Mean Field Approximation of the Policy Iteration Algorithm for Graph-based Markov Decision Processes

Nathalie Peyrard[1] and **Régis Sabbadin**[2]

Abstract. In this article, we consider a compact representation of multidimensional Markov Decision Processes based on Graphs (GMDP). The states and actions of a GMDP are multidimensional and attached to the vertices of a graph allowing the representation of local dynamics and rewards. This approach is in the line of approaches based on *Dynamic Bayesian Networks*. For policy optimisation, a direct application of the *Policy Iteration* algorithm, of exponential complexity in the number of nodes of the graph, is not possible for such high dimensional problems and we propose an approximate version of this algorithm derived from the GMDP representation. We do not try to approximate directly the value function, as usually done, but we rather propose an approximation of the *occupation measure* of the model, based on the *mean field* principle. Then, we use it to compute the value function and derive approximate policy evaluation and policy improvement methods. Their combination yields an approximate *Policy Iteration* algorithm of linear complexity in terms of the number of nodes of the graph. Comparisons with the optimal solution, when available, and with a naive short-term policy demonstrate the quality of the proposed procedure.

1 Introduction

Markov Decision Processes (MDP, [9]) have become a standard model for decision-theoretic planning. However, they reach their limits when dealing with spatial applications (e.g. control in ecology, epidemiology, computer networks...). In such contexts, the size of the problem induces two difficulties: the representation and the time complexity for the MDP resolution. To circumvent the first one, factored representations have been developed. Several methods of aggregation/decomposition for large problems have been proposed in the AI community. These methods ([2, 7]) use a compact representation of probability and reward functions involved in the MDP description and propose algorithms adapted for these factored descriptions. However, these representations and algorithms usually only consider the structure of the states space, not of the actions space.
In this article, we exploit a particular form of compact representation of MDP, based on graphs modelling both states and actions spaces in a common structure, that we will refer to as Graph MDP (GMDP). In this framework, several states and actions variables are attached to the vertices of a graph allowing the representation of local dependencies. The dynamics of each state variable only depends on the action applied locally and on the current values of the neighbouring state variables. The classical *Policy Iteration* algorithm for policy optimisation does not scale to the size of factored MDP or GMDP: the time complexity is exponential in the number of nodes in the graph.

[1] INRA, Avignon - France, email: peyrard@avignon.inra.fr
[2] INRA, Toulouse - France, email: sabbadin@toulouse.inra.fr

We propose an approximate version of the *Policy Iteration* algorithm derived from the GMDP representation and from a *mean field* approximation of the stochastic model, leading to an approximation of the *occupation measure* of the model. Then, we propose approximate policy evaluation and policy improvement steps, based on the occupation measure. Their combination yields an approximate *Policy Iteration* algorithm and we discuss the computational complexity of the proposed method. Then we assess its performance in terms of precision by comparing experimentally the method to the exact *Policy Iteration* algorithm when possible, and to a greedy policy when not, on toy examples.

2 Graph-Based Markov Decision Processes

2.1 Definitions

In its classical formulation [9], a finite stationary MDP is described by a four-tuple $< \mathcal{X}, \mathcal{A}, p, r >$, where $\mathcal{X}$ represents the finite set of admissible states of the system, $\mathcal{A}$ the finite set of applicable actions, $p : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \rightarrow [0, 1]$ the transition probabilities between states and $r : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}$ an "immediate" reward function. In a MDP, the state $x' \in \mathcal{X}$ at step $t + 1$ depends only on the state $x \in \mathcal{X}$ at time t and on the action $a \in \mathcal{A}$ applied at this instant (Markov property). The transition probability from x to x' given a is $p(x'|x, a)$. In addition, a *reward* function r is defined as the reward obtained when action a is applied on the system in the current state. When the process is stationary, both $p(x'|x, a)$ and $r(x, a)$ are independent of t.
In this article, we consider the situation where the state $x \in \mathcal{X}$ is multidimensional, and the coordinates are not independent. They are locally interacting and the interaction network can be represented by a graph. The transition probabilities and the rewards are local according to the graph structure. This representation has already been exploited in [4] : a GMDP is defined by a 5-tuple $< \mathcal{X}, \mathcal{A}, p, r, G >$, the state space is a Cartesian product $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$, and the action space is a Cartesian product $\mathcal{A} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_n$. $G = (V, E)$ is an oriented graph, defined by a set of vertices $V = \{1, \ldots, n\}$ and a set of (oriented) edges $E \subseteq V^2$. An edge (i, j) means that node i influences node j (i is a *parent* of j). A neighbourhood function N is defined over V by the set of parents of a given node :

Definition 1 (Neighbourhood function)
$N : V \rightarrow 2^V$ *is defined by :* $\forall i \in V, N(i) = \{j \in V, (j, i) \in E\}$.

