Multi-Objective MDPs with Conditional Lexicographic Reward Preferences

Kyle Hollins Wray[1] **Shlomo Zilberstein**[1] **Abdel-Illah Mouaddib**[2]

[1] School of Computer Science, University of Massachusetts, Amherst, MA, USA
[2] GREYC Laboratory, University of Caen, Basse-Normandie, France

Abstract

Sequential decision problems that involve multiple objectives are prevalent. Consider for example a driver of a semi-autonomous car who may want to optimize competing objectives such as travel time and the effort associated with manual driving. We introduce a rich model called Lexicographic MDP (LMDP) and a corresponding planning algorithm called LVI that generalize previous work by allowing for conditional lexicographic preferences with slack. We analyze the convergence characteristics of LVI and establish its game theoretic properties. The performance of LVI in practice is tested within a realistic benchmark problem in the domain of semi-autonomous driving. Finally, we demonstrate how GPU-based optimization can improve the scalability of LVI and other value iteration algorithms for MDPs.

1 Introduction

Stochastic planning problems designed to optimize multiple objectives are widespread within numerous domains such as management of smart homes and commercial buildings (Kwak et al. 2012), reservoir water control (Castelletti, Pianosi, and Soncini-Sessa 2008), and autonomous robotics (Mouaddib 2004; Calisi et al. 2007). Current approaches often use a scalarization function and a weight vector to project the multi-objective problem to a single-objective problem (Roijers et al. 2013; Natarajan and Tadepalli 2005; Perny and Weng 2010; Perny et al. 2013). While these approaches leverage effectively the vast existing work on single-objective optimization, they have several drawbacks. Choosing a projection is often too onerous to use in practice since there are many viable Pareto optimal solutions to the original multi-objective problem, making it hard to visualize and analyze alternative solutions. Often there is no clear way to prefer one over another. In some cases, a simple lexicographic order exists among the objectives; for example, using plan safety as primary criterion and cost as secondary. But lexicographic order of objectives can be too rigid, not allowing any trade-offs between objectives (e.g., a large cost reduction for taking a minimal risk).

Recent work by Mouaddib (2004) used a strict lexicographic preference ordering for multi-objective MDPs. Others have also developed lexicographic orderings over value functions, calling this technique *ordinal dynamic programming* (Mitten 1974; Sobel 1975). Mitten assumed a specific preference ordering over outcomes for a finite horizon MDP; Sobel extended this model to infinite horizon MDPs. Ordinal dynamic programming has been explored under reinforcement learning (Gábor, Kalmár, and Szepesvári 1998; Natarajan and Tadepalli 2005), with the notion of a minimum criterion value.

We propose a natural extension of sequential decision making with lexicographic order by introducing two added model components: conditioning and slack. Conditioning allows the lexicographic order to depend on certain state variables. Slack allows a small deviation from the optimal value of a primary variable so as to improve secondary value functions. The added flexibility is essential to capture practical preferences in many domains. For example, in manufacturing, there is always a trade-off among cost, quality, and time. In critical states of the manufacturing process, one may prefer to optimize quality with no slack, whereas in less important states one may optimize cost, but allow for some slack in order to improve time and quality.

Our model is motivated by work on planning for semi-autonomous systems (Zilberstein 2015). Consider a car that can operate autonomously under certain conditions, for example, maintaining safe speed and distance from other vehicles on a highway. All other roads require manual driving. A driver may want to minimize both the time needed to reach the destination and the effort associated with manual driving. The concepts of conditional preference and slack are quite useful in defining the overall objective. To ensure safety, if the driver is tired, then roads that are autonomous-capable are preferred without any margin of slack; however, if the driver is not tired, then roads that optimize travel time are preferred, perhaps with some slack to increase the inclusion of autonomous-capable segments. We focus on this sample domain for the remainder of the paper.

The general use of preference decomposition is popular, as found in Generalized Additive Decomposable (GAI) networks (Gonzales, Perny, and Dubus 2011) or Conditional Preference Networks (CP-Nets) (Boutilier et al. 2004). Constrained MDPs (CMDPs) can also capture complex preference structures, as well as slack, and are potentially a more general representation than LMDPs (Altman 1999). Various other forms of slack are also commonly found in the liter-

ature (Gábor, Kalmár, and Szepesvári 1998). However, as we show, LMDPs offer a good trade-off between expressive power and computational complexity, allowing a new range of objectives to be expressed without requiring substantially more complex solution methods.

Our primary contributions include formulating the Lexicographic MDP (LMDP) model and the corresponding Lexicographic Value Iteration (LVI) algorithm. They generalize the previous methods mentioned above with our formulation of slack variables and conditional state-based preferences. Additionally, we show that LMDPs offer a distinct optimization framework from what can be achieved using a linearly weighted scalarization function. Furthermore, we introduce a new benchmark problem involving semi-autonomous driving together with general tools to experiment in this domain. Finally, we develop a Graphic Processing Unit (GPU) implementation of our algorithm and show that GPU-based optimization can greatly improve the scalability of Value Iteration (VI) in MDPs.

Section 2 states the LMDP problem definition. Section 3 presents our main convergence results, bound on slack, and an interesting relation to game theory. Section 4 presents our experimental results. Finally, Section 5 concludes with a discussion of LMDPs and LVI.

