# Multi-Objective Policy Generation for Mobile Robots Under Probabilistic Time-Bounded Guarantees

**Bruno Lacerda, David Parker and Nick Hawes**
School of Computer Science, University of Birmingham
Birmingham, United Kingdom
{b.lacerda, d.a.parker, n.a.hawes}@cs.bham.ac.uk

## Abstract

We present a methodology for the generation of mobile robot controllers which offer probabilistic time-bounded guarantees on successful task completion, whilst also trying to satisfy soft goals. The approach is based on a stochastic model of the robot's environment and action execution times, a set of soft goals, and a formal task specification in co-safe linear temporal logic, which are analysed using multi-objective model checking techniques for Markov decision processes. For efficiency, we propose a novel two-step approach. First, we explore policies on the Pareto front for minimising expected task execution time whilst optimising the achievement of soft goals. Then, we use this to prune a model with more detailed timing information, yielding a time-dependent policy for which more fine-grained probabilistic guarantees can be provided. We illustrate and evaluate the generation of policies on a delivery task in a care home scenario, where the robot also tries to engage in entertainment activities with the patients.

## Introduction

Decision making under uncertainty has been widely used for the control of systems where the state evolution and outcome of actions is non-deterministic. In particular, models such as Markov decision processes (MDPs) have been successfully used to control robot systems. In this paper, we propose a multi-objective sequential decision making technique to generate policies for MDPs that combine the achievement of a primary task with the execution of as many secondary tasks as possible, while maintaining a high probability of achieving the primary task within a user-defined time-bound. Given a primary task specified in co-safe linear temporal logic (LTL), to be executed within a deadline, and a utility function for a set of secondary tasks, we present a method that tries to gather as much utility from the secondary tasks as possible, whilst maintaining a guaranteed threshold on the probability of successful, timely primary task completion. Time bounds are important in robotic applications, as they allow robots' long-term behaviour to be scheduled reliably against user requirements and temporal environmental variations. By taking such time bounds into account, we are able to generate policies that can reason about elapsed time during execution, allowing for more complex robot behaviour that takes into account the available time for primary task completion when deciding whether to execute secondary tasks.

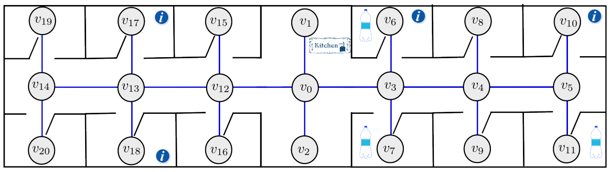

Figure 1: A care home environment.

**Example 1.** The scenario that will be used to motivate our approach, and illustrate its main concepts, is as follows. Consider a care home environment, as shown in Fig. 1, where the central area ($v_0 - v_2$) represents a common entrance and kitchen area, the right-hand area ($v_3 - v_{11}$) represents offices for administrative staff, and the left-hand area ($v_{12} - v_{20}$) represents patient rooms. The robot starts in the middle of the central area ($v_0$), and can navigate between different locations in the environment according to the depicted navigation graph. Furthermore, at certain times the central area is overcrowded (e.g., due to people having lunch). Thus, there is uncertainty on the expected time to navigate from/to $v_0$. In the kitchen area ($v_1$), the robot can retrieve water bottles. Given the high uncertainty of grasping, the time needed to do so is also uncertain. Finally, the robot cannot carry more than 2 bottles at a time.

As part of its routine, the robot has to execute a water delivery task, navigating to a subset $O$ of the staff offices, and asking if the person there wants water. If so, the robot needs to bring it to them. We also want the robot to be ready to carry other objects once the task is finished, so we require that the robot should not be holding bottles when it finishes the task. We provide a bound of $d \in \mathbb{N}_{>0}$ seconds for the task such that we can provide a guarantee on the latest time when people that asked for water will have it delivered.

Apart from being in charge of water delivery, the robot can also engage in interactions with the patients and staff in its spare time. Such interactions can be, for example, providing the lunch menu, or showing an entertainment video on its screen. Thus, in certain rooms $I$ where the occupant is interested in the robot, it can execute an $interact$ action.

In order to accurately model this scenario, one needs to take several factors into account, such as:

- As stated above, there is uncertainty on the outcome and timing of certain actions. In order to effectively cope with these uncertainties, one needs to appropriately include them in the planning model. Thus, we use *timed* MDPs, which explicitly model probabilistic action outcomes and discrete action durations. We use discrete action duration distributions in order to reduce complexity of the solution method, when compared to the use of continuous distributions. Note that one can choose finer discretisations in order to keep the accuracy of the model high. Finally, while the illustrative example presented in this paper has hand written distributions, for real robot deployments one can use machine learning techniques to learn the probability distributions such as those associated with the uncertainties above.
- The natural language description of the primary task of water delivery is reasonably complex, requiring one to take different responses from the staff into account, and also requiring that the robot finishes the task holding no bottles. Specifying such a task using a reward structure, as is more common for MDP planning, is far from straightforward. However, this can represented intuitively using co-safe LTL, a flexible formal specification language for which there are known techniques for automatic verification and policy generation. With co-safe LTL, one can not only specify "classical" planning goals of reaching a specific state, but also, for example, state that a particular sequence of states must be achieved, or that a set of "forbidden" states must be avoided.
- The (secondary) interaction tasks, which should only be executed if the robot has enough time to do so, are related to the execution of a specific behaviour at a given location. These can be more naturally thought of as preferences, and require an associated utility function to guide the choice between them. In this case, such utility can be a measure of how much the different people appreciate interacting with the robot (e.g., obtained from a questionnaire), or can be calculated when building a new problem instance, as a function of how long since the last time a specific person was engaged by the robot. We therefore use a reward structure over state-action pairs to encode this utility function. In other scenarios these preferences can also drawn from structured task knowledge, e.g. information gain for exploration (Korein, Coltin, and Veloso 2014; Santos et al. 2016), or progression towards an implicitly defined co-safe LTL secondary goal (Lacerda, Parker, and Hawes 2015b), or extra tasks or actions that the robot should execute, if there is enough time to do so.

In order to achieve precise guarantees on the behaviour of the policies generated to control our robots, we apply formal verification techniques. In particular we use multi-objective probabilistic model checking, to generate policies that capture the trade-off between timely task completion and gathering of secondary task reward. For the former, most existing approaches measure the *expected time* for task completion, but the usefulness of this can be limited if there is a large variance in execution time. Furthermore, expected time minimisation does not allow reasoning during execution about how long certain actions have taken, or adapting behaviour taking into account how much time for task satisfaction still remains. In this paper, we work with *probabilistic time-bounded* guarantees which give precise information about whether a task can be completed within a deadline, thus allowing robot behaviour to be more accurately scheduled. Such guarantees also allow for adaptive, time-aware behaviour that takes into account how much time for task satisfaction still remains. However, computing time-bounded properties is known to be computationally difficult, because the states of the MDP need to be augmented with the elapsed execution time, limiting feasibility for larger sized problems.

