Handling Advice in MDPs for Semi-Autonomous Systems

Abdel-Illah Mouaddib[1] **and Laurent Jeanpierre**[1] **and Shlomo Zilberstein**[2*†]

Abstract

This paper proposes an effective new model for decision making in situations where full autonomy is not feasible due to inability to fully model and reason about the domain. To overcome this limitation, we consider a human operator who can supervise the system and guide its operation by providing high level advice. We define a rich representation for advice and describe an effective algorithm for generating a new policy that conforms to the given advice. Advice is designed to improve the efficiency and safety of the system by imposing constraints on state visitation (either encouraging or discouraging the system to visit certain states). Coupled with the standard reward maximization criterion for MDPs, advice poses a complex multi-criteria decision problem. We present and analyze an effective algorithm for optimizing the policy in the presence of advice.

Introduction

There has been significant progress in recent years with the construction of autonomous systems for a wide range of domains from household products such as the iRobot's Roomba vacuum cleaners to space exploration vehicles. But limitations of the prevailing sensing and reasoning techniques still limit the deployment of autonomous systems in uncertain environments where a variety of unexpected events may occur. Maintaining a safe and robust behavior in such environments is a considerable challenge. In general, systems use approximate and incomplete models for planning. Even if they compute an optimal policy, the approximate nature of the model makes it hard to produce reliable operation, particularly in application domains where uncontrollable events can lead to catastrophic damage or permanent failure of the system.

A general approach to address this challenge is to develop *semi-autonomous* systems that work under the supervision of a human operator who may have more complete knowledge of the domain and better sensing abilities. The operator may intervene to correct the behavior of the system when a deviation from the desired behavior is detected, according to their knowledge. There are many examples of research efforts to support such capabilities. The multiagent systems community has long been exploring various forms of *adjustable autonomy*, allowing autonomous agents or robots to get help from humans (Côté et al. 2012; Dorais et al. 1999; Goodrich et al. 2001; Mouaddib et al. 2010). Human help could come in different forms such as *teleoperation* (Goldberg et al. 2000) or guidance in the form of *goal bias* (Côté et al. 2012). Tools to facilitate human supervision of robots have been developed. Examples include a single human operator supervising a team of robots that can operate with different levels of autonomy (Bechar and Edan 2003), or robots that operate in hazardous environments under human supervision, requiring teleoperation in difficult situations (Ishikawa and Suzuki 1997). There has also been research on mobile robots that can *proactively seek help from people* in their environment to overcome their limitations (Hüttenrauch and Severinson Eklundh 2006; Rosenthal and Veloso 2012). In robotics, researchers have started to develop robots that can autonomously identify situations in which a human operator must perform a subtask (Shiomi et al. 2008) and design suitable interaction mechanisms for the collaboration (Yanco, Drury, and Scholtz 2004).

But human intervention is often costly and should be minimized, and permanent supervision or tele-operation of a system is not desired. Therefore, it is important to develop mechanisms that allow external input to help a semi-autonomous system avoid permanent failures, situations that require external help or the activation of costly recovery mechanisms.

In this paper, we develop a formal framework allowing an operator to intervene and guide the behavior of a semi-autonomous system. Interventions consist of *advice* sent by the operator to the system to improve its behavior. Advice takes the form of a list of states to target by the system for efficiency reasons and a list of states to avoid for safety reasons. To this end, we define *advice* as sets of forbidden states, desired states and undesired states. We develop a model and algorithms to integrate such advice with an MDP-based framework and to compute on-line a new policy upon the arrival of a new advice, considering the advice as hard and soft constraints on the policy. In this paper, we focus on

[*1]University of Caen basse-Normandie, CNRS, ENSICaen, Campus II, BP5186, 14032 Caen Cedex, France {abdel-Illah.mouaddib, Laurent.Jeanpierre}@unicaen.fr

[†2]School of Computer Science, University of Massachusetts, Amherst, USA shlomo@cs.umass.edu

MDPs with terminal goal states with no loops (cyclic policies including self-loops are left for future work).

Contributions We propose a new decision-making model in which a semi-autonomous system operates while interacting with an operator in charge of extending the abilities of the system by sending it advice to improve the safely and efficiently of the mission. The advice is based on external knowledge that is not explicitly available to the system. To this end, we consider an underlying Markov decision process (MDP) where the standard reward-maximizing objective is augmented by advice from a supervisor (operator) that guides the system in avoiding risky states or visiting desired states. The paper provides:

- A formal definition of advice in terms of desired states, forbidden states and undesired states and desired actions at some states;
- A formal definition of MDP with advice, called $\mathcal{A}$-MDP;
- A formal definition of policy properties in terms of completeness, safety and satisfaction;
- A multi-criteria approach to solve an $\mathcal{A}$-MDP as a multi-criteria MDP using a lexicographic regret-based techniques;
- An efficient algorithm to derive an advice-based policy respecting the properties defined above.