2 Problem Definition

A Multi-Objective Markov Decision Process (MOMDP) is a sequential decision process in which an agent controls a domain with a finite set of states. The actions the agent can perform in each state cause a stochastic transition to a successor state. This transition results in a reward, which consists of a vector of values, each of which depends on the state transition and action. The process unfolds over a finite or infinite number of discrete time steps. In a standard MDP, there is a single reward function and the goal is to maximize the expected cumulative discounted reward over the sequence of stages. MOMDPs present a more general model with multiple reward functions. We define below a variant of MOMDPs that we call Lexicographic MDP (LMDP), which extends MOMDPs with lexicographic preferences to also include conditional preferences and slack. We then introduce a Lexicographic Value Iteration (LVI) algorithm (Algorithm 1) which solves LMDPs.

Definition 1. A **Lexicographic Markov Decision Process (LMDP)** is a represented by a 7-tuple $\langle S, A, T, \mathbf{R}, \delta, \mathcal{S}, o \rangle$:

- S is a finite set of n states, with initial state $s_0 \in S$
- A is a finite set of m actions
- $T : S \times A \times S \to [0, 1]$ is a state transition function which specifies the probability of transitioning from a state $s \in S$ to state $s' \in S$, given action $a \in A$ was performed; equivalently, $T(s, a, s') = Pr(s'|s, a)$
- $\mathbf{R} = [R_1, \ldots, R_k]^T$ is a vector of reward functions such that $\forall i \in K = \{1, \ldots, k\}$, $R_i : S \times A \times S \to \mathbb{R}$; each specifies the reward for being in a state $s \in S$, performing action $a \in A$, and transitioning to a state $s' \in S$, often written as $\mathbf{R}(s, a, s') = [R_1(s, a, s'), \ldots, R_k(s, a, s')]$
- $\delta = \langle \delta_1, \ldots, \delta_k \rangle$ is a tuple of slack variables such that $\forall i \in K, \delta_i \geqslant 0$

Algorithm 1 Lexicographic Value Iteration (LVI)

```
V ← 0
V' ← 0
while ‖V − V'‖_∞^S > ε(1−γ)/γ do
    V' ← V
    V^fixed ← V
    for j = 1, ..., ℓ do
        for i = o_j(1), ..., o_j(k) do
            while ‖V_i' − V_i‖_∞^{S_j} > ε(1−γ)/γ do
                V_i'(s) ← V_i(s), ∀s ∈ S_j
                V_i(s) ← B_i V_i'(s), ∀s ∈ S_j
            end while
        end for
    end for
end while
return V'
```

- $\mathcal{S} = \{S_1, \ldots, S_\ell\}$ is a set which forms an ℓ-partition over the state space S
- $o = \langle o_1, \ldots, o_\ell \rangle$ is a tuple of strict preference orderings such that $L = \{1, \ldots, \ell\}$, $\forall j \in L$, o_j is a k-tuple ordering elements of K

In the interest of clarity, we limit this paper to *infinite horizon* LMDPs (i.e., $h = \infty$), with a *discount factor* $\gamma \in [0, 1)$. The finite horizon case follows in the natural way. A *policy* $\pi : S \to A$ maps each state $s \in S$ to an action $a \in A$.

Let $\mathbf{V} = [V_1, \ldots, V_k]^T$ be a set of *value functions*. Let each function $V_i^\pi : S \to \mathbb{R}$, $\forall i \in K$, represent the value of states S following policy π. The stochastic process of MDPs enable us to represent this using the expected value over the reward for following the policy at each stage.

$$\mathbf{V}^\pi(s) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \mathbf{R}(s^t, \pi(s^t), s^{t+1}) \Big| s^0 = s, \pi \Big]$$

This allows us to recursively write the value of the state $s \in S$, given a particular policy π, in the following manner.

$$\mathbf{V}^\pi(s) = \sum_{s' \in S} T(s, \pi(s), s')(\mathbf{R}(s, \pi(s), s') + \gamma \mathbf{V}^\pi(s'))$$

Lexicographic Value Iteration

LMDPs lexicographically maximize $V_{o_j(i)}(s)$ over $V_{o_j(i+1)}(s)$, for all $i \in \{1, \ldots, k-1\}$, $j \in L$, and $s \in S$, using $V_{oj(i+1)}$ to break ties. The model allows for slack $\delta_{o_j(i)} \geqslant 0$ (deviation from the overall optimal value). We will also refer to $\eta_{o_j(i)} \geqslant 0$ as the deviation from optimal for a single action change. We show that the classical value iteration algorithm (Bellman 1957) can be easily modified to solve MOMDPs with this preference characterization.

For the sake of readability, we use the following convention: Always assume that the ordering is applied, e.g., $V_{i+1} \equiv V_{o_j(i+1)}$ and $\{1, \ldots, i-1\} \equiv \{o_j(1), \ldots, o_j(i-1)\}$. This allows us to omit the explicit ordering $o_j(\cdot)$ for subscripts, sets, etc.

