Anytime Algorithms for Solving Possibilistic MDPs and Hybrid MDPs

Bauters, K., Liu, W., & Godo, L. (2016). Anytime Algorithms for Solving Possibilistic MDPs and Hybrid MDPs. In Foundations of Information and Knowledge Systems: 9th International Symposium, FoIKS 2016, Linz, Austria, March 7-11, 2016. Proceedings. (pp. 24-41). (Lecture Notes in Computer Science; Vol. 9616). Springer-Verlag. DOI: 10.1007/978-3-319-30024-5_2

Published in:
Foundations of Information and Knowledge Systems: 9th International Symposium, FoIKS 2016, Linz, Austria, March 7-11, 2016. Proceedings

Document Version:
Peer reviewed version

Anytime Algorithms for Solving Possibilistic MDPs and Hybrid MDPs

Kim Bauters[1], Weiru Liu[1] and Lluís Godo[2]

[1] Queen's University Belfast, Belfast, UK, {k.bauters, w.liu}@qub.ac.uk
[2] IIIA, CSIC, Bellaterra, Spain,godo@iiia.csic.es

Abstract. The ability of an agent to make quick, rational decisions in an uncertain environment is paramount for its applicability in realistic settings. Markov Decision Processes (MDP) provide such a framework, but can only model uncertainty that can be expressed as probabilities. Possibilistic counterparts of MDPs allow to model imprecise beliefs, yet they cannot accurately represent probabilistic sources of uncertainty and they lack the efficient online solvers found in the probabilistic MDP community. In this paper we advance the state of the art in three important ways. Firstly, we propose the first online planner for possibilistic MDP by adapting the Monte-Carlo Tree Search (MCTS) algorithm. A key component is the development of efficient search structures to sample possibility distributions based on the DPY transformation as introduced by Dubois, Prade, and Yager. Secondly, we introduce a hybrid MDP model that allows us to express both possibilistic and probabilistic uncertainty, where the hybrid model is a proper extension of both probabilistic and possibilistic MDPs. Thirdly, we demonstrate that MCTS algorithms can readily be applied to solve such hybrid models.

1 Introduction

A Markov Decision Process (MDP) [2] is a successful framework for dealing with sequential decision problems under uncertainty, particularly when the uncertainty is due to underlying stochastic processes. However, when dealing with uncertainty due to a lack of knowledge it is often easier to find acceptable qualitative estimates. Possibilistic counterparts of MDPs [17], referred to as π-MDP, have been introduced in recent years to tackle this problem. In some situations, optimal strategies to compute the policy of a π-(PO)MDP have even been shown to give better results than their probabilistic counterparts [3]. A limitation of π-MDP, though, is that current solvers for π-MDP rely on offline algorithms to compute the optimal policy. Conversely, state-of-the-art MDP planners are online approximate anytime planners (e.g. [12,15]). Since such planners only have to determine the next best "*enough*" action instead of coming up with a complete optimal policy, they are considerably faster. Furthermore, these online planners are often simpler to integrate with, for example, BDI systems [16] where the online nature fits well with the reactive nature of such systems. MDP and π-MDP also share a common downside: both frameworks only allow for a single

kind of representation for all sources of uncertainty, either all probabilistic, or all possibilistic. This is often at odds with realistic settings, where the underlying causes of uncertainty are diverse. Consider this example:

Example 1. Patrols are being used in a wildlife preserve to deter poachers. Some areas, such as the grassland, have rich statistical data available about the effects of patrolling near herds. These areas are easy to observe and various sensors are installed to monitor the herd movement. In the grassland, we know that moving a patrol to the grassland will prevent poaching in 82% of the cases if the herd is there. Furthermore, in 11% of the situations the herd will move to the marshes and in 7% to the mountains when we have a patrol near them on the grassland. Otherwise, the herd stays on the grassland for the remainder of the day.

For the other areas, only the rangers' experience is available to predict the effectiveness of patrols, and their result on herd movements. For instance, in the mountains we know that it is entirely possible that the herd will stay there or that the herd moves back to the grassland. However, it is only slightly possible that the herd moves to the marches. It is only somewhat possible to deter poaching if we have a patrol around since tracking and finding the herd is hard in the mountains. The terrain with its many hiding spots also makes it entirely possible for poachers to succeed even when we have a patrol nearby.

The rewards themselves can also be quantitative or qualitative in nature. For example, we can easily express that we prefer states in which no animals are poached, but only in the grassland do we have exact numbers of how many animals would be saved by patrolling. □

Modelling examples like these is difficult in MDP or π-MDP as it involves different types of uncertainty. In this paper we advance the state of the art in three important ways. *Firstly*, we adapt the Monte-Carlo Tree Search algorithm used in e.g. UCT [14] and PROST [12] to the π-MDP setting. To achieve this, we present in Section 3.1 a tractable way to sample possibility distributions. The resulting algorithm is the first online planner for π-MDP, applicable to both brave and cautious reasoning. *Secondly*, we propose a hybrid MDP framework where state transitions can be described as either possibilistic or probabilistic distributions. Such a hybrid MDP, which is a proper extension of both MDP and π-MDP, allows a precise and elegant way of modelling problems such as the ones expressed in Example 1. *Thirdly*, by combining the results from our first contribution with classical MCTS we arrive at an MCTS algorithm that can be used to solve hybrid MDP. We furthermore impose rational restrictions on how qualitative and quantitative utilities of trajectories can be combined in order to guide the search. The resulting machinery provides us with a way to solve hybrid MDPs using efficient online anytime algorithms.

The remainder of the paper is organised as follows. Preliminaries are discussed in Section 2. We discuss how to adapt UCT to π-MDP in Section 3, crucially depending on our method to efficiently sample possibilistic distributions. The hybrid MDP model is presented in Section 4, and we also discuss an MCTS algorithm to solve such hybrid models. Related work is discussed in Section 6 and we draw conclusions in Section 7.

