Budget Allocation using Weakly Coupled, Constrained Markov Decision Processes

Craig Boutilier
Google, Inc.
Mountain View, CA
cboutilier@google.com

Tyler Lu
Google, Inc.
Mountain View, CA
tylerlu@google.com

Abstract

We consider the problem of budget (or other resource) allocation in sequential decision problems involving a large number of concurrently running sub-processes, whose only interaction is through their consumption of budget. Our first contribution is the introduction of *budgeted MDPs (BMDPs)*, an MDP model in which policies/values are a *function of available budget*, (c.f. constrained MDPs which are solved given a *fixed* budget). BMDPs allow one to explicitly trade off allocated budget and expected value. We show that optimal value functions are concave, non-decreasing in budget, and piecewise-linear in the finite horizon case, and can be computed by dynamic programming (and support ready approximation). Our second contribution is a method that exploits BMDP solutions to allocate budget to a large number of independent BMDPs, coupled only by their common budget pool. The problem can be cast as a *multiple-choice knapsack problem*, which admits an efficient, optimal greedy algorithm. Empirical results in an online advertising domain confirm the efficacy of our methods.

1 INTRODUCTION

Markov decision processes (MDPs) [25, 7] are used widely throughout AI; but in many domains, actions consume limited resources and policies are subject to resource constraints, a problem often formulated using *constrained MDPs (CMDPs)* [2]. MDPs and CMDPs are even more complex when multiple independent MDPs, drawing from the same resources, must be controlled jointly, since the state and action spaces are formed by the cross-product of the individual subprocesses [23]. *Online advertising* is a domain with such properties: Archak et al. [4] propose a constrained MDP model for the optimal allocation of advertiser budget over an extended horizon that captures the sequential effect of multiple ads on a user's behavior.

Archak et al. [4] assume a fixed, predetermined budget for each user, and focus on optimally advertising to a user subject to this budget constraint. This formulation, however, does not determine a suitable budget, nor does it allow for making budget tradeoffs across different users, user types, or campaigns (e.g., different MDPs or different states of the same MDP). With this as motivation, we address these challenges by: (a) introducing *budgeted MDPs (BMDPs)*, which are solved as *a function of the (expected) budget available*; and (b) addressing budget tradeoffs across users using a *weakly coupled MDP* formulation [23] and optimally solving the allocation problem with a greedy algorithm that exploits local user BMDP solutions.

Our first contribution lies in the formulation and solution of BMDPs. The usual approach to resource constraints is CMDPs [2, 13]. While valuable models, CMDPs require *a priori* specification of fixed "budget" (e.g., a daily per-customer cap on ad spend). By contrast, BMDPs compute optimal policies and value functions (VFs) as a function of the budget made available. Effectively, we solve a CMDP *for all possible budget levels*, allowing one to explore the tradeoff between (optimal) expected value and allocated budget. Treating the budget b as a parameter, we show that, for any fixed state s of the BMDP, the optimal VF $V(s, b)$ is concave, non-decreasing in b; and for any finite horizon the VF is *piecewise-linear and concave (PWLC)*, defined by a finite set of *useful* resource levels. We derive a dynamic programming (DP) algorithm to compute this PWLC representation that supports approximation.

Our second contribution is a method for piecing together BMDP solutions to determine a joint policy over a set of BMDPs (e.g., for different users), subject to a global resource/budget constraint. Since the MDP is *weakly coupled* [23]—specifically, the individual customer BMDPs evolve independently, linked only through the consumption of shared budget—our aim is to determine an allocation of budget to each customer, which is turn dictates the optimal policy for that customer. We show that the budget allocation problem can be formulated as a *(multi-item variant of a) multiple-choice knapsack problem (MCKP)* [29], for which a straightforward greedy method can be used to construct the optimal budget allocation. We also discuss circumstances in which the dynamic, online *reallocation* of budget may be valuable, an approach rendered viable by the real-time nature of our method.

2 MDPS FOR BUDGET ALLOCATION

We describe an MDP model for engagement with users from a large, heterogeneous population. We use online advertising as our main motivation, though our techniques apply to the control of any large, distributed set of fully observable MDPs where actions consume limited resources. We abstract away a number of factors that arise in realistic ad domains to focus on budget allocation itself (e.g., partial observability and hidden state, control lag, incentives).

2.1 A WEAKLY COUPLED FORMULATION

We assume an advertiser has a fixed budget to spend on *advertising actions* for a target user population. Each action has a cost and is targeted to a specific user (e.g., a search, in-app or web page ad). Users respond stochastically in a way that may depend on their features (e.g., demographics), past actions (e.g., ad exposures) and past responses (e.g., click/purchase behavior).

Following Archak *et al.* [4], we model this as an MDP. We have a finite set of *users* $i \leq M$, who may be segmented into *types* reflecting static, observable characteristics that influence their responses. For ease of exposition, we assume all users have the same type; but the extension to multiple types is straightforward. We have a finite set S of *user states* $j \leq N$. At any time, user i is some state $s[i] \in S$. Let $\mathbf{S} = S^M$ be the *joint state space*, with the joint state denoted $\mathbf{s} = \langle s[1], \ldots, s[M] \rangle \in \mathbf{S}$. A user's state captures all relevant characteristics and history that influence her behavior. S may be small, or quite large in some contexts (e.g., the most recent search keyword on which the advertiser bid, or sufficient statistics summarizing historical interactions and user responses). A finite set A of *advertising actions* is available. At each stage, the advertiser selects an action $a[i]$ to apply to user i. Letting $\mathbf{A} = A^M$, a joint action is $\mathbf{a} = \langle a[1], \ldots, a[M] \rangle \in \mathbf{A}$.

Stochastic user response is captured by a *transition model* $P : S \times A \to \Delta(S)$, where $P(i, a, j) = p^a_{ij}$ is the probability that a user in state i moves to j when subjected to action a. *Reward* $R(i, a) = r^a_i$ reflects costs/payoffs when action a is applied to a user in state i. We decompose reward as $r^a_i = U(i) - C(i, a) = u_i - c^a_i$: *cost* C reflects action costs (e.g., cost of placing an ad, potential annoyance, etc.) and *utility function* U reflects benefits/payoffs (e.g., sales revenue, value of brand exposure, etc.).

The advertiser has a maximum *(global) budget* B that can be spent over the planning horizon. This global budget may be a hard limit in some settings; but we will require only that policies meet this constraint *in expectation*. We assume that the set of users is known, but our model easily handles new users who enter the system according to a known distribution over initial states. Different users will occupy many distinct states at any stage.

The optimal policy is defined w.r.t. the *joint constrained MDP*, with state space $\mathbf{S} = S^M$, action set $\mathbf{A} = A^M$, and a transition model, and cost, utility and reward functions defined as follows:

$$P(\mathbf{s}, \mathbf{a}, \mathbf{t}) = \prod_{i \leq M} P(s[i], a[i], t[i]); \quad U(\mathbf{s}) = \sum_{i \leq M} U(s[i]);$$

$$C(\mathbf{s}, \mathbf{a}) = \sum_{i \leq M} C(s[i], a[i]); \quad R(\mathbf{s}, \mathbf{a}) = U(\mathbf{s}) - C(\mathbf{s}, \mathbf{a}).$$

Joint transitions reflect the natural independence of user transitions. Costs and utilities are additive across users.

