Probabilistic Planning with Prioritized Preferences over Temporal Logic Objectives

Lening Li[1] , **Hazhar Rahmani**[2] , **Jie Fu**[2]
[1]Lening Li is with the Department of Robotics, Worcester Polytechnic Institute, Worcester, MA 01609, USA.
[2]Hazhar Rahmani and Jie Fu are with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32605, USA.
lli4@wpi.edu, {h.rahmani, fujie}@ufl.edu *

Abstract

This paper studies temporal planning in probabilistic environments, modeled as labeled Markov decision processes (MDPs), with user preferences over multiple temporal goals. Existing works reflect such preferences as a prioritized list of goals. This paper introduces a new specification language, termed prioritized qualitative choice linear temporal logic on finite traces, which augments linear temporal logic on finite traces with prioritized conjunction and ordered disjunction from prioritized qualitative choice logic. This language allows for succinctly specifying temporal objectives with corresponding preferences accomplishing each temporal task. The finite traces that describe the system's behaviors are ranked based on their dissatisfaction scores with respect to the formula. We propose a systematic translation from the new language to a weighted deterministic finite automaton. Utilizing this computational model, we formulate and solve a problem of computing an optimal policy that minimizes the expected score of dissatisfaction given user preferences. We demonstrate the efficacy and applicability of the logic and the algorithm on several case studies with detailed analyses for each.

1 Introduction

In this work, we study preference-based planning given a preference order over temporal goals, i.e. , ordered goals specified in temporal logics. Temporal logic are expressive and rigorous languages for specifying complex tasks and mission objectives. Planning with temporal logic goals [Pnueli, 1981] has seen studied for robotic systems [Kantaros *et al.*, 2022; Bradley *et al.*, 2021; He *et al.*, 2020; Vasile *et al.*, 2020; Yang *et al.*, 2020; Wang *et al.*, 2020; Hekmatnejad and Fainekos, 2018; He *et al.*, 2015; Li *et al.*, 2021] and other intelligent systems [Kasenberg *et al.*, 2020; De Giacomo and Vardi, 2015; Camacho *et al.*, 2017; Mallett *et al.*, 2021; Zhou *et al.*, 2022; Zhao *et al.*, 2022].

*This material is based upon work supported by Air Force Office of Scientific Research under award number FA9550-21-1-0085 and by NSF under Grant No. 2024802 and Grant No.2144113.

Specifying preferences over temporal goals gives the decision maker flexibility to revise the task and achieve the most preferred outcomes when not all constraints/subtasks can be satisfied. Early works consider deterministic systems—modeled as finite, discrete systems or system with deterministic dynamics. Several works [Tumova *et al.*, 2013b; Tumova *et al.*, 2013a; Wongpiromsarn *et al.*, 2021; Vasile *et al.*, 2017] proposed minimum violation planning methods that decide which low-priority constraints should be violated in a deterministic system. [Mehdipour *et al.*, 2021] associate weights with Boolean and temporal operators in signal temporal logic to specify the importance of satisfying the subformula and priority in the timing of satisfaction. They develop algorithms to maximize the weighted satisfaction in deterministic dynamical systems. [Rahmani and O'Kane, 2019; Rahmani and O'Kane, 2020] studied temporal planning given both hard and soft specifications of the goal, using linear temporal logic (LTL) and linear dynamic logic on finite traces (LDL_f). [Cai *et al.*, 2020] consider minimizing the deviations from infeasible LTL specifications while maximizing the total rewards.

Recently, preference-based probabilistic planning with temporal logic constraints are studied. [Li *et al.*, 2020] study preference-based planning for Markov decision process (MDP) subject to an ordered list of probabilistic temporal logic formula. The algorithm enumerates tasks one by one in a prioritized order until a policy that satisfies the most preferred task is found. [Lahijanian and Kwiatkowska, 2016] studied syntactically co-safe LTL planning with infeasible specifications in environments modeled by MDPs. They compute a policy that maximizes the probability of satisfying a revised formula and minimizes the cost of revision. [Lacerda *et al.*, 2015] considered a similar problem where the aim is to synthesize a policy that, in decreasing order of priority, maximizes the probability of completing the task, maximizes the probability of progressing toward completion, and minimizes the expected cost.

Despite the existing work on probabilistic preference-based planning, the connection between preference specification in AI and preferences over temporal goals is yet to be established. We propose a new language that extends temporal logic with fuzzy logic representation of preferences. Specifically, we consider qualitative choice logic (QCL) pro-

posed in [Brewka *et al.*, 2004] and its extension prioritized qualitative choice logic (PQCL) [Benferhat and Sedki, 2007; Benferhat and Sedki, 2008]. QCL extends propositional logic with a new logical connective called *ordered disjunction*, denoted $\overrightarrow{\times}$. Formula $A \overrightarrow{\times} B$ means if possible then A, but if A is not possible then at least B. PQCL introduced prioritized conjunction/disjunction to QCL by allowing the user to express priorities in a user's preference. Combined, if $(A \overrightarrow{\times} B)\&(C \overrightarrow{\times} D)$ where $\&$ is the prioritized conjunction, then the preference of $A \overrightarrow{\times} B$ is more important to be satisfied than the preference $C \overrightarrow{\times} D$.

The preference language proposed herein, called prioritized qualitative choice linear temporal logic on finite traces (PQCLTL$_f$), integrates PQCL with a subclass of LTL over finite traces. In particular, we introduce LTL formulas for atomic preference and employ PQCL to represent a preference over the temporal goals. We assign a dissatisfaction score for each outcome (temporal sequence of states) in light of semantics for PQCL. This scoring function enables us to formulate a preference-based probabilistic planning objective, that is, to minimize the expected dissatisfaction score in a stochastic system, modeled as a labeled MDP.

However, this compact, logical representation of preferences alone is not sufficient for probabilistic planning, which generally requires a computational model. Based on the relation between LTL and automata, we developed a procedure that translates a prioritized qualitative choice temporal logic (PQCTL) formula into a *weighted deterministic finite automaton*. This weighted automaton ensures for each path that satisfies the preference to a degree k, the sum of weights of the corresponding induced run on the weighted automaton is exactly k. Augmenting the planning state space with the state set of the weighted automata using a product operation, we show that the most preferred policy in the stochastic system can be obtained by solving a product MDP. The correctness of the solution hinges upon the definition of a reward function based on the weights on transitions in the weighted automaton. We formally prove that the reward-maximizing policy in the product MDP minimizes the expected degree of dissatisfaction in the original MDP given the PQCTL formula. In experiments, we employ several examples of robotic motion planning example to demonstrate the efficacy and applicability of the method and provide a detailed comparison of preference-based planning and traditional probabilistic planning with a monolithic temporal logic formula.

2 Preliminaries

Notations: The set of all probability distributions over a finite set X is denoted $\mathcal{D}(X)$.

We introduce necessary preliminaries and notations next.

System model We model the interaction between the planning agent (a robot) and its stochastic environment as a variant of MDP.

Definition 1 (Labeled Markov Decision Process with a terminating state)**.** A labeled Markov decision process with a teminating state (TLMDP) is a tuple $M = \langle S, A := \cup_{s\in S} A_s, P, s_0, s_\perp, \mathcal{AP}, L\rangle$ where S is a finite set of states; A is a finite set of actions, where for each state $s \in S$, A_s is the set of available actions at s; A includes a special *terminating action* $a_\perp$ and for any $s \in S$, $a_\perp \in A_s$. $P\colon S\times A\times S \to [0,1]$ is the transition probability function, where for each $s, s' \in S$ and $a \in A$, $P(s' \mid s, a)$ is the probability that the MDP transitions to s' after taking action a at s; $s_0 \in S$ is the initial state; $s_\perp \in S$ is the *terminating state*, which is a unique *sink* state. For any $s \in S$, $P(s, a_\perp, s_\perp) = 1$. That is, if an agent selects the terminating action $a_\perp$, then a terminating state $s_\perp$ can be reached surely. The set $\mathcal{AP}$ is a finite set of atomic propositions; and $L\colon S \to 2^{\mathcal{AP}} \cup \{\ltimes\}$ is a labeling function that assigns to each state $s \in S \setminus \{s_\perp\}$, the set of atomic propositions $L(s) \subseteq \mathcal{AP}$ that hold in s. Only the terminating state is labeled the empty string, i.e., $L(s_\perp) = \ltimes$.