2 Preliminaries

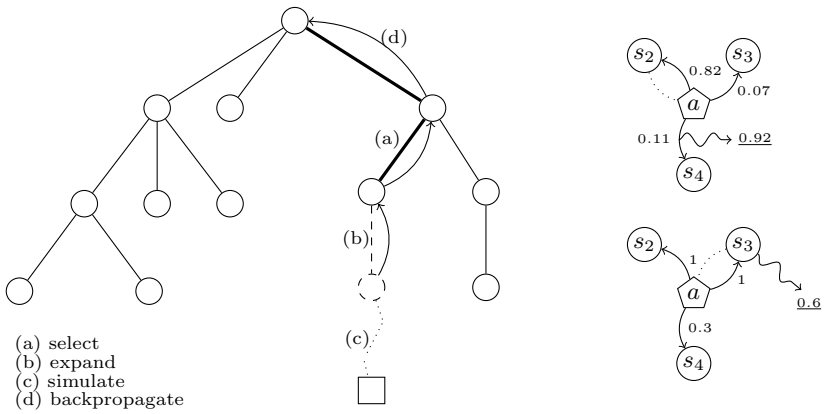
MDP: a probabilistic MDP model is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$. We have that $\mathcal{S}$ is a finite set of *states* and $\mathcal{A}$ is a finite set of *actions*. The *transition function* T gives the probability distributions over states $(s_t, s_{t+1}) \in \mathcal{S}^2$ and an action $a_t \in \mathcal{A}$, such that $T(s_t, a_t, s_{t+1}) = P(s_{t+1} \,|\, s_t, a_t)$, i.e. the (stochastic) uncertainty of reaching the state s_{t+1} from state s_t by taking a_t. A *reward function* R associates the immediate reward value $R(s_t, a_t, s_{t+1})$ with transitioning from state s_t to s_{t+1} by using action a_t. A *discounting factor* $0 \leq \gamma \leq 1$ is used to discount rewards that can only potentially be obtained in the future (or, alternatively, a finite horizon is assumed). A *trajectory* τ is a sequence of states $(s_1, ..., s_h)$, where we use T_h to denote all trajectories of size h. A *policy* (δ) is a sequence of decision rules $\delta : \mathcal{S} \rightarrow \mathcal{A}$ indexed by t, i.e. $\delta(s_t) = a_t$ is the action to execute at time t. The *value*, or quantitative utility, of a policy is given by:

$$v((\delta), s_0) = E\left(\sum_{t=0}^{h-1} \gamma^t \cdot R(s_t, \delta_t(s_t), s_{t+1})\right) \tag{1}$$

with h the *horizon*, $E(\cdot)$ the *expected reward*, and s_{t+1} the stochastic outcome of applying a_t in s_t. A policy applied in the initial state thus describes a trajectory of size h. The optimal policy is the one that maximises $v(\cdot, s_0)$, i.e. at each step t it takes the best available action to maximise the sum of future rewards.

Finding these optimal policies is an intractable problem, so approaches have been developed that can return "*good enough*" actions. One such an approach for solving MDPs is the UCT algorithm [14], which combines Monte-Carlo Tree Search (MCTS) with an upper confidence bound (UCB1) policy. MCTS is an anytime algorithm in which a search tree is built by iteratively sampling the decision space. During each iteration, the algorithm (a) selects an expandable child node; (b) expands this node using an available action; (c) performs a rollout/simulation from the expanded node to a terminal node; and (d) backpropagates the result of the simulation up to the root node to update the statistics for future searches (see Figure 1a). The MCTS algorithm closely relates to Equation 1. The reward of a given trajectory is accumulated during the (c) rollout phase. The probability of the trajectory (implicitly assumed in Equation 1 through the expected value) is respected during the (a) node selection/(b) expansion/(c) rollout. Finally, the total reward from traversing a node/number of times a node has been traversed, are updated during the (d) backpropagation. This allows an approximation of the expected reward of a node to be computed.

The UCB1 policy [1] can considerably speed up MCTS by addressing the exploration-exploitation trade-off during the child node selection. The action to perform in each node is the one that maximises $\overline{X}_j + B\sqrt{(\log n)/n_j)}$ with $\overline{X}_j \in [0, 1]$ the average future reward by taking action a_j, B a bias parameter, n_j the number of times a_j has been selected in this node, and $n = \sum_{j=1}^{k} n_j$, i.e. the total number of times any of the available actions $\{a_1, ..., a_k\}$ in this node have been taken. Intuitively, the term on the left of the sum encourages exploitation, while the term on the right encourages exploring less-visited paths.



(a) basic MCTS procedure

(b) MDP and π-MDP transitions and rewards

Fig. 1: *(left)* basic MCTS procedure with the 4 distinct phases in each iteration *(right)* MDP and π-MDP differences, where transitions are either probabilities or possibilities, rewards are over transitions, and preferences are over states

π-MDP: the π-MDP model [18,17] is the possibilistic counterpart of the MDP model where the uncertainty is modelled as a qualitative possibility distribution. It is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, T^\pi, M, \mathcal{L} \rangle$. Here $\mathcal{L}$ is a *possibility scale*, i.e. a finite and totally ordered set whose *greatest element* is $1_\mathcal{L}$ and whose *least element* is $0_\mathcal{L}$. Typically, it is taken as $\mathcal{L} = \{0, {}^1\!/_k, {}^2\!/_k, ..., {}^{k-1}\!/_k, 1\}$ for some $k \in \mathbb{N}^+$ and it will be required to define the transition function. The possibility distribution over $\mathcal{S}$ is a function $\pi : \mathcal{S} \to \mathcal{L}$ such that $\max_s \pi(s) = 1_\mathcal{L}$, i.e. at least one state is entirely possible. Whenever $\pi(s) < \pi(s')$ it implies that s' is more plausible than s. The transition function T^π is defined over a pair of states $(s_t, s_{t+1}) \in \mathcal{S}^2$ and an action $a_t \in \mathcal{A}$ as $T^\pi(s_t, a_t, s_{t+1}) = \pi(s_{t+1} \,|\, s_t, a_t)$, i.e. the possibility of reaching s_{t+1} conditioned on the current state s_t and the action a_t. This reflects that the uncertainty of the effects of action a_t are due to a lack of information. Furthermore, a function $M : \mathcal{S} \to \mathcal{L}$ models the qualitative utility, or *preference*, of each state (see Figure 1b). The *qualitative utility* of a policy in π-MDP is defined in the *cautious* setting as:

$$u_*((\delta), s_0) = \min_{\tau \in T_h} \max \{1 - \Pi(\tau \,|\, s_0, (\delta)), M(s_h)\} \tag{2}$$

or, in the *brave* setting, as:

$$u^*((\delta), s_0) = \max_{\tau \in T_h} \min \{\Pi(\tau \,|\, s_0, (\delta)), M(s_h)\} \tag{3}$$

with $\Pi(\tau \,|\, s_0, (\delta)) = \min_{t=0}^{h-1} \Big(\pi(s_{t+1} \,|\, s_t, \delta_t(s_t)) \Big)$, i.e. the possibility of the trajectory $\tau = (s_1, ..., s_h)$. The brave setting evaluates a policy based on whether there is at least one possible path that is good. The cautious setting, conversely, evaluates a policy based on how good all of the possible paths are (indeed, it selects

the worst path for its utility). The utility is based on the possibility/necessity of the trajectory which starts from s_0, and the preference of the final state s_h (assuming a finite horizon). Some algorithms have been proposed to compute the solutions of π-MDP models for both decision criteria [17,3]. However, these approaches compute optimal solutions and are therefore only applicable to rather small problem spaces, contrary to the online MCTS algorithm available for MDP.