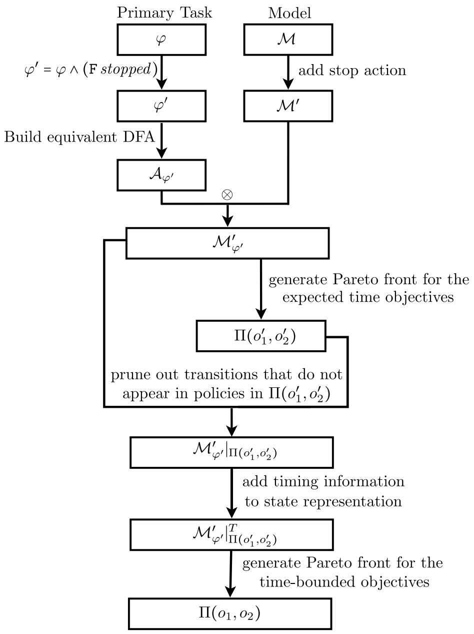
Primary Task
Model
$\varphi$
$\mathcal{M}$
$\varphi' = \varphi \wedge (\mathtt{F}\, stopped)$
add stop action
$\varphi'$
$\mathcal{M}'$
Build equivalent DFA
$\mathcal{A}_{\varphi'}$
$\otimes$
$\mathcal{M}'_{\varphi'}$
generate Pareto front for the expected time objectives
$\Pi(o'_1, o'_2)$
prune out transitions that do not appear in policies in $\Pi(o'_1, o'_2)$
$\mathcal{M}'_{\varphi'}|_{\Pi(o'_1,o'_2)}$
add timing information to state representation
$\mathcal{M}'_{\varphi'}|^{T}_{\Pi(o'_1,o'_2)}$
generate Pareto front for the time-bounded objectives
$\Pi(o_1, o_2)$

Figure 2: Depiction of the overall approach. The primary task is encoded in objectives $o_1$ and $o'_1$ , and the secondary tasks are encoded in objectives $o_2$ and $o'_2$.

In this paper we present a two-step approximation method to tackle the state space explosion associated with multi-objective planning in MDPs with time-bounded properties, depicted in Fig. 2 (a more precise explanation of the meaning of the different steps of the approach will be provided throughout the paper).

The first step in our approach is to prune the actions available at each state of the system. This pruning is based on a solution to a the multi-objective problem *without* elapsed execution time, i.e. a "time-bound relaxed" problem over a product MDP $\mathcal{M}'_{\varphi'}$ with an additional action that explicitly models the decision to stop gathering reward for the secondary tasks. For this relaxed problem one objective ($o'_1$) is minimising the expected time to fulfil the primary co-safe LTL task, and the other objective ($o'_2$) is the infinite-horizon reward maximisation for the secondary tasks. We calculate the Pareto front for these two objectives, and remove actions from the model that do not appear in any of the deterministic policies associated with the Pareto front. Intuitively, actions which do not appear in any policy on the Pareto front are actions which lead to either increased execution time on the primary task (the first objective), decreased reward on the secondary tasks (the second objective), or both. In the second step of our approach we add elapsed time to the states in this pruned model, and calculate the Pareto front for the time-bounded objectives $(o_1, o_2)$ over this smaller model. This Pareto front is suboptimal for the original problem. However, as illustrated in our empirical evaluation, the loss of optimality is low. More importantly, the Pareto front is composed of policies for which we can provide precise probabilistic guarantees on timely primary task completion, which are crucial for reliable real life deployments of autonomous systems.

We proceed with framing our work in the relevant literature, introducing the modelling formalisms and specification languages (TMDPs, co-safe LTL, multi-objective problems) that we will use, and frame the problem described in Example 1 within such formalisms. Then, we describe an optimal solution for our problem and, motivated by its high complexity, we present our two-step approximation approach. Finally, we show how our approach can yield significant state space gains, whilst still finding good approximations to the optimal Pareto front for the time-bounded problem, by evaluating it in the problem described in Example 1.

## Related Work

In the planning community, the term *oversubscription planning* is used to describe a problem in which the planning agent does not have the resources necessary to achieve all the goals given to it (i.e. it is oversubscribed), and instead must determine which goals it should achieve along with a plan to achieve them (Aghighi and Jonsson 2014). In order to assist the choice between goals, a metric is often used to define a the *net benefit* of the selected goals given the cost of the actions to achieve them (Benton, Do, and Kambhampati 2005). Structuring the problem further, some systems deal with *hard goals* (or hard constraints) which must always be achieved and *soft goals* (or soft constraints) which can be selected from to yield additional net benefit (Benton, Do, and Kambhampati 2005).

Our work can be considered in this latter context, with the primary LTL task as a hard goal, and the secondary tasks as soft goals. Our approach solves the net benefit problem given these secondary tasks by maximising the reward from them without harming primary task completion. This is done in the context of a time bound which places a hard constraint on the achievement of the primary goal (thus also limiting time available for gathering secondary reward). Given this, our work is most directly comparable to Sapa$^{ps}$ (Benton, Do, and Kambhampati 2005) and OPTIC (Benton, Coles, and Coles 2012) which can handle hard and soft constraints with numeric rewards/preferences. However we go beyond these planners by tackling this problem under uncertainty.

Adding soft goals that represent the execution of information gathering actions for environment exploration has also been studied in the robotics context. For example, in (Spaan, Veiga, and Lima 2015), an information reward is awarded when the agent reaches a certain level of belief regarding a state feature, and (Korein, Coltin, and Veloso 2014) investigate how to fill gaps in a schedule for a mobile robot with exploration actions.

In recent years, there has been much work on generating MDP policies under temporal logic specifications (Wolff, Topcu, and Murray 2012; Yoo, Fitch, and Sukkarieh 2013; Ding, Pinto, and Surana 2013; Ding et al. 2014; Lacerda, Parker, and Hawes 2015b). These rely on maximising the probability of task success and/or minimising expected costs to do so, and do not use time-bounded specifications, nor deal with maximising a utility related to soft goals.

Other approaches use MDPs for motion planning, and policy generation for goals expressed in time-bounded temporal logics such as bounded LTL (Cizelj and Belta 2014) or metric temporal logic (Fu and Topcu 2015). These works focus on a single objective, not dealing with trade-offs between different objectives. Also, we use LTL specifications extended with a time-bound, which allows us to avoid the high complexity of the time-bounded logics used in these works. Reaching goals under a deadline has also been investigated in the context of stochastic shortest path problems (Nie and Wu 2009; Cao et al. 2017). These works are more focused on finding computationally efficient solution methods, and deal only with simpler, single objective, goal reachability problems when compared to our use of co-safe LTL and multi-objective problem formulation. Finally, also using multi-objective model checking techniques, (Lahijanian and Kwiatkowska 2016) present an approach to manage the trade-off between probabilities of satisfaction of a (revised) co-safe LTL specification and the cost for such a revision of the original specification. This work focuses on the co-safe LTL specification, with the trade-off being on how much the specification needs to be "changed" in order to be achieved with high probability by the robot in its environment.