A finite *run* in this MDP is a sequence $\varrho = s_0 a_0 s_1 a_1 \cdots s_{k-1} a_{k-1} s_k$, in which, s_0 is the initial state and for each $0 \le i \le k-1$, $P(s_{i+1} \mid s_i, a_i) > 0$. The path associated with this run is the sequence $\rho = s_0 s_1 \cdots s_k \in S^*$ an the *trace* of this path is defined as $\mathrm{trace}(\rho) = L(s_0)L(s_1)L(s_2)\cdots L(s_k) \in (2^{\mathcal{AP}})^*$. A path which ends at $s_\perp$ is called terminating.

A finite-memory, randomized policy in the MDP is a function $\pi : S^* \to \mathcal{D}(A)$ that maps a state sequence into a distribution over actions. A Markovian, or memoryless, randomized policy in the MDP is a function $\pi : S \to \mathcal{D}(A)$ that maps the current state into a distribution over actions. We denote the set of all Markovian randomized policies as Π.

A finite-memory, randomized policy $\pi : S^* \to \mathcal{D}(A)$ induces a Markov chain $M^\pi = \langle S^*, P^\pi\rangle$ over S^* as follows: For any $\rho \in S^*$, $s \in S$,

$$P^\pi(\rho s|\rho) = \sum_{a\in A} P(s|\mathsf{Last}(\rho), a)\cdot \pi(\rho, a), \qquad (1)$$

where $\mathsf{Last}(\rho)$ is the last state given the sequence ρ.

The stochastic process induced by a Markov policy is a Markov chain $M^\pi = \langle S^*, P^\pi\rangle$, where P^π can be obtained as a special case of (1). The probability of a path ρ in the Markov chain M^π is denoted by $\mathbf{Pr}(\rho; M^\pi)$.

Planning objectives We are interested in probabilistic planning subject to a preference over a set of temporal goals, captured using linear temporal logic on finite traces (LTL$_f$) formulas.

Definition 2 (LTL$_f$ Syntax [De Giacomo and Vardi, 2013])**.** Given a finite set $\mathcal{AP}$ of atomic propositions, the syntax of LTL$_f$ formulas is defined as follows:

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \bigcirc \varphi \mid \varphi \,\mathsf{U}\, \varphi,$$

where $p \in \mathcal{AP}$, negation ($\neg$) and conjunction ($\wedge$) are standard Boolean operators, and "Next" ($\bigcirc$) and "Until" (U) are temporal operators.

Informally, formula $\bigcirc \varphi$ states that φ holds at the next time instant, and $\varphi_1 \,\mathsf{U}\, \varphi_2$ means there is a future time instant at which φ_2 holds and for all time instants from the current time until that future time, φ_1 holds true. The temporal operator "Eventually" ($\Diamond$) is defined using "Until" as $\Diamond \varphi := \mathsf{true}\,\mathsf{U}\,\varphi$. The dual of this operator is "Always" ($\Box$), which is defined

as $\Box\varphi := \neg\Diamond\neg\varphi$. Formula $\Diamond\varphi$ means there is some future time instant at which φ holds, while $\Box\varphi$ is interpreted that φ is true at all future time instants. The semantics of LTL$_f$ is given as interpretations over finite traces and can be found in [De Giacomo and Vardi, 2013].

The language of an LTL$_f$ formula φ, denoted $\mathcal{L}(\varphi)$, is the set of finite traces over the alphabet $2^{\mathcal{AP}}$ that satisfy φ. For notational simplicity, let $\Sigma := 2^{\mathcal{AP}}$ in the following context. The set of all finite words over a finite alphabet Σ is denoted by Σ^*. The language of LTL$_f$ formula φ can be represented by the set of words accepted by a deterministic finite automaton (DFA) $\mathcal{A}_\varphi = \langle Q, \Sigma, \delta, q_0, F\rangle$, where Q is a finite set of states; $\Sigma = 2^{\mathcal{AP}}$ is a finite set of symbols called the alphabet; $\delta : Q \times \Sigma \to Q$ is a transition function such that $\delta(q, \sigma) = q'$ is the state reached upon reading input σ from state q; $q_0 \in Q$ is an initial state; and $F \subseteq Q$ is a set of accepting/final states. A transition function is recursively extended in the general way: $\delta(q, \sigma w) = \delta(\delta(q, \sigma), w)$ for given $\sigma \in \Sigma$ and $w \in \Sigma^*$. A word w is *accepting* if and only if $\delta(q, w) \in F$. The DFA $\mathcal{A}_\varphi$ accepts the exact set of words satisfying φ given the semantics of LTL$_f$.

3 Preference Language: Integration of Prioritized Qualitative Choice Logic and Temporal Logic

In this section, we present a new task specification language to describe a subset of preferences over temporal goals. We call this language *PQCLTL$_f$*, which combines LTL$_f$ with PQCL [Benferhat and Sedki, 2007]—a propositional logic for representing ranked objectives.

PQCL augments propositional logic with a connective $\overrightarrow{\times}$, called *ordered disjunction*: A formula $\phi_1 \overrightarrow{\times} \phi_2$ means that if possible then ϕ_1, and if ϕ_1 is not possible then ϕ_2. The operator $\overrightarrow{\times}$ is left-associative, and therefore $\phi_1 \overrightarrow{\times} \phi_2 \overrightarrow{\times} \ldots \overrightarrow{\times} \phi_n = \phi_1 \overrightarrow{\times} (\phi_2 \overrightarrow{\times} (\ldots \overrightarrow{\times} \phi_n) \ldots)$. In addition to ordered disjunction, PQCL introduces *prioritized conjunction*: A formula $\phi_1 \& \phi_2$ defines the lexicographical ordering between individual satisfaction of ϕ_1 and ϕ_2.

Definition 3 (Prioritized Qualitative Choice Linear Temporal Logic on Finite Traces)**.** Let Φ be a set of LTL$_f$ formulas over a set of atomic propositions $\mathcal{AP}$. A PQCLTL$_f$ fragment over $\mathcal{AP}$ (without negation) is defined by

$$\varphi := \psi \mid \varphi \overrightarrow{\times} \varphi \mid \varphi \& \varphi,$$

in which $\psi \in \Phi$.

In comparison to PQCL, we do not include negation operation and thereby exclude the prioritized disjunction. Negation is only allowed in the construction of LTL$_f$ formulas. The reason of not including negation is mainly due to ambiguity: A negation of flight $\overrightarrow{\times}$ train can mean the two options are indifferent, incomparable, or train is preferred to flights.

Different words satisfy a PQCLTL$_f$ formula to different degrees, which motivates us to quantify the number of alternatives a PQCLTL$_f$ formula can be satisfied.

Definition 4 (Optionality, extended from [Benferhat and Sedki, 2007])**.** Given an PQCLTL$_f$ formula φ, the optionality of φ, denoted $opt(\psi)$, is the number of ways φ can be satisfied, and is computed recursively as follows:

- If φ is an LTL$_f$ formula, then $opt(\varphi) = 1$;
- If $\varphi = \varphi_1 \overrightarrow{\times} \varphi_2$, then $opt(\varphi) = opt(\varphi_1) + opt(\varphi_2)$;
- If $\varphi = \varphi_1 \& \varphi_2$, then $opt(\varphi) = opt(\varphi_1) \cdot opt(\varphi_2)$.

Associated with this definition of optionality, for each word $w \in \Sigma^*$ and a PQCLTL$_f$ formula, the word satisfies the formula to a certain degree.

Definition 5 (Satisfaction Degree, extended from [Benferhat and Sedki, 2007])**.** Let φ be a PQCLTL$_f$ formula over $\mathcal{AP}$ and $w \in \Sigma^*$ (recall $\Sigma = 2^{\mathcal{AP}}$) be a finite word. We write $w \models_k \varphi$ for some positive integer $k > 0$ to denote that the satisfaction degree of w with respect to φ is k, and use $w \not\models \varphi$ to denote that w does not satisfy φ.

