On Finding Compromise Solutions in Multiobjective Markov Decision Processes

Patrice Perny and **Paul Weng**[1]

Abstract. A Markov Decision Process (MDP) is a general model for solving planning problems under uncertainty. It has been extended to multiobjective MDP to address multicriteria or multiagent problems in which the value of a decision must be evaluated according to several viewpoints, sometimes conflicting. Although most of the studies concentrate on the determination of the set of Pareto-optimal policies, we focus here on a more specialized problem that concerns the direct determination of policies achieving well-balanced tradeoffs. We first explain why this problem cannot simply be solved by optimizing a linear combination of criteria. This leads us to use an alternative optimality concept which formalizes the notion of best compromise solution, i.e. a policy yielding an expected-utility vector as close as possible (w.r.t. Tchebycheff norm) to a reference point. We show that this notion of optimality depends on the initial state. Moreover, it appears that the best compromise policy cannot be found by a direct adaptation of value iteration. In addition, we observe that in some (if not most) situations, the optimal solution can only be obtained with a randomized policy. To overcome all these problems, we propose a solution method based on linear programming and give some experimental results.

1 INTRODUCTION

In many practical applications of Artificial Intelligence such as path planning, game search, preference-based configuration, the comparison of solutions involves several aspects or points of view that must be analyzed to make a decision. This is typically the case of some state-space search problems where any action that transitions from a state to another is evaluated with respect to different criteria, agents or scenarios, thus leading to a multi-dimensional evaluation system, non-necessarily reducible to a scalar reward or cost. For this reason, several vector-valued generalizations of search algorithms have been investigated such as MOA* [16, 12], a multiobjective extension of A^* [10] finding all Pareto-optimal solution paths to reach a goal node from an initial node in an implicit state space graph. In the field of planning under uncertainty, the same type of concern is present. This explains the current interest fot multicriteria or multiagent extensions of Markov Decision Processes [13, 2, 1, 8].

When several objectives must be optimized simultaneously, most of the studies on Markov Decision Processes concentrate on the determination of the entire set of Pareto-optimal solutions, i.e. policies having a cost vector that cannot be improved on a criterion without being downgraded on another criterion. However, the size of the Pareto set is often very large due to the combinatorial nature of the set of deterministic policies, its determination induces prohibitive response times and requires very important memory space as the number of states and/or criteria increases. Fortunately, there is generally no need to determine the entire set of Pareto-optimal policies, but only specific compromise policies achieving a good tradeoff between the possibly conflicting objectives (minimize time, energy consumption, risk...).

This notion of compromise is natural in multiobjective decision support and has counterparts in other optimisation contexts involving several dimensions. For example, in the context of multi-agent optimisation problems, the notion of compromise solution refers to equity or fairness [7]. In the context of optimisation under multiple scenarios, compromise solutions refer to the idea of robustness [11]. The quality of the compromise achieved can be measured using a scalarizing function discriminating between Pareto-optimal solutions [15, 4].

Motivated by such examples, we study in this paper the determination of well-balanced policies in multiobjective Markov Decision Processes. The paper is organized as follows: In Section 2, we recall the basic notions related to Markov decision processes and their multiobjective extension. In Section 3, we discuss the choice of scalarizing functions to generate compromise solutions. This leads us to adopt the augmented Tchebycheff norm as a proper scalarizing function. Section 4 is devoted to the search of Tchebycheff-optimal policies. Finally, Section 5 presents some experimental results showing the effectiveness of our approach in reaching well-balanced policies.

2 BACKGROUND

2.1 Markov Decision Processes

A Markov Decision Process (MDP) [14] is described as a tuple (S, A, T, R) where:

- S is a finite set of states,
- A is a finite set of actions,
- $T\colon S \times A \to \mathbf{Pr}(S)$ is a transition function, giving for each state and each action, a probability distribution over states (in the sequel, we write $T(s, a, s')$ for $T(s, a)(s')$),
- $R\colon S \times A \to \mathbb{R}$ is a reward function giving the immediate reward for executing a given action in a given state.

In this context, a *decision rule* δ is a procedure that determines which action to choose in each state. A decision rule can be *deterministic*, i.e. defined as a function: $\delta : S \to A$, or more generally, *randomized*, i.e. defined as a function $\delta : S \to \mathbf{Pr}(A)$ where $\mathbf{Pr}(A)$ is the set of probability distributions over A.

A *policy* π is a sequence of decision rules $(\delta_0, \delta_1, \ldots, \delta_t, \ldots)$ that indicates which decision rule to apply at each step. It is said to be

[1] LIP6, UPMC, France, email: firstname.surname@lip6.fr

deterministic if all the decision rules are deterministic and *randomized* otherwise. If the same decision rule δ is applied at each step, the policy is said to be *stationary* and is denoted δ^∞.