Our aim is to find a policy that maximizes expected discounted reward subject to the budget constraint: in expectation, the policy should spend no more than B. The optimal solution to this joint CMDP can be found by linear programming (LP) or DP. However, the exponential size of the state and action spaces (e.g., $O(N^M)$ states, or $\binom{M+N-1}{N-1}$ states if we use the "user count" for each state) makes this intractable. Fortunately, the MDP is *weakly coupled* [23]: users transitions are independent, with the local MDPs only coupled by their reliance on a single global budget. We take advantage of this below, solving the local MDPs such that their solutions can be effectively "pieced together" to form an approximate solution to the joint problem.

2.2 RELATED WORK

Budget Optimization. Allocation of advertising budgets is well-studied in marketing [19] with customer behavior and responses often modeled as a discrete or continuous Markov process [8, 24, 15]. Constrained budget optimization is of course critical to maximizing ROI in keyword auctions as well. [6, 16, 17, 12, 20].

Relatively little work has considered online budget optimization based on *sequential* user behavior. Markov models of web browsing and user response to online ads have been studied [10, 21]. Archak et al. [4] also assume Markovian user behavior in sponsored search—this is the behavioral model we adopt above. They propose a constrained MDP model and simple greedy algorithm that determines the optimal ad policy for a given user, assuming a fixed budget *for that user*. The same authors [5] demonstrate that user web search exhibits a "general to specific" behavior that is approximately Markovian. Other sequential work includes: budget optimization with advertiser response learning [3]; and reinforcement learning for optimal online ad strategies [27, 31] without budgets.

Constrained MDPs. We can extend the standard MDP model using *constrained MDPs (CMDPs)* [2, 13]: actions consume one or more resources (e.g., budget, energy) and policies must use no more than some set level of each resource in expectation. We outline a basic (one-dimensional) CMDP model.

Assume an underlying (local, not joint) MDP with states, actions, transitions, utilities and costs as above. Assuming a discounted infinite-horizon problem with discount factor

$0 \leq \gamma < 1$, our aim is to find an optimal (stationary, deterministic) policy that maximizes the expected sum of discounted rewards. The (unique) *optimal value function (VF)* $V^* : S \to \mathbb{R}$ satisfies the Bellman equation for all $i \in S$:

$$V^*(i) = \max_{a \in A} r_i^a + \gamma \sum_{j \in S} p_{ij}^a V^*(j),$$

while the optimal policy $\pi^*(i)$ selects the argmax of this expression. We will often use *Q-functions*, $Q(i,a) = r_i^a + \gamma \sum_{j \in S} p_{ij}^a V^*(j)$. The optimal VF and policy can be computed using DP algorithms (e.g., value/policy iteration) or LP methods [25, 7].

CMDPs extend this model by introducing constraints on the resources used by a policy [2]. We explicate the model using budgets (but it applies to any resource). A *budget constraint* B limits the cost of allowable policies, where the *expected discounted cost* of policy π at state i is:

$$C^\pi(i) = c_i^{\pi(i)} + \gamma \sum_{j \in S} p_{ij}^{\pi(i)} C^\pi(j).$$

An initial state (or distribution) is required for the constraint to be well-posed in general. The optimal solution (policy, VF) can be computed in poly-time using an LP [2]. There is always an optimal stationary policy for a CMDP, but unlike unconstrained MDPs, it may be stochastic.

Satisfying the budget constraint *in expectation* is suitable when a policy is executed many times, perhaps under different conditions, and budget is fungible over these instances, as in our ad domain. In other settings, the budget constraint may be *strict*—our model below easily accommodates strict budgets as well. We also allow costs in the budget constraint to be undiscounted (see below).

Weakly Coupled MDPs. The decomposition of MDPs into independent or semi-independent processes can often be used to mitigate the curse of dimensionality. Challenges lie in discovering a suitable decomposition structure and in determining how best to use the sub-process solutions to construct a (generally approximate) global solution. Many approaches have this flavor, in both standard MDPs and *decentralized (DEC)* MDPs and POMDPs [28, 1, 30, 22, 14]. The approach most related to ours is the decomposition method for *weakly-coupled MDPs* of [23]. There a joint MDP is comprised of a set of independent subprocesses, each itself a "local" MDP. Each local MDP reflects the task or objective of a specific agent, but the local policies require resources, both consumable and non-consumable. Their method: solves the local MDPs independently to produce local VFs parameterized by the resources available; uses the local VFs to assign resources to each local MDP; and reassigns unconsumed resources at each stage given the observed joint state. Our approach to budget decomposition is similar, but we use of the more standard *expected budget* constraint, and guarantee optimal composition. Our dynamic budget reallocation scheme is based on the reallocation mechanism of [23].

3 BUDGETED MDPS

We introduce *budgeted MDPs*, a variant of CMDPs in which budgets, or other resources, are (implicitly) treated as a part of the state, so that VFs/policies can vary with both the state *and available budget*. This allows budget-value tradeoffs to made quickly and easily.

3.1 THE BUDGETED MDP MODEL

A *(finite, one-dimensional) budgeted Markov decision process (BMDP)* $M = \langle S, A, P, U, C, B_{\max}, \gamma \rangle$ has the same components as a CMDP, but without a budget constraint. We allow an optional constant $B_{\max}$ that sets a plausible upper bound on useful or available budgets at any stage. We write $U(i) = u_i$ for the terminal utility at i if no action is taken (e.g., end of the planning horizon). We assume that, for each i, there is an a s.t. $c_i^a = 0$ (so a proper policy exists even with no budget).

We seek an optimal policy which maximizes expected reward over some horizon, but which consumes no more than some budget $b \leq B_{\max}$ in expectation. Unlike CMDPs, however, the policy should be a *function* of b.

3.2 DETERMINISTIC POLICIES

We begin by analyzing deterministic policies for finite-horizon problems, which develops useful intuitions. Define the *optimal deterministic t-stage-to-go VF* as follows. For all $i \in S, b \leq B_{\max}$, let $V_D^0(i,b) = u_i$, and define:

$$V_D^t(i,b) = \max_{\substack{a \in A \\ \mathbf{b} \in \mathbb{R}_+^n}} r_i^a + \gamma \sum_{j \leq n} p_{ij}^a V_D^{t-1}(j, b_j) \qquad (1)$$

$$\text{subj. to } c_i^a + \gamma \sum_{j \leq n} p_{ij}^a b_j \leq b \qquad (2)$$

The optimal policy $\pi_D^t(i,b)$ is obtained by taking the argmax. The VF reflects that doing a at i consumes c_i^a of the budget b, while optimal consumption at each reachable next state must be such that the *expected* budget used is no more than the remaining $b - c_i^a$. We discount the expected spend as is standard in CMDPs, but removing γ from Eq. 2 is also possible (we use undiscounted constraints below).

