by identifying optimal policies for problem instances whose sizes are orders of magnitude beyond what conventional backward induction can tractably manage. These results point to BBI as a useful solution methodology for media selection problems and for max-min DPs in general.

9 Conclusion

The utility of conventional backward induction is limited by the curse of dimensionality. For many max-min DPs of practical interest, conventional backward induction is an intractable solution method. BBI shifts the problem of dimensionality to the task of developing strong and tractable bounds. The upper and lower bounds we develop are applicable to general max-min DPs. The n-tuple dual bounds facilitate a tradeoff between quality and computation, the ensuing lookahead policies come with performance guarantees, and our analysis of budget scenario sets eases the tasks of policy evaluation and dual bound calculation. Using our bounds, BBI solves media selection problems orders of magnitude larger than what is tractable with conventional backward induction. This success points toward BBI as a useful solution methodology for max-min DPs.

The generality of BBI offers opportunities to explore and expand the method. Dual bounds and policies can be mixed and matched, for example, by moving from one n-tuple to another

during various parts of decision tree construction. This could provide some flexibility for a given computational budget by placing better bounds where they are most helpful. Or, instead of leaning on the bounds proposed in this paper, new bounds could be developed. BBI does not require that bounds be general. The theory permits problem-specific bounds. For instance, a variety of reinforcement learning techniques and functional approximations, tailored to a given problem, could yield improvements in bound quality and computation time, thereby facilitating a smaller decision tree with fewer demands on computational resources. In this sense, BBI is a framework that can be customized.

Beyond working within our analysis of BBI, it may be possible to extend it. While budget scenario sets offer certain methodological advantages, they are not the only way to model uncertainty. Other types of scenario sets should be explored. Additionally, although the BBI framework requires a finite horizon, it may be possible to incorporate bounds into a solution methodology for infinite horizon problems. Finally, it may be possible to adapt the general notion of eliminating suboptimal actions via bounds to a stochastic DP framework.

References

Adelman, D. and A. Mersereau (2008). Relaxations of weakly coupled stochastic dynamic programs. *Operations Research 56*(3), 712–727.

Balseiro, S. R. and D. B. Brown (2019). Approximations to stochastic dynamic programs via information relaxation duality. *Operations Research 67*(2), 577–597.

Bertsekas, D. (2017). *Dynamic programming and optimal control* (4th ed.), Volume I. Belmont, MA: Athena Scientific.

Bertsekas, D. (2019a). *Reinforcement Learning and Optimal Control*. Athena Scientific.

Bertsekas, D. (2021). Distributed asynchronous policy iteration for sequential zero-sum games and minimax control. `https://arxiv.org/abs/2107.10406`, Accessed on July 11, 2024.

Bertsekas, D. (2022). Reinforcement learning course at ASU. `https://www.youtube.com/watch?v=pM6i_E9f_jQ`, Accessed on July 11, 2024.

Bertsekas, D. P. (2019b). Robust shortest path planning and semicontractive dynamic programming. *Naval Research Logistics (NRL) 66*(1), 15–37.

Bertsimas, D. and D. Brown (2011). Theory and applications of robust optimization. *SIAM Review 53*(3), 464–501.

Betsimas, D. and A. Thiele (2006). Robust and data-driven optimization: Modern decision-making under uncertainty. In M. Johnson, B. Norman, N. Secomandi, and P. Gray (Eds.), *Tutorials on Operations Research*. INFORMS.

Brown, D. and J. Smith (2014). Information relaxations, duality, and convex stochastic dynamic programs. *Operations Research 62*(6), 1394–1415.

Brown, D., J. Smith, and P. Sun (2010). Information relaxations and duality in stochastic dynamic programs. *Operations Research 58*(4), 785–801.

De Kluyver, C. (1980). Media selection by mean-variance analysis. *European Journal of Operational Research 5*(2), 112–117.

Delage, E. and D. A. Iancu (2015). Robust multistage decision making. In *Tutorials in Operations Research: The Operations Research Revolution*, pp. 20–46.

Delage, E. and S. Mannor (2010). Percentile optimization for markov decision processes with parameter uncertainty. *Operations Research 58*(1), 203–213.

Denardo, E. (2003). *Dynamic programming: models and applications*. Mineola, NY: Dover Publications.

Goyal, V. and J. Grand-Clément (2023). Robust markov decision processes: Beyond rectangularity. *Mathematics of Operations Research 48*(1), 203–226.

Haugh, M. and C. Wang (2015). Information relaxations and dynamic zero-sum games. `https://arxiv.org/abs/1405.4347`, Accessed on July 12, 2024.

Iyengar, G. N. (2005). Robust dynamic programming. *Mathematics of Operations Research 30*(2), 257–280.

Kwak, N., C. Lee, and J. Kim (2005). An mcdm model for media selection in the dual consumer/industrial market. *European Journal of Operational Research 166*, 25–265.

Little, J. and L. Lodish (1969). A media planning calculus. *Operations Research 17*(1), 1–35.

Maggioni, F. and G. C. Pflug (2016). Bounds and approximations for multistage stochastic programs. *SIAM Journal on Optimization 26*(1), 831–855.

Nilim, A. and L. El Ghaoui (2005). Robust control of markov decision processes with uncertain transition matrices. *Operations Research 53*(5), 780–798.

Pearl, J. (1980). Asymptotic properties of minimax trees and game-searching procedures. *Artificial Intelligence 14*(2), 113–138.

Pearl, J. (1982). The solution for the branching factor of the alpha-beta pruning algorithm and its optimality. *Communications of the ACM 25*(8), 559–564.

Powell, W. (2022). *Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions*. John Wiley and Sons.

Puterman, M. (1994). *Markov decision processes: discrete stochastic dynamic programming*. New York, NY: Wiley.

Saen, R. (2011). Media selection in the presence of flexible factors and imprecise data. *Journal of the Operational Research Society 62*, 1695–1703.

Shapiro, A. (2011). A dynamic programming approach to adjustable robust optimization. *Operations Research Letters 39*, 83–87.

Srinivasan, V. (1976). Decomposition of a multi-period media scheduling model in terms of single period equivalents. *Management Science 23*(4), 349–360.

Sutton, R. and A. Barto (2020). *Reinforcement Learning: An Introduction* (2nd ed.). The MIT Press.

Wilson, C. (1962). Linear programming basics. In *Proceedings of the 8th annual conference of the advertising resesarch foundation*, pp. 78–100.

Xu, H. and S. Mannor (2006). The robustness-performance tradeoff in markov decision processes. In B. Schölkopf, J. Platt, and T. Hoffman (Eds.), *Advances in Neural Information Processing Systems*, Volume 19. MIT Press.

Ye, F., H. Zhu, and E. Zhou (2018). Weakly coupled dynamic program: Information and lagrangian relaxations. *IEEE Transactions on Automatic Control 63*(3), 698–713.

Zufryden, F. (1975). Media scheduling and solution approaches. *Operational Research Quarterly 26*(2), 283–295.