First, Equation 1 defines $Q_i(s, a)$, the value of taking an action $a \in A$ in a state $s \in S$ according to objective $i \in K$.

$$Q_i(s, a) = \sum_{s' \in S} T(s, a, s')(R_i(s, a, s') + \gamma V_i(s')) \quad (1)$$

With this definition in place, we may define a restricted set of actions for each state $s \in S$. For $i = 1$, let $A_1(s) = A$ and for all $i \in \{1, \ldots, k-1\}$ let $A_{i+1}(s)$ be defined as:

$$A_{i+1}(s) = \{a \in A_i(s) | \max_{a' \in A_i(s)} Q_i(s, a') - Q_i(s, a) \leqslant \eta_i\} \tag{2}$$

For reasons explained in Section 3, we let $\eta_i = (1-\gamma)\delta_i$. Finally, let Equation 3 below be the *Bellman update equation* for MOMDPs with lexicographic reward preferences for $i \in K$, using slack $\delta_i \geqslant 0$, for all states $s \in S$. If $i > 1$, then we require V_{i-1} to have converged.

$$V_i(s) = \max_{a \in A_i(s)} Q_i(s, a) \tag{3}$$

Within the algorithm, we leverage a modified value iteration with slack Bellman update equation (from Equation 3) denoted as B_i. We either use V_i for $s \in S_j \subseteq S$ or $V_i^{fixed}(s)$ for $s \in S \backslash S_j$, as shown in Equation 4 below, with $[\cdot]$ denoting Iverson brackets.

$$B_i V_i'(s) = \max_{a \in A_i(s)} \sum_{s' \in S} T(s, a, s')(R_i(s, a, s') + \gamma \bar{V}_i(s')) \tag{4}$$

$$\bar{V}_i(s') = V_i'(s')[s \in S_j] + V_i^{fixed}(s')[s \notin S_j] \tag{5}$$

3 Theoretical Analysis

This section provides a theoretical analysis of LVI in three parts. First, we provide a strong lower bound on slack to complete the definition of LVI. Second, we prove convergence given an important assumption, and explain why the assumption is needed. Third, we show two properties of LMDPs: a game theory connection and a key limitation of scalarization methods in comparison to LMDPs.

Strong Bound on Slack

First we show in Proposition 1 that η_i from Equation 2 may be defined as $(1-\gamma)\delta_i$ to bound the final deviation from the optimal value of a state by δ_i, for $i \in K$. This is designed to be a worst-case guarantee that considers each state selects an action as far from optimal as it can, given the slack allocated to it. The accumulation of error over all states is bounded by δ. In practice, this strong bound can be relaxed, if desired.

Proposition 1. For all $j \in L$, for $i \in K$, assume $1, \ldots, i-1$ has converged. Let V^η be the value functions returned following Equation 4; Lines 7-10. Let V^π be the value functions returned by value iteration, following the resulting optimal policy π, starting at V^η. If $\eta_i = (1-\gamma)\delta_i$ then $\forall s \in S_j$, $V_i^\eta(s) - V_i^\pi(s) \leqslant \delta_i$.

Proof. For any $i \in K$, the full (infinite) expansion of value iteration for V_i^η is as follows ($t \to \infty$).

$$\begin{aligned} V_i^\pi(s) = \sum_{s^t \in S} T(s, \pi(s), s^t)\Big(R_i(s, \pi(s), s^t) + \gamma\Big(\cdots \\ + \gamma\Big(\sum_{s^1 \in S} T(s^2, \pi(s^2), s^1)\Big(R_i(s^2, \pi(s^2), s^1) \\ + \gamma\Big(V_i^0(s^1)\Big)\Big)\Big)\cdots\Big)\Big) \end{aligned}$$

Since value iteration admits exactly one unique fixed point, any initial value of V_i^0 is allowed; we let $V_i^0 = V_i^\eta$. From this, $Q_i^\eta(s, \pi(s))$ (Equation 1) exists within the above equation for $V_i^\pi(s)$. By Equation 2, $V_i^\eta(s) - Q_i^\eta(s, \pi(s)) \leqslant \eta_i$ since $\pi(s) \in A_k(s) \subseteq \cdots \subseteq A_{i+1}(s)$. Equivalently, $Q_i^\eta(s, \pi(s)) \geqslant V_i^\eta(s) - \eta_i$. Combine all of these facts and bound it from below. The η_i falls out of the inner equation. Also, recall that $\sum_{s^2 \in S} T(s^3, \pi(s^3), s^2) = 1$ and $\gamma\eta_i$ is a constant. This produces:

$$\begin{aligned} \geqslant \sum_{s^t \in S} T(s, \pi(s), s^t)\Big(R_i(s, \pi(s), s^t) + \gamma\Big(\cdots \\ + \gamma\Big(\sum_{s^2 \in S} T(s^3, \pi(s^3), s^2)\Big(R_i(s^3, \pi(s^3), s^2) \\ + \gamma\Big(V_i^\eta(s^2)\Big)\Big) - \gamma\eta_i\Big)\cdots\Big)\Big) \end{aligned}$$

We again recognize $Q_i^\eta(s^3, \pi(s^3))$ and place a lower bound on the next one with $V_i^\eta(s^3) - \eta_i$. This process repeats, each time introducing a new η_i, with one less γ multiplied in front of it, until we reach the final equation. We obtain the following inequality, and note that if $\eta_i \geqslant 0$, then we may subtract another η_i in order to obtain a geometric series (i.e., the sum may begin at $t = 0$).