In standard MDPs, the value of a policy π is defined by a function $v^\pi : S \to \mathbb{R}$, called *value function*, which gives the expected discounted total reward yielded by applying π from each initial state. For $\pi = (\delta_0, \delta_1, \ldots, \delta_t, \ldots)$, they are given by: $\forall s \in S$, $\forall h > 0$, $\forall t = 1, \ldots, h$,

$$\begin{aligned} v_0^\pi(s) &= 0 \\ v_t^\pi(s) &= R(s, \delta_{h-t}(s)) + \gamma \sum_{s' \in S} T(s, \delta_{h-t}(s), s') v_{t-1}^\pi(s') \end{aligned}$$

where $\gamma \in [0, 1[$ is the discount factor. This sequence converges to the value function of π.

In this standard framework, there exists an optimal stationary policy that yields the best expected discounted total reward in each state. Solving an MDP amounts to finding one of those policies and its associated value function. The optimal value function $v^* : S \to \mathbb{R}$ can be determined by solving the *Bellman equations*:

$$\forall s \in S, \;\; v^*(s) = \max_{a \in A} R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') v^*(s') \qquad (1)$$

There are three main approaches for solving MDPs. Two are based on dynamic programming: value iteration and policy iteration. The last approach is based on linear programming. We now recall value iteration and the linear programming approach as they are needed for the exposition of our results.

Value iteration consists in computing the solution of the Bellman equations using the following recursive sequence: $\forall s \in S, \forall t > 0$,

$$v_0^*(s) = 0 \qquad (2)$$

$$v_t^*(s) = \max_{a \in A} R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') v_{t-1}^*(s') \qquad (3)$$

This sequence converges to the optimal value function and the optimal stationary policy can be recovered by a greedy optimization.

VALUE ITERATION

$\forall s \in S, v(s) \leftarrow 0$
repeat
  for all $s \in S$ **do**
    for all $a \in A$ **do**
      $q(s, a) \leftarrow R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') v(s')$
    end for
    $v(s) \leftarrow \max_{a \in A} q(s, a)$
  end for
until convergence of v

An MDP can also be solved by linear programming. The Bellman equations state that value functions satisfying the following inequalities are upper bounds of the optimal value function:

$$\forall s \in S, \forall a \in A, \;\; v(s) \geq R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') v(s')$$

The linear program ($\mathcal{P}$) can then be written as follows:

$$\begin{aligned} \min \quad & \sum_{s \in S} \mu(s) v(s) \\ \text{s.t.} \quad & v(s) - \gamma \sum_{s' \in S} T(s, a, s') v(s') \geq R(s, a) \\ & \forall s \in S, \forall a \in A \end{aligned}$$

where weights μ could be interpreted as the probability of starting in a given state. Any positive μ can in fact be chosen to determine the optimal value function.

Writing the dual of this program ($\mathcal{D}$) is interesting as it uncovers the dynamic of the system:

$$\begin{aligned} \max \quad & \sum_{s \in S} \sum_{a \in A} R(s, a) x(s, a) \\ \text{s.t.} \quad & \sum_{a \in A} x(s, a) - \gamma \sum_{s' \in S} \sum_{a \in A} x(s', a) T(s', a, s) = \mu(s) \\ & \forall s \in S \\ & x(s, a) \geq 0 \quad \forall s \in S, \forall a \in A \end{aligned}$$

To give an interpretation to variables $x(s, a)$, we recall the two following propositions that relate feasible solutions of the dual linear program to stationary randomized policies in the MDP [14].

Proposition 1. *For a policy π, if x^π is defined as:*

$$x^\pi(s, a) = \sum_{t=0}^{\infty} \gamma^t p_t^\pi(s, a) \quad \forall s \in S, \forall a \in A \qquad (4)$$

where $p_t^\pi(s, a)$ is the probability of reaching state s and choosing action a at step t, then x^π is a feasible solution of the dual linear program.

Proposition 2. *If $x(s, a)$ is a solution of the dual problem, then the stationary randomized policy δ^∞, defined by:*

$$\delta(s, a) = \frac{x(s, a)}{\sum_{a \in A} x(s, a)} \quad \forall s \in S, \forall a \in A \qquad (5)$$

defines values $x^{\delta^\infty}(s, a)$ as in Equation 4, that are equal to $x(s, a)$.

Thus, the basic solutions of the dual program $\mathcal{D}$ correspond to deterministic policies. Moreover, the basic solutions of the primal program $\mathcal{P}$ correspond to the value functions of deterministic policies. Those of randomized policies are in the convex hull of those basic solutions. Note that in an MDP, any feasible value function can be obtained with a randomized policy [14].