## Preliminaries

### Timed Markov Decision Processes

We model the problem as a *timed MDP* (TMDP), an MDP augmented with stochastic durations for each transition. A TMDP is a tuple $\mathcal{M} = \langle S, \overline{s}, A, \delta_{\mathcal{M}}, T, \Lambda, AP, Lab \rangle$, where: $S$ is a finite set of discrete states; $\overline{s} \in S$ is the initial state; $A$ is a finite set of actions; $\delta_{\mathcal{M}} : S \times A \times S \to [0,1]$ is a probabilistic transition function, where $\sum_{s' \in S} \delta_{\mathcal{M}}(s,a,s') \in \{0,1\}$ for all $s \in S$, $a \in A$; $T = \{t_1, ...t_n\} \subset \mathbb{N}_{>0}$ is a finite set of *discrete action execution times*; $\Lambda = \{\lambda_{s,a,s'} \mid \delta_{\mathcal{M}}(s,a,s') > 0\}$, where each $\lambda_{s,a,s'} : T \to [0,1]$ is a probability distribution over (integer) durations representing the time taken to execute action $a$ from state $s$ and finish in $s'$; $AP$ is a set of atomic propositions; and $Lab : S \to 2^{AP}$ is a labelling function, such that $p \in Lab(s)$ iff $p$ is true in $s \in S$.

A TMDP represents the possible evolutions of the state of a system where, in each state $s$, any action $a$ from the *enabled actions* $A_s = \{a \in A \mid \delta_{\mathcal{M}}(s,a,s') > 0 \text{ for some } s' \in S\}$ can be selected. The probability of evolving to successor state $s'$ is then $\delta_{\mathcal{M}}(s,a,s')$, and the probability that the time duration of this is $t \in T$ is $\lambda_{s,a,s'}(t)$. A TMDP can be seen as a special case of a semi-MDP (Howard 1960), restricted to integer, not real, duration times. For a mobile robot, these probability distributions can be learned by gathering data during robot execution and a clustering algorithm applied to group them into discrete classes. One can obtain more accurate models by increasing the number of such clusters, at the price of increased model sizes. During execution, assuming an accurate enough model where there is low action duration variability within each cluster, one can associate the observed action duration to the closest cluster representative.

A (finite or infinite) *path* through the TMDP is a sequence $\sigma = s_0(a_0,t_0)s_1(a_1,t_1)\dots$ where $s_0 = \overline{s}$ and, for all $i$, $\delta_{\mathcal{M}}(s_i,a_i,s_{i+1}) > 0$ and $\lambda_{s_i,a_i,s_{i+1}}(t_i) > 0$. We denote the sets of all finite and infinite paths of $\mathcal{M}$ by $FPath_{\mathcal{M}}$ and $IPath_{\mathcal{M}}$. The choice of action to take at each step of execution is made by a *policy*, which can base its decision on the history of $\mathcal{M}$ up to the current state, mapping finite paths of $\mathcal{M}$ to distributions over the set of actions. Formally, a policy is a function $\pi : FPath_{\mathcal{M}} \to Dist(A)$ such that, for any finite path $\sigma$ ending in state $s_n$, if $\pi(\sigma)(a) > 0$ then $a \in A_{s_n}$.

Important classes of policy include those that are *deterministic* (where $\pi(\sigma)$ is a point distribution), *memoryless* (which only base their choice on the current state $s$), and *finite-memory* (which need to track only a finite set of "modes"). The set of all policies for $\mathcal{M}$ is denoted by $\Pi_{\mathcal{M}}$.

Under a particular policy $\pi$, all nondeterminism is resolved and the behaviour of $\mathcal{M}$ is fully probabilistic. This allows for the definition of a probability measure $Pr^{\pi}_{\mathcal{M}}$ over the set of infinite paths $IPath_{\mathcal{M}}$, which determines the probability of certain events holding under policy $\pi$. For a property $q : IPath_{\mathcal{M}} \to \{0,1\}$, we write $Pr^{\pi}_{\mathcal{M}}(q)$ for the probability of $q$ holding under $\pi$. Given a (measurable) function $f : IPath_{\mathcal{M}} \to \mathbb{R}$, we can define the *expected value* $E^{\pi}_{\mathcal{M}}(f)$ of $f$ with respect to the probability measure $Pr^{\pi}_{\mathcal{M}}$. We can then consider the maximum probabilities or expected values over all policies: $Pr^{\max}_{\mathcal{M}}(q) = \sup_{\pi} Pr^{\pi}_{\mathcal{M},s}(q)$ and $E^{\max}_{\mathcal{M}}(f) = \sup_{\pi} E^{\pi}_{\mathcal{M}}(f)$, respectively. Minimum values $Pr^{\min}_{\mathcal{M}}(q)$ or $E^{\min}_{\mathcal{M}}(f)$ are defined analogously.

**Example 2.** The scenario represented in Example 1 can be modelled as a TMDP with a *factored*[1] state space defined as

$$\begin{aligned} S = &\langle L, n\_water, (checked\_water_o)_{o \in O}, \\ &(wants\_water_o)_{o\in O}, (delivered\_water_o)_{o\in O}, \\ &(interacted_i)_{i \in I}\rangle, \end{aligned}$$

where $L \in \{v_0, ..., v_{20}\}$ represents the robot location, $n\_water \in \{0,1,2\}$ represents the number of bottles the robot is currently holding, $wants\_water_o \in \{-1,0,1\}$ represents that the robot either does not know if the staff member wants water, or it knows that the staff member does not want water, or it knows that the staff member wants water. The other state factors are boolean values representing, for example, if the robot already asked if the staff member in office $o$ wants water, or if the robot has delivered the water at office $o$. The initial state is $\overline{s} = \langle v_0, 0, (0)_{o\in O}, (-1)_{o\in O}, (0)_{o \in O}, (0)_{i\in I}\rangle$. The action space comprises the different navigation actions between nodes, plus actions for delivering water, providing interaction, etc. For example, the action $check\_if\_water_o \in A$ is such that:

$$\begin{aligned} \delta_{\mathcal{M}}(&\langle checked\_water_o = 0, wants\_water_o = -1\rangle, \\ &check\_if\_water_o, \\ &\langle checked\_water_o = 1, wants\_water_o = 0\rangle) = 0.2 \end{aligned}$$

$$\begin{aligned} \delta_{\mathcal{M}}(&\langle checked\_water_o = 0, wants\_water_o = -1\rangle, \\ &check\_if\_water_o, \\ &\langle checked\_water_o = 1, wants\_water_o = 1\rangle) = 0.8 \end{aligned}$$

Also, there is different timing information associated with the transitions. For example, we have:

$$\begin{aligned} \delta_{\mathcal{M}}(L = 0, move_{01}, L = 1) = 1, \\ \lambda_{(L=0, move_{01}, L=1)}(15) = 0.8, \\ \lambda_{(L=0, move_{01}, L=1)}(90) = 0.2, \end{aligned}$$

meaning that 80% of the time, the robot can travel between location 0 and 1 in around 15 seconds, but 20% of the time it needs around 90 seconds, due to the presence of many people. Finally, we define the set of atomic propositions as:

$$\begin{aligned} AP = \{&holding\_0\_bottles, asked\_if\_water_o, \\ &wants\_water_o, delivered\_water_o\}, \end{aligned}$$

with $holding\_0\_bottles \in Lab(s)$ iff $s \in \{s' \in S \mid s'[n\_water] = 0\}$, where $s'[n\_water]$ represents the value of state feature $n\_water$ in $s'$. The labelling function for the other atomic propositions is defined by labelling states for which the corresponding state feature value is 1.