Background on Markov decision process

A Markov decision process (MDP) is a mathematical tool for robust sequential decision making and planning under uncertainty. Formally, an MDP is specified by a tuple $\langle S, A, T, R, H\rangle$ as follows :

- S is a set of possible system states
- A is the set of actions
- T is the transition function: $T : S \times A \times S \rightarrow [0, 1]$ where $T(s, a, s')$ represents the probability of reaching a state s' when taking action a in state s
- R is the reward function: $R : S \rightarrow \Re$ representing the reward of the agent in state s.
- H is the horizon, representing the number of decision steps. When H is infinite, we say that the MDP is with an infinite horizon and when H is finite but unknown, we say that the MDP is with indefinite horizon.

The core decision problem for MDPs is to find a "policy" for the decision maker: a function π that specifies the action $\pi(s)$ that the decision maker should choose when in state s. The goal is to find a policy π that will maximize the expected cumulative value of the rewards, possibly discounting future rewards with a factor of γ per decision cycle. The Bellman equation defines a value function over states, $V^*(s)$, from which an optimal policy π^* can be extracted:

$$V^*(s) = argmax_{a\in A}[R(s) + \gamma \sum_{s'} T(s, a, s') \cdot V^*(s')] \quad (1)$$

We consider MDPs with indefinite horizon and terminal goal states. In this case, a process unfolds over a finite, but possibly unknown, number of steps and ends when the system reaches one of the terminal states. Many algorithms have been developed to solve such MDPs and derive an optimal policy such as value and policy iteration (Sutton and Barto 1998).

$\mathcal{A}$-MDP definition and policy properties

Definition of advice

Advice can come in the form of state visitation (telling the system to visit or not visit certain states during plan execution), such as "avoid the bridge" or "visit the park", or in the form of actions to perform or not to perform in some states, such as "don't cross the bridge" or "don't take the highway, go through the forest". To capture such guidance, we define advice $\mathcal{A}$ to be a tuple$\langle S_f, S_d, S_u, \hat{\pi}\rangle$ such that:

- $S_f \subset S$ is a set of *forbidden states* that the system must avoid;
- $S_d \subset S$ is a set of *desired* states whose visitation is preferred;
- $S_u \subset S$ is a set of *undesired* states whose avoidance is preferred;
- $\hat{\pi}$ is a *partial policy* recommending some actions in some states.

An MDP-based model with advice

An $\mathcal{A}$-MDP is a classical MDP where some states are labeled desired, undesired and forbidden and some actions are recommended at some state. More formally, an $\mathcal{A}$-MDP is a pair defined by $\langle \text{MDP}, \mathcal{A}\rangle$. A goal-based $\mathcal{A}$-MDP is defined by $\langle \text{MDP}, \mathcal{A}\rangle$ and $\mathcal{G}$, a set of terminal goal states that the agent has to reach. In this paper, we address goal-based $\mathcal{A}$-MDPs with no loops. Self-looping and cyclic policies are left for future work.

A goal-based $\mathcal{A}$-MDP presents a multi-criteria problem where we have to consider first $\hat{\pi}$, then $(\mathcal{G}, \mathcal{S}_f)$ as hard constraints to satisfy and then $(\mathcal{S}_d, \mathcal{S}_u)$, which are soft constraints to satisfy. The best policies should guarantee that a goal is reached, all forbidden states are avoided, desired states are visited and undesired states are avoided as much as possible. Hence, we face a multi-criteria MDP problem with ordered criteria.

Different techniques have been dedicated to solving such MDPs. Most of them are focused on the determination of the set of Pareto-optimal solutions. However, this set could be very large making its computation highly complex. The good news is that policies offering a good well-balanced tradeoff between criteria (Roijers, Vamplew, and Whiteson 2013) or fairly sharing the expected rewards among agents (Mouaddib, Boussard, and Bouzid 2007) present promising solution techniques to this class of MDPs. Minimizing the ordered weighted average of regrets (OWR) (Roijers, Vamplew, and Whiteson 2013) has been proposed to compute such policies. This approach is considered an extension of minmax regret technique, relaxing egalitarianism with a milder notion of fairness. The OWR approach overcomes the

limitation of minmax and weighted sum methods, which are known to reach respectively pessimistic or non-balanced solutions. OWR is then a good alternative, but it suffers from initial state dependence (its optimality depends on the initial state) and violation of the Bellman optimality principle (that each subpolicy of an optimal policy is optimal).

To this end, we propose an approach that takes advantage of the problem structure, particularly that our criteria are ordered and the initial state is known, thereby transforming OWR into a Regret Tchebychev-like measure where ideal and worst regret measures are used to guarantee the Bellman optimality principle (Mouaddib 2004; Roijers, Vamplew, and Whiteson 2013). In the following, we formally define the key concepts and policy properties using our regret value function. With such characteristics (ordered criteria, a know initial state and our regret function), the solutions we find using classical techniques such as value-iteration or linear programming are regret-optimal solutions (Roijers, Vamplew, and Whiteson 2013). We define the key ingredients of the approach below.