$$V_i^\pi(s) \geqslant V_i^\eta(s) - \sum_{t=0}^{\infty} \gamma^t \eta_i \geqslant V_i^\eta(s) - \frac{\eta_i}{1-\gamma}$$

$$V_i^\eta(s) - V_i^\pi(s) \leqslant \frac{\eta_i}{1-\gamma}$$

Therefore, let $\eta_i = (1-\gamma)\delta_i$. This guarantees that error *for all* states $s \in S$ is bounded by δ_i. □

Convergence Properties

In order to prove the convergence of Algorithm 1, we first prove that the value iteration component over a partition with slack is a contraction map. The proof follows from value iteration (Bellman 1957) and from the suggested proof by Russel and Norvig (2010). We include it because of its required modifications and it explains exactly why Assumption 1 is needed. Finally, we include the domain in *max norms*, i.e., $\|\cdot\|_\infty^Z = \max_{z \in Z} |\cdot|$.

Proposition 2. For all $j \in L$, for $i \in K$, assume $1, \ldots, i-1$ has converged, with discount factor $\gamma \in [0, 1)$. B_i (Equation 4) is a contraction map in the space of value functions over $s \in S_j$, i.e., $\|B_i V_1 - B_i V_2\|_\infty^{S_j} \leqslant \gamma \|V_1 - V_2\|_\infty^{S_j}$.

Proof. Let the space $Y_i = \mathbb{R}^z$ be the *space of value functions for* i for $z = |S_j|$, i.e., we have $V_i = [V_i(s_{j1}), \ldots, V_i(s_{jz})]^T \in Y_i$. Let the distance metric d_i be the *max norm*, i.e., $\|V_i\|_\infty = \max_{s \in S_j} |V_i(s)|$. Since $\gamma \in [0, 1)$, the metric space $M_i = \langle Y_i, d_i \rangle$ is a *complete normed metric space* (i.e., *Banach space*).

Let the lexicographic Bellman optimality equation for i (Equation 4) be defined as an operator B_i. We must show that the operator B_i is a contraction map in M_i for all $i \in K$, given either that $i = 1$ or that the previous $i - 1$ has converged to within ϵ of its fixed point.

Let $V_1, V_2 \in Y_i$ be any two value function vectors. Apply Equation 4. For $s \in S_j$, if $i = 1$ then $A_i(s) = A$; otherwise, $A_i(s)$ is defined using $A_{i-1}(s)$ (Equation 2) which by construction has converged.

$$\|B_iV_1 - B_iV_2\|_\infty^{S_j} = \max_{s \in S_j} |\max_{a \in A_i(s)} Q_1(s,a) - \max_{a \in A_i(s)} Q_2(s,a)|$$

As part of the $Q(\cdot)$ values, we distribute $T(\cdot)$ to each $R(\cdot)$ and $V(\cdot)$ in the summations, then apply the property: $\max_x f(x) + g(x) \leqslant \max_x f(x) + \max_x g(x)$, twice. We may then pull out γ and recall that for any two functions f and g, $|\max_x f(x) - \max_x g(x)| \leqslant \max_x |f(x) - g(x)|$.

$$\leqslant \gamma \max_{s \in S_j} \max_{a \in A_i(s)} \sum_{s' \in S} T(s,a,s') \Big| \bar{V}_1(s') - \bar{V}_2(s') \Big|$$

Now, we can apply Equation 5. Note the *requirement* that $\|V_1^{fixed} - V_2^{fixed}\|_\infty^{S \setminus S_j} \leqslant \|V_1 - V_2\|_\infty^{S_j}$. Informally, this means that the largest error in the fixed values is bounded by the current iteration's largest source of error from partition j. This is trivially true when performing VI in Algorithm 1 on Lines 6-13 because V^{fixed} is the same for V_1 and V_2, implying that $0 \leqslant \|V_1 - V_2\|_\infty^{S_j}$. Therefore, we are left with the difference of V_1 and V_2 over S_j.

$$\leqslant \gamma \max_{s \in S_j} \max_{a \in A_i(s)} \sum_{s' \in S_j} T(s,a,s') \Big| V_1(s') - V_2(s') \Big|$$

$$\leqslant \gamma \max_{s \in S_j} \Big| V_1(s) - V_2(s) \Big| \leqslant \gamma \|V_1 - V_2\|_\infty^{S_j}$$

This proves that the operator B_i is a contraction map on metric space M_i, for all $i \in K$. □

We may now guarantee convergence to within $\epsilon > 0$ of the fixed point for a specific partition's value function. We omit this proof because it is the same as previous proofs and does not produce more assumptions (unlike Proposition 2).

Proposition 3. For all $j \in L$, for $i \in K$, assume $1, \ldots, i-1$ has converged. Following Equation 4; Lines 7-10, for any $i \in K$, B_i converges to within $\epsilon > 0$ of a unique fixed point once $\|V_i^{t+1} - V_i^t\|_\infty^{S_j} < \epsilon \frac{1-\gamma}{\gamma}$ for iteration $t > 0$.