2.2 Multiobjective MDP

MDP has been extended to take into account multiple dimensions or criteria. A multiobjective MDP (MMDP) is defined as an MDP where the reward function has been replaced by:

- R: $S \times A \to \mathbb{R}^n$ where n is the number of criteria, $R(s, a) = (R_1(s, a), \ldots, R_n(s, a))$ and $R_i(s, a)$ is the immediate reward for criterion i.

Now, a policy π is valued by a value function $V^\pi : S \to \mathbb{R}^n$, which gives the expected discounted total reward vector in each state. To compare the value of policies in a given state s, the basic model adopted in most previous studies [5, 17, 18] is *Pareto dominance* defined as follows: For any two policies π, π', for any state s, $V^\pi(s)$ Pareto-dominates $V^{\pi'}(s)$, denoted $V^\pi(s) \succ_P V^{\pi'}(s)$ if and only if:

$$V^\pi(s) \neq V^{\pi'}(s) \text{ and } \forall i = 1 \ldots n, V_i^\pi(s) \geq V_i^{\pi'}(s) \qquad (6)$$

Due to the incompleteness of Pareto dominance, there may exist several optimal vectors in a given state. For a set $X \subset \mathbb{R}^n$, the set of

Pareto optimal vectors of X is defined by $M(X, \succ_P) = \{x \in X : \forall y \in X, \text{ not } y \succ_P x\}$.

Standard methods for MDPs can be extended to solve MMDPs although some problems remain open for methods based on dynamic programming, as noted by [18]. As it is needed later for the exposition of our results, we recall the linear program that has been proposed by [17] for finding non-dominated solutions in a MMDP.

The dual linear program formulated for MDPs can be extended to a multiobjective linear program as the dynamic of a MDP and that of a MMDP are identical. We only need to change the objective function of $\mathcal{D}$ by its vector counterpart. Thus we obtain the following multiobjective linear program $v\mathcal{D}$:

$$\begin{aligned} \text{v-max} \quad & \sum_{s\in S}\sum_{a\in A} R(s,a)x(s,a) \\ \text{s.t.} \quad & \sum_{a\in A} x(s,a) - \gamma \sum_{s'\in S}\sum_{a\in A} x(s',a)T(s',a,s) = \mu(s) \\ & \qquad \forall s \in S \\ & x(s,a) \geq 0 \quad \forall s \in S, \forall a \in A \end{aligned}$$

where v-max is a vector maximization with respect to Pareto dominance. Looking for all non-dominated solutions can be difficult and time-consuming as there could be many non-dominated solutions. In fact, there exists instances of problems where the number of solutions is exponential in the number of states. We illustrate this point by adapting an example proposed by [9].

Example 1. *Let $N > 0$. Consider the following MMDP defined by (S, A, T, R) where $S = \{0, 1, \ldots, N\}$, $A = \{a, b\}$, $\forall i = 1, \ldots, N-1$, $T(i, a, i+1) = 1$, $T(i, b, i+1) = 1$, and the only non null reward vectors are defined by $R(i, a, i+1) = (2^i, 0)$, $R(i, b, i+1) = (0, 2^i)$. Here, we can take $\gamma = 1$ as state N is absorbing.*

In this example, there are 2^{N+1} stationary deterministic policies. Stationary deterministic policies that only differ from one another on the choice of the action in the last state N have the same value functions as the reward and the transition in those states for both actions are identical. The remaining policies induce 2^N different valuation vectors, of the form $(x, 2^N - 1 - x)$ for $x = 0, 1, \ldots, 2^N - 1$. Those different vectors are in fact all Pareto-optimal as they are on the line $x + y = 2^N - 1$. It is then infeasible in such a case to determine all non-dominated solutions.

Besides, in practice, one is generally only interested in one particular solution among all the non-dominated solutions that gives interesting tradeoffs between all the criteria. A more interesting approach for MMDP would be to directly search for that particular solution instead of finding first all the non-dominated solutions.