Probability of visitation

Definition 1 Probability of visitation *The probability of visitation of a state s when following a policy π and starting at state s_0 is:*

$$P_{vis}(s|s_0,\pi) = \sum_{t\in S} T(t,\pi(t),s)\cdot P_{vis}(t|s_0,\pi)$$

with conditions:

$$\forall \text{ terminal state } g \; P_{vis}(g|g,\pi) = 1$$

$$P_{vis}(s_0|s_0,\pi) = 1$$

Admissibility conditions of a policy

Goal states

Definition 2 *We say that a policy π is* **proper and complete** *when: $P_{vis}(\mathcal{G}|s_0,\pi) = \sum_{g\in\mathcal{G}} P_{vis}(g|s_0,\pi) = 1$*

Forbidden states

Definition 3 *We say that a policy π is* **safe** *when*

$$P_{vis}(S_f|s_0,\pi) = 0$$

$$P_{vis}(S_f|\pi) = \sum_{f\in S_f} P_{vis}(f|s_0,\pi)$$

Thus, $\forall f \in S_f, P_{vis}(f|s_0,\pi) = 0$

Admissible policy

Definition 4 *We say that a policy π is admissible when π is* **proper** *and* **complete** *and when it is* **safe**. *More formally:*
$\forall \pi : \pi$ *is admissible when*

$$P_{vis}(\mathcal{G}|s_0,\pi) = 1 \;\; and \;\; P_{vis}(S_f|s_0,\pi) = 0$$

When admissible policies don't exist, we consider policies that maximize the probability of reaching a goal in $\mathcal{G}$ and minimize the probability of visiting S_f. To this end, we define value functions $V^{\mathcal{G},\pi}$ and $V^{S_f,\pi}$ as follows:

$$V^{\mathcal{G},\pi} = P_{vis}(\mathcal{G}|s_0,\pi)$$

$$V^{S_f,\pi} = P_{vis}(S_f|s_0,\pi)$$

We present in the following sections methods to solve these equations when an admissible policy doesn't exist.

Optimization conditions

Desired states

Definition 5 Desired states *are the states whose visitation is preferred. We say that a policy is satisfying when the expected number of visitation $\mathcal{N}_{vis}$ of S_d is maximized. The expected number of visitation $\mathcal{N}_{vis}$ of S_d when starting at state s_0 and following a policy π is as follows:*

$$\mathcal{N}_{vis}(S_d|s_0,\pi) = \sum_{s\in S_d} P_{vis}(s|s_0,\pi)$$

Undesirable states

Definition 6 Undesirable states *are the states for which avoiding visitation is preferred. We say that a policy is satisfying when the expected number of visitation $\mathcal{N}_{vis}$ of S_u is minimized. The expected number of visitation $\mathcal{N}_{vis}$ of S_u when starting at state s_0 and following a policy π is as follows:*

$$\mathcal{N}_{vis}(S_u|s_0,\pi) = \sum_{s\in S_u} P_{vis}(s|s_0,\pi)$$

Perfect satisfying condition

Definition 7 *We say that a policy π is* **perfectly satisfying** *when the following conditions hold:*
$[\forall \pi : \pi$ *is perfectly satisfying when*

$$\pi^* = argmin_\pi \mathcal{N}_{vis}(S_u|s_0,\pi) \text{ and}$$
$$\pi^* = argmax_\pi \mathcal{N}_{vis}(S_d|s_0,\pi)$$

Solving these equations can lead to many solutions or none. When there are many solution policies, we have to use the value function to select solution policies that maximize the value. However, the more interesting and more likely case is that no solution is available for the above two equations. We introduce a multi-criteria optimization technique to deal with such cases.

Partial Policy

Definition 8 Desirable actions at some states *advice recommend some actions at some states, such as "by night take the highway". Such advice is considered a partial definition of the policy π_{advice} which specifies the actions to perform at some states defined by the set S_{advice}. For such advice, the policy to follow is defined as follows:*

$$\forall s \in S_{advice}, \pi(s) = \pi_{advice}(s)$$
$$\forall s \in S - S_{advice},$$
$$\pi(s) = argmax_{a\in A}(R(s) + \sum_{s'} P(s,a,s')\cdot V(s'))$$

Overall principles and algorithms

An MDP with an advice $< \mathcal{G}, S_f, S_d, S_u, \hat{\pi} >$ can be seen as a Multi-criteria MDP where criteria have a partial order to be respected such that we should solve the MDP considering the constraints in the following order $\hat{\pi} > (\mathcal{G}, S_f) > (S_d, S_u)$. Thus, solving an MDP with advice consists of:

1. Considering $\hat{\pi}$ as a constraint on possible policies;
2. Verifying if policies exist satisfying $\mathcal{G}, S_f$ with potability 1, named admissible;
3. if admissible policies exist then satisfying S_d, S_u by maximizing $\mathcal{N}_{vis}$ of S_d and minimizing $\mathcal{N}_{vis}$ of S_u using a multi-criteria optimization based on a regret-based algorithm inspired by OWR (Roijers, Vamplew, and Whiteson 2013).
4. If admissible policies don't exist, select policies maximizing the probability to visit a goal in $\mathcal{G}$ and minimizing the probability of visiting states in S_f and then optimize S_d, S_u. To this end, we use the OWR approach where regret criteria on $\mathcal{G}$ and S_f satisfaction and on S_d, S_u satisfaction are computed and a lexicographic order over these two regret criteria is used. The use of this lexicographic method over ordered criteria guarantees a fair optimization (Roijers, Vamplew, and Whiteson 2013).