The satisfaction degree of w with respect to φ is computed as follows:

- If φ is an LTL$_f$ formula, then $w \models_1 \varphi$ if $w \in \mathcal{L}(\varphi)$, and $w \not\models \varphi$ if $w \notin \mathcal{L}(\varphi)$.
- If $\varphi = \varphi_1 \overrightarrow{\times} \varphi_2$, then $w \models_k \varphi_1 \overrightarrow{\times} \varphi_2$ if either
  - $w \models_k \varphi_1$; or
  - $w \models_n \varphi_2$, $w \not\models \varphi_1$, and $k = n + opt(\varphi_1)$.
- If $\varphi = \varphi_1 \& \varphi_2$, then $w \models_k \varphi_1 \& \varphi_2$ if
  - there exist $i, j > 0$ such that $w \models_i \varphi_1$, $w \models_j \varphi_2$, and $k = opt(\varphi_2) \times (i - 1) + j$;
  - otherwise, if $w \not\models \varphi_1$ or $w \not\models \varphi_2$, then $w \not\models \varphi_1 \& \varphi_2$.

The definition of satisfaction degree induces a total order only on the set of all the words that satisfy the PQCLTL$_f$ formula, but it does not rank those words who does not satisfy the formula. For planning purposes, we introduce a metric whose range of values is circumscribed between 0 and 1.

Definition 6 (Dissatisfaction Score)**.** The dissatisfaction score function is a function $\mathsf{d} : \Sigma^* \times \Phi \to (0, 1]$ that assigns to each word $w \in \Sigma^*$ and PQCLTL$_f$ formula $\varphi \in \Phi$, a positive real value in $(0, 1]$, called the *dissatisfaction score of w with respect to φ*, which is computed as follows:

- If $w \not\models \varphi$, then $\mathsf{d}(w, \varphi) = 1$;
- If $w \models_k \varphi$ for $k > 0$, then $\mathsf{d}(w, \varphi) = \frac{k}{opt(\varphi)+1}$.

The lower the score, the more satisfied is the word. Note that the score is always greater than 0. In the following context, when the formula φ is clear from the context, we simply write $\mathsf{d}(w)$ for the dissatisfaction score of w w.r.t. φ.

Preference model generated from a PQCLTL$_f$ formula Each PQCLTL$_f$ formula φ over a set of atomic propositions $\mathcal{AP}$ induces a preference model $\succeq^\varphi$ over $\Sigma^* = (2^{\mathcal{A}})^*$ such that for any two words $w, w' \in \Sigma^*$, w is preferred to w' with respect to φ, i.e. , $w \succeq^\varphi w'$, if and only if $\mathsf{d}(w, \varphi) \leq \mathsf{d}(w', \varphi)$.

It is easy to prove the following property.

Lemma 1. If $w \succeq^{\varphi} w'$, then one of the following conditions holds: 1. $w \models_k \varphi$ and $w' \not\models \varphi$; or 2. $w \models_n \varphi$, $w' \models_m \varphi$, and $n \leq m$.

The preference model over Σ^* directly translates to a preference model over S^*—the set of finite paths in a labeled MDP such that path $\rho \in S^*$ is preferred to $\rho' \in S^*$ if and only if $\mathsf{d}(L(\rho), \varphi) \leq \mathsf{d}(L(\rho'), \varphi)$. Thus, given a policy in a labeled MDP, we introduce the following measure to evaluate how preferred a policy is with respect to a PQCLTL$_f$ formula.

Definition 7 (Expected Dissatisfaction Score)**.** Let π be a finite-memory, randomized policy for a given MDP, $M^\pi = \langle S^*, P^\pi \rangle$ be its induced Markov chain, and φ be a PQCLTL$_f$ formula. The *expected dissatisfaction score* of π with respect to φ, denoted by $\mathsf{d}(\pi, \varphi)$, is defined

$$\mathsf{d}(\pi, \varphi) = \sum_{\rho \in S^*} \mathbf{Pr}(\rho; M^\pi) \cdot \mathsf{d}(L(\rho), \varphi). \tag{2}$$

We now formally state the probabilistic planning problem:

Problem: Probabilistic Planning with Prioritized Preferences over Temporal Logic Objectives (PPwPPoTLO)

Input: A labeled MDP $M = \langle S, A := \cup_{s \in S} A_s, P, s_0, s_\perp, \mathcal{AP}, L \rangle$ and a PQCLTL$_f$ formula φ.

Output: A policy $\pi \colon S^* \to \mathcal{D}(A)$ that minimizes the expected dissatisfaction score of φ.

4 Optimal planning for PQCLTL$_f$ formulas

We now present a planning algorithm to solve the PPwPPoTLO problem. Our approach consists of two steps: In the first step, we construct an automata-theoretic model for PQCLTL$_f$ formula. In the second step, we show that the optimal policy that minimizes the expected dissatisfaction score of the given formula can be computed by solving a reward-maximizing MDP with augmented states.

4.1 Automata-theoretic modeling of PQCLTL$_f$ formulas

We focus on constructing a computational model for a given PQCTL formula φ. The choice of such a computational model for representing the subclass of PQCLTL$_f$ formulas is a *weighted deterministic finite automaton*.

Definition 8 (Weighted Deterministic Finite Automaton [Droste and Gastin, 2009])**.** A *weighted deterministic finite automaton* is a tuple $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, \mathbf{w} \rangle$, where Q is a finite set of states; $\Sigma \cup \{\ltimes\}$ is a finite set of symbols (alphabet); and $\ltimes$ is a unique symbol representing the end of a string [1]; $\delta : Q \times (\Sigma \cup \{\ltimes\}) \to Q$ is a deterministic transition function; q_0 is the initial state; and $\mathbf{w} : Q \times (\Sigma \cup \{\ltimes\}) \times Q \to \mathbb{R}$ is a weight function that assigns each transition (q, σ, q') to a real value, called the weight of this transition.

[1] In general, one can include $\rtimes$ as the beginning of a finite string and $\ltimes$ as the ending of a finite string. The beginning symbol $\rtimes$ is omitted as it is clear from the context.

Consider a finite word $w = \sigma_0 \sigma_1 \dots \sigma_{n-1} \ltimes$, let $w[i]$ be the i-th symbol of this word. The run ρ generated by word w is $\rho := q_0 \sigma_0 q_1 \dots \sigma_{n-1} q_n$ that satisfies $q_{i+1} = \delta(q_i, w[i])$, for $i = 0, \dots, n-1$. We write $\mathsf{Word}(\rho) = w$ to denote the word associated with the run ρ. The total weight is $\mathbf{w}(\rho) = \sum_{i=0}^{n-1} \mathbf{w}(q_i, w[i], q_{i+1})$.

First, we show how to construct the weighted deterministic finite automaton (WDFA) for an LTL$_f$ φ.

Definition 9 (WDFA for an LTL$_f$ Formula φ)**.** Let $\mathcal{A}_\varphi = \langle Q, \Sigma, \delta, q_0, F \rangle$ be a DFA encoding φ. A WDFA for encoding φ is constructed from $\mathcal{A}_\varphi$ as a tuple

$$\mathcal{A} = \langle Q \cup \{\mathsf{sink}\}, \Sigma \cup \{\ltimes\}, \delta', q_0, \mathbf{w} \rangle$$

in which for each $q \in Q \cup \{\mathsf{sink}\}$ and $\sigma \in \Sigma \cup \{\ltimes\}$,

$$\delta'(q, \sigma) = \begin{cases} \delta(q, \sigma) & \text{if } q \neq \mathsf{sink} \text{ and } \sigma \neq \ltimes \\ \mathsf{sink} & \text{otherwise,} \end{cases} \tag{3}$$

and for each $q, q' \in Q \cup \{\mathsf{sink}\}$ and $\sigma \in \Sigma \cup \{\ltimes\}$,

$$\mathbf{w}(q, \sigma, q') = \begin{cases} 1 & \text{if } q \in F \text{ and } \sigma = \ltimes \text{ and } q' = \mathsf{sink} \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Intuitively, the WDFA $\mathcal{A}$ extends the DFA $\mathcal{A}_\varphi$ with a sink state sink. For any state $s \in S$ of the original DFA $\mathcal{A}_\varphi$, a transition to sink is made with an input symbol $\ltimes$. A weight one is received only if the transition is from an accepting state to the sink state upon reading the ending symbol $\ltimes$.

Lemma 2. Given a WDFA $\mathcal{A}$ for an LTL$_f$ formula φ and a finite run $\rho = q_0 \sigma_0 q_1 \dots \sigma_{n-1} q_n$, if $\mathbf{w}(\rho) = 1$ then $\mathsf{Word}(\rho) \models_1 \varphi$.