It is easy to see that $V_D^t(i,b)$ is monotone non-decreasing in b. While the optimal VF involves a continuous dimension b, it has a concise finite representation. For a fixed stage-to-go t and state i, define budget $0 \leq b \leq B_{\max}$ to be *(deterministically) useful* iff $V_D^t(i,b) > V_D^t(i, b - \varepsilon)$ for all $\varepsilon > 0$, and *useless* otherwise. We observe that:

Proposition 1. *For any finite t and state i, $V_D^t(i,\cdot)$ has a finite number of deterministically useful budget levels.*

We describe an algorithm to compute useful budgets (from which the proof of Prop. 1 follows).[1] Let $b_0^{i,t} = 0 < b_1^{i,t} < \ldots < b_M^{i,t}$ be the useful budget levels for (i,t). Prop. 1

[1]All proofs, and more detailed exposition, are available in a longer version of the paper, available at each author's web page.

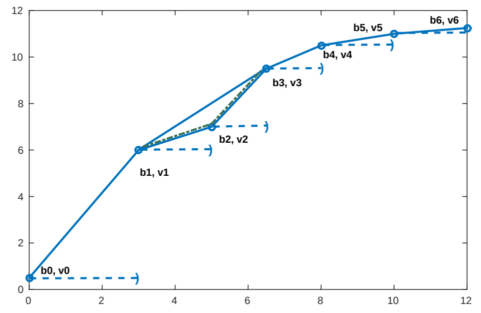


Fig. 1: An example VF $V_D(i,b)$. Useful budget levels are shown (the points (b,v)) along with: (a) the induced piecewise constant VF $V_D(i,b)$ (dashed line); and (b) the PWLC VF (solid line) for a randomized policy. Randomizing among b_0, b_1 for any expected spend $b \in (b_0, b_1)$ dominates deterministic spend b_0; and randomizing among b_1 and b_3 with expected spend b_2 dominates deterministic spend b_2 (dashed-dotted lines).

implies that $V^t_D(i,\cdot)$ is *piecewise constant* and monotone, with $V^t_D(i,b) = V^t_D(i,b^{i,t}_k)$ for all $b \in [b^{i,t}_k, b^{i,t}_{k+1})$, for each $k < M$. We write $[\langle b^{i,t}_0, v^{i,t}_0\rangle, \ldots, \langle b^{i,t}_M, v^{i,t}_M\rangle]$ to denote this piecewise constant function (see Fig. 1).

Let the *reachable set* for state-action pair i, a be $S^a_i = \{j \in S | p^a_{ij} > 0\}$ and the *maximum out-degree* of an MDP be $d = \max_{i\in S, a\in A} |S^a_i|$. We compute the useful budgets for each state-stage pair—hence the optimal VF and policy—using a simple DP algorithm. Assume a piecewise constant representation of V^{t-1}_D (for V^0_D this is trivial); and for ease of exposition, assume each i has $M+1$ useful levels $b^{i,t-1}_0, \ldots, b^{i,t-1}_M$. We compute $V^t_D(i,b)$ as follows:

- Let $\sigma : S^a_i \to [M]$ be an assignment of each reachable state $j \in S^a_i$ to a useful budget level $b^{j,t-1}_{\sigma(j)}$, $\sigma(j) \le M$, with $t-1$ stages-to-go. Let Σ be the set of such mappings. We view $b^{j,t-1}_\sigma(j)$ as the budget to be consumed if we reach j after doing a; implicitly, σ dictates the $t-1$-stage-to-go policy by selecting a budget level at each next state.
- Let the *potentially useful budget levels* for (i,t,a) be:

$$\widetilde{B}^{i,t}_a = \{c^a_i + \sum_{j\in S^a_i} p^a_{ij} b^{j,t-1}_{\sigma(j)} \mid \sigma \in \Sigma\} \cap \{b \le B_{\max}\}.$$

  Each $b^{i,t}_k \in \widetilde{B}^{i,t}_a$ is determined by some σ. The corresponding expected value is $v^{i,t}_k = r^a_i + \gamma \sum_j p^a_{ij} v^{j,t-1}_{\sigma(j)}$.
- Assume a reindexing of the entries in $\widetilde{B}^{i,t}_a$ so that the budget levels are ascending (ties broken arbitrarily). The *useful budget levels* for (i,t,a) are:

$$B^{i,t}_a = \{b^{i,t}_k \in \widetilde{B}^{i,t}_a : \nexists k' < k \text{ s.t. } v^{i,t}_{k'} \ge v^{i,t}_k\}.$$

  That is, any potentially useful budget that is weakly dominated by a smaller budget is discarded. The useful budgets and corresponding values give us $Q^t(i,a,b)$.
- Let the *potentially useful budget levels* for (i,t) be $\widetilde{B}^{i,t} = \cup_{a\in A} B^{i,t}_a$, and let $B^{i,t}$ be the *useful budget levels*, obtained by pruning $\widetilde{B}^{i,t}$ as above. The useful budget levels and corresponding values give the VF $V^t(i,b)$.

While finite, the number of useful budget levels can grow exponentially in the horizon:

Proposition 2. *For any finite t and state i, $V^t_D(i,\cdot)$ has at most $O((|A|^d)^t)$ useful budget levels, where d is the maximum out-degree of the underlying MDP.*

This motivates approximating this set, as we discuss below in the context of stochastic policies.

3.3 STOCHASTIC POLICIES

Stochastic policies can offer greater value than deterministic policies due to their inherent flexibility, and thus have a rather different structure. Suppose at state-stage pair (i,t) the available budget b lies strictly between two deterministically useful levels, $b^{i,t}_k < b < b^{i,t}_{k+1}$. A stochastic policy that spends $b^{i,t}_{k+1}$ (and takes the corresponding action) with probability $p = \frac{b - b^{i,t}_k}{b^{i,t}_{k+1} - b^{i,t}_k}$, and $b^{i,t}_k$ with $1-p$, provides greater expected value for (expected) spend b than the PW constant value offered by the optimal deterministic policy (see Fig. 1).

The optimal VF for such "single-stage randomized" policies is given by the *convex hull* of the useful budgets for (i,t) (see Fig. 1). Given useful budget set $B^{i,t} = \{0 = b^{i,t}_0 < b^{i,t}_1 < \ldots < b^{i,t}_M\}$, we say $b^{i,t}_k$ is *dominated* if there are two budgets $b^{i,t}_{k^-}, b^{i,t}_{k^+}$ $(k^- < k < k^+)$ s.t. $(1-p)v^{i,t}_{k^-} + pv^{i,t}_{k^+} > v^{i,t}_k$. The convex hull is *piecewise linear and concave (PWLC)* and monotone, comprising the PWL function formed by the *non-dominated* points.

This PWLC structure is preserved by Bellman backups, and can be computed effectively in two stages: first, a simple *greedy algorithm* assigns budget incrementally to reachable next states, giving a PWLC representation of the Q-functions for each a; second, we compute the backed up VF by taking the convex hull of the union of these Q-functions.