**Single-objective Problems** We introduce the set of problems over TMDPs that are relevant for our work. Let $d \in \mathbb{N}_{>0} \cup \{\infty\}$ be a *time-bound*. The *probabilistic reachability problem* is the problem of calculating $Pr^{\max}_{\mathcal{M}}(reach^{\leq d}_{S'})$, along with a corresponding optimal policy. Given $S' \subseteq S$, we define the event of reaching a state in $S'$ within bound $d$ as $reach^{\leq d}_{S'} : IPath_{\mathcal{M}} \to \{0,1\}$ where $reach^{\leq d}_{S'}(s_0(a_0,t_0)s_1(a_1,t_1)\dots) = 1$ iff exists $i$ such that $s_i \in S'$ and $\sum_{k=0}^{i-1} t_k \leq d$. The *expected cumulative reward problem* is the problem of calculating $E^{\max}_{\mathcal{M}}(cumul^{\leq d}_{r})$, along with a corresponding optimal policy. Let $r : S \times A \to \mathbb{R}_{\geq 0}$ be a *reward structure* over $\mathcal{M}$. We define the cumulative reward within bound $d$ $cumul^{\leq d}_r : IPath_{\mathcal{M}} \to \mathbb{R}_{\geq 0}$ as $cumul^{\leq d}_r(s_0(a_0,t_0)s_1(a_1,t_1)\dots) = \sum_{i=0}^{k_d} r(s_i,a_i)$ where $k_d = \infty$ if $d = \infty$, and $k_d = \max\{k \in \mathbb{N}_{>0} \mid \sum_{i=0}^{k} t_i \leq d\}$ otherwise. $E^{\min}_{\mathcal{M}}(cumul^{\leq d}_r)$ is defined analogously, and in this case we write $c$ instead of $r$, and call it a *cost structure*.

When $d = \infty$, we call the above *time-unbounded* problems. Such problems can be solved via standard MDP algorithms such as value or policy iteration (Puterman 1994). Furthermore, the existence of an optimal memoryless policy for them is guaranteed. Since the value for the expected cumulative reward problem might not converge, we first perform a graph analysis to identify states for which the reward is infinite, and end components with zero cost (Forejt et al. 2011).

When $d \neq \infty$, we call the above *time-bounded* problems. Time-boundedness implies that, in order to make optimal

[1] Factored state representations are usual for MDP modelling, given that they provide a more compact representation of the domain. We will also define transitions over the factored representation, where if a state factor is not present in the transition then it will be defined regardless of the value of that state factor.

decisions, one needs to keep track of how much time has passed during execution. Thus, the optimal policies might not be memoryless any more, and can be finite-memory. To solve these problems optimally, we need to explicitly add an elapsed time component to the state representation. For this extended state representation, the policies are again memoryless. However, the addition of the elapsed time component incurs a large state-space explosion and severely limits the size of the problems that can be solved.

Finally, given $S' \subseteq S$ such that $Pr^{\max}_{\mathcal{M}}(\mathit{reach}^{\leq d}_{S'}) = 1$, and a cost function $c$, we can define the *stochastic shortest path* problem as finding a policy that reaches a state in $S'$ with minimal expected cumulative cost, which we denote as $E^{\min}_{\mathcal{M}}(\mathit{cumul}^{\mathit{reach}_{S'}}_c)$. The stochastic shortest path problem can be reduced to a time-unbounded expected cumulative cost problem with guaranteed convergence, by appropriately changing the cost function.

**Example 3.** We define a reward and a cost structure over the TMDP presented in Example 2. First, we assume that the robot can gather a reward for interacting with people in rooms $i \in I$. The value of this reward is given by how much the person in each room $i$ values interacting with the robot. In our case, to simplify presentation, we assume reward 10 for the first interaction with each person:

$$r(s,a) = \begin{cases} 10 & \text{if } a = \mathit{interact}_i \text{ and } s[\mathit{interacted}_i] = 0 \\ & \text{for some } i \in I \\ 0 & \text{otherwise} \end{cases}$$

The above reward structure is an example of a soft goal specification, defined over (state,action) pairs. Note that one can also define soft goals as just state based rewards. Furthermore, we assume there is a finite number of soft goals, i.e., the amount of soft goal reward that can be accumulated is finite. For soft goals that are only accumulated once, one can easily modify the MDP model in order to guarantee the finiteness of the soft goal reward, by adding an extra state feature representing whether the reward for a given soft goal has been gathered or not.

We can also define a cost structure, obtained from the time distributions in the TMDP, that represents the *expected time* to execute a given action:

$$c(s,a) = \sum_{s' \in S} \delta_{\mathcal{M}}(s,a,s')(\sum_{t \in T'} \lambda'_{s,a,s'}(t)t)$$

**Multi-objective Problems** In this paper, we also consider *multi-objective* properties, which optimise two or more distinct objectives. To mix objective values, it is common to consider their *Pareto front*. Let $o_1, \ldots, o_n$ be different maximisation[2] objectives (i.e., $o_i(\pi) = Pr^{\max}_{\mathcal{M}}(q)$ or $o_i(\pi) = E^{\max}_{\mathcal{M}}(f)$). Given policies $\pi$, $\pi' \in \Pi_{\mathcal{M}}$, $\pi$ *Pareto-dominates* $\pi'$, denoted $\pi \succ \pi'$, if $\pi$ is at least as good as $\pi'$ for all objectives, and strictly better than $\pi'$ for at least one of them:

$$\pi \succ \pi' \text{ iff } \forall_i o_i(\pi) \geq o_i(\pi') \text{ and } \exists_i o_i(\pi) > o_i(\pi')$$

The Pareto front for $o_1, \ldots, o_n$ is then defined as the set of policies which are *not* dominated by any other policy:

$$Par(o_1, \ldots, o_n) = \{\pi \in \Pi_{\mathcal{M}} \mid \nexists_{\pi' \in \Pi_{\mathcal{M}}} \pi' \succ \pi\}$$

The Pareto front can be represented as a finite set of deterministic policies $\Pi(o_1, \ldots, o_n)$: Any value for $o_1, ..., o_n$ corresponding to a policy in $Par(o_1, \ldots, o_n)$ can be attained by a randomised policy constructed by a convex combination of two elements of $\Pi(o_1, \ldots, o_n)$, e.g., a policy that behaves like $\pi_1 \in \Pi(o_1, \ldots, o_n)$, with probability $\alpha$, and behaves like $\pi_2 \in \Pi(o_1, \ldots, o_n)$, with probability $1 - \alpha$, $0 \leq \alpha \leq 1$. We can calculate such a set $\Pi(o_1, \ldots, o_n)$ using the approach presented in (Forejt, Kwiatkowska, and Parker 2012).

## Syntactically Co-Safe Linear Temporal Logic

*Linear temporal logic* (LTL) (Pnueli 1981) is an extension of propositional logic that provides a formal, convenient and powerful way to specify a variety of qualitative properties over a system's execution. In this work, we use the *syntactically co-safe* class of LTL formulas, for which formula $\varphi$ over propositions $AP$ is defined using the grammar:

$$\varphi ::= \mathit{true} \mid p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathtt{X}\,\varphi \mid \mathtt{F}\,\varphi \mid \varphi \,\mathtt{U}\, \varphi,\ p \in AP.$$

The $\mathtt{X}$ operator is read "next", meaning that the formula it precedes will be true in the next state. The $\mathtt{F}$ operator is read "eventually", meaning that the formula it precedes must become true in some future state. The $\mathtt{U}$ operator is read "until", meaning that its second argument will eventually become true in some state, and the first argument will be continuously true until this point. See, e.g., (Pnueli 1981) for the formal semantics of the logic.