The proofs of Lemmas 2, 3 4, and 5 can be found in Appendix A.

Next, we define the construction process of WDFAs for ordered disjunction and prioritized conjunction of PQCLTL$_f$ formulas.

Definition 10 (WDFA for Ordered Disjunction of PQCLTL$_f$ Formulas)**.** Let $\mathcal{A}_i = \langle Q_i \cup \{\mathsf{sink}_i\}, \Sigma \cup \{\ltimes\}, \delta_i, q_{0i}, \mathbf{w}_i \rangle$ for $i = 1, 2$ be two WDFA's that respectively encode two PQCLTL$_f$ formulas φ_1, φ_2. One can constrcut from them, a WDFA for $\varphi_1 \overset{\rightarrow}{\times} \varphi_2$ as a tuple $\mathcal{A} = \langle Q_1 \times Q_2 \cup \{\mathsf{sink}\}, \Sigma \cup \{\ltimes\}, \delta, (q_{01}, q_{02}), \mathbf{w} \rangle$, in which, the transition function is defined as, for any $(q_1, q_2) \in Q_1 \times Q_2$ and $\sigma \in \Sigma \cup \{\ltimes\}$,

$$\delta((q_1, q_2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)) & \text{if } \sigma \neq \ltimes \\ \mathsf{sink} & \text{otherwise,} \end{cases}$$

and the weight function is defined as,

- For any $(q_1, q_2) \in Q_1 \times Q_2$, input $\sigma \in \Sigma$

$$\mathbf{w}((q_1, q_2), \sigma, (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))) = 0$$

- For any $(q_1, q_2) \in Q_1 \times Q_2$, input $\ltimes$,

$$\mathbf{w}((q_1, q_2), \ltimes, \mathsf{sink}) = \begin{cases} \mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) & \text{if } \mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) > 0 \\ \mathbf{w}_2(q_2, \ltimes, \mathsf{sink}) + opt(\varphi_1) & \text{if } \mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) = 0 \\ & \text{and } \mathbf{w}_2(q_2, \ltimes, \mathsf{sink}) > 0 \\ 0 & \text{otherwise,} \end{cases}$$

Lemma 3. Given a WDFA $\mathcal{A}$ for $\varphi = \varphi_1 \overrightarrow{\times} \varphi_2$ and a finite run $\rho = \mathbf{q}_0\sigma_0\mathbf{q}_1 \dots \mathbf{q}_{n-1}\sigma_{n-1}\mathbf{q}_n$, if $\mathbf{w}(\rho) = k$ for a $k > 0$, then $\mathsf{Word}(\rho) \models_k \varphi$, else $\mathsf{Word}(\rho) \not\models \varphi$.

Definition 11 (WDFA for Prioritized Conjunction of PQCLTL$_f$ Formulas)**.** Let $\mathcal{A}_i = \langle Q_i \cup \{\mathsf{sink}_i\}, \Sigma \cup \{\ltimes\}, \delta_i, q_{0_i}, \mathbf{w}_i\rangle$ for $i = 1, 2$ be two WDFA's that respectively encode two PQCLTL$_f$ formulas φ_1, φ_2. One can construct from them, a WDFA for $\varphi_1 \& \varphi_2$ as a tuple $\mathcal{A} = \langle Q_1 \times Q_2 \cup \{\mathsf{sink}\}, \Sigma \cup \{\ltimes\}, \delta, (q_{0_1}, q_{0_2}), \mathbf{w}\rangle$, in which, the transition function is defined as, for any $(q_1, q_2) \in Q_1 \times Q_2$ and $\sigma \in \Sigma \cup \{\ltimes\}$,

$$\delta((q_1, q_2), \sigma) = \begin{cases} (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma)) & \text{if } \sigma \neq \ltimes, \\ \mathsf{sink} & \text{otherwise.} \end{cases}$$

and the weight function is defined as,

- For any $(q_1, q_2) \in Q_1 \times Q_2$, for $\sigma \in \Sigma$,
$$\mathbf{w}((q_1, q_2), \sigma, (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))) = 0$$
- For any $(q_1, q_2) \in Q_1 \times Q_2$, for input $\ltimes$, if $\mathbf{w}_i(q_i, \ltimes, \mathsf{sink}) > 0$ for both $i = 1, 2$, then
$$\mathbf{w}((q_1, q_2), \ltimes, \mathsf{sink}) = \mathbf{w}_2(q_2, \ltimes, \mathsf{sink}) + opt(\varphi_2) \cdot (\mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) - 1),$$
else $\mathbf{w}((q_1, q_2), \ltimes, \mathsf{sink}) = 0$.

Lemma 4. Given a WDFA $\mathcal{A}$ for $\varphi = \varphi_1 \& \varphi_2$, and a finite run $\rho = \mathbf{q}_0\sigma_0\mathbf{q}_1 \dots \mathbf{q}_{n-1}\sigma_{n-1}\mathbf{q}_n$, if $\mathbf{w}(\rho) = k$ for a $k > 0$, then $\mathsf{Word}(\rho) \models_k \varphi$, else $\mathsf{Word}(\rho) \not\models \varphi$.

Given the above construction methods of WDFAs for PQCLTL$_f$ formulas, the WDFA for a more complex PQCLTL$_f$ formulas can be constructed recursively.

Lemma 5. Given a PQCLTL$_f$ formula φ for which $\mathcal{L}(\varphi) \neq \emptyset$ and the constructed WDFA $\mathcal{A}$, the optionality of φ is the maximal weight of all transitions in $\mathcal{A}$. That is

$$opt(\varphi) = \max\{\mathbf{w}(q, a, q') \mid \delta(q, a, q') \text{ is defined.}\}$$

An example to illustrate the construction of WDFAs is given in the Appendix B.

5 Probabilistic Planning to Minimizing Dissatisfaction Score

In this section, we show how to leverage the WDFA for solving Problem PPwPPoTLO. Similar to probabilistic planning with linear temporal logic constraints, a product operation between the labeled MDP and the WDFA allows us to keep track of temporal objectives.

Definition 12 (The product between the labeled MDP and a WDFA)**.** The product of a given WDFA $\mathcal{A} = \langle Q \cup \{\mathsf{sink}\}, \Sigma \cup \{\ltimes\}, \delta, q_0, \mathbf{w}\rangle$ and a terminating labeled MDP $M = \langle S, A := \cup_{s\in S} A_s, P, s_0, s_\perp, \mathcal{AP}, L\rangle$ is an MDP

$$\mathcal{M} = M \otimes \mathcal{A} = (V, A := \bigcup_{v\in V} A_v, \mathcal{P}, v_0, R)$$

in which

- $V = S \times Q$ is the state space,
- A is the set of actions, and for each $v = (s, q) \in V$, $A_v = A_s$ is the v' set of available actions,
- $\mathcal{P}$ is the probabilistic transition function, where for each states $(s, q), (s', q') \in V$ and action $a \in A$,
$$\mathcal{P}((s, q), a, (s', q')) = P(s, a, s') \cdot \mathbf{1}(\delta(q, L(s')) = q')$$
- $v_0 = (s_0, \delta(q_0, L(s_0)))$ is the initial state.
- $R : V \times A \to \mathbb{R}$ is the reward function, where for each $(s, q) \in V$ and $a \in A$, if $a = a_\perp$ and $\mathbf{w}(q, \ltimes, \mathsf{sink}) > 0$, then $R((s, q), a_\perp) = opt(\varphi) - \mathbf{w}(q, \ltimes, \mathsf{sink}) + 1$, else $R((s, q), a) = 0$.

Given a finite run $h = v_0a_0v_1a_1 \dots v_n$ in the product MDP, the total reward is $R(h) = \sum_{i=0}^{n-1} R(v_i, a_i)$. Since a run h in the product MDP corresponds to a run ρ in the original MDP except that each state in ρ is augmented with an automaton state, we use $\mathsf{Proj}_S(h)$ to compute the projection of the run $h = (s_0, q_0)(s_1, q_1) \dots (s_n, q_n) \in V^*$ to a run $s_0s_1s_2 \dots s_n \in S^*$ whose labeling is $L(s_0s_1 \dots s_n) = L(s_0)L(s_1) \dots L(s_n)$. We denote the set of finite runs in $\mathcal{M}$ by $Runs(\mathcal{M})$.