3 SEARCH FOR COMPROMISE SOLUTIONS

We introduce the notion of *scalarizing function* that will be used to discriminate between Pareto-optimal vectors in a given state. Formally, a scalarizing function is a function $\psi : \mathbb{R}^n \to \mathbb{R}$ that defines an *overall value function* $v : S \to \mathbb{R}$ from a vector value function $V : S \to \mathbb{R}^n$:

$$v(s) = \psi(V_1(s), \ldots, V_n(s)) \tag{7}$$

3.1 Weighted Sum

The most straightforward choice for ψ seems to be the weighted sum. In this case, $v(s) = \sum_{i=1}^n \lambda_i V_i(s)$ where $\lambda_i > 0, \forall i = 1 \ldots n$ so as to preserve the monotonicity with respect to Pareto dominance. By linearity of the mathematical expectation and the weighted sum, optimizing v is equivalent to solving the standard MDP obtained from the MMDP where the reward function is defined as: $r(s, a) = \sum_{i=1}^n \lambda_i R_i(s, a), \forall s, a$. In that case, an optimal stationary deterministic policy exists and standard solution methods can then be applied. However, using a weighted sum is not a good procedure for reaching balanced solutions as weighted sum is fully compensatory operator that does not encode the idea of balanced solutions. This is well illustrated by the two following examples:

Example 2. *Consider the following MMDP with two criteria using the same valuation scale and characterized by (S, A, T, R) where $S = \{1\}$, $A = \{a, b\}$, $T(1, a, 1) = T(1, b, 1) = 1$, $R(1, a) = (1, 9)$ and $R(1, b) = (5, 5)$. In this MDP, there only exists two deterministic stationary policies depending on the choice of the action in state 1. They are thus denoted a and b respectively. Their value functions are given by: $V^a(1) = (1/(1-\gamma), 9/(1-\gamma))$ and $V^b(1) = (5/(1-\gamma), 5/(1-\gamma))$. If the rewards of the agents are simply summed (i.e. we give equal weights to each criterion), then both policies are optimal and have the same value functions. However, the policy choosing action b yields a much better balanced vector and should be preferred.*

From the previous example, one could think that by choosing appropriate weights, one could reach a balanced solution. This is not the case. There exists instances where well-balanced solutions cannot be obtained by optimizing a weighted sum.

Example 3. *Consider the MDP defined by (S, A, T, R) where $S = \{1\}$, $A = \{a, b, c\}$, $T(1, a, 1) = T(1, b, 1) = T(1, c, 1) = 1$, $R(1, a) = (1, 9)$, $R(1, b) = (4, 4)$, $R(1, c) = (9, 1)$. Here, there are three stationary deterministic policies. They are denoted a, b and c respectively. Their value functions are given by: $V^a(1) = (1/(1-\gamma), 9/(1-\gamma))$, $V^b(1) = (4/(1-\gamma), 4/(1-\gamma))$ and $V^c(1) = (9/(1-\gamma), 1/(1-\gamma))$. By giving equal weights to both criteria, the policy that chooses action a and the one choosing action c are equivalent. As soon as the weights are different, only one of those two policies is optimal and it therefore yields an unbalanced valuation vector. However, the policy choosing action b is not dominated by the other policies. As it yields a much better balanced valuation vector, one could arguably prefer that solution.*

In fact, it is possible to reach balanced solutions that are even better than those corresponding to deterministic policies. Consider Figure 1 which represents value functions of deterministic policies in an initial state for a given MMDP ($n = 2$). If a weighted sum is used to combine the two criteria, the only solutions that can be found are those represented by points a, b and d because they are on the boundary of the convex hull of feasible vectors. However, one could argue that point c brings a better trade-off between the two criteria. Yet, if we consider randomized policies, we can do much better as shown in Figure 2.

The valuation vectors of randomized policies are in the convex hull of the valuation vectors of deterministic policies, represented by the greyed zone. The dark greyed zone represents all feasible valuation vectors that are preferred to point c. The dotted lines linking points a, b and d represent all Pareto-optimal valuation vectors. It is then easy to see that point c is dominated by all the randomized policies that are both in the dark greyed zone and on the dotted line. The next subsection explains how to reach such points.

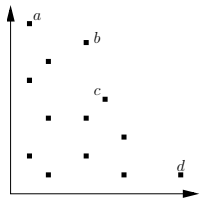


Figure 1. Valuation vectors

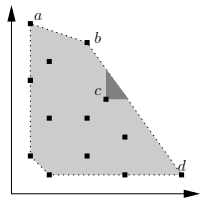


Figure 2. Better solutions

3.2 Tchebycheff Scalarizing Function

In multiobjective optimization, the question of finding balanced solutions among the non-dominated ones is a crucial issue. The standard way of generating compromise solutions in the Pareto-optimal set is resorting to the so-called reference point approach that consists in finding a feasible vector that minimizes a distance to a prescribed reference point [15, 19, 4]. The standard approach for that is to minimize the *Tchebycheff scalarizing function* defined for all $x \in \mathbb{R}^n$ by:

$$\psi(x, r, \lambda) = \max_{i=1\ldots n} \lambda_i |r_i - x_i| + \varepsilon \sum_{i=1\ldots n} \lambda_i |r_i - x_i| \qquad (8)$$