Computing Q-functions. Assume $V^{t-1}(j,\cdot)$ is PWLC for all $j \in S$, with points $[\langle b^{j,t-1}_0, v^{j,t-1}_0\rangle, \ldots, \langle b^{j,t-1}_M, v^{j,t-1}_M\rangle]$, where each $b^{j,t-1}_k$ is non-dominated (for ease of exposition, assume each j has $M+1$ non-dominated levels). Let $B(S^i_a) = \cup_{j\in S^i_a} B^{j,t-1}$, and re-index $B(S^i_a)$ in decreasing order of *bang-per-buck ratio (BpB)*:

$$BpB(b^{j,t-1}_k) = \frac{v^{j,t-1}_k - v^{j,t-1}_{k-1}}{b^{j,t-1}_k - b^{j,t-1}_{k-1}} = \frac{\Delta v^{j,t-1}_k}{\Delta b^{j,t-1}_k}.$$

(For $k=0$, we leave BpB undefined, since $b^{j,t-1}_0 = 0$ for all j.) This BpB expresses the (per-unit budget) increase in value when increasing budget at $(j,t-1)$ from $b^{j,t-1}_{k-1}$ to $b^{j,t-1}_k$. Let $j(m), k(m), \Delta b(m), \Delta v(m)$ denote, resp., the state j, the useful budget index for j, the budget increment, and the value increment associated with the mth element of (sorted) $B(S^i_a)$. Let $M^*_i = |S^i_a|M$.

We define Q-functions as follows:

Definition 1. Let V^{t-1} be a VF such that, for all $j \in S$, $V^{t-1}(j,b)$ is bounded, monotone and PWLC in b with a finite number of budget points. Define:

- the 0th useful budget for a at i to be $b^{i,t}_{a,0} = c^a_i$, which gives value $v^{i,t}_{a,0} = r^a_i + \gamma \sum_{j \in S^a_i} p^a_{ij} v^{j,t-1}_0$;
- the mth useful budget, for $0 < m \leq M^*_i$, to be $b^{i,t}_{a,m} = \sum_{\ell \leq m} p^a_{i,j(\ell)} \Delta b(\ell)$ which gives value $v^{i,t}_{a,m} = r^a_i + \gamma \sum_{\ell \leq m} p^a_{i,j(\ell)} \Delta v(\ell)$.

For any $0 < m \leq M^*_i$, and b s.t. $b^{i,t}_{a,m-1} < b < b^{i,t}_{a,m}$, let $p^{i,t,a}_b = \frac{b - b^{i,t}_{a,m-1}}{b^{i,t}_{a,m} - b^{i,t}_{a,m-1}}$. Define *the Q-function for action a at stage t* as follows:

$$Q^t(i,a,b) = \begin{cases} \text{undefined} & \text{if } b < b^{i,t}_{a,0} \\ v^{i,t}_{a,m} & \text{if } b = b^{i,t}_{a,m} \\ p^{i,t,a}_b v^{i,t}_{a,m} + (1 - p^{i,t,a}_b) v^{i,t}_{a,m-1} & \text{if } b^{i,t}_{a,m-1} < b < b^{i,t}_{a,m} \\ v^{i,t}_{a,M^*_i} & \text{if } b > b^{i,t}_{a,M^*_i}. \end{cases}$$

This Q-function is optimal:

Theorem 3. *$Q^t(i,a,b)$ in Defn. 1 is the maximum expected value achievable when taking action a at state i with budget b with future value given by V^{t-1}.*

Q-functions can thus be computed with a simple greedy algorithm that allocates budget to "next states" in BpB-order (i.e., using a simple sorted merge of the linear segments of all reachable-state VFs, with each segment scaled by its transition probability; see Fig. 2 for an illustration). The intuitions are straightforward: once taking a at i, cost c^a_i is incurred, and the remaining budget $b' = b - c^a_i$ must be "allocated" to states in S^a_i. The first units of b' are most effectively used by state $j(1)$—i.e., the first in $B(S^i_a)$—since it has the greatest initial BpB. Budget up to $\Delta b(1) = b^{j(1),t-1}_{k(1)}$ to $j(1)$ gives an expected (future) value improvement of $\Delta v(1) = \Delta v^{j(1),t-1}_{k(1)}$ with probability $p^a_{i,j(1)}$, and has an expected spend of $p^a_{i,j(1)} \Delta b(1)$. This is the greatest expected (future) value attainable at i for any $b' \leq p^a_{ij} b^{j(1),t-1}_{k(1)}$. Similarly, the next $\Delta b(2)$ units of b' should be allocated to $j(2)$, giving a return of $BpB(b^{j(2),t-1}_{k(2)})$ per unit, with expected spend and return occurring with probability $p^a_{i,j(2)}$. This continues until all useful budgets from states in S^a_i have been allocated (reaching the max useful budget for (i,t)).

Computing Value Functions. Given the PWLC representation of the Q-functions, we can construct a similar PWLC representation of $V^t(i,b) = \max_a Q^t(i,a,b)$. We simply take the union of the points that determine each Q-function, and remove any dominated points (analogous to the move from deterministic to stochastic policies). More precisely, assuming a fixed state-stage (i,t), let Q_a be the set of budget-value points in a's Q-function, and let $Q_* = \cup_a Q_a$ be the union of of these points—we annotate each point with the action from which it was derived, so each has the form (b,v,a). We say (b,v,a) is *dominated* in Q_* if there are two (different) points $(b_1,v_1,a_1),(b_2,v_2,a_2) \in Q_*$ such that $b_1 \leq b \leq b_2$ and $(1-\alpha)v_1 + \alpha v_2 > v$, where $\alpha = \frac{b-b_1}{b_2-b_1}$. Removing all dominated points from Q_* leaves the set of points that form the useful budget levels in the PWLC representation of $V^t(i,\cdot)$. In other words, we form the convex hull of Q_*. Clearly no dominated point (b,v,a) is useful in a stochastic policy, since a greater value can be attained, using the same expected budget, by α-mixing between actions a_1 and a_2 (and the corresponding budgets).

The construction above shows:

Theorem 4. *For any finite t and state i, $V^t(i,b)$ is piecewise linear, concave and monotone in b.*

The PWLC representation of $V^t(i,\cdot)$ can be constructed using any convex hull method. A basic *Graham scan* [18] is appropriate here, since Q-budget points are maintained in sorted order, and has complexity $O(|Q_*| \log |Q_*|)$.

Again, the number of useful levels grows exponentially, but is no greater than the number of deterministically useful levels, i.e., $V^t(i,\cdot)$ has size at most $O((|A|^d)^t)$. For infinite horizon problems, we may not have finitely many useful budgets, but the VF remains concave:

Theorem 5. *For any state i, the optimal infinite-horizon VF $V(i,b)$ is concave and monotone in b.*

Standard bounds apply when using the finite-horizon V^t to approximate the infinite-horizon VF: if Bellman error of V^t is ε, $||V^* - V^t|| \leq \frac{\varepsilon}{1-\gamma}$.

Approximation. The complexity of VF computation depends on the number of useful budget points. The VF can be approximated by removing non-dominated points that are "close" to lying strictly inside the convex hull. For instance, in Fig. 2(c), deletion of the second and third points results in a simpler Q-function, with a single segment from $(pb_0 + p'b'_0, pv_0 + p'v'_0)$ to $(pb_2 + p'b'_1, pv_2 + p'v'_1)$ replacing three true segments. It closely approximates the true Q-function since the slopes (BpBs) of all deleted segments are nearly identical.