3 Adapting UCT to π-MDP

We now develop a way to apply UCT, or MCTS in general, to π-MDP. As discussed in Section 2, MCTS builds a search tree by iteratively sampling the decision space. The concept of sampling plays an important role in the rollout phase, but also in the selection and expansion phase due to the non-deterministic nature of the underlying model. In the probabilistic setting, the law of large numbers makes sampling straightforward. In the possibilistic setting, however, we do not have a concept similar to the law of larger numbers. Still, the idea remains similar. Through sampling, we want to select one of the effects of a given action in a π-MDP model, in accordance with the possibility associated with the various effects of that action. This idea is closely related to the idea of transforming a possibility distribution into a probability distribution, where we can sample directly from the latter. A compelling possibilistic-probabilistic transformation, the DPY transformation, was first introduced by Kaufmann in French [10], and later independently by Dubois and Prade [6] and Yager [22]. Not only has this transformation been independently developed by other authors, both in the setting of possibility theory [5,13] and Dempster-Shafer theory [20], but it also has a large number of desirable properties (see [8]). In Section 3.1 we focus on how we can use the DPY transformation to sample a possibility distribution, and how we can do so in a tractable way. In Section 3.2 we look at some of the intricacies of backpropagation and node selection in the π-MDP setting. Together, these components will allow us to use an MCTS-style algorithm to solve π-MDP.

3.1 Possibilistic Sampling

As an initial step towards sampling a possibility distribution, we transform such a distribution into an intermediate data structure with a tractable algorithm:

Definition 1. *Let π be a possibility distribution over $\mathcal{S}$. Let $\mathcal{S}_0$ be those states with a strictly positive possibility, $\mathcal{S}_0 = \{s \mid \pi(s) > 0, s \in \mathcal{S}\}$. Furthermore, we rank order elements in $\mathcal{S}_0$ as $\pi(s_0) \geq ... \geq \pi(s_k)$. Let $\mathcal{C}_\pi$ then be the list of tuples sorted in ascending order on i such that $\langle i, p_i \rangle \in \mathcal{C}_\pi$ whenever $\pi(s_i) > \pi(s_{i+1})$ with $p_i = \pi(s_i) - \pi(s_{i+1})$ and, by default, $\pi(s_{k+1}) = 0$.*

The structure $\mathcal{C}_\pi$ provides a compact representation of the DPY transformation of a possibility distribution π to a basic belief assignment m.[3] Indeed, the tuple

[3] A basic belief assignment, or bba, is a function of the form $m : 2^{\mathcal{S}} \to [0, 1]$ satisfying $m(\emptyset) = 0$ and $\sum_{A \in 2^{\mathcal{S}}} m(A) = 1$.

$\langle i, p_i \rangle$ marks that $\{s_0, ..., s_i\}$ is the α-cut of π with $\alpha = \pi(s_i)$. The value p_i is the probability mass associated with this α-cut in m on the subset $A = \{s_0, ..., s_i\}$.

Example 2. Consider the following possibility distribution π:

$$\pi(s_0) = 1 \qquad \pi(s_1) = 0.7 \qquad \pi(s_2) = 0.7$$
$$\pi(s_3) = 0.3 \qquad \pi(s_4) = 0.1 \qquad \pi(s_5) = 0$$

We have $\mathcal{C}_\pi = (\langle 0, 0.3 \rangle, \langle 2, 0.4 \rangle, \langle 3, 0.2 \rangle, \langle 4, 0.1 \rangle)$.

An algorithm to compute C_π from π is given in Algorithm 1.
Given a compact representation $\mathcal{C}_\pi$, we can readily determine the probability distribution associated with π:

Definition 2. *Let π be a possibility distribution and $\mathcal{C}_\pi$ as in Definition 1. For every $s_i \in \mathcal{S}_0$ the probability of s_i is:*

$$p(s_i) = \sum_{\substack{\langle j, p_j \rangle \in \mathcal{C}_\pi \\ j \geq i}} {}^{p_j}/_{j+1}$$

and $p(s) = 0$ for all $s \in (\mathcal{S} \setminus \mathcal{S}_0)$.

Of course, we only want to sample *according* to the associated probability distribution *without explicitly computing* the distribution. This can be achieved by randomly selecting a $\langle s_i, p_i \rangle \in \mathcal{C}_\pi$ according to probability masses $\mathcal{P}$ using the principle of Probability Proportional to Size (PPS), followed by a random selection with a uniform probability ${}^{1}/_{(i+1)}$ of an element $s \in \{s_0, ..., s_i\}$. This idea is reflected in Algorithm 2.

Example 3. Consider π and $\mathcal{C}_\pi$ from Example 2. Following Definition 2, we have the probability distribution p such that:

$$p(s_0) = 0.5033\ldots \qquad p(s_1) = 0.2033\ldots \qquad p(s_2) = 0.2033\ldots$$
$$p(s_3) = 0.07 \qquad p(s_4) = 0.02 \qquad p(s_5) = 0$$

where e.g. $p(s_0) = {}^{0.3}/_{1} + {}^{0.4}/_{3} + {}^{0.2}/_{4} + {}^{0.1}/_{5} = 0.5033$. In other words: assume a PPS selection of $\langle s_i, p_i \rangle \in \mathcal{C}_\pi$ has been made, followed by a random selection with a uniform probability of $s_k \in \{s_0, ..., s_i\}$. In 50.33...% of the cases, this procedure will return s_0, i.e. $k = 0$.

We now prove that using the compact representation allows for tractable sampling of a possibility distribution.