As part of Proposition 2's proof, there is an implicit constraint underlying its application; essentially, for a partition to properly converge, there must be a bound on the fixed values of other partitions. This is trivially true for the inner while loop on Lines 8-11, but not necessarily true for the outer while loop. We formally state the assumption below.

Assumption 1 (Proper Partitions). Let $j \in L$ be the partition which satisfies $\|V_1^{fixed} - V_2^{fixed}\|_\infty^{S \setminus S_j} \leqslant \|V_1 - V_2\|_\infty^{S_j}$ from Proposition 2. Also, let $i \in K$ such that $1, \ldots, i-1$ has converged. Assume that this inequality holds until j has converged to within ϵ and then remains fixed.

Intuitively, with respect to LVI, the assumption takes the partition which is the source of the largest error, and assumes that it converges first and then remains fixed thereafter. For example, this can arise in scenarios with goal states, since the partition without the goal state implicitly depends on the value functions of the partition with the goal state. We prove that LVI converges under this assumption.

Proposition 4. LVI (Algorithm 1) converges to a unique fixed point V^* under Assumption 1 with $\gamma \in [0, 1)$.

Proof. We must show that for each partition, all value functions converge following the respective orderings, in sequence, to a unique fixed point. We do this by constructing a metric space of the currently converged values, and the ones that are "converging" (i.e., satisfying Assumption 1). Then, we apply Proposition 2 which proves that G is a contraction map on this space. Finally, since this is true for all partitions and their value functions, and Assumption 1 guarantees converged values remain stable, it converges to the entire metric space over the entire partitions and their values.

Let Y be the space of all value functions over partitions $\forall j \in L' \subseteq L$, value functions $K'_j \subseteq K$, and their states S which have converged or satisfy Assumption 1. Note that the base case includes only the first $o_j(1) \in K'_j$ from each $j \in L'$. Let distance metric d be the *max norm*, i.e., $\|V\|_\infty = \max_{j \in L'} \max_{i \in K'_j} \max_{s \in S_j} |V_i(s)|$. Thus, the metric space $M = \langle Y, d \rangle$ is a *Banach space*.

Let G be an operator in M following Lines 4-13 in Algorithm 1, i.e., $V' = GV$ using the algorithm's variables: V and V'. We must show that G is a contraction map. Let $V_1, V_2 \in Y$ be any two value function vectors. Rewrite in terms of B_i and then apply Proposition 2.

$$\begin{aligned}
\|GV_1 - GV_2\|_\infty &= \max_{j \in L'} \max_{i \in K'_j} \max_{s \in S_j} |(GV_1)_i(s) - (GV_2)_i(s)| \\
&= \max_{j \in L'} \max_{i \in K'_j} \max_{s \in S_j} |B_iV_{1i}(s) - B_iV_{2i}(s)| \\
&\leqslant \gamma \max_{j \in L'} \max_{i \in K'_j} \max_{s \in S_j} |V_{1i}(s) - V_{2i}(s)| \\
&\leqslant \gamma \|V_1 - V_2\|_\infty
\end{aligned}$$

Therefore, G is a contraction map. By *Banach's fixed point theorem*, since M is a complete metric space and G is a contraction map on Y, G admits a unique fixed point $V^* \in Y$. The convergence of G in M produces a new expanded metric space, which converges to a new unique fixed point in the larger space. This process repeats to completion until we reach the entire space over $\mathbb{R}^{k \times n}$, yielding a final unique fixed point V^*. □

Relation to Other Models

Interestingly, there is a connection between LMDPs and game theory. This link bridges the two often distinct fields of decision theory and game theory, while simultaneously providing another description of the optimality of an LMDP policy. Furthermore, it opens the possibility for future work on game theoretic solvers which produce solutions that map back from the *normal form game* to an LMDP policy.

Definition 2 describes this mapping from an LMDP to a normal form game. Intuitively, each partition can be thought of as a player in a game. The action a partition can take are any sub-policy for the states in the partition. We select any ℓ value functions, one from each partition, and use those as the payoffs for a normal form game. We show in Proposition 5 that the resulting LMDP's optimal policy computed from LVI is, in fact, a Nash equilibrium.

Definition 2. Let LMDP $\langle S, A, T, \mathbf{R}, \delta, \mathcal{S}, o \rangle$ have value functions V^{π} for corresponding optimal policy π, the optimal value functions V^{η}, computed via LVI (Algorithm 1). Let $\bar{s} = \langle s_1, \ldots, s_\ell \rangle$ be a state tuple such that $\forall z \in L$, $s_z \in S_z$. Let $\bar{i} = \langle i_1, \ldots, i_\ell \rangle$ be any tuple of indices $i_z \in K$.

The **LMDP's Normal Form Game** $\langle L, \mathcal{A}, U \rangle$ is:

- $L = \{1, \ldots, \ell\}$ is a set of agents (partitions)
- $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_\ell$ is a finite set of joint actions, such that $\forall z \in L, x = |S_z|, S_z = \{s_{z1}, \ldots, s_{zx}\}, \mathcal{A}_z = A_{i_z}(s_{z1}) \times \cdots \times A_{i_z}(s_{zx})$
- $U = \langle u_1, \ldots, u_\ell \rangle$ is a set of utility functions, such that $\forall z \in L$, $\forall a \in \mathcal{A}$, $s_z \in \bar{s}$, $u_z(a_z, a_{-z}) = \min\{V_{i_z}^{\pi, a_z}(s_z), V_{i_z}^{\eta}(s_z) - \delta_{i_z}\}$.