Based on the reward function, the expected total reward of a nonstationary policy $\pi : V^* \to \mathcal{D}(A)$ for an initial state $v \in V$ is defined as

$$J_\pi(v) = \lim_{N\to\infty} \sup J_{\pi,N}(v),$$

with $J_{\pi,N}(v)$ being the expected N-stage reward of π for state v:

$$J_{\pi,N}(v) = E\left[\sum_{t=0}^{N} R(V_k, \pi(V_0 \dots V_k)) \mid V_0 = v\right],$$

where V_k is the state at time k. The expectation is with respect to the probability distribution of paths in Markov chain $\mathcal{M}^\pi$.

Lemma 6. For any policy $\pi : V^* \to \mathcal{D}(A)$ of the product MDP $\mathcal{M}$, for any $v \in V$, $J_\pi(v) < \infty$.

The proof is in Appendix A.

The optimal value function is defined to be

$$J^*(v) = \arg\max_\pi J_\pi(v).$$

For optimal planning to maximize the total reward, $J^*(v)$ can be attained by a Markovian policy [Puterman, 2014]. Therefore, in the following, we only consider Markovian policies. We also consider the Bellman operator T, defined by

$$TJ(v) = \max_{\pi\in\Pi} \sum_{a\in A_v} [R(v, a)\pi(a \mid v) + \sum_{v'\in V} \mathcal{P}(v'|v, a)\pi(a \mid v)J(v')],$$

and the optimal value function satisfies $TJ^* = J^*$.

Among all the Markovian policies for the product MDP, we consider only the *proper* ones.

Definition 13 (Extended from [Bertsekas and Yu, 2013])**.** A policy π for the MDP $\mathcal{M}$ is *proper* if it guarantees that the sink state $(s_\perp, \mathsf{sink})$ will be reached with probability one.

Lemma 7. The optimal value $J^*(v)$ for any $v \in V$ can be obtained by a proper, Markovian policy of product MDP $\mathcal{M}$.

The proof is in Appendix A.

Thus, to search an optimal policy, we need to consider only proper, Markovian policies. We now relate the reward maximizing problem in the product MDP to the planning objective of minimizing the expected dissatisfaction score.

Lemma 8. For each path $h = (s_0, q_0)(s_1, q_1)\dots(s_n, q_n) \in V^*$, it holds that,

$$\mathsf{d}(L(\mathsf{Proj}_S(h)), \varphi) = \begin{cases} 1 - \frac{R(h)}{opt(\varphi)+1} & \text{if } R(h) > 0, \\ 1 & \text{if } R(h) = 0. \end{cases}$$

The proof is in Appendix A.

Theorem 1. Let $\pi : V \to \mathcal{D}(A)$ be a policy for the product MDP $\mathcal{M}$. Construct from π, a policy $\pi' : S^* \to \mathcal{D}(A)$ for M such that for each $\rho : s_0 s_1 \cdots s_n \in S^*$, $\pi'(\rho) = \pi((s_n, \delta(q_0, L(\rho))))$. If π is an optimal policy for $\mathcal{M}$, then π' is an optimal policy that minimizes the expected dissatisfaction score, i.e. , the solution to PPwPPoTLO.

Proof. We establish a connection between the expected dissatisfaction score of π' and the value of π. First, we use (2) to expand the expected dissatisfaction score of π':

$$\mathsf{d}(\pi', \varphi) = \sum_{\rho \in S^*} \mathbf{Pr}(\rho; M^{\pi'}) \cdot \mathsf{d}(L(\rho), \varphi) \tag{5}$$

Next, we expand the value of π.

$$\begin{aligned} J_\pi(v_0) &= \sum_{h \in V^*} \mathbf{Pr}(h; \mathcal{M}^\pi) \cdot R(h) \\ &= \sum_{h \in V^* : R(h) = 0} \mathbf{Pr}(h; \mathcal{M}^\pi) \cdot 0 \\ &+ \sum_{h \in V^* : R(h) \neq 0} \mathbf{Pr}(h; \mathcal{M}^\pi) \cdot R(h) \end{aligned} \tag{6}$$

Using the result of Lemma 8, we write this summation as:

$$\begin{aligned} J_\pi(v_0) = &\sum_{h \in V^* : R(h) \neq 0} \mathbf{Pr}(h; \mathcal{M}^\pi) \cdot (1 - \mathsf{d}(L(\mathsf{Proj}_S(h)))) \\ &\cdot (opt(\varphi) + 1)) \\ &[\text{ replace } opt(\varphi)+1) \text{ by } K, \mathbf{Pr}(h; \mathcal{M}^\pi) \text{ by } \mathbf{Pr}^\pi(h).] \\ =&(K \cdot \sum_{h \in V^* : R(h) \neq 0} \mathbf{Pr}^\pi(h) \\ &- K \cdot \sum_{h \in V^* : R(h) \neq 0} \mathbf{Pr}^\pi(h) \mathsf{d}(L(\mathsf{Proj}_S(h))) \\ =&K \cdot \sum_{h \in V^*} \mathbf{Pr}^\pi(h) - K \cdot \sum_{h \in V^* : R(h) = 0} \mathbf{Pr}^\pi(h) \\ &- K \cdot \sum_{h \in V^* : R(h) \neq 0} \mathbf{Pr}^\pi(h) \mathsf{d}(L(\mathsf{Proj}_S(h))) \qquad (7) \\ =&(K - K \cdot \sum_{h \in V^* : R(h) = 0} \mathbf{Pr}^\pi(h) \cdot 1 \\ &- K \cdot \sum_{h \in V^* : R(h) \neq 0} \mathbf{Pr}^\pi(h) \mathsf{d}(L(\mathsf{Proj}_S(h)) \qquad (8) \\ =&K - K \cdot \sum_{h \in V^*} \mathbf{Pr}^\pi(h) \cdot \mathsf{d}(L(\mathsf{Proj}_S(h))) \qquad (9) \end{aligned}$$

From (7) to (8), we use the probability axiom that $\sum_{h \in V^*} \mathbf{Pr}(h; \mathcal{M}^\pi) = 1$. From (8) to (9), we use Lemma 8 that if $R(h) = 0$ then $\mathsf{d}(L(\mathsf{Proj}_S(h))) = 1$. Thus, relating (9) and (5), we have

$$J_\pi(v_0) = K - K \cdot \mathsf{d}(\pi', \varphi), \tag{10}$$

and therefore $\operatorname{argmax}_\pi J_\pi(v_0) = \operatorname{argmin}_{\pi'} \mathsf{d}(\pi', \varphi)$, that is, a policy π that maximizes J yields a policy π' that minimizes the dissatisfaction score d. □

6 Complexity Analysis

The first step of the algorithm constructs a WDFA that encodes φ. The constructed DFA from LTL$_f$ formulas is double-exponential in the size of the formulas in the worst case [Wolper, 2001; De Giacomo and Favorito, 2021]. However, in practice this translation is tractable for commonly seen LTL$_f$ formulas in robotic planning. The construction of automata for ordered disjunction $\varphi_1 \vec{\times} \varphi_2$ and prioritized conjunction $\varphi_1 \& \varphi_2$ using Def. 10 and Def. 11, respectively, takes a polynomial time to the sizes of the WDFA's for subformulas φ_1 and φ_2. Constructing the product MDP $\mathcal{M}$ takes a polynomial time to the size of the WDFA and the original MDP. And computing an optimal policy for $\mathcal{M}$ takes a time polynomial in the size of the product MDP $\mathcal{M}$, using standard techniques (value/policy iteration or linear programming).

7 Experiment

We show the efficacy of the proposed algorithm using several examples of probabilistic robotic motion planning. [2]

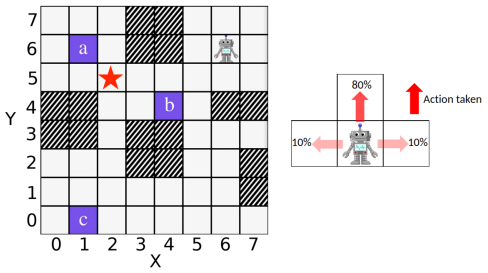


Figure 1: A 8×8 stochastic gridworld g_1 (without the red star) and the transition probabilities when an action "N" is taken. The red star is an additional hole introduced in gridworld g_2.