Several simple pruning criteria can be added to the insertion step of the Graham scan (i.e., when transforming Q-functions to VFs). When inserting (b_{new}, v_{new}) with BpB β_{new}, we can delete the previously inserted point, (b_k, v_k) with BpB β_k, if $\beta_{new} \geq \beta_k - \varepsilon$, for some tolerance ε. This introduces a max-norm error of at most $\varepsilon(b_k - b_{k-1})$. Recursively applying this rule gives additive accumulation: s consecutive pruning steps gives error at most $s\varepsilon(b_k - b_{k-s})$. Since error also depends on the length of the segments being pruned, we can use both *slope* pruning and *length* pruning, or pruning dictated by their product, i.e., terminated when this product bound reaches some threshold τ. Standard MDP approximation bounds can be derived.

Policy Execution. Given the optimal VF V^*, we can readily determine the ideal level of (expected) spend b_0 given initial state i_0 (see our discussion of sweet spots below). While one could then solve the corresponding CMDP with budget b_0, the BMDP solution, in fact, embeds an optimal

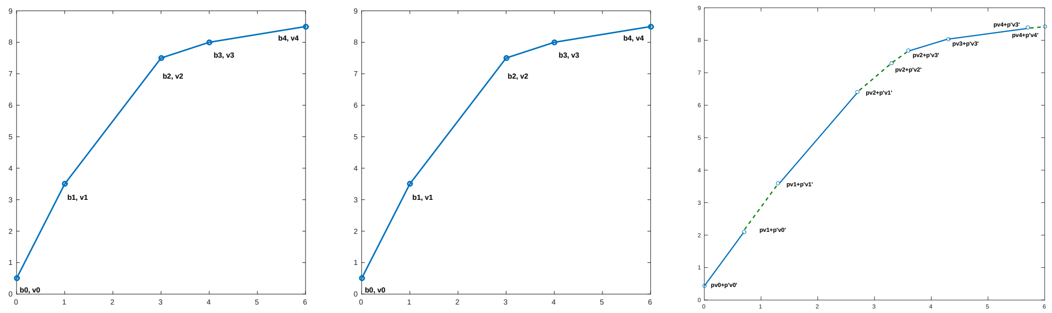
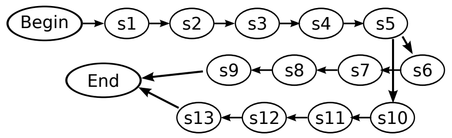


Fig. 2: (a) VF for state j; (b) VF for j'; (c) Q-function (future component) for a reaching j with prob. p and j' with $p' = 1 - p$.



Search: s1: unint; s2: general int; s3: search1, s4: search2, s5:search3
Advertiser: s6: interest1; s7: interest2; s8: interest3, s9: conversion
Compt'r: s10: interest1; s11: interest2; s12: interest3, s13: conversion

Fig. 3: A synthetic ad MDP. "Begin" reflects the beginning of a customer search (transitions from it encode a prior over interest states). From an advertiser's perspective, the MDP for a specific customer can begin at any state of the process.

(non-stationary) CMDP solution for any budget level. The *execution* of the BMDP policy π^* is somewhat subtle, but straightforward. The optimization in Eqns. 1 and 2 generally assigns some future states a *greater* budget than what remains, $b_0 - c^a_{i_0}$, and some future states less—it is only *expected spend* that must not exceed the remaining budget. If state j's "assigned budget" b_j differs from $b_0 - c^a_{i_0}$, whether greater or less, then we must execute action $\pi^*(j, b_j)$ to ensure we achieve expected value $V^*(i_0, b_0)$. This requires that we record the optimal budget mapping σ at the final Bellman backup (assuming an infinite horizon problem) for each useful budget point at each state. When we reach any next state j, we "pass forward" its assigned budget $b_\sigma(j)$, at which point we take action $\pi^*(j, b_\sigma(j))$.

As in CMDPs, there can be substantial variance in actual spend when implementing the BMDP policy $\pi^*(i, b)$ (variance can be computed exactly during DP). We discuss strategies for mitigating the impact of variance below. A simple DP algorithm can be used to compute VFs with *strict* budgets [23], but allowing some variance in spend can improve expected value significantly.

3.4 EMPIRICAL EVALUATION

We evaluate the effectiveness of our DP method on several BMDPs, and measure the impact of approximation.

Synthetic Ad MDP. We begin with a small synthetic MDP that reflects the influence of ads on product search behavior. Its small size allows detailed examination of its optimal VF structure. The MDP (see Fig. 3) has 15 states reflecting various levels of customer interest in an advertiser's (or competitors') product, and five actions for different levels of ad intensity. Transitions for "nominal" (stochastic) progress through the search funnel are shown, with others omitted for clarity. More intense ad actions are more costly, but increase the odds of progressing in the funnel, and lower the odds of abandoning purchase intent.

We solve the BMDP ($\gamma = 0.975$, horizon 50) with four different degrees of approximation: exact (no pruning); mild pruning (slope/length pruning set to 0.01); aggressive pruning (both set to 0.05); and hybrid (mild pruning for 45 iterations, then exact computation for five). The following table shows the average (and min–max) number of segments in the PWLC VF over the 15 MDP states—the number of segments determines the complexity of the VFs—and computation time for each regime. For approximation schemes, we show the maximum absolute and relative errors (and the optimal value at which this error occurs).

	No prun.	Mild	Aggr.	Mild+No
Segments	3066 (0–5075)	18.3 (0–47)	10.4 (0–26)	480.8 (0–877)
Max Err	—	4.84 (26.61)	4.84 (26.61)	0.21 (58.77)
Max RelErr	—	40.9% (4.24)	48.7% (1.54)	2.3% (0.55)
CPU (s.)	1055.4	17.54	10.36	28.67

The optimal VF has a large number of segments per state but can be approximated quite well with very few segments. With mild/aggressive pruning, the VF is very compact, but has large maximum error (4.84, which is 18% rel. error at the point at which it occurs); the relative error is also significant, though it occurs at points with low value (hence gives small abs. error). The hybrid scheme works very well—by exploiting the contraction properties of the MDP, error associated with initial pruning is almost entirely overcome. It reduces the number of VF segments by an order of magnitude, and computation time by nearly two orders of magnitude, but gives very small max absolute (0.21, or 0.36%) and relative error (2.3%, or 0.0013).[2] Determining suitable pruning thresholds and schedules in general is an interesting open question.

Advertiser MDP. We next study two larger MDPs derived from the contact data of a large advertiser. The data consists of sequences of cross-channel touch points with users—each touch is labeled with a contact event (e.g., display ad, email, paid search ad, direct navigation to web site) or type of activity on the advertiser's web site (including transactions or "conversions"). There are 28 event types, and 3.6M trajectories comprising 10M events.

From this data, we learn (using MLE) several *variable-order Markov chain (VOMC) models* [9, 11] that predict

[2]CPU time using a simple Python prototype, on a 3.5GHz CPU with 32Gb of RAM.

transitions induced by the advertiser's policy. Predictions are based on a small sufficient history involving up to 10 preceding events. The histories comprising each VOMC form the state space of a Markov chain. From these we derive action-conditional transition models by substituting history end points with one of a small number of "controllable" events and using the VOMC model for these altered histories for transition predictions. We consider four actions (no-op, email, paid search, display ad), and two different models: the first is a VOMC model with a maximum state-order of 10 and an average state-order of five, giving an MDP with 451,582 states. The second is a two-component mixture of smaller VOMC models (maximum order of 3); we use the first component, with 1469 states.