Note that we only consider pure strategy profiles. Thus, a player's *strategy set* will be used synonymously with its *action set*. Similarly, since we consider one stage games, *strategy profile* will be synonymous with *action profile*.

Proposition 5. For an LMDP, the optimal policy π is a weak pure strategy Nash equilibrium of the LMDP's equivalent normal form.

Proof. Let LMDP $\langle S, A, T, \mathbf{R}, \delta, \mathcal{S}, o \rangle$ have optimal policy π. Let $f(\pi) = \langle \omega_1, \ldots, \omega_\ell \rangle$ so that $\forall z \in L$, $x = |S_z|$, $S_z = \{s_{z1}, \ldots, s_{zx}\}$, $\omega_z = \langle \pi(s_{z1}), \ldots, \pi(s_{zx}) \rangle$. After applying the transformation in Definition 2, we must show that the strategy (action) profile $f(\pi) = a = (a_z, a_{-z}) \in \mathcal{A}$ is a weak pure strategy Nash equilibrium.

By the definition of a (weak) Nash equilibrium, we must show that $\forall z \in L$, $\forall a' \in \mathcal{A}_z$, $u_z(a_z, a_{-z}) \geqslant u_z(a', a_{-z})$. Let π' be the corresponding policy for $f(\pi') = \langle a_1, \ldots, a_{z-1}, a', a_{z+1}, \ldots, a_\ell \rangle \in \mathcal{A}$. Let $V^{\pi'}$ be the value functions after value iteration has converged for each value function in K following policy π'. Note that $\mathbf{V}_x^{\pi} = \mathbf{V}_x^{\pi'}$, $\forall x \in \{1, \ldots, i_z - 1\}$, and therefore by Equation 2, $A_{i_z}^{\pi} = A_{i_z}^{\pi'}$. This ensures we may use Proposition 1, by considering a MOMDP with a reduced number of rewards up to i_z.

By Proposition 1, $V_{i_z}^{\pi}(s_z) \geqslant V_{i_z}^{\eta}(s_z) - \delta_i$. Apply the fact that π is defined following action a_z, so $V_{i_z}^{\pi, a_z}(s_z) = V_{i_z}^{\pi}(s_z)$. Thus, by Definition 2: $u_z(a_z, a_{-z}) = \min\{V_{i_z}^{\pi}(s_z), V_{i_z}^{\eta}(s_z) - \delta_i\} = V_{i_z}^{\eta}(s_z) - \delta_i$. There are two cases for $V_{i_z}^{\pi'}(s_z)$; regardless, the inequality $u_z(a_z, a_{-z}) \geqslant u_z(a', a_{-z})$ is true. Therefore, the strategy (action) profile π is a Nash equilibrium. □

We now show that optimal solutions for LMDPs may not exist in the space of solutions captured by MOMDPs with linearly weighted scalarization functions. Hence, LMDPs are a new distinct multi-objective optimization problem, while still being related to MOMDPs. Figure 1 demonstrates the proof's main idea: Construct a MOMDP with conflicting weight requirements to produce the LMDP's optimal policy.

Proposition 6. The optimal policy of an LMDP π may not exist in the space of solutions captured by its corresponding scalarized MOMDP's policy $\pi_{\mathbf{w}}$.

Proof. Consider the MOMDP depicted in Figure 1, with $S = \{s_1, s_2, s_3, s_4\}$, $A = \{stay, leave\}$, $T(s, stay, s') = 1$ if $s = s'$, $T(s, leave, s') = 1$ if $s \neq s'$, $T(s, a, s') = 0$ otherwise, and rewards $\mathbf{R} = [R_1, R_2]^T$ as shown.

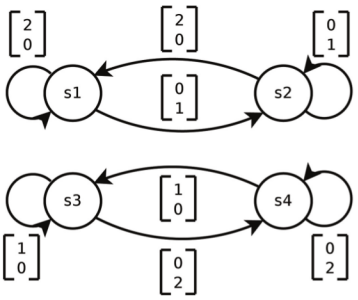

Figure 1: The example MOMDP for Proposition 6.

For states s_1 and s_2, $w_1 < 1/3$ and $w_2 > 2/3$ produce the policy $\pi_{\mathbf{w}}(s_1) = leave$ and $\pi_{\mathbf{w}}(s_2) = stay$. Similarly, weights $w_1 > 1/3$ and $w_2 < 2/3$ produce the policy $\pi_{\mathbf{w}}(s_1) = stay$ and $\pi_{\mathbf{w}}(s_2) = leave$. Only weights set to $w_1 = 1/3$ and $w_2 = 2/3$ allows for ambiguity; a carefully selected tie-breaking rule alows for the policy $\pi_{\mathbf{w}}(s_1) = \pi_{\mathbf{w}}(s_2) = stay$.

The exact same logic holds for states s_3 and s_4 with the weights reversed at $w_1 = 2/3$ and $w_2 = 1/3$ to allow a tie-breaking rule to produce $\pi_{\mathbf{w}}(s_3) = \pi_{\mathbf{w}}(s_4) = stay$.