where $r \in \mathbb{R}^n$ is the reference point, $\lambda \in \mathbb{R}^n$ is a positive weighting vector and ε is a positive real chosen arbitrarily small. The best compromise solution $V^{T*} : S \to \mathbb{R}^n$, called *Tchebycheff-optimal*, can then be computed with:

$$V^{T*} = \operatorname*{argmin}_{V} \psi(\sum_{s \in S} \mu(s) V(s), r, \lambda) \qquad (9)$$

where μ is a distribution probability over initial states. To apply this equation, we need to set properly those parameters. Generally, reference point r is taken as the *ideal point*. It can be computed with n different one-dimensional optimizations, that is we solve the MMDP as a standard MDP successively with reward function R_i for $i = 1 \ldots n$. We denote V^{i*} the vectorial value function optimal for the i-th criterion. In a state s, the ideal point then can be formally defined as follows: $v_i^I = \sum_{s \in S} \mu(s) V_i^{i*}(s)$ for all $i = 1 \ldots n$. With the V^{i*}'s, a second point can also be defined: $v_i^A = \sum_{s \in S} \mu(s) \min_{j=1\ldots n} V_i^{j*}(s)$ which is in fact an approximation of the Nadir point, i.e. a lower bound of the Pareto-optimal solutions. We use an approximation of the Nadir point as it is generally difficult to determine exactly [3]. Then the weights are defined as follows:

$$\lambda_i = \frac{w_i}{|v_i^I - v_i^A|} \qquad (10)$$

where $w \in \mathbb{R}^n$ represents the weights of criteria. The construction of a compromise solution using Equations 8-10 is illustrated in Figure 3. The greyed zone represents the feasible solutions. The point I (resp. A) is the ideal point (resp. approximate Nadir point). The best compromise solution is the non-dominated point that is on the line linking I and A when using equal weights w_i. Actually, using Equation 10 with equal weights amounts to projecting point I on the Pareto-optimal frontier in the direction of A. The direction of projection can be modified with weights w_i to get other tradeoffs as illustrated on Figure 4.

As shown in [19], this way of constructing compromise solutions guarantees some nice properties. Contrary to the weighted sum, here, any Pareto-optimal solution can be reached by minimizing the Tchebycheff scalarizing function with a proper choice of w. Moreover, any Tchebycheff-optimal solution is Pareto-optimal. These properties justified the use of Tchebycheff optimality in multiobjective path planning problems [6].

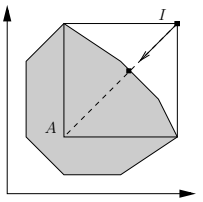


Figure 3. Compromise solution

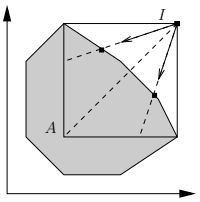


Figure 4. Different weights

Coming back to Example 3, the ideal point in state 1 is $(9/(1-\gamma), 9/(1-\gamma))$, the approximated Nadir point is $(1/(1-\gamma), 1/(1-\gamma))$ and λ can be taken $((1-\gamma)/8, (1-\gamma)/8)$. Then the Tchebycheff-optimal value, which is close to $1/2$ when ε is close to 0, is reached by the policy choosing action b. This simple example explains why the Tchebycheff scalarizing function is preferred to any weighted sum in multiobjective optimization.

As mentioned previously, the quality of compromise solutions found is even better if we consider randomized policies. In the next section, we present the problems that we need to overcome for finding Tchebycheff-optimal (possibly randomized) policies and we propose a solution method for compromise search in MMDPs.

4 SOLUTION METHOD

4.1 Tchebycheff Optimality is State-Dependent

In a standard MDP, an optimal policy is optimal in every initial state. However, here, the optimality notion based on the Tchebycheff function depends on the initial state, i.e. a best compromise policy in a given initial state may not be a best compromise solution in another state. We illustrate this point with a simple example:

Example 4. *Consider an MMDP defined by* (S, A, T, R) *where* $S = \{1, 2\}$, $A = \{a, b\}$, T *and* R *are defined as follows:* $T(1, a, 1) = 1$, $T(1, b, 2) = 1$, $T(2, a, 2) = 1$, $T(2, b, 2) = 1$ *and* $R(1, a) = (0, 6)$, $R(1, b) = (5, 0)$, $R(2, a) = (0, 5)$, $R(2, b) = (2, 2)$. *Take* $\gamma = 0.5$.