We also introduce the following characteristics of the GMDP: $\sigma = \max_i |\mathcal{X}_i|$, $\alpha = \max_i |\mathcal{A}_i|$ and $\nu = \max_i |N(i)|$ (ν is the maximum degree of a node in the graph).
In a GMDP, transition probabilities and rewards are local according to G:

Definition 2 (Local transitions) *Let* $< \mathcal{X}, \mathcal{A}, p, r, G >$ *be a GMDP. Transitions are said to be* local *iff for all* $x = (x_1 \dots x_n), x' = (x'_1 \dots x'_n) \in \mathcal{X}, a = (a_1 \dots a_n) \in \mathcal{A}$,

$$p(x'|x, a) = \prod_{i=1}^{n} p_i(x'_i|x_{N(i)}, a_i),$$

where $\forall I \subseteq \{1, \dots, n\}, \quad x_I = \{x_i\}_{i \in I}$. With this factored representation, the space complexity of the representation of p is now $O(n \cdot \sigma^{\nu+1} \cdot \alpha)$, instead of $O((\sigma^2 \cdot \alpha)^n)$ for a classical MDP.

Definition 3 (Local rewards) *Let* $< \mathcal{X}, \mathcal{A}, p, r, G >$ *be a GMDP. Rewards are said to be* local *when* $\forall x = (x_1 \dots x_n) \in \mathcal{X}, \forall a = (a_1 \dots a_n) \in \mathcal{A}$,

$$r(x, a) = \sum_{i=1}^{n} r_i(x_{N(i)}, a_i).$$

A function $\delta : \mathcal{X} \to \mathcal{A}$ assigning an action to every state is called a policy. In the general case, policies for a GMDP take the form $\delta = (\delta_1, \dots, \delta_n)$, where $\delta_i : \mathcal{X} \to \mathcal{A}_i$. Nevertheless, global policies can take space in $O(n \cdot \sigma^n)$ at most to be expressed, which can be prohibitive, except for very low dimensionality problems. Some special policies, called *local policies* are of special interest since they take space in $O(n \cdot \sigma^\nu)$.

Definition 4 (Local policy) *In a GMDP* $< \mathcal{X}, \mathcal{A}, p, r, G >$*, a policy* $\delta : \mathcal{X} \to \mathcal{A}$ *is said to be* local *iff* $\delta = (\delta_1, \dots, \delta_n)$ *where* $\delta_i : \mathcal{X}_{N(i)} \to \mathcal{A}_i$.

2.2 GMDP example

Let us consider the graph in Figure 1, representing crop fields infected or not by a disease. Each node i, representing a field, has two possible states : $x_i = 1$ if uninfected and $x_i = 2$ if infected ($\mathcal{X}_i = \{1, 2\}$). Edges represent possible paths for contamination from neighbour locations. The disease passes through an edge with fixed probability p (in this case the graph in non-oriented). In addition, every location has probability ε of being infected through long range contamination (from another region). Decisions are taken yearly and two actions are possible for each node : $a_i = 1$ if a normal culture mode is used, and $a_i = 2$ if the field is left fallow and treated ($\mathcal{A}_i = \{1, 2\}$). Yields are affected by the state of the field and the applied action: the disease halves the yearly yield, while leaving the crop fallow generates no yield. Transition probabilities and rewards are summarised in Table 1. The objective in such context is to optimise the choice of a long-term policy in terms of expected yield.

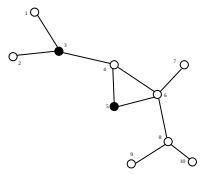

Figure 1. An epidemiological problem state: white (resp. black) nodes represent uninfected (resp. infected) fields.