In the next sections, we present different algorithms to satisfy advice using a labeling algorithm to efficiently determine whether admissible policies exist or not and then we use a regret-based algorithm to optimally satisfy the other constraints.

Solving an MDP with $(\mathcal{G}, S_f)$ constraints when an admissible policy exists

When the advice comes in the form of (G, S_f) constraints, the solving algorithms of the MDP should derive a policy respecting these constraints. To do that, we propose the following labeling and pruning algorithm.

The algorithm we propose labels states and actions according to the satisfaction of $(\mathcal{G}, S_f)$ constraints. To this end, we use the following principle: (1) all terminal goal states are labeled $+1$; (2) all forbidden states are labeled -1 ; (3) all terminal non-goal states are labeled -1. For other states, we use the following state and action labeling approach:

1. Assign a label $+1$ to goal terminal states and -1 to non-goal terminal states and forbidden states.
2. Assign a label $+1$ to an action when it leads to states labeled $+1$. Otherwise, the label of the action is -1. This is a $\min$ operator over the label of reached states with this action.
3. Assign to states the label $\max$ of the labeled action. This means that once an action ends at a goal state without visiting forbiden states this action is labeled $+1$.
4. prune all states labeled -1 and their corresponding actions.

More formally: $\forall s \in S, \forall a \in A$,

$$Label(s) = \max_{a \in A} Label(s, a)$$

input : S, A, $(\mathcal{G}, S_f)$
output : Labeled safe state and action spaces

for $s \in S_f$ **do**
  $Label(s) = -1$
end
for *terminal state s* **do**
  if $s \in \mathcal{G}$ **then**
    $Label(s) = +1$
  end
  $Label(s) = -1$
end
for $s \in S$ **do**
  for $a \in A$ **do**
    $Label(a) = \min_{s' \in S:T(s,a,s')>0} Label(s')$
  end
  $Label(s) = \max_{a \in A} Label(a)$
end
Return Labeled A and Labeled S

Algorithm 1: The $(\mathcal{G}, S_f)$-labeling algorithm

$$\forall a \in A, \forall s \in S, Label(s, a) = \min_{s' \in S_{safe}: T(s,a,s')>0} Label(s')$$

Thus,

$$Label(s) = \max_{a \in A} \min_{s' \in S: T(s,a,s')>0} Label(s')$$

Theorem 1 *An admissible policy exists when the label of the initial state is* $+1$.

Proof Let s_0 be the initial state and $Label(s_0) = +1$

$$Label(s^0) = \max_{a \in A} Label(a) = +1$$

Thus $Label(a) = +1$

$$Label(a) = \min_{s^1 \in S: T(s_0,a,s_1)>0} Label(s_1) = +1$$

Thus, $Label(s_1) = +1$ Iteratively, we can say $t \in \{0 \dots H\}$, we have $Label(s_t) = +1$ and $Label(a_1) = +1$. Note that the only states s_H with label +1 are the goal states, thus $s_H \in \mathcal{G}$. Then, when the label of initial state is +1, all the trajectories $\tau = \{s_0, a_0, s_1, a_1, \dots, s_{H-1}, a_{H-1}, s_H = goal$ have states and actions labeled +1. Then, any policy π following these trajectories will reach a state $s_H = goal$. Thus $P_{vis}(s_H = goal | s_0, \pi) = 1$. The policy π is proper and complete.

Assume that π is not safe. This means that $P_{vis}(s_f \in S_f | s_0, \pi) \neq 0$. $P_{vis}(s_f \in S_f | s_0, \pi) \neq 0$ and $P_{vis}(s_f \in S_f | s_0, \pi) = \sum_{t \in S} T(t, \pi(s_f), s_f) P_{vis}(t | s_0, \pi) \rightarrow T(t, \pi(t), s_f) > 0$. By definition, $Label(\pi(t)) = +1$, then $Label(\pi(t)) = \min_{s' \in S: T(t,\pi(t),s')>0} Label(s') = 1 \rightarrow Label(s') = 1$, thus $Label(s_f) = 1$. This contradicts the fact that $s_f \in S_f$ and $Label(s_f)$ should be -1. Thus π is safe.

Solving an MDP with $(\mathcal{G}, S_f)$ with no admissible policy

In this section, we discuss the situation where an admissible policy doesn't exist. In this case, we use the labeling algorithm to guide the search of the best satisfying policy. To define the best satisfying policy, we define the so-called ideal

input : Labeled S, Labeled A
output : New state and action spaces
for $t \in \{0 \dots H-1\}$ **do**
  for $s^t \in S$ **do**
    for $a^t \in A$ **do**
      if $label(a^t) = -1$ **then**
        Prune a^t
        for $s^{t+1} : T(s^t, a^t, s^{t+1}) > 0$ **do**
          Prune_subtree_with_root(s^{t+1})
        end
      end
    end
  end
end
Return new state and action spaces

Algorithm 2: The $(\mathcal{G}, S_f)$-Pruning algorithm

policy which maximizes the value of states labeled $+1$ and minimizes, the value of states labeled -1.