Point I *in state* 1 *is* $(7, 12)$, *that in state* 2 *is* $(4, 10)$. *Point* A *is equal to* $(0, 2)$ *in state* 1 *and* $(0, 4)$ *in state* 2. *Thus, the weights are* $(1/7, 1/10)$ *in state* 1 *and* $(1/4, 1/6)$ *in state* 2.

Now, the compromise solution in state 2 *is given by* $(2, 7)$, *obtained by choosing actions* a *or* b *with probability* 0.5. *If we applied this strategy in state* 2, *then the best compromise in state* 1 *is obtained by choosing action* a *with probability* $99/169$ *and action* b *with probability* $70/169$, *which yields approximately* $(3.515, 7.021)$. *The distance to the ideal is equal to* $99/199 \approx 0.498$.

However, if we had chosen to take action a *in state* 2, *we could have obtained a better compromise solution in state* 1. *Indeed by choosing action* a *with probability* $29/64$ *and action* b *with probability* $35/64$, *we could have obtained a valuation vector roughly equals to* $(3.536, 7.051)$ *whose distance to the ideal point equals to* $49/99 \approx 0.495$.

From this observation, we can conclude that the best compromise policy viewed from one state is not necessarily a best compromise solution viewed from another state.

Therefore, to use the Tchebycheff optimality, one needs to know the initial state. This is not, in our opinion, a very demanding requirement as for most problems, this information is available. Moreover, when the initial state is unknown, we can instead consider the average of a value function over all states.

4.2 Dynamic Inconsistency

Due to the non-linearity of the Tchebycheff scalarizing function, we cannot transform the MMDP into a standard MDP. More specifically, solving an MDP obtained by aggregating the vector rewards of an MMDP with the Tchebycheff scalarizing function is not equivalent to optimizing Equation 9 in the original MMDP.

As another consequence of the non-linearity of the scalarizing function, we can no longer rely on algorithms based on dynamic programming, such as value iteration. Indeed, the natural counterpart of value iteration for Tchebycheff optimization should be:

1: $\forall s \in S, V(s) \leftarrow (0, \ldots, 0)$
2: **repeat**
3: **for all** $s \in S$ **do**
4: **for all** $a \in A$ **do**
5: $Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V(s')$
6: **end for**
7: $V(s) \leftarrow \text{argmin}_{a \in A}\, \psi(Q(s,a), v^I(s), \lambda(s))$
8: **end for**
9: **until** convergence of V

where ψ is defined by Equation 8. Let us apply this algorithm on an example.

Example 5. *Consider the MMDP defined by* (S, A, T, R) *where* $S = \{1, 2\}$*,* $A = \{a, b, c\}$*,* T *and* R *are defined as follows:* $T(1, a, 1) = 1$*,* $T(1, b, 1) = 1$*,* $T(1, c, 2) = 1$*,* $T(2, a, 2) = 1$*,* $T(2, b, 2) = 1$*,* $T(2, c, 2) = 1$ *and* $R(1, a) = (0, 5)$*,* $R(1, b) = (5, 0)$*,* $R(1, c) = (1, 1)$*,* $R(2, a) = R(2, b) = R(2, c) = (0, 0)$*. Set* $\gamma = 0.5$*.*

So, state 2 *is an absorbing state and the only valuation vector that can be obtained in that state is* $(0, 0)$*. In state* 1*, the ideal point is defined by* $(10, 10)$*, point* A *by* $(0, 0)$ *and we set the weights to* $(1/10, 1/10)$*.*

Initialize $v_0(1) = (0, 0)$*. The value of applying action* a *in state* 1 *is given by* $Q(1, a) = (0, 5)$*. For action* b*, we get* $Q(1, b) = (5, 0)$*. And finally, for action* c*, we get* $Q(1, c) = (1, 1)$*.* $Q(1, a)$ *and* $Q(1, b)$ *are at a distance of about* 1 *from the ideal point while* $Q(1, c)$ *is at a distance of about* 0.9 *from the ideal point. If we choose action* c*,* $v_1(1) = (1, 1)$*. We can easily see that the next iteration would give the same computations. Therefore, the algorithm would converge in two steps and it gives as optimal valuation vector* $(1, 1)$ *for state* 1*. However, it is easy to see that choosing actions* a *or* b *with probability* 0.5 *would have yield much better rewards with respect to Tchebycheff optimality.*

In this example, the ideal point has been computed at the infinite horizon. We could have also used instead the ideal point for a given horizon. It is easy to check the same problem occurs.

This last observation motivates us to propose a solving method based on linear programming.