2.3 Search for an optimal policy

Let $< \mathcal{X}, \mathcal{A}, p, r, G >$ be a GMDP, discounted with discount factor γ. In the stationary case, the problem of searching an optimal policy

$p(x'_i \mid x_{N(i)}, a_i)$				
	$a_i = 1$		$a_i = 2$	
	$x_i = 1$	$x_i = 2$	$x_i = 1$	$x_i = 2$
$x'_i = 1$	$1 - P(\varepsilon, p)$	0	1	q
$x'_i = 2$	$P(\varepsilon, p)$	1	0	$1 - q$

$r(x_i, a_i)$		
	$a_i = 1$	$a_i = 2$
$x_i = 1$	r	0
$x_i = 2$	$r/2$	0

Table 1. Epidemiological problem: transition probabilities and rewards, $P(\varepsilon, p) = \varepsilon + (1 - \varepsilon)(1 - (1 - p)^{n_i})$, with n_i the number of parents of i which are infected.

with respect to the infinite horizon-discounted value of a policy δ, $V_\delta(x) = E\Big[\sum_{t=0}^{+\infty} \gamma^t r(X^t, \delta(X^t)) \mid X^0 = x\Big]$, can be written as:

$$\text{Find } \delta^*, \mathcal{X} \to \mathcal{A}, V_{\delta^*}(x) \geq V_\delta(x), \forall x \in \mathcal{X}, \forall \delta \in \mathcal{A}^{\mathcal{X}}$$

(NB: In this section and in the following, we will use upper case letters to represent random variables, and lower case ones for realisations of these random variables.)

This problem is classically solved by Stochastic Dynamic Programming methods [9] such as the *Backwards Induction*, *Policy Iteration* and *Value Iteration* algorithms.

The *Policy Iteration* (PI) algorithm consists in the alternation of an *evaluation* phase of the current policy and an *improvement* phase of this policy.

Policy Evaluation step: Solve

$$V_\delta(x) = r(x, \delta(x)) + \gamma \cdot \sum_{x' \in \mathcal{X}} p(x'|x, \delta(x)) \cdot V_\delta(x'), \forall x.$$

Policy improvement step: Update the current policy δ into a policy δ' by applying for all x :

$$\delta'(x) = \arg\max_{a \in \mathcal{A}} \Big(r(x, a) + \gamma \cdot \sum_{x' \in \mathcal{X}} p(x'|x, a) \cdot V_\delta(x')\Big).$$

It is guaranteed that $V_{\delta'}(x) \geq V_\delta(x), \forall x$, after the improvement phase. Furthermore when $V_{\delta'}(x) = V_\delta(x), \forall x$ we have the guarantee that δ is an optimal policy [9].

Generally, the PI algorithm converges within a very small number of iterations, but each iteration is costly since policy evaluation needs solving a system of equations (complexity in $O(\sigma^{3n})$) and policy improvement has complexity in $O((\sigma^2 \cdot \alpha)^n)$. This exponential complexity limits severely the size of GMDP that can be efficiently solved. In this paper we propose an approximation of the two steps of the PI algorithm for a GMDP, based on the search of an (a priori sub-optimal) policy among the local ones, which leads to a time complexity linear in n (but exponential in ν).

3 Approximate PI for GMDP

3.1 Policy Value and Occupation Measures

There exists an alternative way of expressing a policy value, based on the occupation measure, which has been used for instance in [1]. The occupation measure is related to the time a stochastic process spends in a given state, knowing the initial state. By extension, occupation measures have also been used in the framework of discounted MDP.

Definition 5 (Occupation measure) *Let* $< \mathcal{X}, \mathcal{A}, p, r >$ *be a stationary MDP,* γ *a discount factor. Let* $\delta : \mathcal{X} \to \mathcal{A}$ *be a policy and*

$x^0 \in \mathcal{X}$ *an initial state. The occupation measure* $P_{x^0,\delta,\gamma} : \mathcal{X} \to [0,1]$ *is defined as:*

$$\forall y \in \mathcal{X}, P_{x^0,\delta,\gamma}(y) = (1-\gamma)\sum_{t=0}^{+\infty} \gamma^t \cdot P_\delta(X^t = y | X^0 = x^0).$$