Consider a small stochastic gridworld g_1 shown in Fig. 1. For each state $s \in S$, the robot has four actions: "N", "W",

[2] All experiments are executed on an Ubuntu 20.04 machine with AMD Ryzen 9 5900X CPU and 32 GB RAM. We use the Gurobi solver for planning in MDP. The computational times of solving the optimal planning problem for any 8×8 gridworlds with different formulas are no more than 0.1 seconds. The code can be found in the supplementary file.

"S", "E". After taking an action from a state, the robot transits to the *intended* cell with probability 0.8 and slips to *unintended* cells with probability 0.1. If the robot takes an action and reaches the boundary wall, then it stays in the original cell. The initial state of the robot is $(6, 6)$. The shaded areas denote holes. Once the robot enters holes, it gets stuck. Regions of interest are labeled a, b, and c. Accordingly, $\mathcal{AP} = \{a, b, c\}$. Each of these atomic propositions holds at a time instant when the robot is in the region labeled by the corresponding atomic proposition.

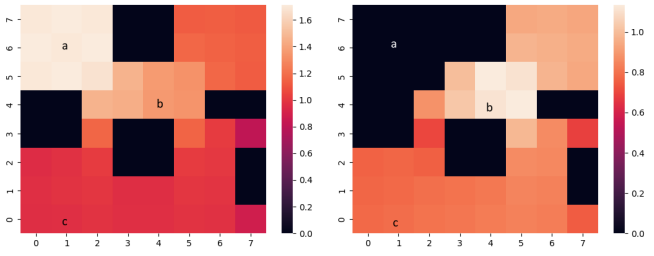
Given the set $\mathcal{AP}$ of atomic propositions, we consider the following preference formula (see Appendix B for the WDFA.)

$$\Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c).$$

We computed the optimal policy π_* that minimizes the expected dissatisfaction score. To see the difference of ordered disjunction and regular disjunction, we also compute a optimal policy that maximizes the probability of satisfying formula $\Diamond b \vee (\Diamond a \vee \Diamond c)$. We denote this policy as $\pi_\vee$. We plot the optimal values for different initial states in Fig 2b. Then we perform policy evaluation of π_* and $\pi_\vee$ against $\Diamond b$ and $\Diamond a \vee \Diamond c$, separately. The probabilites of satisfying formula ϕ for $\phi \in \{\Diamond b, \Diamond a \vee \Diamond c\}$ for different initial states are shown in Fig. 2c, 2d, 2e and 2f. Comparing Fig. 2c against Fig. 2e for the formula $\Diamond b$, π_* achieves higher values in the most areas of the gridworld, especially at the top left corner. On the other side, comparing Fig. 2d against Fig. 2f, we can see that $\pi_\vee$ achieves higher probability of satisfying $\Diamond a \vee \Diamond c$ than that of policy π_* in most areas of the gridworld, especially at the top right corner. This comparison indicates that when $\Diamond b$ is preferred to $\Diamond a \vee \Diamond c$, the preference-based policy gravitates towards satisfying $\Diamond b$.

Next, we consider the following formula that has prioritized conjunction and nested ordered disjunctions: $\varphi_3 = \varphi_1 \& \varphi_2$, where $\varphi_1 = \Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c)$ and $\varphi_2 = \Diamond (a \wedge \Diamond (b \wedge \Diamond c)) \overset{\rightarrow}{\times} \Diamond (a \wedge \Diamond c) \vee \Diamond (b \wedge \Diamond c)$. This task formula describes that the system needs to satisfy φ_1 and φ_2 both, with φ_1 having a higher priority than φ_2.

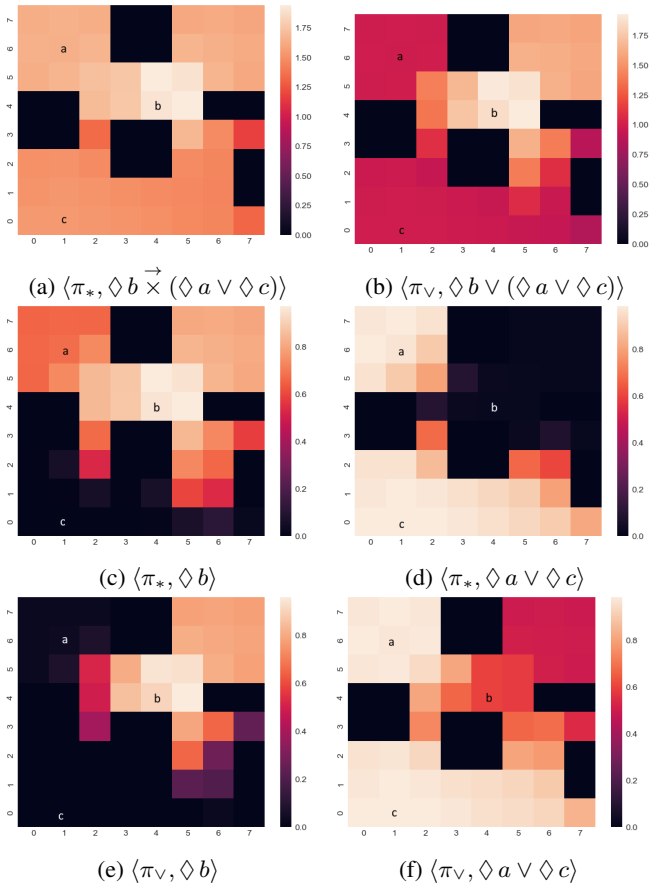
For this case, we consider an additional gridworld g_2 which includes an additional hole at the position $(2, 5)$, which blocks the access to a. Given the formula φ_3, we compute the optimal policies $\pi_\ddagger$ when region a is accessible and $\pi_\ddagger^-$ when region a is inaccessible.



(a) (b)

Figure 3: (a) The value $J((\cdot, \mathbf{q}_0); \langle \pi_\ddagger, \varphi_3 \rangle)$ in gridworld g_1. (b) The value $J((\cdot, \mathbf{q}_0); \langle \pi_\ddagger^-, \varphi_3 \rangle)$ in gridworld g_2.

We plot the heatmaps of φ_3 for these two gridworlds in Fig. 3a and 3b. The following observation is made: When a is accessible, starting from the upper left corner, the agent receives higher values with the optimal policy. But if a is not accessible, the upper left corner states have values zero. This is because the formulas φ_2 cannot be satisfied as the agent cannot reach region c when starting from the upper left corner. Therefore, φ_3 is not satisfiable. The state values under the optimal policy given a accessible are higher than the state values when a is not accessible, indicating the agent can achieve a more preferred outcome in the gridworld g_1.



(a) $\langle \pi_*, \Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c) \rangle$ (b) $\langle \pi_\vee, \Diamond b \vee (\Diamond a \vee \Diamond c) \rangle$

(c) $\langle \pi_*, \Diamond b \rangle$ (d) $\langle \pi_*, \Diamond a \vee \Diamond c \rangle$

(e) $\langle \pi_\vee, \Diamond b \rangle$ (f) $\langle \pi_\vee, \Diamond a \vee \Diamond c \rangle$

Figure 2: Each subfigure with the subcaption $\langle \pi, \phi \rangle$ is the value $J((\cdot, \mathbf{q}_0); \langle \pi, \phi \rangle)$ of policy evaluation of policy π given the formula ϕ starting from different initial states in gridworld g_1.

8 Conclusion

In this paper, we introduced a new specification language, termed prioritized qualitative choice linear temporal logic on finite traces (PQCLTL$_f$), for compactly specifying a temporal goal along with the user's preferences on sub-goals. We presented an automatic translation from this language to weighted deterministic finite automaton. We used this translation in solving the problem of computing a policy that minimizes the expected dissatisfaction score of a given PQCTL formula in a stochastic environment modeled by an MDP. By bridging the gap between preferences in AI and temporal logic planning, this work enables future study that incorporates preference elicitation and learning from positive/negative data and adaptive planning in sequential

decision-making problems.

References

[Benferhat and Sedki, 2007] Salem Benferhat and Karima Sedki. A revised qualitative choice logic for handling prioritized preferences. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 635–647. Springer, 2007.

[Benferhat and Sedki, 2008] Salem Benferhat and Karima Sedki. Two alternatives for handling preferences in qualitative choice logic. *Fuzzy Sets and Systems*, 159(15):1889–1912, August 2008.

[Bertsekas and Yu, 2013] Dimitri P Bertsekas and Huizhen Yu. Stochastic shortest path problems under weak conditions. *Lab. for Information and Decision Systems Report LIDS-P-2909, MIT*, 2013.