Proposition 1. *Constructing $\mathcal{C}_\pi$ for π a possibility distribution over $\mathcal{S}$ requires an $O(n \log n)$ algorithm with $n = |\mathcal{S}|$. Sampling of π based on $\mathcal{C}_\pi$ can be done in constant time.*

Proof. Algorithm 1 requires the sorting of the possibility distribution π, with $O(n \log n)$ time complexity, followed by an iteration over all the states in $\mathcal{S}_0$, with $O(n)$ time complexity. Hence, the algorithm is an algorithm with $O(n \log n)$ time complexity. Sampling based on Algorithm 2 relies on a PPS selection and a selection using a uniform distribution, both of which can be implemented with $O(1)$ time complexity [21]. □

input : a possibility distribution π over $\mathcal{S}$
result : a compact representation $\mathcal{C}_\pi$ of the possibility distribution π as given in Definition 1

initialise a list $\mathcal{C}_\pi$
sort π such that $\pi(s_0) \geq \pi(s_2) \geq ... \geq \pi(s_{n-1})$
if $\pi(s_{n-1}) > 0$ **then** $\pi(s_n) \leftarrow 0$
for $i \in [0, k]$ *with* $k = |\mathcal{S}_0|$ **do**
  if $\pi(s_i) > \pi(s_{i+1})$ **then**
    append $\langle i, \pi(s_{i+1}) - \pi(s_i)\rangle$ to $\mathcal{C}_\pi$
end
return $\mathcal{C}_\pi$

Algorithm 1: tractable construction of $\mathcal{C}_\pi$

input : a possibility distribution π over $\mathcal{S}$ and its compact transformation $\mathcal{C}_\pi$
result : a state $s \in \mathcal{S}$

$\langle i, p_i\rangle \leftarrow$ PPS selection from $\mathcal{C}_\pi$ given probability masses $\mathcal{P} = \{p_j \mid \langle j, p_j\rangle \in \mathcal{C}_\pi\}$
$idx \leftarrow$ random selection from $[1, i]$ using uniform probability $1/(i+1)$
return s_{idx}

Algorithm 2: constant time sampling using $\mathcal{C}_\pi$

3.2 Backpropagation and Node Selection

With an efficient way of sampling a possibility distribution, only a few other issues need to be resolved in order to apply a MCTS-style algorithm to π-MDP. The first issue has to do with the information we keep track of, and which we update during the backpropagation. Particularly, each node will need to keep track of a tuple of the form $\langle u_*, u^*\rangle$ instead of a single quantitative reward as in the MDP setting. Here, u_* denotes the cautious qualitative utility of that particular node onwards, while u^* denotes the brave qualitative utility. From Equation 2 and Equation 3 we know that we also have to keep track of the possibility $\Pi(\tau \mid s_0, (\delta))$ of the trajectory τ, i.e. we have to compute $\min_{t=0}^{h-1}\big(\pi(s_{t+1} \mid s_t, \delta_t(s_t))\big)$. This can be achieved during the selection, expansion and rollout phase by keeping track of the minimum of the possibility of the transition for each chance node we encounter.[4] Further in accordance with Equation 2 and Equation 3, once a terminal state is encountered, or the horizon is reached, the preference $M(s_h)$ for this state s_h is determined. Based on the values $\Pi(\tau \mid s_0, (\delta))$ and $M(s_h)$, we can readily compute a tuple $\langle u_*, u^*\rangle$ with $u_* = \max(1 - \Pi(\tau \mid s_0, (\delta)), M(s_h))$ and $u^* = \min(\Pi(\tau \mid s_0, (\delta)), M(s_h))$. This information is then backpropagated where the tuple $\langle {}_j u_*, {}_j u^*\rangle$ of each node j is updated to $\langle \min(u_*, {}_j u_*), \max(u^*, {}_j u^*)\rangle$.

[4] To deal with uncertainty in MCTS, a dual-layered approach is used in the search tree. A *decision node*, or state, allows us to choose which action to perform. A *chance node*, or action, has a number of stochastic effects which are outside our control.

The second issue is how to select the best node to expand during the selection phase. As it turns out, we can readily apply UCB1 for this, even in the possibilistic setting. However, we do have to decide for each search whether we are pursuing a cautious or brave search. For a brave search, we use ${}_ju^*$ instead of $\overline{X}_j$. For a cautious search, we instead use ${}_ju_*$. When the computational budget is reached, an action is selected for which its node is a direct child of the root and there does not exist another such node with a higher quantitative utility ${}_ju^*$ (resp. ${}_ju_*$). An MCTS algorithm to solve π-MDP is given in Algorithm 3.[5]

Note that while a possibility-probability transformation is used for the purpose of sampling a possibility distribution, the induced probability distribution is not used at any other stage. Indeed, during selection/expansion/rollout it is the *possibility* of the trajectory that is computed. This possibility is combined with the preference of the terminal state to determine the *brave and cautious qualitative utility* of the trajectory. Similarly, backpropagation only takes these qualitative utilities into account when updating the node information, and it are these qualitative utilities that guide the choice of the best node to expand during the selection phase. This ensures that no unnecessary transformation bias is introduced.

input : a π-MDP model, $\mathcal{C}_\pi^a$ for every π describing the outcome of an action a, and root state s_0
result : the next best action a' to execute

create root node n with state s_0
while *within computational budget* **do**
  $\Pi(\tau) \leftarrow 1$
  while *n has untried actions and n' has children* **do**
    $a \leftarrow$ `select`(n')
    `/* select next state by sampling` $\mathcal{C}_\pi^a$ `*/`
    $n' \leftarrow$ `sample`$(n, a, \mathcal{C}_\pi^a)$
    $\Pi(\tau) \leftarrow \min(\Pi(\tau), \pi(n' \,|\, n, a))$
    $n \leftarrow n'$
  end
  if *n' has untried actions* **then**
    $a \leftarrow$ select an untried action
    $n'' \leftarrow$ *sample*$(n', a, \mathcal{C}_\pi^a)$ `/* expand node */`
  end
  `/* rollout by sampling from` $\mathcal{C}_\pi$ `as needed */`
  $n_{\texttt{end}} \leftarrow$ `rollout`(n'')
  `backpropagate` $(n_{\texttt{end}}, \max(\Pi(\tau), M(n_{\texttt{end}})))$
end
return *best_action*(n_0)

Algorithm 3: MCTS algorithm for brave π-MDP

[5] An implementation of the algorithm proposed in Algorithm 3 is also available online, at https://github.com/kimbauters/sparsepi .