The value function V_δ of any policy can be computed from the probability distribution $P_{x^0,\delta,\gamma}$ as

$$\forall x^0 \in \mathcal{X}, V_\delta(x^0) = \frac{1}{1-\gamma} \cdot \sum_{y \in \mathcal{X}} P_{x^0,\delta,\gamma}(y) \cdot r(y, \delta(y)).$$

In particular, $\forall \delta$ a local policy, the value function can be decomposed as $V_\delta(x^0) = \sum_{i=1}^n V_\delta^i(x^0)$, where

$$V_\delta^i(x^0) = \sum_{t=0}^{+\infty} \gamma^t \cdot \sum_{y \in \mathcal{X}} P_\delta(X^t = y | X^0 = x^0) \cdot r_i(y_{N(i)}, \delta_i(y_{N(i)})). \quad (1)$$

As such, this expression does not help to compute efficiently optimal or near optimal policies. However, we propose to look for an approximation $\hat{C}$ of the conditional probability $P_\delta(X^t = y | X^0 = x^0)$ with simpler structure, which will result in an approximation of $V_\delta^i(x^0)$ by a function depending only on $x^0_{N(i)}$. Conditional probabilities (from time 0 to time t) can be derived from the transition probabilities $p_i(x_i^t \mid x^{t-1}_{N(i)}, \delta(x^{t-1}_{N(i)}))$. If we are able to find approximate transition probabilities of the form $\hat{Q}_\delta^{t,i}(x_i^t | x_i^{t-1})$ then the corresponding approximate conditional probability will be, $\forall t \geq 2$,

$$\begin{aligned} \hat{C}_\delta^t(y|x^0) &= \sum_{x^{t-1}} \hat{Q}_\delta^t(y|x^{t-1}).\hat{C}_\delta^{t-1}(x^{t-1}|x^0) \\ &= \prod_{i=1}^n \sum_{x_i^{t-1}} \hat{Q}_\delta^{t,i}(y_i|x_i^{t-1}).\hat{C}_\delta^{t-1,i}(x_i^{t-1}|x_i^0). \quad (2) \end{aligned}$$

the recursive definition being initialised by $\hat{C}_\delta^1(y|x^0) = \prod_i \hat{Q}_\delta^{1,i}(y_i|x_i^0)$. And by induction, $\forall t, \hat{C}_\delta^t(y|x^0) = \prod_{i=1}^n \hat{C}_\delta^{t,i}(y_i|x_i^0)$, so that we get $V_\delta^i(x^0) \approx \hat{V}_\delta^i(x^0_{N(i)})$ where

$$\hat{V}_\delta^i(x^0_{N(i)}) = \sum_{t=0}^{+\infty} \gamma^t \cdot \sum_{y_{N(i)}} \left(\prod_{j \in N(i)} \hat{C}_\delta^{t,j}(y_j|x_j^0) \cdot r_i(y_{N(i)}, \delta_i(y_{N(i)})) \right).$$

The main contribution of this work is to propose a solution for the choice of $\hat{Q}_\delta^{t,i}(x_i^t|x_i^0)$, derived from the mean field principle, as detailed in the next section.

3.2 Mean field approximation of the transition probabilities

The mean field principle arises from statistical mechanics [3] and the study of systems of particles in interaction and Gibbs distributions. It has since been successfully used in other domains such as image analysis, graphical models, epidemiology, ecology. The mean field approximation of a multidimensional distribution $P(u_1, \ldots, u_m)$ is defined as the best approximation by a system of independent variables in the sense of the Kullback-Leibler divergence, $KL(Q|P) = E_Q[\log(Q/P)]$. The minimum of $KL(Q|P)$ is thus looked for among the distributions Q with factorisation property: $Q(u_1, \ldots, u_m) = \prod_{i=1}^m Q_i(u_i)$.
In the GMDP framework, the model complexity lies in the spatio-temporal dependence between sites' states. More specifically, if $X^t = \{X_1^t, \ldots, X_n^t\}$ is the random variable corresponding to the system's state at time t then, given a policy δ, the variable X_i^t depends on the variables in its neighborhood at the previous time step, $X_{N(i)}^{t-1}$ through the transition probabilities. So, we propose to apply the mean field principle to the joint distribution of X^{t-1} and X^t, which is fully defined by the distribution of X^{t-1}, and the transition probability of X^t knowing X^{t-1}.
Let us start the iterative approximation procedure at time $t = 1$. We will use the notation P for the exact distributions and the notation Q for the approximate ones. Assuming an initial, factored, distribution P^0 on $\mathcal{X}^0$ (for sake of simplicity and also to ensure the repeatability of the approximation method from one time step to the next one) we have the following forms for the exact and the approximate distributions:

$$P^0(x^0) = \prod_{i=1}^n P^{0,i}(x_i^0), \quad P_\delta(x^1|x^0) = \prod_{i=1}^n p_i(x_i^1 | x^0_{N(i)}, \delta_i(x^0_{N(i)}))$$

$$Q^0(x^0) = P^0(x^0), \quad Q_\delta^1(x^1|x^0) = \prod_{i=1}^n Q_\delta^{1,i}(x_i^1|x_i^0) \quad (3)$$

With the approximate model, given δ, $X^{1,i}$ depends only on $X^{0,i}$. Let us denote $\mathcal{Q}$ the set of all joint distributions of X^{t-1} and X^t which can be decomposed as (3). Among all the elements of $\mathcal{Q}$, we are looking for the best approximation of the exact joint distribution, according to the Kullback-Leibler divergence. Since $Q^0 = P^0$, the approximation is only on Q_δ^1.

Proposition 1 (Mean field approximation of the transitions) *The mean field approximation* $\hat{Q}_\delta^1$*, defined as the solution of*

$$\hat{Q}_\delta^1 = \arg \min_{Q_\delta^1 \in \mathcal{Q}} KL(Q_\delta^1(X^1|X^0).P^0(X^0) | P_\delta(X^1|X^0).P^0(X^0))$$

is

$$\hat{Q}_\delta^{1,i}(x_i^1|x_i^0) \propto \exp E_{P^0}\left[\log p_i(x_i^1|x_i^0, X^0_{N(i)\setminus\{i\}}, \delta_i(x_i^0, X^0_{N(i)\setminus\{i\}}))\right] \quad (4)$$

Proof: Indeed, we have

$$\begin{aligned} & KL(Q_\delta^1(X^1|X^0).P^0(X^0) | P_\delta(X^1|X^0).P^0(X^0)) \\ &= \sum_{x^0, x^1 \in \mathcal{X}^2} Q_\delta^1(x^1|x^0).P^0(x^0).\log \frac{Q_\delta^1(x^1|x^0)}{P_\delta(x^1|x^0)} \end{aligned}$$

Setting the derivates according to the $Q_\delta^{1,i}(x_i^1|x_i^0)$ to zero, we obtain the mean field solution. □

By switching the expectation and the logarithm, we obtain the approximation

$$\hat{Q}_\delta^{1,i}(x_i^1|x_i^0) = E_{P^0}\left[p_i(x_i^1|x_i^0, X^0_{N(i)\setminus\{i\}}, \delta_i(x_i^0, X^0_{N(i)\setminus\{i\}}))\right]$$

It is easy to check that this quantity is normalized. It is easier to compute and more intuitive than expression (4). We will use this approximation in the following as the mean field approximation of the transition probabilities. Note that this factored expression relies on an initial factored a priori distribution P^0. This initial distribution is necessary for technical reasons, but can also help representing a priori knowledge about the initial distribution of the process. If no a priori knwoledge is available, we simply chose P^0 as a product of independent uniform laws on the $\mathcal{X}_i$'s.
Given the approximate transition probabilities between time 0 and

time 1, and given P^0, we can derive the approximate probability distribution of X^1 for a given policy δ:

$$\hat{Q}^1_\delta(x^1) = \sum_{x^0} \hat{Q}^1_\delta(x^1|x^0).P^0(x^0) = \prod_{i=1}^{n} \sum_{x^0_i} \hat{Q}^{1,i}_\delta(x^1_i|x^0_i).P^{0,i}(x^0_i)$$

This approximate distribution at time 1 has also the factorisation property. Thus we can iterate the approximation method (minimisation of the Kullback-Leibler divergence) and obtain for each time step $t \geq 2$ an approximation of the transition probabilities, $\hat{Q}^{t+1}_\delta(x^{t+1}|x^t)$. Note that they depend on t (and on P^0) while this is not the case in the initial model.