We solve the BMDPs for both models (horizon 50, discount 0.975 as above). The table below shows pruning level, number of segments, and computation time for the 1469-state MDP using the same strategies as above.[3]

	No prun.	Mild	Aggr.	Mild+No
Segments	251.5 (74–359)	234.2 (77–342)	25.6 (5–39)	76.8 (18–321)
Max Err	—	5.13 (171.6)	28.9 (169.3)	3.9 (167.6)
Max RelErr	—	3.0% (171.6)	12.3% (169.3)	2.4% (167.6)
CPU (s.)	19918.9	10672.5	1451.8	2390.0

The 452K-state BMDP was solved only with aggressive pruning (slope 2.0, length 0.1), and averaged about 11.67 segments per state and 1168s. per DP iteration.

4 BUDGET ALLOCATED ACROSS MDPS

We now consider the problem facing an advertiser with a fixed maximum budget and a specific set of target customers, each of whom occupies the state of some underlying MDP. To solve the joint MDP above, we use the *weakly coupled* decomposition of [23], but merge the local solutions (and analyze this merge) in a different manner.

Offline Decomposition Our approach to decomposition is straightforward. We first solve each *single-user sub-MDP* as a BMDP with a some natural per-user maximum budget B_u. For ease of exposition, we assume just one user type, hence only one user MDP to solve (i.e., each user has the same dynamics). Users are distinguished only by their MDP state. The BMDP solution gives an optimal single-user policy π and VF V spanning all $s \in S, b \leq B_u$, indicating action choice and value *as a function of the budget available to be spent (in expectation) on that user alone*. We exploit this below. Indeed, this is why we do uses CMDPs, which do not indicate the value of allocating *varying* budgets to a user.

The Budget Allocation Problem Given initial (or current) joint state with M customers is $\mathbf{s} = \langle s[1], \cdots s[M] \rangle$, and budget B, a natural way to exploit the BMDP solution is to allocate some portion $b[i]$ of B to each customer $i \leq M$ s.t. we maximize the sum of expected values $v[i]$ assuming optimal engagement with i with budget $b[i]$. Specifically, the *budget allocation problem (BAP)* is :

$$\max_{b[i]:i\leq M} \sum_{i\leq M} V(s[i], b[i]) \;\text{ subj. to }\; \sum_{i\leq M} b[i] \leq B, \qquad (3)$$

where V is the optimal VF for the underlying BMDP.

BAP determines an allocation $\mathbf{b}^* = \langle b^*[1], \ldots, b^*[M] \rangle$ that maximizes the expected value of *committing* a specific (expected) budget $b^*[i]$ to customer i. By simple linearity of expectation, we have:

Observation 6. *Let V be the optimal VF and π the optimal policy for the user MDP. Let $\mathbf{b}$ be the optimal solution to the budget allocation problem. Then the joint (non-stationary) policy $\overline{\pi}(\mathbf{s}) = \prod_i \pi(s[i], b[i])$ has expected value $\sum_i V(s[i], b[i])$ and expected spend $\sum_{i\leq M} b[i] \leq B$.*

We cannot guarantee this policy is truly optimal for the joint MDP, since this decomposed policy does not admit *recourse* in the execution of one user's MDP that depends on the realized outcome of some other user's MDP. However, we expect it to work well in practice (see below). For MDPs with large numbers of customers, or where the spend variance of the local BMDP policy is low, this form of suboptimality will be small. However, as we discuss below, *repeated online reallocation of budget* can sometimes overcome even this potential suboptimality in practice.

BAP (Eq. 3) can be viewed as a (multi-item variant of a) *multiple-choice knapsack problem (MCKP)* [29]. In the classic MCKP, we are given M classes of items, with each class i containing n_i items. To explain, we first begin with a restricted version of BAP, the *useful-budget assignment problem (UBAP)*. In UBAP, we require each user i be assigned a *useful* budget level from the discrete set $B^{s[i]}$. UBAP is exactly an MCKP: each user i is as an *item class* for whom exactly one budget must be chosen from the set of *items* $B^{s[i]}$ in that class. The *weight* of item $b \in B^{s[i]}$ is b (i.e., the amount of the budget it consumes); and the *profit* of assigning b to i is $V(s[i], b)$. The *capacity* is the global budget B, so total weight (budget assigned) cannot exceed B. We can view this as a *multi-item* variant of MCKP with multiple "copies" of the same class (namely, all users in a state j have the same items, weights and profits).

It is useful to consider an integer programming (IP) model of MCKP (where binary variable x_{ik} indicates that i is allocated the kth useful budget $\beta_{ik} = b_k^{s[i]}$):

$$\max_{x_{ik}} \quad \sum_{i\leq M} \sum_{k\in B^{s[i]}} V(s[i], \beta_{ik}) x_{ik} \qquad (4)$$

$$\text{subject to} \quad \sum_{i\leq M} \sum_{k\in B^{s[i]}} \beta_{ik} x_{ik} \leq B \qquad (5)$$

$$\sum_{k\in B^{s[i]}} x_{ik} = 1, \quad \forall i \leq M \qquad (6)$$

$$x_{ik} \in \{0, 1\} \qquad (7)$$

[3]The "no pruning" optimal benchmark uses slope/length pruning of 0.001/0.0001 for 20 iterations and 0.01/0.001 for 30. Aggressive is slope/length = 0.2/0.01; mild is 0.02/0.001.

We can collapse users in the same class using a counting (integer) variable as well.

Consider the LP relaxation of the IP Eq. 4—its optimal solution gives a (generally) fractional assignment of useful budget to any user i in state $s[i] = j$. Using a minor adaptation of the analysis of Sinha and Zoltners [29], we can show that the optimal allocation to any i is either integral (i.e., assigns a useful budget level to i), or is a mixture of two *consecutive* useful levels, in which case the expected budget $b^*[i]$ and induced expected value $v^*[i]$ in the LP corresponds to a point on the convex hull of the useful points. In other words, it lies on the PWLC VF $V(i, \cdot)$:

Proposition 7. *The optimal solution of the LP relaxation of UBAP IP Eq. 4 is such that, for each i: (a) $x_{ik} = 1$ for one value of k; or (b) there is some k such that only $x_{ik}, x_{i,k+1} > 0$ (i.e., only two budget levels are allocated, and they must be consecutive).*

We immediately obtain:

Corollary 8. *The optimal solution to the LP relaxation of UBAP is an optimal solution to BAP.*

The structure of the LP relaxation of MCKP is very valuable. Sinha and Zoltners [29] show that a simple greedy algorithm can be used to solve the relaxation. We adopt the same method, *greedy budget allocation (GBA)*, to solve BAP. For each state j, and each $1 \leq k \leq L$, define the *bang-for-buck ratio* as before:

$$BpB_{jk} = \frac{V(j, \beta_{jk}) - V(j, \beta_{jk-1})}{\beta_{jk} - \beta_{jk-1}}.$$

GBA assigns budget incrementally to each user in the order given by the BpB ratio. Initially each user i in state $s[i] = j$ is assigned a budget of $\beta_{j0} = 0$, and the unallocated budget is set to B (our budget constraint). At any stage of GBA, let b denote the unallocated budget, let β_{j,k_i} be i's current allocation and i's current ratio to be $BpB[i] = BpB_{jk_i+1}$. Let i be any *best user*, with maximum ratio $BpB[i]$. If sufficient budget remains, we increase i's allocation from β_{j,k_i} to β_{j,k_i+1}, then update i's ratio and the unallocated budget. We continue until the unallocated budget is less than $\beta_{j,k_i+1} - \beta_{j,k_i}$; then we allocate the remaining budget fractionally to the best i (β_{j,k_i} with probability p and β_{j,k_i+1} with $1-p$, for $p = \frac{(\beta_{j,k_i+1} - \beta_{j,k_i}) - b}{(\beta_{j,k_i+1} - \beta_{j,k_i})}$).