Proposition 2. *The failure probability at the root, i.e. the probability of selecting a sub-optimal action, converges to* 0 *as the number of samples grows to infinity for MCTS applied to* π*-MDP.*

Proof. (sketch for $u^*(\cdot, s_0)$) Assume $h = 1$, i.e. a search tree with only one level of alternating action and state nodes. We have that the value associated with each action a_i is given by the maximum of the qualitative utility of each outcome of a_i. This utility is in turn given by the minimum of the possibility degree of the trajectory – which corresponds with the possibility of the outcome – and the preference of the final state. Since the qualitative brave utility associated with an action never decreases through repeated sampling, and since repeated sampling will explore all branches as the number of samples grows to infinity, we know that the qualitative utility of the returned action will never decrease after additional samples and that it converges in the limit to the optimal action. Indeed, since we are assuming a finite horizon and since the number of actions/states is finite, we know that the size of the tree is bounded. For $h > 1$ the only difference is the increased size of the trajectory. However, since the possibility $\Pi(\tau \,|\, s_0, (\delta))$ of the trajectory τ as calculated in the algorithm is the same as the one in Equation 3, the results readily hold as well for $h > 1$. □

Initial experimental results confirm the benefit of the online planner for π-MDP. We first consider a problem space where $|\mathcal{S}| = 2^{10}$, $|\mathcal{A}| = 11$, and assume a 50 iteration budget for the online planner. Both the online planner and the offline planner are used to solve the same problem 100 times, where the offline planner first needs to compute its optimal policy but can then repeatedly use that to quickly solve the problems. The online (offline) planner took on average 34.49ms (9.48ms) to find a solution with an average qualitative utility of 0.602 (0.792). In small scale examples like these, the offline planner benefits from being able to apply its policy near-instantaneously once computed to navigate the problem space. When increasing the state space to $|\mathcal{S}| = 2^{12}$ the online (offline) planner took on average 37.44ms (169.96ms) to find a solution with an average qualitative utility of 0.632 (0.762). When $|\mathcal{S}| = 2^{15}$ the online (offline) planner took on average 37.18ms (9961.50ms) to find a solution with an average qualitative utility of 0.658 (0.742). Although clearly these are not conclusive experiments, they already provide indications that the online planner better qualifies to navigate the increasingly large search space, although this comes at the cost of a reduced utility due to the search space not being fully explored. A more comprehensive experimental evaluation falls beyond the scope of this paper, yet is planned for future work.

4 Hybrid MDPs

As shown in Example 1, real-life problems can have different types of uncertainty that we cannot accurately express using only a single kind of representation. Indeed, possibility theory is not good at modelling uncertainty due to underlying stochastic processes, while probability theory is not well-suited for modelling

uncertainty due to a lack of information.[6] Other theories of uncertainty, such as Dempster-Shafer theory [19], are a proper extension of both possibility and probability theory. However, due to their computational complexity, they lend themselves poorly to the use in online anytime algorithms such as MCTS. Algorithms such as MCTS rely on the repeated exploration of different trajectories, which is made feasible by the tractability of the underlying phases (see Figure 1a). Therefore, we instead present a model in which uncertainty can be explicitly expressed as either a possibility or probability.

In the *hybrid MDP model* the actions are partitioned into those with probabilistic and a possibilistic effects, so that each action can be described using the most appropriate representation. In particular, the transition function associates with every action a either a possibilistic, or a probabilistic distribution. Furthermore, we also keep track of a total reward and preference function, which allows us to derive either a qualitative or quantitative utility as needed.

Definition 3. *A hybrid MDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, T, R, M, \gamma, \mathcal{L} \rangle$ such that (i) $\mathcal{S}$ is a finite set of states; (ii) $\mathcal{A}$ is the set of actions, with $\mathcal{A} = \mathcal{A}_P \cup \mathcal{A}_\pi$, where $\mathcal{A}_P$ is the set of actions with probabilistic effects and $\mathcal{A}_\pi$ the set of actions with possibilistic effects; (iii) T is the transition function, where $T(s_t, a_t, s_{t+1})$ is the conditional probability (resp. possibility) of reaching the state s_{t+1} by performing action a_t at state s_t if $a_t \in \mathcal{A}_P$ (resp. if $a_t \in \mathcal{A}_\pi$); (iv) R and M are totally specified reward and preference functions over $\mathcal{S}$; (v) γ is a discounting factor such that $0 \leq \gamma \leq 1$; and (vi) $\mathcal{L}$ is a possibility scale.*

Example 4. Looking back at Example 1, the hybrid MDP model allows us to describe, on the one hand, the effect of patrolling an area on herd movement for the grassland area as a probabilistic transition and, on the other hand, for the mountain area as a possibilistic transition. We have the action *patrol_grassland* with *herd_grassland* as precondition and the three probabilistic effects $\neg$*poaching*, *herd_mountain*, and *herd_marsh* with resp. probability 0.82, 0.11, and 0.07. We also have the action *patrol_mountain* with *herd_mountain* as precondition and the four possibilistic effects *herd_mountain*, *herd_grassland*, $\neg$*poaching*, and *herd_marsh* with resp. possibilities 1, 1, 0.4, and 0.2. Rewards and preferences are fully specified for all state transitions and states, allowing us to express e.g. that preventing a herd from being poached in the grassland is a preferred state, and it results in a reward r (e.g. based on the number of saved animals). In other words, assuming an original state s, an action a, and a state s' where animals are saved, we can have $M(s') = 1$ and $R(s, a, s') = 15$.

A policy in the hybrid MDP setting is defined in the same way as for MDP and π-MDP. However, unlike in MDP or π-MDP, the value of a policy in a hybrid MDP is not given by a single value but by a tuple consisting of both a reward and (brave/cautious) qualitative utility. We have:

[6] A common approach in probability theory to try to overcome this problem is to use subjective probabilities. However, in the more general POMDP/MOMDP settings this creates difficulties in its own right as subjective probabilities from the transitions are then combined with objective probabilities from the observation function.

Definition 4. *The utility w of a policy (δ) in a hybrid MDP model is given by:*

$$w((\delta), s_0) = \begin{cases} \langle v((\delta), s_0), u_*((\delta), s_0)\rangle & \textit{cautious setting} \\ \langle v((\delta), s_0), u^*((\delta), s_0)\rangle & \textit{brave setting} \end{cases}$$

with $v((\delta), s_0)$, and $u_*((\delta), s_0)$ and $u^*((\delta), s_0)$, computed over respectively the MDP and π-MDP induced by the hybrid MDP, as explained next.

An optimal policy for a hybrid MDP is defined in Section 5. Notice that, since we are using both $u(\cdot)$ and $v(\cdot)$ in the policy, the computation of a qualitative/quantitative reward requires an MDP/π-MDP. We thus need an efficient way of deriving the MDP and π-MDP that underlie a hybrid MDP. Obtaining the MDP $\langle \mathcal{S}, \mathcal{A}, T^*, R, \gamma\rangle$ is straightforward. Indeed, $\mathcal{S}$, $\mathcal{A}$, R, and γ are as specified in the hybrid MDP. For those actions $a \in \mathcal{A}_\pi$ that are possibilistic, we discussed in Section 3.1 how we can use the DPY transformation to transform a possibility distribution $\pi(\cdot) = T(s_t, a, \cdot)$, with $a \in \mathcal{A}_\pi$, into a probability distribution $p^*(\cdot) = \textit{yager}(T(s_t, a, \cdot))$. As before, we do not need to explicitly compute the associated probability distribution, as we can have the MCTS algorithm indirectly rely on probabilities through the sampling process. This time around, and contrary to Section 3, the probabilistic sampling of a possibility distribution is taken into account (indirectly, as part of the sampling process) and used to determine the probability of the trajectory.