Given $\sigma \in IPath_{\mathcal{M}}$, we write $\sigma \models \varphi$ to denote that $\sigma$ satisfies formula $\varphi$. Furthermore, we write $Pr^{\max}_{\mathcal{M}}(\varphi)$ to denote the maximum probability of satisfying $\varphi$ from $s$ in $\mathcal{M}$. It is known that this problem can be reduced to a reachability problem in a *product* MDP (Vardi 1985).

Even though their semantics is defined over infinite sequences, co-safe LTL formulas always have a finite *good prefix* (Kupferman and Vardi 2001). Given an LTL formula $\varphi$ and an infinite sequence of sets of atomic propositions $w = w_0 w_1 ... \in (2^{AP})^{\omega}$ such that $w \models \varphi$, $w$ has a good prefix for $\varphi$ if there exists $n \in \mathbb{N}$ for which the truncated finite sequence $w|_n = w_0 w_1 ... w_{n-1}$ is such that the concatenation $w|_n \cdot w' \models \varphi$ for any $w' \in (2^{AP})^{\omega}$. We denote as $k^w_{\varphi}$ the length of the shortest good prefix of $w$ for $\varphi$.

Furthermore, for any co-safe LTL formula $\varphi$ written over $AP$, we can build a deterministic finite automaton (DFA) $\mathcal{A}_{\varphi} = \langle Q, \overline{q}, Q_F, 2^{AP}, \delta_{\mathcal{A}_{\varphi}} \rangle$, where: $Q$ is a finite set of discrete states; $\overline{q} \in Q$ is the initial state; $Q_F \subseteq Q$ is the set of accepting (i.e., final) states; $2^{AP}$ is the alphabet; and $\delta_{\mathcal{A}_{\varphi}} : Q \times 2^{AP} \to Q$ is a transition function. $\mathcal{A}_{\varphi}$ accepts exactly the good prefixes for $\varphi$ (Kupferman and Vardi 2001). Given that a good prefix satisfies $\varphi$ regardless of how it is "completed', an accepting state $q_F \in Q_F$ is such that $\delta_{\mathcal{A}_{\varphi}}(q_F, \alpha) \in Q_F$ for all $\alpha \in 2^{AP}$.

We finish by adding time bounds to our co-safe LTL specifications. Let $d \in \mathbb{N}_{>0}$ be a *time bound* and $\varphi$ a co-safe LTL formula. We say that $\sigma = s_0(a_0, t_0)s_1(a_1, t_1) \cdots \in IPath_{\mathcal{M}}$

[2] We present the notion of Pareto front only for maximisation objectives in order to simplify the presentation. Its adaptation to a mix of maximisation and minimisation objectives is straightforward.

satisfies $\varphi$ within the time bound $d$, denoted $\sigma \models \varphi^{\leq d}$ iff $w = Lab(s_0)Lab(s_1)... \models \varphi$ and $\sum_{i=0}^{k_\varphi^w - 1} t_i \leq d$. Following the presented notation, we write $Pr_{\mathcal{M}}^{\max}(\varphi^{\leq d})$ to denote the maximum probability of satisfying $\varphi$ in less than $d$ time.

**Example 4.** The water delivery task described in Example 1 can be represented by the following co-safe LTL formula:

$$\varphi = \mathtt{F}(holding\_0\_bottles \wedge \bigwedge_{o \in O} asked\_if\_water_o \wedge \\ \bigwedge_{o \in O}(wants\_water_o \rightarrow (\mathtt{F}\, delivered\_water_o)))$$

One can read the above specification[3] as "eventually reach a state where the robot is not holding water bottles, and has asked if all staff members in offices $O$ want water, and for each of the staff member, if wants water, then eventually deliver him the water."

## Problem Statement

We now formally define the problem tackled in the paper. In the next sections, we describe the optimal solution, and propose a more efficient approximation technique.

Let $\mathcal{M}$ be a TMDP, $\varphi$ a co-safe LTL formula, $d \in \mathbb{N}_{>0}$ a deadline and $r : S \times A \rightarrow \mathbb{R}_{\geq 0}$ a reward structure for soft goals such that, for all $\sigma = s_0(a_0, t_0)s_1 \cdots \in IPath_{\mathcal{M}}$, $\sum_{i=0}^{\infty} r(s_i, a_i) < \infty$. Our goal is to (i) complete the co-safe task within the deadline, while (ii) retrieving as much soft goal reward as possible. Formally, we have the objectives:

$$\text{(i) } o_1 = Pr_{\mathcal{M}}^{\max}(\varphi^{\leq d}) \;\text{ and }\; \text{(ii) } o_2 = E_{\mathcal{M}}^{\max}(cumul_r^{\leq d})$$

and our goal is to generate the Pareto front $Par(o_1, o_2)$.

## Optimal Solution

To solve the problem above, we start by following the standard approach of adding a time component to the states. We build the MDP $\mathcal{M}^T$ by extending the TMDP $\mathcal{M}$ such that:

$$S^T = \{(s,t) \in S \times \mathbb{N} \mid t \leq d + \\ \max\{k \in \mathbb{N}_{>0} \mid \exists_{s' \in S, a \in A} \delta_{\mathcal{M}}(s', a, s)\lambda_{s',a',s}(k) > 0\}\}$$

$$\delta_{\mathcal{M}^T}((s,t), a, (s', t+k)) = \\ \begin{cases} \delta_{\mathcal{M}}(s, a, s')\lambda_{s,a,s'}(k) & \text{if } t \leq d \\ 0 & \text{otherwise} \end{cases}$$

The rest of the structure of $\mathcal{M}^T$ and the soft goal reward $r$ can be extended to $S^T$ straightforwardly. $\mathcal{M}^T$ is such that all possible paths through $\mathcal{M}^T$ end in some terminal state $(s,t)$ where $t > d$. This construction reduces objective $o_2$ in the problem statement to a time-unbounded problem over $\mathcal{M}^T$:

$$E_{\mathcal{M}}^{\max}(cumul_r^{\leq d}) = E_{\mathcal{M}^T}^{\max}(cumul_r^{\leq \infty})$$

The same is not true for $o_1$. In particular, $\mathcal{M}^T$ does not have information on the satisfaction of $\varphi$. Thus, we follow the standard approach of building the *product* MDP $\mathcal{M}_\varphi^T = \mathcal{M}^T \otimes \mathcal{A}_\varphi$ over state space $S_\varphi^T = S^T \times Q$ which behaves like the original MDP but is augmented with information about the satisfaction of $\varphi$. Once a path of $\mathcal{M}_\varphi^T$ reaches an *accepting state* in $acc = \{(s, q_F) \in S^T \times Q_F \mid s[t] \leq d\}$, it is a good prefix for $\varphi$ that was reached within $d$ timesteps, and we are sure that $\varphi$ is satisfied before the deadline. The construction of the product is well known and is such that it preserves the probabilities of the original MDP (see, e.g., (Baier and Katoen 2008)). We can again extend $r$ to $S_\varphi^T$, so $o_2$ can be equivalently solved in $\mathcal{M}^T$ and in $\mathcal{M}_\varphi^T$. Furthermore, $o_1$ can now be reduced to an time-unbounded (due to the addition of a time component to the state) probabilistic reachability (due to the product with $\mathcal{A}_\varphi$) problem:

$$Pr_{\mathcal{M}}^{\max}(\varphi^{\leq d}) = Pr_{\mathcal{M}_\varphi^T}^{\max}(reach_{acc}^{\leq \infty})$$

Since $o_1$ and $o_2$ can be independently solved in $\mathcal{M}_\varphi^T$ using value iteration, we can also solve the corresponding multi-objective problem in $\mathcal{M}_\varphi^T$, and build the Pareto front representing the optimal solutions for the trade-offs between $o_1$ and $o_2$. As stated before, the main drawback of this approach is the very large state-space explosion it entails. Thus, in the next section we present an approach to generate approximations of the optimal Pareto front.