We can show that GBA finds the optimal solution to our BAP (see [29]). GBA can be modified to aggregate all users that lie in the same state, and to account for the stochastic arrival of customers of various types, or at various states.

Dynamic Budget Reallocation The variance in the spend of an optimal policy $\pi(i, b)$ means there is some risk over overspending the global budget. This risk is, of course, greater with small numbers of users than with large numbers. One way to alleviate this risk *dynamic budget reallocation (DBRA)*. Rather than committing to the optimal policy for each user given their initial allocation, we reallocate any remaining budget at each stage. More precisely, given the current joint state $\mathbf{s} = \langle s[1], \ldots, s[M] \rangle$ and remaining budget B, we: (a) use GBA to determine allocation $b[i], i \leq M$ given $(\mathbf{s}, B)$; (b) execute action $\pi(s[i], b[i])$ for each i, incurring the cost c_i^a; and (c) observe the next state $\mathbf{s}'$ and remaining B' and repeat. This approach (virtually) guarantees that the global budget B will not be overspent (if the BMDP policy randomizes, a small chance of a small violation may exist). It also offers a form of recourse; e.g., if the budget for some user is no longer useful (e.g., transition to an unprofitable state), it budget can be reallocated to a more profitable user.

Empirical Evaluation We test the effectiveness of GBA on the BMDPs described in Sec. 3.4. We study expected spend and value of the "implied" joint policies, as well as spend variance.

We consider several ways of implementing the joint policies induced by the GBA solution of BAP. The first is the *BMDP policy*, where once GBA allocates $b[i]$ to each user i, we implement the corresponding BMDP policy starting at state $s[i]$. This ensures that *expected spend* does not exceed $b[i]$, but doesn't guarantee budget satisfaction for any specific user. The second is the *static user budget policy (SUB)*: given the GBA allocation $b[i]$, we implement the first action $a = \pi(s[i], b[i])$ in the BMDP policy at $s[i]$ for each i; but when reaching next state, we take the action as if we only had that user's *remaining* budget $b[i] - c_{s[i]}^a$, ignoring the next state budget mapping from the BMDP policy. SUB thus recalibrates the actual spend to minimize the odds of overspending $b[i]$ on a per-user basis. We also use the reallocation scheme DBRA, which reduces the overspending risk collectively (not per-user).

Solving BMDPs and using GBA to allocate budget allows one to assess budget tradeoffs across different users in different states. Without a BMDP model or our budget-dependent VF, these tradeoffs must be made heuristically. In this case, we can use *uniform budget allocation (UBA)*, which apportions budget equally across all users, and then solves the induced CMDPs (one per occupied state).

Synthetic Ad MDP. In the synthetic MDP, our initial setup has 1000 customers starting in s_0. The following table shows the expected value obtained by 3 different policies for 4 different global budgets:

Total Bud.	BMDP Val.	DBRA Val.	SUB Val.
1000	8210	8579 (830.5)	4106 (707)
2000	10,905	11,019 (964)	4429 (825)
5000	15,692	15,658 (1239)	5270 (830.5)
10,000	18,110	17,942 (—)	6329 (1159)

BMDP value is the true expected value of the optimal BMDP policy. SUB and DBRA values are averaged over 100 trials, which execute the relevant policies for all 1000 users (sample std. dev. also shown). The optimal BMDP policy has a considerable advantage over a static policy that forbids the per-user budget to be exceeded, yielding 2-3 times the return.[4] BMDP values also show a clear pat-

[4]Values are discounted, net of budget spent.

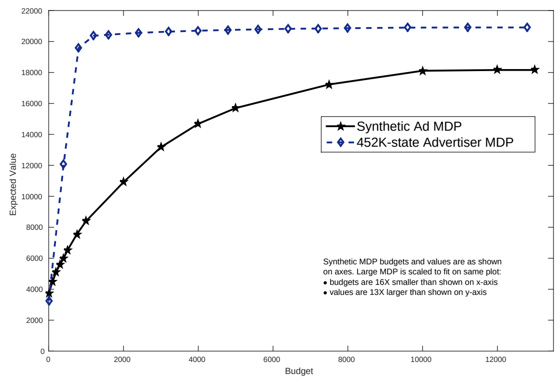


Fig. 4: Sweet spot curves: a plot of expected value vs. global budget for Synthetic and 452K-state MDPs (1000 users). The flat tails show that the maximal useful budget has been reached.

tern of diminishing return with increasing budget. Indeed, our MCKP-based GBA method allows one to *rapidly* assess the value of *optimally* using different global budgets to find the "sweet spot" in spend. Fig. 4 illustrates this *sweet spot curve*) for the joint MDP over 16 budget points.[5] The sweet spot curve is effectively the *Pareto frontier* expressing the tradeoff between two competing objectives, spend and expected return (for an in-depth survey of general multi-objective MDPs, please see [26]).

The optimal BMDP policy has considerable spend variance. In one run of 1000 customers (initial allocation 10 each), the sample average 10.012 is close to expected spend; but the sample std. dev. is 15.24. The empirical odds that a user exceeds the budget of 10 is 32.7%, and the odds of exceeding it by at least 50% is 28.7%. This alone explains the poor performance of the SUB allocation policy. Even with a large user base, overspending is possible: simulating the BMDP policy (1000 users) for 30 trials users, the global budget is exceeded in 13 of 30 trials, in 5 instances by over 3% (the largest overspend is by 11.7%). DBRA alleviates this risk—in all 100 trials the budget constraint is satisfied—while its average return matches or exceeds that of BMDP. With the very constrained budget (1000), DBRA also appears to offer the advantage of reallocating budget to more promising customers over time.

We also compare GBA to UBA on the same MDP, but with 1000 customers uniformly spread among the 12 nonterminal states (GBA and UBA are identical if all users start in the same state). The table below shows (exact) expected value of GBA and UBA for several global budgets.

Total Bud.	GBA Val.	UBA Val.
1000	39818.6	36997.2
2000	44559.5	40311.8
5000	53177.7	47142.4
10,000	58356.8	53773.8

The optimal BMDP solution allows GBA to make budget tradeoffs among customers in different states, giving greater value than a uniform scheme.

[5]The greedy algorithm averages 1.47ms. to compute the optimal allocation at each budget point.