While deriving the MDP that underlies a hybrid MDP is straightforward, it is more complicated to derive the underlying π-MDP $\langle \mathcal{S}, \mathcal{A}, T^{**}, M, \mathcal{L}\rangle$. A simplifying factor – as in the MDP case – is that $\mathcal{S}$, $\mathcal{A}$, M, and $\mathcal{L}$ are as specified in the hybrid MDP model. Hence, the derivation of the underlying π-MDP only requires a transformation of the probability distributions used in the hybrid MDP model into possibility distributions. Many such transformations exist, applying to either objective or subjective probability distributions. Since we assume in the hybrid MDP model that subjective information is best represented using a possibility distribution, we can conversely assume that we are dealing with an objective probability distribution. As such, the transformation of a probability distribution into a possibility distribution should be based on preserving as much information as possible. One such a transformation [9] is defined as follows. Given a probability distribution p such that $p(s_0) \geq p(s_1) \geq \ldots \geq p(s_k)$ we have that the associated possibility distribution π is defined as $\pi(s_j) = \sum_{i=j,\ldots,k} P(s_i)$. Furthermore, for equiprobable elements it is enforced that the corresponding elements are also equipossible [9]. Computing the associated possibility distribution can be done using an $O(n \log n)$ time complexity algorithm and lookups require $O(\log n)$. As a final step, all results are rounded up to the nearest element in $\mathcal{L}$. In conclusion, for those actions $a \in \mathcal{A}_P$ that are probabilistic, we transform the probability distribution $p(\cdot) = T(s_t, a, \cdot)$, with $a \in \mathcal{A}_P$, into a possibility distribution $\pi^{**}(\cdot) = \textit{dubois}(T(s_t, a, \cdot))$ by using the transformation outlined in this paragraph.

Example 5. Consider the probability distribution p such that

$$p(s_0) = 0.7 \qquad p(s_1) = 0.1 \qquad p(s_2) = 0.1$$
$$p(s_3) = 0.07 \qquad p(s_4) = 0.03 \qquad p(s_5) = 0$$

Assuming $\mathcal{L} = \{{}^0\!/_k, {}^1\!/_k, ..., {}^k\!/_k\}$ with $k = 20$ we have the the resulting π:

$$\pi(s_0) = {}^{20}\!/_{20} \qquad \pi(s_1) = {}^{6}\!/_{20} \qquad \pi(s_2) = {}^{6}\!/_{20}$$
$$\pi(s_3) = {}^{2}\!/_{20} \qquad \pi(s_4) = {}^{1}\!/_{20} \qquad \pi(s_5) = {}^{0}\!/_{20}$$

where π is the possibility distribution associated with p.

We can now treat a hybrid MDP as a π-MDP in a similar way as we treated a hybrid MDP as an MDP. The induced possibility distribution is used to determine the possibility of the trajectory, and the the (qualitative) reward is obtained by ignoring the reward values and only relying on preference values. Importantly, it should be noted that the transformation to a possibility distribution is *only* used to compute the possibility $\Pi(\tau \mid s_0, (\delta))$ of the trajectory τ, and *not* for sampling purposes. This is because, in general, transforming a probability distribution into a possibilistic one and back to a probability distribution does not result in the same distribution, thus introducing an avoidable bias.

5 Solving hybrid MDPs

We show in this section that an online anytime MCTS-style algorithm can be used to solve hybrid MDPs. In particular, we show that the main difficulty lies in the selection phase. Indeed, in the selection phase we need to select the next best node to expand based on the utility of a hybrid MDP, which is a tuple composed of a quantitative and qualitative utility. However, selection strategies – such as UCB1 – require a single value to assess the value of a node. We thus need methods that enforce commensurability and allow us to combine both utility values into a single value $w_\downarrow$. This ties in with Definition 4, where we so far have not defined what an optimal policy is in the hybrid MDP setting. To be able to define an optimal policy, and to define $w_\downarrow$, we propose a number of rationality postulates. These postulates are used to motivate reasonable methods to ensure commensurability. Still, as we will see, some degree of freedom remains. This is to be expected given that the meaning of the qualitative utilities, and especially their inter-relation with quantitative utilities, is domain-dependent.

To simplify the description of the postulates, we assume a straightforward transformation of the quantitative utilities to map them onto the interval $[0, 1]$. The maximum reward, or quantitative utility, is given by $\max_v v((\delta), s_0)$ with (δ) the optimal policy. In a similar way, we can define the worst reward. The transformation then consists of rescaling all rewards accordingly onto $[0, 1]$ where a value of 1 (resp. 0) denotes the most (resp. least) preferred outcome. Notice that while this transformation may appear to give the quantitative utilities a

qualitative interpretation, it does not allow us to directly compare quantitative and qualitative utilities (e.g. they still have different neutral elements[7]).

We can now introduce the postulates that motivate the characteristics of $w_{\downarrow}$.

P1: Let $w(a) = \langle 1, 1 \rangle$, then a is the most preferred action, i.e. $\nexists a' \cdot w_{\downarrow}(a') > w_{\downarrow}(a)$.

When an action has the highest possible value and is qualitatively valued higher than others, then it must be the most preferred action.

P2: Let $w(a) = \langle 0, 0 \rangle$, then a is the least preferred action, i.e. $\nexists a' \cdot w_{\downarrow}(a') < w_{\downarrow}(a)$.

When an action has the lowest possible value and is qualitatively valued lower than others, then it must be the least preferred action.

Postulates **P1-P2** are the base cases with a clear consensus among both utilities. The next two postulates identify the monotonicity w.r.t. a single value change:

P3: Let $w(a) = \langle v, u \rangle$ and $w(a') = \langle v', u' \rangle$ with $v = v'$ and $u > u'$, then $w_{\downarrow}(a) > w_{\downarrow}(a')$.

P4: Let $w(a) = \langle v, u \rangle$ and $w(a') = \langle v', u' \rangle$ with $u = u'$ and $v > v'$, then $w_{\downarrow}(a) > w_{\downarrow}(a')$.

When the qualitative (resp. quantitative) utilities of two actions are the same, the action with the highest quantitative (resp. qualitative) utility must be the preferred one.

Commensurability of the qualitative and quantitative utility is the most difficult when there is a level of disagreement, e.g. a high reward obtained in a disliked state. Dissension between the utilities reaches its maximum when they take on their neutral elements, or the neutral elements of the other utility. In such cases, the exact interpretation of the dissension is dependent on whether we are using brave/cautious reasoning for the qualitative utility:

P5: (brave reasoning, u^*) When $w(a) = \langle 0, 1 \rangle$, then we are ignorant (*eq.* have a weak conflict) about a. When $w(a) = \langle 1, 0 \rangle$, then we have a (strong) conflict about a.