## Approximate Solution

### Relaxation of the Time-bounded Objectives

To approximate solutions to our problem, we present a method based on using the Pareto-front for time-unbounded simplifications of $o_1$ and $o_2$ to prune the model. We will define a multi-objective problem that (i) minimises the expected cost to fulfil the co-safe LTL task; *and* (ii) gathers as much soft goal reward as possible. To guarantee convergence for the first objective, we take the common assumption that $\varphi$ can be satisfied with probability 1 from the initial state. We leave relaxing this assumption for future work, building on work on stochastic shortest paths with dead ends (Teichteil-Königsbuch 2012; Kolobov, Mausam, and Weld 2012) and partially satisfiable co-safe LTL (Lacerda, Parker, and Hawes 2015b; 2015a).

Problems $o_1$ and $o_2$ are tightly related, in the sense that both are defined with the same time-bound $d$. When relaxing them to time-unbounded problems, that relation is lost: given that we want to minimise time expectations, and maximise rewards over an infinite horizon, it is not clear when one should stop accumulating time cost, as that depends not only on fulfilling the task, but also on the decision to stop trying to satisfy more soft goals. Thus, to recover the relation between $o_1$ and $o_2$ when relaxing them to time-unbounded problems, we need to add an extra (0-timed) $stop$ action that is active in all the states, and allows us to decide when to stop trying to maximise the soft goal reward structure. For TMDP $\mathcal{M} = \langle S, \overline{s}, A, \delta_{\mathcal{M}}, \Lambda, AP, Lab, \rangle$, we define $\mathcal{M}' = \langle S', \overline{s}', A', \delta_{\mathcal{M}'}, \Lambda', AP', Lab' \rangle$ that has two copies of $S$, i.e., $S' = S^{gather} \cup S^{stop}$, where $S^{gather}$ represents that we are still gathering soft goal reward, and $S^{stop}$ represents that we have decided to stop doing so. All other elements of $\mathcal{M}'$ are simply extended to $S'$. $\mathcal{M}'$ starts in $S^{gather}$, where $stop$ is enabled in all states. When $stop$ is executed, $\mathcal{M}'$

[3] In fact, to be syntactically co-safe, the occurrences of $p \rightarrow (Fq)$ need to be written as $(\neg p) \vee (Fq)$. We write the specification using implications for readability.

evolves to the corresponding state in $S^{stop}$, where it can still act towards fulfilling the task, but no more soft goal reward is available, i.e., slightly abusing notation we define $r'$ over $S^{gather} \cup S^{stopped}$ such that:

$$r'(s,a) = \begin{cases} r(s,a) & \text{if } s \in S^{gather} \\ 0 & \text{otherwise} \end{cases}$$

We also define a cost function over $\mathcal{M}'$, $c' : S' \times A' \to \mathbb{R}_{\geq 0}$, providing the *expected time* to execute $a$ in $s$, as explained in Example 3.

We then extend $\varphi$ to reflect that expected time cost can only stop being accumulated once *stop* has been executed:

$$\varphi' = \varphi \wedge (\mathtt{F}\, stopped)$$

This transformation preserves the probabilities of satisfaction of $\varphi$ over $\mathcal{M}$ and also the timing between states in the TMDP. By performing it, we can now get policies that achieve soft goals *after* satisfying the task, while still accumulating expected time cost. Thus, we calculate the Pareto front for the following two objectives:

$$\text{(i) } o_1' = E^{\min}_{\mathcal{M}'}(cumul^{\models\varphi'}_{c'}) \text{ and (ii) } o_2' = E^{\max}_{\mathcal{M}'}(cumul^{\infty}_{r'})$$

$E^{\min}_{\mathcal{M}'}(cumul^{\models\varphi'}_{c'})$ represents the minimal expected cost to satisfy $\varphi'$. Since $\varphi$ is co-safe, $o_1'$ can be reduced to the stochastic shortest path problem of reaching a state in $acc = S \times Q_F$ in the product MDP $M'_{\varphi'}$. Note that, contrary to the product MDP built for the optimal solution, this product does not have a time component in the states, thus its state space is significantly smaller. Finally, by straightforwardly extending $r'$ and $c'$ to $r'_{\varphi'}$ and $c'_{\varphi'}$ over $M'_{\varphi'}$, we can define both objectives over the same structure:

$$E^{\min}_{\mathcal{M}'}(cumul^{\models\varphi'}_{c'}) = E^{\min}_{\mathcal{M}'_{\varphi'}}(cumul^{reach_{acc}}_{c'_{\varphi'}})$$

$$E^{\max}_{\mathcal{M}'}(cumul^{\infty}_{r'}) = E^{\max}_{\mathcal{M}'_{\varphi'}}(cumul^{\infty}_{r'_{\varphi'}})$$

As mentioned before, the stochastic shortest path problem can be turned into a time-unbounded expected cost minimisation by adjusting the cost function. Furthermore, convergence is ensured if $Pr^{\max}_{\mathcal{M}}(\varphi) = 1$, so we have two basic problems which can be calculated using value iteration. Thus, we can calculate a set $\Pi(o_1', o_2')$ of deterministic memoryless policies representing the Pareto front for $o_1'$ and $o_2'$, using the approach in (Forejt, Kwiatkowska, and Parker 2012).

### Pareto front Induced MDP

Having calculated the set $\Pi(o_1', o_2') = \{\pi_1, \ldots, \pi_m\}$, we can use it to prune $\mathcal{M}'_{\varphi'}$. The basic idea is that, given the relation between the time-unbounded objectives $o_1'$ and $o_2'$, and the original time-bounded objectives $o_1$ and $o_2$, policies on the Pareto front for $o_1'$ and $o_2'$ are, in principle, also good policies for balancing $o_1$ and $o_2$. This means that an MDP consisting of only the transitions present in at least one of these policies obtained for $o_1'$ and $o_2'$ can be a good model for reasoning about the time-bounded problems. Thus, we build the pruned MDP $\mathcal{M}'_{\varphi'}|_{\Pi(o_1', o_2')}$ by removing all transitions $\delta(s, a, s')$ for which $a$ is not in $\pi(s)$ for some policy $\pi \in \Pi(o_1', o_2')$. So, $\mathcal{M}'_{\varphi'}|_{\Pi(o_1', o_2')}$ only has actions that are working towards either the expected time cost minimisation, or the soft goal reward maximisation. We can then find approximations of the optimal value by solving the time-bounded objectives $o_1$ and $o_2$ on $\mathcal{M}'_{\varphi'}|_{\Pi(o_1', o_2')}$, using the MDP $\mathcal{M}'_{\varphi'}|^{T}_{\Pi(o_1', o_2')}$, obtained by adding the time component to the states of $\mathcal{M}'_{\varphi'}|_{\Pi(o_1', o_2')}$ as explained when describing the optimal solution. As we show in the next section, we obtain good approximations of the optimal Pareto curve from MDPs with considerably smaller state spaces. Crucially, we still have precise probabilistic time-bounded guarantees for the resulting policies. Furthermore, the Pareto front that we generate represents a set of policies, and allows us to generate policies for any specified threshold on the probability of timely task completion with very little computational effort. We refer the reader back to Fig. 2 for a diagram of the overall approach.