[Bradley *et al.*, 2021] Christopher Bradley, Adam Pacheck, Gregory J Stein, Sebastian Castro, Hadas Kress-Gazit, and Nicholas Roy. Learning and planning for temporally extended tasks in unknown environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4830–4836. IEEE, 2021.

[Brewka *et al.*, 2004] Gerhard Brewka, Salem Benferhat, and Daniel Le Berre. Qualitative choice logic. *Artificial Intelligence*, 157(1):203–237, August 2004.

[Cai *et al.*, 2020] Mingyu Cai, Hao Peng, Zhijun Li, Hongbo Gao, and Zhen Kan. Receding horizon control-based motion planning with partially infeasible ltl constraints. *IEEE Control Systems Letters*, 5(4):1279–1284, 2020.

[Camacho *et al.*, 2017] Alberto Camacho, Eleni Triantafillou, Christian Muise, Jorge A Baier, and Sheila A McIlraith. Non-deterministic planning with temporally extended goals: Ltl over finite and infinite traces. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[De Giacomo and Favorito, 2021] Giuseppe De Giacomo and Marco Favorito. Compositional approach to translate ltlf/ldlf into deterministic finite automata. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 31, pages 122–130, 2021.

[De Giacomo and Vardi, 2013] Giuseppe De Giacomo and Moshe Y Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *IJCAI'13 Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 854–860. Association for Computing Machinery, 2013.

[De Giacomo and Vardi, 2015] Giuseppe De Giacomo and Moshe Vardi. Synthesis for ltl and ldl on finite traces. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[Droste and Gastin, 2009] Manfred Droste and Paul Gastin. Weighted automata and weighted logics. In *Handbook of weighted automata*, pages 175–211. Springer, 2009.

[He *et al.*, 2015] Keliang He, Morteza Lahijanian, Lydia E Kavraki, and Moshe Y Vardi. Towards manipulation planning with temporal logic specifications. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 346–352. IEEE, 2015.

[He *et al.*, 2020] Binghan He, Jaemin Lee, Ufuk Topcu, and Luis Sentis. Bp-rrt: Barrier pair synthesis for temporal logic motion planning. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1404–1409. IEEE, 2020.

[Hekmatnejad and Fainekos, 2018] Mohammad Hekmatnejad and Georgios Fainekos. Optimal multi-valued ltl planning for systems with access right levels. In *2018 Annual American Control Conference (ACC)*, pages 2363–2370. IEEE, 2018.

[Kantaros *et al.*, 2022] Yiannis Kantaros, Samarth Kalluraya, Qi Jin, and George J Pappas. Perception-based temporal logic planning in uncertain semantic maps. *IEEE Transactions on Robotics*, 2022.

[Kasenberg *et al.*, 2020] Daniel Kasenberg, Ravenna Thielstrom, and Matthias Scheutz. Generating explanations for temporal logic planner decisions. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 449–458, 2020.

[Lacerda *et al.*, 2015] Bruno Lacerda, David Parker, and Nick Hawes. Optimal policy generation for partially satisfiable co-safe ltl specifications. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[Lahijanian and Kwiatkowska, 2016] Morteza Lahijanian and Marta Kwiatkowska. Specification revision for Markov decision processes with optimal trade-off. In *Proc. 55th Conference on Decision and Control (CDC'16)*, pages 7411–7418, 2016.

[Li *et al.*, 2020] Meilun Li, Andrea Turrini, Ernst Moritz Hahn, Zhikun She, and Lijun Zhang. Probabilistic preference planning problem for markov decision processes. *IEEE transactions on software engineering*, 2020.

[Li *et al.*, 2021] Shen Li, Daehyung Park, Yoonchang Sung, Julie A Shah, and Nicholas Roy. Reactive task and motion planning under temporal logic specifications. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12618–12624. IEEE, 2021.

[Mallett *et al.*, 2021] Ian Mallett, Sylvie Thiébaux, and Felipe Trevizan. Progression heuristics for planning with probabilistic ltl constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11870–11879, 2021.

[Mehdipour *et al.*, 2021] Noushin Mehdipour, Cristian-Ioan Vasile, and Calin Belta. Specifying User Preferences Using Weighted Signal Temporal Logic. *IEEE Control Systems Letters*, 5(6):2006–2011, December 2021.

[Pnueli, 1981] Amir Pnueli. The temporal semantics of concurrent programs. *Theoretical computer science*, 13(1):45–60, 1981.

[Puterman, 2014] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[Rahmani and O'Kane, 2019] Hazhar Rahmani and Jason M O'Kane. Optimal temporal logic planning with cascading soft constraints. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2524–2531. IEEE, 2019.

[Rahmani and O'Kane, 2020] Hazhar Rahmani and Jason M O'Kane. What to do when you can't do it all: Temporal logic planning with soft temporal logic constraints. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6619–6626. IEEE, 2020.

[Tumova *et al.*, 2013a] Jana Tumova, Luis I Reyes Castro, Sertac Karaman, Emilio Frazzoli, and Daniela Rus. Minimum-violation ltl planning with conflicting specifications. In *American Control Conference*, pages 200–205. IEEE, 2013.

[Tumova *et al.*, 2013b] Jana Tumova, Gavin C Hall, Sertac Karaman, Emilio Frazzoli, and Daniela Rus. Least-violating control strategy synthesis with safety rules. In *Proc. Int. Conf. on Hybrid systems: Computation and control*, 2013.

[Vasile *et al.*, 2017] Cristian-Ioan Vasile, Jana Tumova, Sertac Karaman, Calin Belta, and Daniela Rus. Minimum-violation scltl motion planning for mobility-on-demand. pages 1481–1488. IEEE, 2017.

[Vasile *et al.*, 2020] Cristian Ioan Vasile, Xiao Li, and Calin Belta. Reactive sampling-based path planning with temporal logic specifications. *The International Journal of Robotics Research*, 39(8):1002–1028, 2020.

[Wang *et al.*, 2020] Yu Wang, Siddhartha Nalluri, and Miroslav Pajic. Hyperproperties for robotics: Planning via hyperltl. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8462–8468. IEEE, 2020.

[Wolper, 2001] Pierre Wolper. Constructing Automata from Temporal Logic Formulas: A Tutorial. In G. Goos, J. Hartmanis, J. van Leeuwen, Ed Brinksma, Holger Hermanns, and Joost-Pieter Katoen, editors, *Lectures on Formal Methods and PerformanceAnalysis*, volume 2090, pages 261–277. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[Wongpiromsarn *et al.*, 2021] Tichakorn Wongpiromsarn, Konstantin Slutsky, Emilio Frazzoli, and Ufuk Topcu. Minimum-violation planning for autonomous systems: Theoretical and practical considerations. In *2021 American Control Conference*, 2021. submitted.

[Yang *et al.*, 2020] Yuanjiang Yang, Xiang Yin, and Shaoyuan Li. A distributed framework for multi-robot task planning with temporal logic specifications. In *2020 IEEE 16th International Conference on Control & Automation (ICCA)*, pages 570–575. IEEE, 2020.

[Zhao *et al.*, 2022] Jiawei Zhao, Xiang Yin, and Shaoyuan Li. Temporal logic robot task planning with active acquisition of information. In *2022 IEEE 17th International Conference on Control & Automation (ICCA)*, pages 1014–1020. IEEE, 2022.

[Zhou *et al.*, 2022] Xiaoyi Zhou, Tiange Yang, Yuanyuan Zou, Shaoyuan Li, and Hao Fang. Multiple sub-formulae cooperative control for multi-agent systems under conflicting signal temporal logic tasks. *IEEE Transactions on Industrial Electronics*, 2022.