P5′: (cautious reasoning, u_*) When $w(a) = \langle 1, 0 \rangle$, then we are ignorant (*eq.* have a weak conflict) about a. When $w(a) = \langle 0, 1 \rangle$, then we have a (strong) conflict about a.

Ignorance, or the situation with weak conflict, reflects that the values are their own neutral elements and thus convey no real information. Strong conflict suggests that the utilities disagree with each other by taking on the strongest (opposing) values in their scale. In the cautious setting, these notions are flipped around.

[7] We use the terminology of a neutral elements loosely here to indicate that a reward of 0, and a preference of 1, are the defaults. Indeed, when rewards (resp. preferences) are omitted these are the values MDPs (resp. π-MDPs) default to.

Interestingly, postulates **P1-P5** tell us nothing about how to relate two actions a, a' for which $w(a) = \langle 0, 1\rangle$ and $w(a') = \langle 1, 0\rangle$ (i.e. when there is a discord between the utility measures), which is the degree of freedom we have. Simple behaviour can be obtained by e.g. defining $w_{\downarrow}(a) = u^n \cdot v$ with $w(a) = \langle v, u\rangle$. Here, n is a parameter indicating the strength of the qualitative utility over the quantitative utility with values of $n > 1$ placing a higher importance on the qualitative utility. An implicit effect of a formula of this form is that it equates ignorance with conflict. Since we are using a product, we are also stressing the weakest component. However, simple methods like these fail to satisfy a final desirable postulate:

P6: A hybrid MDP is a proper extension of both an MDP and a π-MDP.

When the hybrid MDP only models probabilistic (resp. possibilistic) information, the result of solving the hybrid MDP should be identical to solving the identical MDP (resp. π-MDP).

Postulates **P1-P6** suggest a lexicographic ordering, based on whether we are dealing with brave or cautious qualitative reasoning, and on whether or not qualitative information takes priority over quantitative information.

Definition 5 (lexicographic ordering). *Let $w(a) = \langle v, u\rangle$ and $w(a') = \langle v', u'\rangle$. For a probabilistic lexicographic ordering, we have that $w(a) \geq_p w(a')$ iff $v > v'$, or $v = v'$ and $u \geq u'$. For a possibilistic lexicographic ordering, we instead have $w(a) \geq_\pi w(a')$ iff $u > u'$, or $u = u'$ and $v \geq v'$.*

Definition 6 (optimal policy). *Let $w_{\downarrow}$ be a function agreeing with Definition 5, i.e. such that $w_{\downarrow}(a) \geq w_{\downarrow}(a')$ iff $w(a) \geq w(a')$, where the last ordering is either $\geq_p$ or $\geq_\pi$. Then the optimal policy in a hybrid MDP induced by $w_{\downarrow}$ is the one that maximises $w_{\downarrow}(w(\cdot, s_0))$.*

As previously mentioned, the best choice for $w_{\downarrow}$ is application-specific and depends on the domain, and on the probability-to-possibility transformation used.

Proposition 3. *Let $w_{\downarrow}$ be a probabilistic or possibilistic lexicographic ordering. We then have that $w_{\downarrow}$ satisfies postulates **P1-P6**.*

Proof. It readily follows that postulates **P1** and **P2** are satisfied. Indeed, no action will be less preferred than an action a such that $w(a) = \langle 0, 0\rangle$, or more preferred than an action a' such that $w(a') = \langle 1, 1\rangle$ since $v, u \in [0, 1]$ for $w(\cdot) = \langle v, u\rangle$. Postulates **P3** and **P4** are satisfied directly by the choice of the lexicographic ordering. For a probabilistic lexicographic ordering we get, by definition, that $w(a) \geq_p w(a')$ if $v > v'$ which agrees with **P4**. Equivalently, a possibilistic lexicographic ordering agrees with **P3**. Postulates **P5** and **P5'** do not impose any constraints but follow directly from the underlying theories of uncertainty. Postulate **P6** holds since an MDP (resp. π-MDP) represented as a hybrid MDP can assign the neutral element for preference (resp. reward) to the action. Hence, for an MDP with a probabilistic lexicographic ordering, we get for all actions a that $w(a) = \langle 1, u\rangle$. Hence $w(a) > w(a')$ if and only if $u > u'$. The same line of reasoning applies for a π-MDP with a possibilistic lexicographic. $\square$

In the MCTS algorithm, w is used during node selection to merge the qualitative and qualitative utility of the node into a single value to be used by e.g. UCB1. The other three phases in the MCTS algorithm, expansion, rollout, and backpropagation, also need to be altered slightly. During *expansion/rollout*, both the probability and possibility of a trajectory needs to be computed. To compute the probability of the trajectory we can rely on sampling, which is available for both probabilistic and possibilistic transitions given the results in Section 3.1. To obtain the possibility of a trajectory, specifically when encountering a probabilistic transition during the expansion/rollout phase, the associated possibility of the trajectory is computed as discussed in the previous section. Throughout the *expansion/rollout* we also need to keep track of the rewards and preferences. Both are readily available since the reward and preference functions are total. Once a terminal node is reached, given the possibility/probability of the trajectory as well as the rewards/preferences, the qualitative/quantitative utility is calculated. The *backpropagation* phase then simply combines the backpropagation of both the MDP and π-MDP approaches to update both the qualitative/quantitative utility for each node along the trajectory.

6 Related Work and Future Work

One of the first people to discuss possibilistic MDP is Sabbadin [18,17]. In those works, the author introduces the π-MDP model in which either an optimistic or pessimistic form of reasoning can be used. Applying the optimistic approach might lead to unsatisfactory states, whereas the pessimistic version offers some guarantee that this will not happen. However, as discussed later by Drougard et al. [3], for problems where there is no risk of being blocked in an unsatisfactory state the optimistic version is generally preferred. Furthermore, optimality of an algorithm is easier to prove in the optimistic version. Our implementation of π-MDP demonstrates similar behaviour, where a brave version has a high chance of getting trapped in deadlock states, while the cautious version very often reaches its goals notwithstanding.