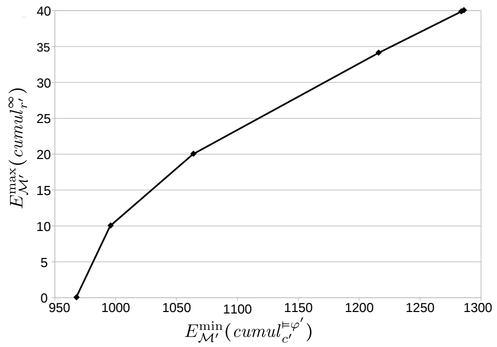

Figure 3: Pareto front for minimising expected task completion time and maximising expected soft goal reward.

## Implementation and Evaluation

We implemented our approach in the PRISM model checker (Kwiatkowska, Norman, and Parker 2011), which supports solving MDPs against LTL properties and generation of Pareto curves (Forejt, Kwiatkowska, and Parker 2012). The tool includes multiple solution engines, based on symbolic data structures such as binary decision diagrams (BDDs), designed to improve scalability. We analyse the performance of our approach on the illustrative example that has been described throughout the paper. In particular, we analyse the state space reduction obtained by our method, versus the amount of sub-optimality it incurs, for $O = \{v_6, v_7, v_{11}\}$ and three instances of $I$: $I_1 = \{v_6, v_{10}, v_{17}, v_{18}\}$, $I_2 = \{v_6, v_7, v_{10}, v_{11}\}$ and $I_3 = \{v_{17}, v_{18}, v_{19}, v_{20}\}$. These three instances cover a range of situations that can occur in this scenario: $I_1$ represents a situation where there is some intersection between rooms for the task, and rooms for interaction with humans; $I_2$ represents a situation where all the offices for the primary task are also relevant for the soft goals; and $I_3$ represents a situation where offices for the primary task and offices for interaction are disjoint, and far apart.

We start by illustrating the generation of the Pareto fronts for $I_1$. In Fig. 3, we depict the Pareto front for $o_1'$ and $o_2'$ with the six points being the values for the deterministic policies. These are used to prune the product before adding the time component. In Fig. 4, we compare the optimal Pareto fronts with the approximated Pareto fronts obtained over the pruned model, for different deadlines. Note that the approximated

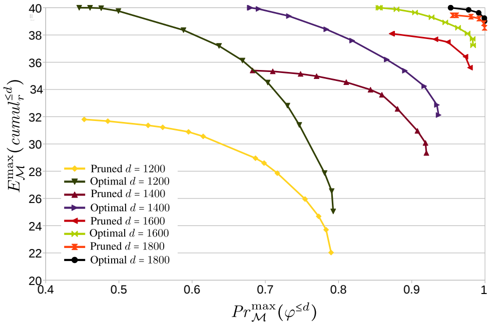

Figure 4: Pareto front balancing the objectives of maximising probability of task completion and maximising expected soft goal reward, both within deadline $d$.

Pareto curve gets closer to the optimal as we give more importance to maximising probability of task completion within the deadline. This is because we give more importance to primary task completion than soft goals, and is due to the multi-objective problem used for pruning: since we calculate expected time for the conjunction of task completion and reward gathering, policies that prioritise only soft goals and ignore the primary task are not present in the Pareto curve for $o'_1$ and $o'_2$. Hence, the approximated Pareto front becomes worse as we give more importance to reward maximisation.

Note that there is a large increase in terms of expected reward when we allow the value for the probability of task completion to be slightly less than the calculated maximum, as the maximum fully ignores the soft goals while there is still a small probability of not fulfilling the task within the time bound, regardless of how small that probability is. Thus, we choose (randomised) policies that achieve $99\%$ of the maximum probability of task satisfaction on the approximated Pareto front. This entails a slight decrease in probability of success, but as a trade-off we have a significant improvement in terms of expected soft goal reward.

Furthermore, the reward for soft goals can also be defined using a notion of information gain from executing certain actions at certain locations. This can allow us to improve the accuracy of the models for future planning instances. Thus, for a robot that is continuously learning its environment in order to improve its action outcomes and duration models, trading-off a small percentage of probability of primary task timely achievement for the achievement of soft goals related with model learning can also induce performance improvements for other primary tasks in the long run. We note that this is a simple approach of choosing a policy from the Pareto front, and more elaborate ones can be defined. In fact, the Pareto fronts can be seen as a fast policy generation mechanism for different trade-offs between our two objectives.

Finally, in Table 1, we depict the model sizes, computation times and sub-optimality incurred for $I_1$, $I_2$ and $I_3$. Sub-optimality is measured as the fraction:

$$\frac{E^{\max}_{\mathcal{M}'_{\varphi'}|_{\Pi(o'_1,o'_2)}}(cumul^{\leq d}_r \mid p_{0.99})}{E^{\max}_{\mathcal{M}}(cumul^{\leq d}_r \mid p_{0.99})}$$

| | $\mathcal{M}^T_{\varphi}$ | | $\mathcal{M}'_{\varphi'}$ | | $\mathcal{M}'_{\varphi'}|^T_{\Pi(o'_1,o'_2)}$ | | Optimality (fraction) |
|---|---|---|---|---|---|---|---|
| $d$ | #states | CT (s) | #states | CT (s) | #states | CT (s) | |
| 1200 | 10,505,498 | 4025 | 165,312 | 461 | 346,525 | 1043 | 0.864 |
| 1400 | 13,811,739 | 4017 | | | 472,678 | 1244 | 0.890 |
| 1600 | 17,117,979 | 4110 | | | 598,599 | 1490 | 0.954 |
| 1800 | 20,424,219 | 4160 | | | 724,529 | 1676 | 0.984 |
| 1200 | 11,524,883 | 3727 | 165,312 | 456 | 189,076 | 605 | 0.860 |
| 1400 | 14,831,123 | 3855 | | | 258,131 | 683 | 0.922 |
| 1600 | 18,137,363 | 3779 | | | 326,863 | 780 | 0.974 |
| 1800 | 21,443,603 | 3826 | | | 395,572 | 877 | 0.996 |
| 1200 | 9,790,652 | 3905 | 165,312 | 460 | 111,601 | 378 | 0.730 |
| 1400 | 13,096,955 | 3914 | | | 152,957 | 474 | 0.844 |
| 1600 | 16,403,195 | 3935 | | | 193,650 | 520 | 0.938 |
| 1800 | 19,709,435 | 4028 | | | 233,684 | 651 | 0.980 |

Table 1: Model sizes, computation times (CT), and incurred sub-optimality, for $I_1$(top), $I_2$ (middle), and $I_3$ bottom.