4.3 Solving Method

Although the Tchebycheff norm is not linear, the dynamic of an MMDP remains identical to a standard MDP and thus is linear. For finding Tchebycheff-optimal solutions, it is therefore possible to adapt the linear program proposed for finding Pareto-optimal solutions in a MMDP. Compromise solutions can be found with the following non linear program:

$$\begin{aligned}
\min \quad & \max_{i=1\ldots n} \lambda_i\big(\mu.v_i^I - R_i.x\big) + \varepsilon \sum_{i=1\ldots n} \lambda_i\big(\mu.v_i^I - R_i.x\big) \\
\text{s.t.} \quad & \sum_{a \in A} x(s,a) - \gamma \sum_{s' \in S} \sum_{a \in A} x(s',a) T(s',a,s) = \mu(s) \\
& \forall s \in S \\
& x(s,a) \geq 0 \quad \forall s \in S, \forall a \in A
\end{aligned}$$

where $\mu.v_i^I = \sum_{s \in S} \mu(s) v_i^I(s)$ and $R_i.x = \sum_{s \in S} \sum_{a \in A} R_i(s,a) x(s,a)$.

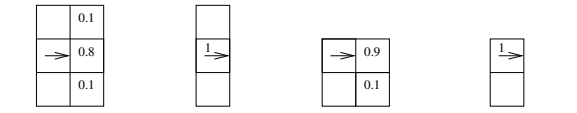
This program can easily be linearized as follows:

$$\begin{aligned}
\min \quad & z + \varepsilon \sum_{i=1\ldots n} \lambda_i\big(\mu.v_i^I - R_i.x\big) \\
\text{s.t.} \quad & z \geq \lambda_i\big(\mu.v_i^I - R_i.x\big) \quad \forall i = 1 \ldots n \\
& \sum_{a \in A} x(s,a) - \gamma \sum_{s' \in S} \sum_{a \in A} x(s',a) T(s',a,s) = \mu(s) \\
& \forall s \in S \\
& x(s,a) \geq 0 \quad \forall s \in S, \forall a \in A
\end{aligned}$$

Our previous observation concerning the state-dependency of the optimality based on best compromise tells us that the Tchebycheff-optimal solution might change with μ, which differs from the classical case. When we do not know the initial state, distribution μ can be chosen as the uniform distribution over the possible initial states. When the initial state s_0 is known, $\mu(s)$ should be set to 1 when $s = s_0$ and to 0 otherwise. The solution found by the linear program does not specify which action to choose for the states that receive a null weight and that are not reachable from the initial state as they do not impact the value of the Tchebycheff-optimal policy.

5 EXPERIMENTAL RESULTS

We tested our solving method on the navigation problem over a grid NxN. In this problem, the robot can choose among four actions: (L)eft, (U)p, (R)ight, (D)own. Figure 5 gives the transitions for action (R)ight. The whole transition function can then be obtained by symmetry.



Moving to three available squares | Moving to a wall | Moving to a corner | Moving to a wall in a corner

Figure 5. Transitions

Rewards are two-dimensional vectors whose components are randomly drawn within interval $[0, 1]$. The discount factor is set to 0.9 and the initial state is set arbitrarily to the upper left corner of the grid. We ran two series of experiments. As in real problems, criteria are generally conflicting. For the first set of experiments, to generate realist random instances, we simulate conflicting criteria with the following procedure: we pick one criterion randomly for each state and action and its value is drawn uniformly in $[0, 0.5]$ and the value of the other is drawn in $[0.5, 1]$. The results over 100 experiments are represented on Figure 6. One point on that figure (a dot when the weighted

sum is used and a circle for the Tchebycheff norm) represents the optimal value function in the initial state for one instance. Naturally, for some instances, the weighted sum can find a balanced solution. But, in most cases, the weighted sum gives a bad compromise solution. Figure 6 actually shows that we do not have any control on tradeoffs obtained with a weighted sum. On the contrary, when using a Tchebycheff norm, the profile of the solutions are always balanced.

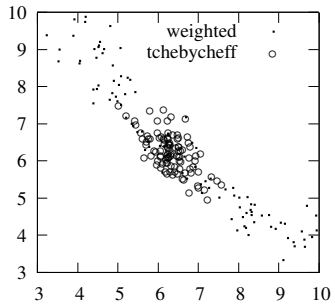


Figure 6. First experiments

To show the effectiveness of our approach, we ran a second set of experiments on pathological instances of the navigation problem. All the rewards are drawn randomly as for the first set of experiments. Then in the initial state, for each action, we choose randomly one of the criteria and add a constant (here, arbitrarily set to 5). Then by construction, the value functions of all deterministic policies in the initial state are unbalanced. Those value functions are represented on Figure 7 with dots as the weighted sum can only reach deterministic policies. Reassuringly, best compromise solutions found by minimizing the Tchebycheff norm are always well-balanced.