Advertiser MDPs. We apply the same four methods to the advertiser-based MDPs. We first use GBA to derive "sweet spot" curves for the large MDP (results are similar for the 1469-state MDP). We assume 1000 customers, with 20 customers each entering the process in the 50 states with the largest "value spans" (i.e., difference in expected value given the minimal and maximal useful budget). Fig. 4 shows the budget-value tradeoff.

These MDPs model behavior that is quite random (i.e., not influenced very strongly by the actions). As a consequence, once the GBA algorithm is run, there is not a great difference between the BMDP and SUB policies. The table below shows results for the 452K-state BMDP for two fairly constrained budget levels (DBRA, SUB are averaged over 50 trials, BMDP and UBA values are exact).

Budg.	BMDP Val.	DBRA Val.	SUB Val.	UBA Val.
15	113358	99236 (3060)	112879 (1451)	106373
25	157228	142047 (3060)	157442 (2589)	149175

Neither BMDP nor SUB exhibit much variance in spend and both have similar expected values. SUB rarely overspends (e.g., maximum overspend for SUB with $B = 25$ is 0.16%). Variance tends to be greater when budgets are tighter. Among the 50 BMDP trials, 14 instances exceed the global budget, though only four instances exceed it by more than 4.0% (and one does so by 8.6%). DBRA eliminates the risk of overspending, but in this problem has a negative impact on expected value. GBA offers greater expected value than UBA (which is the only viable option if the BMDP has not been solved). GBA exceeds UBA by up to 6.5% over a range of constrained budgets.

5 CONCLUDING REMARKS

We have addressed the problem of budget (or other resource) allocation in MDPs so that budget-value tradeoffs can be addressed effectively. Our *budgeted MDP* model offers an alternative view of CMDPs that allows value to be derived as a function of available budget. We characterized the structure of optimal VFs and developed a DP algorithm that exploits the PWLC form of these VFs. Our second contribution was a method for exploiting BMDP solutions to allocate budget across independently operating BMDPs. We cast the problem as multi-item MCKP for which a simple greedy algorithm rapidly allocates budget optimally in a "committed" fashion. We also investigated dynamic reallocation of budget over time.

The extension of our methods to account for dynamic user populations is straightforward, but warrants empirical investigation. Future work includes the further study of and experimentation with our algorithms on richer models of user behavior. We are also interested in extending our models to partially observable settings (e.g., where user type estimation based on behavioral observations is needed).

References

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

[2] E. Altman. *Constrained Markov Decision Processes*. Chapman and Hall, London, 1999.

[3] K. Amin, M. Kearns, P. Key, and A. Schwaighofer. Budget optimization for sponsored search: Censored learning in MDPs. In *UAI-12*, pp.543–553, 2012.

[4] N. Archak, V. Mirrokni, and S. Muthukrishnan. Budget optimization for online campaigns with positive carryover effects. In *WINE-12*, pp.86–99, 2012.

[5] N. Archak, V. Mirrokni, and S. Muthukrishnan. Mining advertiser-specific user behavior using adfactors. In *WWW 2010*, pp.31–40, 2010.

[6] C. Borgs, J. Chayes, O. Etesami, N. Immorlica, K. Jain, and M. Mahdian. Bid optimization in online advertisement auctions. In *Workshop on Sponsored Search Auctions*, 2006.

[7] C. Boutilier, T. Dean, and S. Hanks. Decision theoretic planning: Structural assumptions and computational leverage. *J. Artificial Intel. Res.*, 11:1–94, 1999.

[8] B. Bronnenberg. Advertising frequency decisions in a discrete Markov process under a budget constraint. *J. Marketing Res.*, 35(3):399–406, 1998.

[9] P. Bühlmann and A. Wyner. Variable-length Markov chains. *Annals of Stat.*, 27(2):480–513, 1999.

[10] M. Charikar, R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. On targeting Markov segments. In *STOC-99*, pp.99–108, 1999.

[11] F. Chierichetti, R. Kumar, P. Raghavan, and T. Sarlos. Are web users really Markovian? In *WWW-12*, pp.609–618, 2012.

[12] B. DasGupta and S. Muthukrishnan. Stochastic budget optimization in internet advertising. *Algorithmica*, 65(3):634–661, 2013.

[13] D. Dolgov and E. Durfee. Stationary deterministic policies for constrained MDPs with multiple rewards, costs, and discount factors. In *IJCAI-05*, pp.1326–1331, 2005.

[14] D. Dolgov and E. Durfee. Optimal resource allocation and policy formulation in loosely-coupled Markov decision processes. In *ICAPS-04*, pp.315–324, 2004.

[15] G. Feichtinger, R. Hartl, and S. Sethi. Dynamic optimal control models in advertising: Recent developments. *Mgmt. Sci.*, 40(2):195–226, 1994.

[16] J. Feldmann, S. Muthukrishnan, M. Pál, and C. Stein. Budget optimization in search-based advertising auctions. In *ACM EC'07*, pp.40–49, 2007.

[17] A. Ghosh, P. McAfee, K. Papineni, and S. Vassilvitskii. Bidding for representative allocations for display advertising. In *WINE-09*, pp.208–219, 2009.

[18] R. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Inf. Proc. Letters*, 1(4):132–133, 1972.

[19] D. Holthausen, Jr. and G. Assmus. Advertising budget allocation under uncertainty. *Mgmt. Sci.*, 28(5):487–499, 1982.

[20] C. Karande, A. Mehta, and R. Srikant. Optimizing budget constrained spend in search advertising. In *WSDM-13*, pp.697–706, 2013.

[21] T. Li, N. Liu, J. Yan, G. Wang, F. Bai, and Z. Chen. A Markov chain model for integrating behavioral targeting into contextual advertising. In *ADKDD-09*, pp.1–9, 2009.

[22] F. Melo and M. Veloso. Decentralized MDPs with sparse interactions. *Artificial Intel.*, 175(11):1757–1789, 2011.

[23] N. Meuleau, M. Hauskrecht, K. Kim, L. Peshkin, L. P. Kaelbling, T. Dean, and C. Boutilier. Solving very large weakly coupled Markov decision processes. In *AAAI-98*, pp.165–172, 1998.

[24] P. Otter. On dynamic selection of households for direct marketing based on Markov chain models with memory. *Marketing Letters*, 18(1):73–84, 1981.

[25] M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.

[26] D. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *J. Artificial Intel. Res.*, 48:67–113, 2013.

[27] D. Silver, L. Newnham, D. Barker, S. Weller, and J. McFall. Concurrent reinforcement learning from customer interactions. In *ICML-13*, pp.924–932, 2013.

[28] S. Singh and D. Cohn. How to dynamically merge Markov decision processes. In *NIPS 10*, pp.1057–1063, 1998.

[29] P. Sinha and A. Zoltners. The multiple-choice knapsack problem. *Op. Res.*, 27(3):503–515, 1979.

[30] M. Spaan and F. Melo. Interaction-driven Markov games for decentralized multiagent planning under uncertainty. In *AAMAS-08*, pp.525–532, 2008.

[31] G. Theocharous, P. Thomas, and M. Ghavamzadeh. Personalized ad recommendation systems for lifetime value optimization with guarantees. In *IJCAI-15*, pp.1806–1812, 2015.