Construct the corresponding LMDP with $\mathcal{S} = \{S_1, S_2\}$, $S_1 = \{s_1, s_3\}$, $S_2 = \{s_2, s_4\}$, $o = \langle o_1, o_2 \rangle$, $o_1 = \langle 1, 2 \rangle$, $o_2 = \langle 2, 1 \rangle$, and $\delta = \langle 0, 0 \rangle$. The policy returned by LVI is $\pi(s) = stay$ for all $s \in S$. This policy is unique since it cannot be produced by any selection of weights. □

4 Experimentation

Semi-autonomous systems require efficient policy execution involving the transfer of control between human and agent (Cohn, Durfee, and Singh 2011). Our experiments focus on a semi-autonomous driving scenario in which transfer of control decisions depend on the driver's level of fatigue. The multi-objective model we use is motivated by an extensive body of engineering and psychological research on monitoring driver fatigue and risk perception (Ji, Zhu, and Lan 2004; Pradhan et al. 2005). We use real-world road data from OpenStreetMap[1] for sections of the 10 cities in Table 1.

The Semi-Autonomous Driving Domain

LMDP states are formed by a pair of intersections (previous and current), driver tiredness (true/false), and autonomy (enabled/disabled). The intersection pair captures the direction of travel. Actions are taken at intersections, as found in path planning in graphs (e.g., GPS). Due to real world limitations, autonomy can only be enabled on main roads and highways; in our case, this means a speed limit greater than or equal to 30 miles per hour. Actions are simply which road to take at each intersection and, if the road allows, whether or not to enable autonomy. The stochasticity within state transitions model the likelihood that the driver will drift from attentive to tired, following probability of 0.1.

[1] http://wiki.openstreetmap.org

Figure 2: The policy for attentive (above) and tired (below).

Rewards are the negation of costs. Thus, there are two cost functions: *time* and *fatigue*. The time cost is proportional to the time spent on the road (in seconds), plus a small constant value of 5 (seconds) which models time spent at a traffic light, turning, or slowed by traffic. The fatigue cost is proportional to the time spent on the road, but is only applied if the driver is tired and autonomy is disabled; otherwise, there is an ϵ-cost. The absorbing goal state awards zero for both.

It is natural to minimize the time cost when the driver is attentive and the fatigue cost when the driver is tired. We allow for a 10 second slack in the expected time to reach the goal, in order to favor selecting autonomy-capable roads.

Figure 2 shows an example optimal policy returned by LVI for roads north of Boston Commons. Each road at an intersection has an action, denoted by arrows. Green arrows denote that autonomy was disabled; purple arrows denote that autonomy was enabled. The blue roads show autonomy-capability. We see that the policy (green) correctly favors a more direct route when the driver is attentive, and favors the longer but safer autonomous path when the driver is tired. This is exactly the desired behavior: Autonomy is favored for a fatigued driver to improve vehicle safety.

We created simple-to-use tools to generate a variety of scenarios from the semi-autonomous driving domain. This domain addresses many of the existing concerns regarding current MOMDP evaluation domains (Vamplew et al. 2011). In particular, our domain consists of real-world road data, streamlined stochastic state transitions, both small and large state and action spaces, policies which are natural to understand, and the ability for both qualitative and quantitative comparisons among potential algorithms.

GPU-Optimized LVI

We explore the scalability of LVI in a GPU-optimized implementation using CUDA[2]. VI on GPUs has been shown to improve performance and enable scalability (Jóhannsson 2009; Boussard and Miura 2010). Our approach for LVI exploits this structure of Bellman's optimality equation: independent updates of the values of states. LVI also allows us to run each partition separately, further improving its scalability.

[2]https://developer.nvidia.com/about-cuda

City	$\|S\|$	$\|A\|$	$\mathbf{w}$	CPU	GPU
Chicago	400	10	1.5	3.3	3.9
Denver	616	10	6.2	12.9	6.4
Baltimore	676	8	5.6	14.1	5.7
Pittsburgh	864	8	7.3	15.4	7.9
Seattle	1168	10	29.2	63.5	14.2
Austin	1880	10	99.4	433.3	29.8
San Franc.	2016	10	4235.8	4685.7	159.4
L.A.	2188	10	118.7	273.5	37.8
Boston	2764	14	10595.5	11480.9	393.2
N.Y.C.	3608	10	14521.8	16218.5	525.7

Table 1: Computation time in seconds for weighted scalarized VI ($\mathbf{w}$), and LVI on both the CPU and GPU.

Our implementation first transfers the entire LMDP model from the host (CPU) to the device (GPU). Each partition is executed separately, with multiple Bellman updates running in parallel. After convergence, the final policy and values are transferred back from the device to the host. One of the largest performance costs comes from transferring data between host and device. LVI avoids this issue since transfers occur outside each iteration. Our experiments shown in Table 1 were executed with an Intel(R) Core(TM) i7-4702HQ CPU at 2.20GHz, 8GB of RAM, and an Nvidia(R) GeForce GTX 870M graphics card using C++ and CUDA(C) 6.5.