Hence, we define $V^{\mathcal{G},*}$ and $V^{S_f,*}$ two ideal values to optimally satisfying $\mathcal{G}$ and S_f as follows :

$$V^{\mathcal{G},*}(s) = \max_{\pi} \sum_{g \in \mathcal{G}} P_{vis}(g|s,\pi) \tag{2}$$

$$V^{S_f,*}(s) = \min_{\pi} \sum_{f \in S_f} P_{vis}(f|s,\pi) \tag{3}$$

Definition 9 *We say that a policy π^* is perfect when its values are the two ideal values $V^{\mathcal{G},*}$ and $V^{S_f,*}$*

Theorem 2 *Any admissible policy is perfect.*

Proof When π is admissible, by definition of the value function, we have: $V^{\mathcal{G},\pi}(s) = 1$, which is the upper bound of $V^{\mathcal{G},*}(s)$ and $V^{S_f,\pi}(s) = 0$, which is the lower bound of $V^{S_f,*}(s)$.

To assess the value of a *non-admissible* policy π we compute its values $V^{\mathcal{G},\pi}$ and $V^{S_f,\pi}$ as follows:

$$V^{\mathcal{G},\pi}(s) = \sum_{s', label(s')=+1} T(s,\pi(s),s') \cdot V^{\mathcal{G},\pi}(s') \tag{4}$$

$$V^{S_f,\pi}(s) = \sum_{s', label(s')=-1} T(s,\pi(s),s') \cdot V^{S_f,\pi}(s') \tag{5}$$

with $\forall g \in \mathcal{G} : V^{\mathcal{G},\pi}(g) = 1$ and $\forall f \in S_f : V^{S_f,\pi}(f) = 0$

We then compare the value vectors ($V^{\mathcal{G},\pi}$, $V^{S_f,\pi}$) and ($V^{\mathcal{G},*}$, $V^{S_f,*}$). This comparison leads to a multi-criteria optimization process. To this end, we propose to use a regret Tchebychev measure to compare policies by preferring a policy over another one when it minimizes the regret. To this end, we consider ($V^{\mathcal{G},*}$, 0) respectively the best value without considering the second criterion ($V^{S_f,\pi}$) and an approximation of the worst value of $V^{\mathcal{G},\pi}$ to compute a regret measure of getting $V^{\mathcal{G},\pi}$ by policy π at a state:

$$regret(V^{\mathcal{G},\pi}) = \frac{V^{\mathcal{G},*} - V^{\mathcal{G},\pi}}{V^{\mathcal{G},*}}$$

with $regret(V^{\mathcal{G},\pi}) = 0$ when $V^{\mathcal{G},*}$ meaning there is no regret when it's impossible to accomplish the goals.

The regret of $V^{\mathcal{S}_f,\pi}$ considers ($V^{\mathcal{S}_f,*}$, 1) respectively the best value without considering the second criterion ($V^{\mathcal{G},\pi}$) and an approximation of the worst value of $V^{\mathcal{S}_f,\pi}$.

$$regret(V^{S_f,\pi}) = \frac{V^{S_f,*} - V^{S_f,\pi}}{V^{S_f,*} - 1}$$

with $regret(V^{S_f,\pi}) = 0$ when $V^{S_f,*} = 1$ meaning there is no regret when it's impossible to avoid forbidden states.

Finally, we compute the Tchebychev measure as follows:

$$regret^{\mathcal{G},S_f}(\pi) = regret(V^{\mathcal{G},\pi}) + regret(V^{S_f,\pi})$$

The best preferred policy is given by:

$$\pi^* = argmin_{\pi} regret^{\mathcal{G},S_f}(\pi)$$

Solving an $\mathcal{A}$-MDP with (S_d, S_u) soft constraints

To exploit the previous algorithms, we propose a new labeling algorithm as follows:

$$\forall s \in S_d : NLabel(s) = +1$$
$$\forall s \in S_u : NLabel(s) = -1$$

and for all states belonging to $\mathcal{G}$ and S_f are neutral and thus we label them accordingly. Thus, we propose, the following new labeling: If an admissible policy doesn't exist then

$$\forall s \in \mathcal{G} \bigcup S_f : NLabel(s) = 0$$

otherwise all states of S_f are pruned and,

$$\forall s \in \mathcal{G} : NLabel(s) = 0$$

and, for the other states we use the $minmax$ labeling algorithm as above.