Another significant contribution in [17] is that the author also introduced the π-POMDP model, and shows that a finite translation of a π-POMDP to a π-MDP exists. This is unlike the probabilistic setting where no finite translation exists between a POMDP and an MDP, and it allows algorithms used to solve a π-MDP to be used without modifications to also solve π-POMDP. In practice, however, the exponential size of the π-POMDP makes it infeasible to find solutions in a reasonable amount of time for anything but the smallest problems. These problems were addressed in [3], and later in [4], where the authors present the π-MOMDP framework. In this new framework only a subset of the states are partially observable. This significantly reduces the belief space, allowing for optimal algorithms to be applicable in practice. One such algorithm was presented in [4] based on using a factored representation of the original problem. In future work we intend to explore whether the online algorithm presented in the current paper can similarly be applied to π-MOMDP and π-POMDP.

The work in [11] is one of the earliest works to discuss how sampling techniques can be used to solve MDPs. The strength of such techniques lies in their ability to find near-optimal solutions in only a fraction of the time of other approaches. However, only after the seminal work by Kocsis and Szepesvári [14] were sampling-based planning taken seriously in the community. The main improvement proposed in [14] is to employ a multi-bandit approach, as described by the UCB1 procedure [1], to offer an effective balance between exploration and exploitation. This allows to considerably speed up the search process as the most promising part of the tree is more quickly and more profoundly explored. Not long after, the term *Monte-Carlo Tree Search* was coined to describe these sampling-based approaches to planning.

The problem of how to transform a probability distribution into a possibility distribution, and vice versa, has been addressed in a large body of papers (e.g. [22,5,20,13,8,9]). Most generally, a possibility distribution is seen as a (very large) family of probability distributions. The problem therefore boils down to choosing, and motivating, one probability distribution. Intrinsically, such a choice is based on extra information that is external to the possibility distribution. Transforming a probability distribution to a possibility distribution always implies some loss of information, leading to a range of different transformations based on different consistency principles. A concise overview of these and other issues related with both directions of the transformation is given in [7]. How the use of different transformations than the one used in this paper affects the results in a hybrid MDP setting is a topic of interest for future work.

7 Conclusions

This paper introduced a novel approximate way for solving possibilistic MDP (π-MDP) models, based on the established Monte-Carlo Tree Search (MCTS) algorithms for solving MDPs. We found that the applicability of MCTS for solving π-MDP depends on the ability to quickly sample possibility distributions. By introducing a new compact data structure that represents the DPY transformation of a possibility distribution into a probability distribution, we showed that constant time sampling is indeed feasible. Furthermore, we proposed a hybrid MDP model in which we can encode both probabilistic, as well as possibilistic transitions. This allows us to express different facets of uncertainty in the hybrid MDP model. In addition, we showed how a modified version of MCTS can also be applied to solve hybrid MDP models. A central component of this modification is the need to relate the qualitative and quantitative utility. We showed that, while the exact procedure to combine these utilities is application-dependent, such procedures should adhere to a number of rationality postulates. In particular, the postulates enforce that any algorithm to solve a hybrid MDP can also be used to solve either a π-MDP and MDP. Finally, algorithms and computational complexity results of all the main components are presented throughout the paper to highlight the applicability of our approach.

Acknowledgements

This work is partially funded by EPSRC PACES project (Ref: EP/J012149/1). Special thanks to Steven Schockaert who read an early version of the paper and provided invaluable feedback. We also like to thank the reviewers for taking the time to read the paper in detail and provide feedback that helped to further improve the quality of the paper.

References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine Learning 47(2-3), 235–256 (2002)
2. Bellman, R.: A Markovian decision process. Indiana University Mathematics Journal 6, 679–684 (1957)
3. Drougard, N., Teichteil-Königsbuch, F., Farges, J., Dubois, D.: Qualitative possibilistic mixed-observable MDPs. In: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI'13) (2013)
4. Drougard, N., Teichteil-Königsbuch, F., Farges, J., Dubois, D.: Structured possibilistic planning using decision diagrams. In: Proceedings of the 28th AI Conference on Artificial Intelligence (AAAI'14). pp. 2257–2263 (2014)
5. Dubois, D., Prade, H.: On several representations of an uncertain body of evidence. Fuzzy information and decision processes pp. 167–181 (1982)
6. Dubois, D., Prade, H.: Unfair coins and necessity measures: towards a possibilistic interpretation of histograms. Fuzzy sets and systems 10(1), 15–20 (1983)
7. Dubois, D., Prade, H.: Possibility theory and its application: Where do we stand? Mathware and Soft Computing 18(1), 18–31 (2011)
8. Dubois, D., Prade, H., Sandri, S.: On possibility/probability transformation. In: Proceedings of the 4th International Fuzzy Systems Association Congress (IFSA'91). pp. 50–53 (1991)
9. Dubois, D., Prade, H., Smets, P.: New semantics for quantitative possibility theory. In: Proceedings of the 6th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU01). Lecture Notes in Computer Science, vol. 2143, pp. 410–421 (2001)
10. Kaufmann, A.: La simulation des sous-ensembles flous. In: Table Ronde CNRS-Quelques Applications Concrètes Utilisant les Derniers Perfectionnements de la Théorie du Flou (1980)
11. Kearns, M., Mansour, Y., Ng, A.: A sparse sampling algorithm for near-optimal planning in large markov decision processes. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI'99). pp. 1324–1231 (1999)
12. Keller, T., Eyerich, P.: PROST: Probabilistic planning based on UCT. In: Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS'12) (2012)
13. Klir, G.: A principle of uncertainty and information invariance. International Journal Of General System 17(2-3), 249–275 (1990)
14. Kocsis, L., Szepesvári, C.: Bandit based monte-carlo planning. In: Proceedings of the 17th European Conference on Machine Learning (ECML'06). pp. 282–293 (2006)

15. Kolobov, A., Mausam, Weld, D.: LRTDP versus UCT for online probabilistic planning. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI'12) (2012)
16. Rao, A., Georgeff, M.: Modeling rational agents within a BDI-architecture. In: Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). pp. 473–484 (1991)
17. Sabbadin, R.: A possibilistic model for qualitative sequential decision problems under uncertainty in partially observable environments. In: Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'99). pp. 567–574 (1999)
18. Sabbadin, R., Fargier, H., Lang, J.: Towards qualitative approaches to multi-stage decision making. International Journal of Approximate Reasoning 19(3), 441–471 (1998)
19. Shafer, G., et al.: A mathematical theory of evidence. Princeton university press Princeton (1976)
20. Smets, P.: Constructing the pignistic probability function in a context of uncertainty. In: Proceedings of the 5th Annual Conference on Uncertainty in Artificial Intelligence (UAI'89). pp. 29–40 (1989)
21. Vose, M.: A linear algorithm for generating random numbers with a given distribution. IEEE Transactions on Software Engineering 17(9), 972–975 (1991)
22. Yager, R.: Level sets for membership evaluation of fuzzy subset. Fuzzy Sets and Possibility Theory : Recent Developments pp. 90–97 (1982)