A Proofs

Proof of Lemma 2

Proof. The run ρ has a weight 1 if and only if $q_{n-1} \in F$, $\sigma_{n-1} = \ltimes$, and $q_n = \mathsf{sink}$. Due to the acceptance condition for DFAs, the word $\mathsf{Word}(\rho) = \sigma_0\sigma_1 \dots \sigma_{n-1}$ is accepted and thus satisfies the LTL_f formula φ. □

Proof of Lemma 3

Proof. Given the penultimate state $\mathbf{q}_{n-1} = (q_1, q_2)$, if $\mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) = \mathbf{w}_2(q_2, \ltimes, \mathsf{sink}) = 0$, then $\mathsf{Word}(\rho) \not\models \varphi$, meaning it does not satisfy φ. If $\mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) = 0$, but $\mathbf{w}_2(q_2, \ltimes, \mathsf{sink}) > 0$, then $\mathsf{Word}(\rho)$ satisfies φ_2 to a positive degree but does not satisfy φ_1. The satisfaction degree w.r.t. $\varphi_1 \overset{\rightarrow}{\times} \varphi_2$ is the sum of the satisfaction degree w.r.t. φ_2 and the optionality of φ_1. Else, if $\mathbf{w}_1(q_1, \ltimes, \mathsf{sink}) > 0$, then the satisfaction degree w.r.t. $\varphi_1 \overset{\rightarrow}{\times} \varphi_2$ is the satisfaction degree w.r.t. φ_1. □

Proof of Lemma 4

Proof. The proof is by construction and similar to the proof of Lemma 3. Thus, it is omitted. □

Proof of Lemma 5

Proof. The property can be shown based on the recursive definition. First, it is clear that if the PQCLTL_f formula is an LTL_f formula, then the optionality is one and the maximal weight of all defined transitions is one. Consider two PQCLTL_f formulas φ_1, φ_2, and their corresponding WDFAs $\mathcal{A} = \langle Q_i \cup \{\mathsf{sink}\}, \Sigma \cup \{\ltimes\}, \delta_i, q_{0i}, \mathbf{w}_i \rangle$ that satisfies $opt(\varphi_i) = \max\{\mathbf{w}_i(q, a, q') \mid \delta_i(q, a, q') \text{ is defined.}\}$.

In the WDFA of the ordered disjunction $\varphi_1 \overset{\rightarrow}{\times} \varphi_2$, the maximal weight by construction is $\max\{\mathbf{w}_2(q_2, \ltimes, \mathsf{sink}) + opt(\varphi_1)\} = \max\{\mathbf{w}_2(q_2, \ltimes, \mathsf{sink})\} + opt(\varphi_1) = opt(\varphi_2) + opt(\varphi_1)$, which is consistent with Def. 4.

In the WDFA of the prioritized conjunction $\varphi_1 \& \varphi_2$, the maximal weight by construction is $\max_{i,j}\{opt(\varphi_2) \times (i-1) + j\}$ where $0 < i \leq \max\{\mathbf{w}_1(q_1, \ltimes, \mathsf{sink})\} = opt(\varphi_1)$ and $0 < j \leq \max\{\mathbf{w}_2(q_2, \ltimes, \mathsf{sink})\} = opt(\varphi_2)$. Therefore, $\max_{i,j}\{opt(\varphi_2) \times (i-1) + j\} = opt(\varphi_2) \times (opt(\varphi_1) - 1) + opt(\varphi_1) = opt(\varphi_2) \times opt(\varphi_1)$. This is again consistent with Def. 4. □

Proof of Lemma 6.

Proof. A finite run $\rho = v_0 a_0 v_1 a_1 \dots v_n$ receives a nonzero reward only if there exists $0 \leq k \leq n$, $v_k = (s_\perp, \mathsf{sink})$, and for all $j \leq k$, $v_j \neq (s_\perp, \mathsf{sink})$. The total reward of ρ is upper bounded by $opt(\varphi)$. Therefore, for any policy π and any state v, the limit of $J_{\pi,N}(v)$ as $N \to \infty$ exists and is upper bounded by $opt(\varphi)$. □

Proof of Lemma 7.

Proof. We show that for every improper, optimal Markovian policy, there is a proper, Markovian policy that obtains the same value. Consider an improper, optimal Markovian policy $\pi^\dagger$ under which there is an infinite run. Since the reward is only obtained by reaching state $(s_\perp, \mathsf{sink})$, the infinite run h will have a reward of zero. Thus, a proper policy π^* that has the same value $J_{\pi^\dagger}(v) = J_{\pi^*}(v)$ can be constructed by copying $\pi^\dagger$ for all finite runs. For all infinite runs, π^* is obtained from $\pi^\dagger$ by terminating at any state with a zero reward. □

Proof of Lemma 8.

Proof. Prove by construction. For the first case, let us recall $R(h) = \sum_i^{n-1} R(v_i, a_i)$. If $L(\mathsf{Proj}_S(h)) \models_k \varphi$ for some $k > 0$, then $R(h) = opt(\varphi) - \mathbf{w}(q_n, \ltimes, \mathsf{sink}) + 1$. Plug in $R(h)$, and we have $\mathsf{d}(L(\mathsf{Proj}_S(h)), \varphi) = 1 - \frac{R(h)}{opt(\varphi)+1} = \frac{\mathbf{w}(q_n, \ltimes, \mathsf{sink})}{opt(\varphi)+1}$, complying with Lemma 2, 3, and 4. For the second case, if $L(\mathsf{Proj}_S(h)) \not\models \varphi$, then $R(h = 0)$, then $\mathsf{d}(L(\mathsf{Proj}_S(h)) = 1$, complying with Def. 6. □

B Example of weighted automata construction

We illustrate the construction of WDFA using an example.

Example 1. Given two LTL_f formulas $\Diamond b$ and $\Diamond a \vee \Diamond c$ and a PQCLTL_f formula $\Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c)$, reading "if possible, eventually satisfy b, and if not possible, eventually satisfy a or c." The WDFAs for the LTL_f formulas are shown in Fig. 4a and 4b, and the WDFA is shown in Fig. 4c. For clarity, we use propositional logic formulas instead of $2^{\mathcal{AP}}$ as the symbols for the transitions. For example, $b \wedge (a \vee c) : 0$ stands for $\{b, a\} : 0, \{b, c\} : 0$, and $\{b, a, c\} : 0$. From Fig. 4c, we see that the weight transits from $(0, 1)$ to sink is 2, that is because by triggering that transition the satisfied formula $\Diamond a \vee \Diamond c$ is less preferred.

In Table 1 we list the satisfaction degrees given different words. From the dissatisfaction scores, we have $w_1 \succeq^\phi w_2 \succeq^\phi w_3$, where $\phi = \Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c)$.

Table 1: Dissatisfaction Scores for Words *w.r.t.* $\Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c)$

Words	Word descriptions	$w \models_i \Diamond b$	$w \models_j \Diamond a \vee \Diamond c$	$w \models_k \Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c)$	$\mathsf{d}(w, \Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c))$
w_1	$\{b\}\{a\}\ltimes$	1	1	1	$1/3$
w_2	$\emptyset\emptyset\{a\}\ltimes$	$\not\models$	1	2	$2/3$
w_3	$\emptyset\emptyset\ltimes$	$\not\models$	$\not\models$	$\not\models$	1

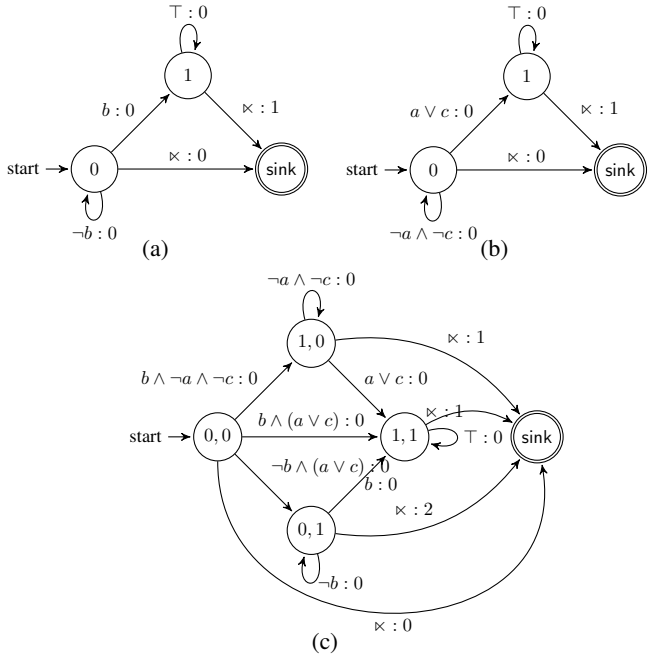


Figure 4: (a) The WDFA accepting the formula $\Diamond b$. (b) The WDFA accepting the formula $\Diamond a \vee \Diamond c$. (c) The WDFA accepting the formula $\Diamond b \overset{\rightarrow}{\times} (\Diamond a \vee \Diamond c)$.