From this new labeling we can compute the perfectly satisfying policy using the same principle by computing the ideal values to satisfy (S_d, S_u). Similarly to the previous section, we get:

$$V^{S_d,*}(s) = \max_{\pi} \mathcal{N}_{vis}(S_d|s_0,\pi) \tag{6}$$

$$V^{S_u,*}(s) = \min_{\pi} \mathcal{N}_{vis}(S_u|s_0,\pi) \tag{7}$$

and,

$$regret(V^{S_d,\pi}) = \frac{V^{S_d,*} - V^{S_d,\pi}}{V^{S_d,*}}$$

with $regret(V^{S_d,\pi}) = 0$ when $V^{S_d,*} = 0$

$$regret(V^{S_u,\pi}) = \frac{V^{S_u,*} - V^{S_u,\pi}}{V^{S_u,*} - 1}$$

with $regret(V^{S_u,\pi}) = 0$ when $V^{S_u,*} = 1$

$$regret^{S_d,S_u}(\pi) = regret(S_d,\pi) + regret(S_u,\pi)$$

Note that for (S_d, S_u) a simple weighted sum could be considered but to find a good balance of (S_d, S_u), we use the regret measure.

An overall solution method for $\mathcal{A}$-MDPs with advice $(\hat{\pi}, \mathcal{G}, S_f, S_d, S_u)$

Taking the constraint $\hat{\pi}$ into account, consists in redefining the value function where the value of states where the policy is defined are the expectation of applying this policy at these states. More formally,
Let $S_{\hat{\pi}}$ be the states where the policy is given and $S_{\not\hat{\pi}}$ are the other states such that $S_{\not\hat{\pi}} = S - S_{\hat{\pi}}$.
The value function for a policy π where $\hat{\pi}$ is considered is as follows:

$$V^{\pi}(s) = \begin{cases} s \in S_{\hat{\pi}} : R(s) + \sum_{s' \in S} \gamma T(s, \hat{\pi}(s), s') \cdot V^{\pi}(s') \\ s \in S_{\not\hat{\pi}} : R(s) + \sum_{s' \in S} \gamma T(s, \pi(s), s') \cdot V^{\pi}(s') \end{cases} \quad (8)$$

Then, we consider the other constraints as follows. The algorithm uses the $(\mathcal{G}, S_f)$-labeling and pruning algorithm to detect the existence of admissible policy and then considers the (S_d, S_u) soft constraints using the regret-based method to derive an admissible perfectly satisfying policy or a perfect eligible satisfying policy. This works as follows:

1. Assign to each state an action using $\hat{\pi}$ and let NS be the state spaces non-assigned.
2. Use the $(\mathcal{G}, S_f)$-labeling algorithm.
3. If the label of the initial state is $+1$, then optimize according to (S_d, S_u) in NS space by minimizing $regret^{S_d, S_u}$.
4. If the label of the initial state is -1 (no admissible policy exists), then optimize according to $(\mathcal{G}, S_f, S_d, S_u)$ in NS by using a lexicographic order on $(regret^{\mathcal{G}, S_f}, regret^{S_d, S_u})$.

More formally:

$$\pi^* = \begin{cases} \forall s \in S_{\hat{\pi}} : \hat{\pi} \\ \forall s \notin S_{\hat{\pi}} : \\ argmin_{\pi}\ regret_{\pi}^{S_d, S_u} \quad \textbf{if } label(s_0) = +1; \\ arglexmin_{\pi}(regret_{\pi}^{\mathcal{G}, S_f}, regret_{\pi}^{S_d, S_u}) \quad \textbf{otherwise} \end{cases}$$

Experimental Results

Real-robot target application

We have developed a robotic system for exploration and recognition to map and recognize an area and we developed a user-interface, Figure 1, allowing an operator to express its advice. The operator with a simple clic can show the desired, undesired or forbidden location on the map (shown in the middle of the image. This interface allows also the operator to send actions to perform. This system is dedicated to security application where robots evolve in an enemy area and receive advice on the zone to avoid (undesired), to head (desired), not to visit (forbidden) and the destination (goal).

To show how the policy computation are performed in such real examples, a grid and maze based environments have been used to assess the advice-based MDP in comparison with a classical MDP. Different situations have been developed where goals, desired, undesired and forbidden states (cells) have been expressed to evaluate and to compare the policies derived from an $\mathcal{A}$-MDP and from a classical MDP.

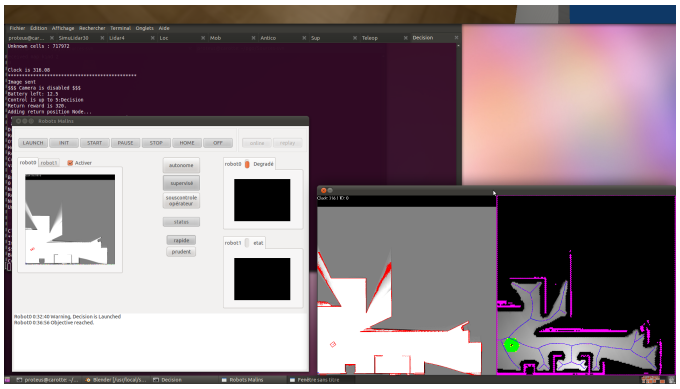

Figure 1: Interface to express advice

Model/Config	1G	1G2D2U	1G4D4U	1G8F	4G	8G
MDP	$25,9$	24	$25,5$	$25,7$	$26,3$	$25,6$
$\mathcal{A}$-MDP	$9,9$	$9,6$	$9,5$	$9,8$	$9,7$	$9,6$