In the above, we write $p_{0.99}$ to denote that we are maximising the expected value of cumulative reward only over policies $\pi$ such that $Pr^{\pi}_{\mathcal{M}'_{\varphi'}|_{\Pi(o'_1,o'_2)}}(\varphi^{\leq d}) \geq 0.99$. The experiments were run on an Intel® Core™ i7-4930K CPU @ 3.40GHz, with 16GB of RAM. We obtain state space savings of more than $95\%$, while providing good approximations of the optimal Pareto curve. Our worst approximation provides an expected reward that is $73\%$ of the optimal expected reward for the chosen value for probability of timely task satisfaction. In terms of running times, we also achieve significant speed-ups, being twice as fast in the worst case, and generally achieving speed-ups in the order of three to four.

We finish by noting that variability in execution time in this example stems from two factors: the underlying uncertainty on action durations, and how many people ask for water. Our approach allows for the robot to decide whether to engage in more or less interactions with humans depending on these factors, i.e., the final outcome is a rich set of policies that adapt the robot behaviour to different situations during execution. This type of reasoning is not possible when just minimising expected time for task completion.

## Conclusions

We have presented an approach for the generation of policies for time-bounded specifications, showing how multi-objective model checking techniques can be used to generate policies that balance the achievement of a primary task with the achievement of a set of soft goals, and are guaranteed to fulfil the primary task specification with a given probability. In order to deal with time-bounded specifications, we introduce a pruning operation that removes transitions from the MDP before adding a time component to the states. We showed that this pruning can mitigate the state-space explosion. We have solved the pruned MDP optimally using probabilistic model checking techniques, but a relevant topic for future work is applying approximated solution techniques to our model. We expect the savings in state space provided by our pruning technique, along with approximated solutions, to allow us to solve much larger models. Finally, here we assume time-independent soft goal rewards. We will also tackle time-dependent rewards in the future.

## Acknowledgements

This work has received funding from the EU Seventh Framework Programme (FP7/2007-2013) under grant agreement No 600623, STRANDS, and from the PRINCESS project, funded by the DARPA BRASS programme.

# References

Aghighi, M., and Jonsson, P. 2014. Oversubscription planning: Complexity and compilability. In *Proc. of the 28th AAAI Conference on Artificial Intelligence (AAAI)*.

Baier, C., and Katoen, J.-P. 2008. *Principles of Model Checking*. MIT Press.

Benton, J.; Coles, A. J.; and Coles, A. 2012. Temporal planning with preferences and time-dependent continuous costs. In *Proc. of the 22nd Int. Conf. on Automated Planning and Scheduling (ICAPS)*.

Benton, J.; Do, M. B.; and Kambhampati, S. 2005. Oversubscription planning with numeric goals. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI)*.

Cao, Z.; Guo, H.; Zhang, J.; Oliehoek, F.; and Fastenrath, U. 2017. Maximizing the probability of arriving on time: A practical Q-learning method. In *Proc. of the 31st AAAI Conf. on Artificial Intelligence (AAAI)*.

Cizelj, I., and Belta, C. 2014. Control of noisy differential-drive vehicles from time-bounded temporal logic specifications. *Int. Journal of Robotics Research* 33(8).

Ding, X.; Smith, S.; Belta, C.; and Rus, D. 2014. Optimal control of Markov decision processes with linear temporal logic constraints. *IEEE Trans. on Automatic Control* 59(5).

Ding, X.; Pinto, A.; and Surana, A. 2013. Strategic planning under uncertainties via constrained Markov decision processes. In *Proc. of the 2013 IEEE Int. Conf. on Robotics and Automation (ICRA)*.

Forejt, V.; Kwiatkowska, M.; Norman, G.; and Parker, D. 2011. Automated verification techniques for probabilistic systems. In *Formal Methods for Eternal Networked Software Systems (SFM)*, volume 6659 of *LNCS*. Springer.

Forejt, V.; Kwiatkowska, M.; and Parker, D. 2012. Pareto curves for probabilistic model checking. In Chakraborty, S., and Mukund, M., eds., *Proc. 10th Int. Symp. on Automated Technology for Verification and Analysis (ATVA'12)*, volume 7561 of *LNCS*, 317–332. Springer.

Fu, J., and Topcu, U. 2015. Computational methods for stochastic control with metric interval temporal logic specifications. In *Proc. of the 54th IEEE Conf. on Decision and Control (CDC)*.

Howard, R. 1960. *Dynamic Programming and Markov Processes*. MIT Press.

Kolobov, A.; Mausam; and Weld, D. 2012. A theory of goal-oriented MDPs with dead ends. In *Proc. of 28th Conf. on Uncertainty in Artificial Intelligence (UAI)*.

Korein, M.; Coltin, B.; and Veloso, M. 2014. Constrained scheduling of robot exploration tasks. In *Proc. of the 2014 Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*.

Kupferman, O., and Vardi, M. 2001. Model checking of safety properties. *Formal Methods in System Design* 19(3).

Kwiatkowska, M.; Norman, G.; and Parker, D. 2011. PRISM 4.0: Verification of probabilistic real-time systems. In *Computer Aided Verification (CAV)*, volume 6806 of *LNCS*. Springer.

Lacerda, B.; Parker, D.; and Hawes, N. 2015a. Nested value iteration for partially satisfiable co-safe LTL specifications. In *Proc. of AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents (SDMIA)*.

Lacerda, B.; Parker, D.; and Hawes, N. 2015b. Optimal policy generation for partially satisfiable co-safe LTL specifications. In *Proc. of the 24th Int. Joint Conf. on Artificial Intelligence (IJCAI)*.

Lahijanian, M., and Kwiatkowska, M. 2016. Specification revision for Markov decision processes with optimal trade-off. In *Proc.of the 55th IEEE Conf. on Decision and Control (CDC)*.

Nie, Y. M., and Wu, X. 2009. Shortest path problem considering on-time arrival probability. *Transportation Research Part B: Methodological* 43(6):597–613.

Pnueli, A. 1981. The temporal semantics of concurrent programs. *Theoretical Computer Science* 13.

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

Santos, J.; Krajník, T.; Fentanes, J. P.; and Duckett, T. 2016. Lifelong information-driven exploration to complete and refine 4D spatio-temporal maps. *IEEE Robotics and Automation Letters* 1(2).

Spaan, M. T. J.; Veiga, T. S.; and Lima, P. U. 2015. Decision-theoretic planning under uncertainty with information rewards for active cooperative perception. *Autonomous Agents and Multi-Agent System* 29(6):1157–1185.

Teichteil-Königsbuch, F. 2012. Stochastic safest and shortest path problems. In *Proc. of 26th AAAI Conf. on Artificial Intelligence (AAAI)*.

Vardi, M. 1985. Automatic verification of probabilistic concurrent finite state programs. In *Proc. of 26th IEEE Annual Symp. on Foundations of Comp. Sci. (FOCS)*.

Wolff, E. M.; Topcu, U.; and Murray, R. M. 2012. Robust control of uncertain Markov decision processes with temporal logic specifications. In *IEEE 51st Conference on Decision and Control (CDC)*.

Yoo, C.; Fitch, R.; and Sukkarieh, S. 2013. Provably-correct stochastic motion planning with safety constraints. In *Proc. of the 2013 IEEE Int. Conf. on Robotics and Automation (ICRA)*.