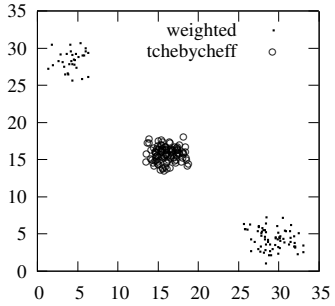


Figure 7. Second experiments

Finally, we give the average execution time in function of the problem size in Table 5. All the experiments were run using CPLEX 12.1 on a PC (Intel Core 2 Quad CPU 2.83Ghz) with 4Go of RAM. The first column n gives the number of objectives. The second column gives the number of states of the problem. Finally, column TW gives the execution time for the weighted sum approach while column TT gives the execution time for the Tchebycheff criterion. All the times are averages over 20 experiments and are given in seconds.

6 CONCLUSION

In this paper, we have presented the search of best compromise solutions in MMDPs with the use of a Tchebycheff distance. Although this is a non-linear criterion, we have provided a LP-solvable formulation of the problem. Experiments have shown the practical feasibility of the approach on difficult instances specially designed to exhibit conflicting criteria. In all experiments, the Tchebycheff criterion significantly outperforms the weighted sum concerning the quality of compromises found. Interestingly enough, this way of incorporating non-linear functions in MMDPs could be extended to other non-linear criteria. For instance, our approach can be applied to multiagent problems with a non-linear social-welfare function to determine policies that fairly share rewards among agents.

n	Size	TW	TT
2	400	0.3	0.5
2	2500	6.1	13.8
2	10000	111.3	319.4
4	400	1.1	1.1
4	2500	6.1	24.6
4	10000	128.7	614.2
8	400	0.3	1.4
8	2500	6.2	42.8
8	10000	131.3	1115.4

Table 1. Average execution time in seconds

REFERENCES

[1] C. Boutilier, 'Sequential optimality and coordination in multiagent systems', in *Proc. IJCAI*, (1999).

[2] K. Chatterjee, R. Majumdar, and T.A. Henzinger, 'Markov decision processes with multiple objectives', in *STACS*, (2006).

[3] M. Ehrgott and D. Tenfelde-Podehl, 'Computation of ideal and Nadir values and implications for their use in MCDM methods', *EJOR*, **151**, 119–139, (2003).

[4] Matthias Ehrgott, *Multicriteria optimization*, Springer, 2005.

[5] N. Furukawa, 'Vector-valued markovian decision processes with countable state space', *Ann. Math. Stat.*, **36**, (1965).

[6] L. Galand and P. Perny, 'Search for compromise solutions in multiobjective state space graphs', in *ECAI*, pp. 93–97, (2006).

[7] B. Golden and P. Perny, 'Infinite order lorenz dominance for fair multiagent optimization', in *AAMAS*, (2010).

[8] C. Guestrin, D. Koller, and R. Parr, 'Multiagent planning with factored mdps', in *NIPS*, (2001).

[9] P. Hansen, *Bicriterion Path Problems*, chapter Bicriterion Path Problems, 109–127, Springer, 1980.

[10] P. E. Hart, N. J. Nilsson, and B. Raphael, 'A formal basis for the heuristic determination of minimum cost paths', *IEEE Trans. Syst. and Cyb.*, **4**(2), 100–107, (1968).

[11] P. Kouvelis and G. Yu, *Robust discrete optimization and its applications*, Kluwer Academic Publisher, 1997.

[12] L. Mandow and J.L. Pérez de la Cruz, 'A new approach to multiobjective A* search', in *IJCAI*, pp. 218–223, (2005).

[13] A.I. Mouaddib, 'Multi-objective decision-theoretic path planning', in *IEEE Int. Conf. Robotics and Automation*, volume 3, pp. 2814–2819, (2004).

[14] M.L. Puterman, *Markov Decision Processes - Discrete Stochastic Dynamic Programming*, John Wiley and Sons, 1994.

[15] R.E. Steuer, *Multiple criteria optimization*, John Wiley, 1986.

[16] B.S. Stewart and C.C. White III, 'Multiobjective A*', *J. ACM*, **38**(4), 775–814, (1991).

[17] B. Viswanathan, V.V. Aggarwal, and K.P.K. Nair, 'Multiple criteria Markov decision processes', *TIMS Studies in the management sciences*, **6**, 263–272, (1977).

[18] D.J. White, 'Multi-objective infinite-horizon discounted Markov decision processes', *Journal of mathematical analysis and applications*, **89**, 639–647, (1982).

[19] A.P. Wierzbicki, 'On the completeness and constructiveness of parametric characterizations to vector optimization problems', *OR Spektrum*, **8**, 73–87, (1986).