Figure 2: Table summarizing computation time (in seconds) of $\mathcal{A}$-MDP and an MDP for a 40×40 grid with 40 time units (deadline)

Performance

In this section, we evaluate the performance of $\mathcal{A}$-MDP in terms of computation time in comparison with a classical solving algorithm of MDPs to show that our approach computes policies in reasonable time that allow us to consider on-line policy computation. We also show the ability of the approach to scale up well by considering very large Maze and grid environments. We developed experiments with grids 100×100 with 100 time units leading to one million states solved in 3,3 second with our approach and 25s with a classical MDP. In general, our approach scales up well and solves different instances (as show in Table 2) in reasonable time more quickly than a classical MDP. The main reason to that is the fact that with different kind of states (goals, desired, undesired, forbidden) leads to a structured problem which is exploited with labeling and pruning algorithms to speed up the resolution. A summary of results are given in the following table with different configurations (where notation xGyDzUtF, in the table, means x goal states, y desirable states, z undesirable states and t forbidden states):

Results

We conducted a set of experiments in a grid-based environments with different situations. We present the results conducted in an environment where a goal is at the bottom right cell of the grid, two undesired cells in the bottom of the grid and along the frontier of the grid we set a set of desired cells. According to different deadlines, we can notice that when the deadlines are tight (deadline =4, Figure 3) both approaches behave in the same way by reaching the goal but violating the undesired states since the policy leads to visit them. When the deadline is larger (6 time units), Figure 4, our approach allows to reach the goal and not visit-

ing undesired states. As soon as the deadlines become larger (deadlines 8 and 10, Figures 5 and 6), $\mathcal{A}$-MDP behaves in a better way since the policy allows the robot to avoid undesired cells, reaching the goal and visiting some desired states (1 desired state for deadline 8, Figure 5 and 3 desired states for deadline 10, Figure 6). The MDP approach is an average expectation based approach, the undesired states are often visited.

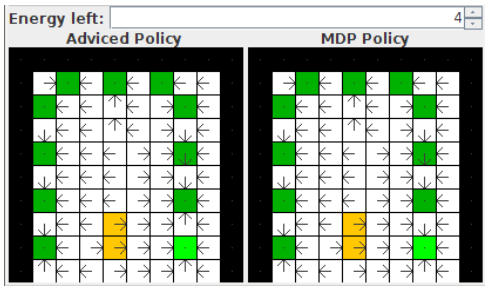

Figure 3: Derived policies from $\mathcal{A}$-MDP and classical MDP with tight deadlines in a grid (4 time units)

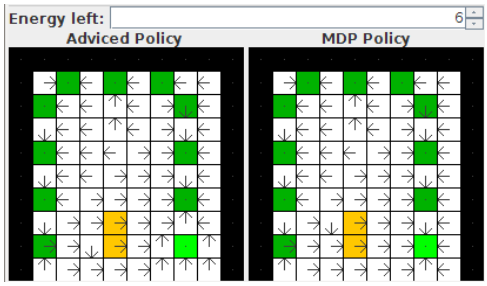

Figure 4: Derived policies from $\mathcal{A}-$MDP and classical MDP with tight deadlines in a grid (6 time units)

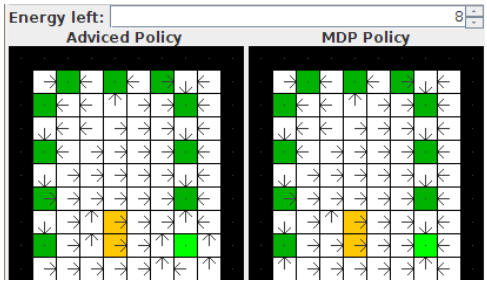

Figure 5: Derived policies from $\mathcal{A}-$MDP and classical MDP with large deadlines in a grid (deadline 8)

We conducted a set of experiments in a Maze-based environments with two situations : first situation concerns one goal state, two desired states and one undesired state and the second situation is similar the first one but we replace the undesired state with forbidden state. In the first situation $\mathcal{A}$-MDP behaves gracefully and leads to a desired policy where the undesired state is often avoided and the desired states are visited (Figure 7). In the second situation, Figure 8, the policy respects the hard constraint (forbidden state) even if this

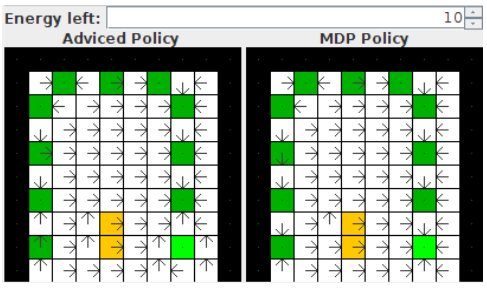

Figure 6: Derived policies from $\mathcal{A}-$MDP and classical MDP with large deadlines in a grid (deadline 10)

avoid the robot to visit the goal state while the MDP-based approach can violate this constraint.