Proposition 6 clearly shows that optimal polices for an LDMP may differ from those of a scalarized MOMDP. One valid concern, however, is that scalarization methods may still return a reasonable policy much faster than LVI. For this reason, we evaluated the computation times of both weighted VI, as well as LVI run on both a CPU and GPU. As shown in the table, the runtimes of LVI and scalarized VI differ by a small constant factor. The GPU version of LVI vastly improves the potential scales of LMDPs, reaching speeds roughly 30 times that of the CPU.

5 Conclusion

Our work focuses on a novel model for state-dependent lexicographic MOMDPs entitled LMDPs. This model naturally captures common multi-objective stochastic control problems such as planning for semi-autonomous driving. Competing approaches such as linearly weighted scalarization involve non-trivial tuning of weights and—as we show—may not be able to produce the optimal LMDP policy for an LMDP. Our proposed solution, LVI, converges under a partition-dependent assumption and is amenable to GPU-based optimization, offering vast performance gains. Within the domain of semi-autonomous driving, policies returned by LVI successfully optimize the multiple state-dependent objectives, favoring autonomy-capable roads when the driver is fatigued, thus maximizing driver safety.

In future work, we plan to enrich the semi-autonomous driving domain and make the scenario creation tools public to facilitate further research and comparison of algorithms for multi-objective optimization. We also plan to further investigate the analytical properties and practical applicability of LVI and its GPU-based implementation.

6 Acknowledgements

We thank Laurent Jeanpierre and Luis Pineda for fruitful discussions of multi-objective MDPs. This work was supported in part by the National Science Foundation grant number IIS-1405550.

References

Altman, E. 1999. *Constrained Markov Decision Processes*, volume 7. CRC Press.

Bellman, R. E. 1957. *Dynamic Programming*. Princeton, NJ: Princeton University Press.

Boussard, M., and Miura, J. 2010. Observation planning with on-line algorithms and GPU heuristic computation. In *ICAPS Workshop on Planning and Scheduling Under Uncertainty*.

Boutilier, C.; Brafman, R. I.; Domshlak, C.; Hoos, H. H.; and Poole, D. 2004. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research* 21:135–191.

Calisi, D.; Farinelli, A.; Iocchi, L.; and Nardi, D. 2007. Multi-objective exploration and search for autonomous rescue robots. *Journal of Field Robotics* 24(8-9):763–777.

Castelletti, A.; Pianosi, F.; and Soncini-Sessa, R. 2008. Water reservoir control under economic, social and environmental constraints. *Automatica* 44(6):1595 – 1607.

Cohn, R.; Durfee, E.; and Singh, S. 2011. Comparing action-query strategies in semi-autonomous agents. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, 1287–1288.

Gábor, Z.; Kalmár, Z.; and Szepesvári, C. 1998. Multi-criteria reinforcement learning. In *Proceedings of the 15th International Conference on Machine Learning*, volume 98, 197–205.

Gonzales, C.; Perny, P.; and Dubus, J. P. 2011. Decision making with multiple objectives using GAI networks. *Artificial Intelligence* 175(78):1153 – 1179.

Ji, Q.; Zhu, Z.; and Lan, P. 2004. Real-time nonintrusive monitoring and prediction of driver fatigue. *IEEE Transactions on Vehicular Technology* 53(4):1052–1068.

Jóhannsson, Á. P. 2009. GPU-based Markov decision process solver. Master's thesis, School of Computer Science, Reykjavík University.

Kwak, J.-Y.; Varakantham, P.; Maheswaran, R.; Tambe, M.; Jazizadeh, F.; Kavulya, G.; Klein, L.; Becerik-Gerber, B.; Hayes, T.; and Wood, W. 2012. SAVES: A sustainable multiagent application to conserve building energy considering occupants. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 21–28.

Mitten, L. G. 1974. Preference order dynamic programming. *Management Science* 21(1):43–46.

Mouaddib, A.-I. 2004. Multi-objective decision-theoretic path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, 2814–2819.

Natarajan, S., and Tadepalli, P. 2005. Dynamic preferences in multi-criteria reinforcement learning. In *Proceedings of the 22nd International Conference on Machine learning*, 601–608.

Perny, P., and Weng, P. 2010. On finding compromise solutions in multiobjective Markov decision processes. In *Proceedings of the European Conference on Artificial Intelligence*, 969–970.

Perny, P.; Weng, P.; Goldsmith, J.; and Hanna, J. 2013. Approximation of Lorenz-optimal solutions in multiobjective Markov decision processes. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, 92–94.

Pradhan, A. K.; Hammel, K. R.; DeRamus, R.; Pollatsek, A.; Noyce, D. A.; and Fisher, D. L. 2005. Using eye movements to evaluate effects of driver age on risk perception in a driving simulator. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 47(4):840–852.

Roijers, D. M.; Vamplew, P.; Whiteson, S.; and Dazeley, R. 2013. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research* 48:67–113.

Russell, S., and Norvig, P. 2010. *Artificial Intelligence: A modern approach*. Upper Saddle River, New Jersey: Prentice Hall.

Sobel, M. J. 1975. Ordinal dynamic programming. *Management Science* 21(9):967–975.

Vamplew, P.; Dazeley, R.; Berry, A.; Issabekov, R.; and Dekker, E. 2011. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning* 84(1-2):51–80.

Zilberstein, S. 2015. Building strong semi-autonomous systems. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.