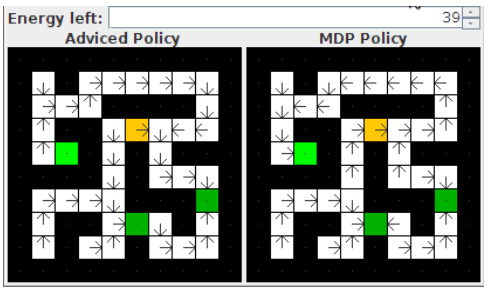

Figure 7: Derived policies from $\mathcal{A}$-MDP and classical MDP with large deadlines in a Maze with 1 goal, 2 desired states and 1 undesired state

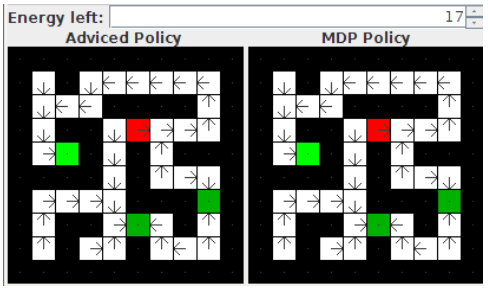

Figure 8: Derived policies from $\mathcal{A}$-MDP and classical MDP with large deadlines in a Maze with 1 goal, 2 desired states and 1 forbidden

Conclusion and future works

In this paper we present a new framework for handling advice in MDP to allow autonomous systems to consider advice from an external entity to overcome difficulties of sensing or acting. This framework presents a definition of advice in terms of goal, desired, undesired and forbidden states and how these advice could be integrated in an MDP and how to transform an MDP with advice into a multi-criteria decision-making problem. Finally, an efficient algorithm to solve such MDPs has been developed, implemented and evaluated in grid and maze environments and how it can be used in an existing robotic system. Experimental results show that $\mathcal{A}$-MDP outperforms classical MDP and that this decision

model is suitable to semi-autonomous systems and bridging the cap between them and full autonomous ones.

Acknolwledgement

This work was supported in part by NSF (USA) and ANR/DGA (France).

References

Bechar, A., and Edan, Y. 2003. Human-robot collaboration for improved target recognition of agricultural robots. *Industrial Robot: An International Journal* 30(5):432–436.

Côté, N.; Canu, A.; Bouzid, M.; and Mouaddib, A.-I. 2012. Humans-robots sliding collaboration control in complex environments with adjustable autonomy. In *Proceedings of Intelligent Agent Technology*.

Dorais, G. A.; Bonasso, R. P.; Kortenkamp, D.; Pell, B.; and Schreckenghost, D. 1999. Adjustable autonomy for human-centered autonomous systems. In *IJCAI Workshop on Adjustable Autonomy Systems*, 16–35.

Goldberg, K.; Chen, B.; Solomon, R.; Bui, S.; Farzin, B.; Heitler, J.; Poon, D.; and Smith, G. 2000. Collaborative teleoperation via the Internet. In *IEEE International Conference on Robotics and Automation*, 2019–2024.

Goodrich, M. A.; Olsen, Jr., D. R.; Crandall, J. W.; and Palmer, T. J. 2001. Experiments in adjustable autonomy. In *IEEE International Conference on Systems, Man and Cybernetics*, 1624–1629.

Hüttenrauch, H., and Severinson Eklundh, K. 2006. Beyond usability evaluation: Analysis of human-robot interaction at a major robotics competition. *Interaction Studies* 7(3):455–477.

Ishikawa, N., and Suzuki, K. 1997. Development of a human and robot collaborative system for inspecting patrol of nuclear power plants. In *Robot and Human Communication*, 118–123.

Mouaddib, A.-I.; Zilberstein, S.; Beynier, A.; and Jeanpierre, L. 2010. A decision-theoretic approach to cooperative control and adjustable autonomy. In *Proceedings of the Nineteenth European Conference on Artificial Intelligence*, 971–972.

Mouaddib, A.-I.; Boussard, M.; and Bouzid, M. 2007. Towards a formal framework for multi-objectiove multi-agent planning. In *International Joint conference on Autonomous and MultiAgent Systems (AAMAS)*, 123–130.

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

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

Rosenthal, S., and Veloso, M. M. 2012. Mobile robot planning to seek help with spatially-situated tasks. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Shiomi, M.; Sakamoto, D.; Kanda, T.; Ishi, C. T.; Ishiguro, H.; and Hagita, N. 2008. A semi-autonomous communication robot: a field trial at a train station. In *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, 303–310. New York, NY, USA: ACM.

Sutton, R. S., and Barto, A. G. 1998. *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press.

Yanco, H. A.; Drury, J. L.; and Scholtz, J. 2004. Beyond usability evaluation: Analysis of human-robot interaction at a major robotics competition. *Human–Computer Interaction* 19(1-2):117–149.