3.3 Approximate Policy Evaluation

Using the mean field approximation of the occupation measure to perform the evaluation step, we recall that we obtain an approximation of each component of the policy value of the following form:

$$\begin{aligned} \hat{V}^i_\delta(x) &\approx \hat{V}^i_\delta(x_{N(i)}) \\ &\approx \sum_{t=0}^{+\infty} \gamma^t \cdot \sum_{y_{N(i)}} \Big(\prod_{j \in N(i)} \hat{C}^{t,j}_\delta(y_j|x_j) \Big) \cdot r_i(y_{N(i)}, \delta_i(y_{N(i)})) \end{aligned}$$

In this approximation, each component $\hat{V}^i_\delta$ depends on $x_{N(i)}$ only. This results in a reduced time complexity, exponential in the maximum number of neighbours of a node, as opposed to exponential in the number of nodes for the exact PI.

Proposition 2 (Complexity of the Approximate Policy Evaluation) *Let ϵ be the stopping threshold in the computation of the infinite sum over time. Then the complexity of the Approximate Policy Evaluation step is in $O(\ln(1/\epsilon) \cdot n \cdot \sigma^{3\nu} \cdot \nu^2)$.*

Proof: There are $n \cdot \sigma^\nu$ quantities $\hat{V}^i_\delta(x_{N(i)})$ to compute. Then, for a given site i and a given configuration $x_{N(i)}$, the computation of $\gamma^t \sum_{y_{N(i)}} \Big(\prod_{j \in N(i)} \hat{C}^{t,j}_\delta(y^t_j|x_j) \Big) \Big).r_i(y_{N(i)}, \delta_i(y_{N(i)}))$ requires at most σ^ν summations, ν products, and the evaluation of $\hat{C}^{t,j}_\delta(y^t_j|x_j)$. This last evaluation is of complexity $\nu \cdot \sigma \cdot \sigma^{\nu-1}$: sum over each possible state of site i at the previous time step and computation of the approximated transition probabilities as expectations of products of $\nu+1$ terms. The discount factor γ^t ensures that the precision required is reached for a time $t > K\frac{\ln(\epsilon)}{\ln(\gamma)}$. Finally, the complexity of the Approximate Policy Evaluation is $O(\ln(1/\epsilon) \cdot n \cdot \sigma^{3\nu} \cdot \nu^2)$.□

3.4 Approximate Policy Improvement

In order to reduce the search space, we limit the search of approximately optimal policies among local ones (even if in general there is no guarantee that there exist a local policy which is optimal). This allows a reduction in representation space and in time complexity. Using the mean field approximation of V_δ and the local properties of rewards and transition probabilities, the exact maximisation

$$\delta'(x) = \arg\max_{a \in \mathcal{A}} \Big(r(x,a) + \gamma \cdot \sum_{y \in \mathcal{X}} p(y|x,a) \cdot V_\delta(y) \Big), \forall x.$$

is replaced by the search of the arguments of the maximum of the expressions :

$$\sum_i r_i(x_{N(i)}, a_i) + \gamma \sum_{y_{N(i)}} \Big(\prod_{j \in N(i)} p_j(y_j|x_{N(j)}, a_j) \hat{V}^i_\delta(y_{N(i)}) \Big).$$

Let us consider a particular site i. The terms which depend on a_i are

$$r_i(x_{N(i)}, a_i) + \gamma \sum_{k/i \in N(k)} \sum_{y_{N(k)}} \Big(\prod_{j \in N(k)} p_j(y_j|x_{N(j)}, a_j) \cdot \hat{V}^i_\delta(y_{N(k)}) \Big).$$

In order to ensure that the argument of the maximum is local, first we adopt a parallel scheme, initialised by δ, to compute each component of the improved policy:

$$\forall i, \tilde{\delta}^{n+1}_i(x) = \arg\max_{a_i \in \mathcal{A}_i} r_i(x_{N(i)}, a_i) + \gamma \sum_{k \text{ s.t.} i \in N(k)} \sum_{y_{N(k)}} \Big[\Big(\prod_{j \in N(k), j \neq i} p_j(y_j|x_{N(j)}, \tilde{\delta}^n_j(x_{N(j)})) \Big) \cdot p_i(y_i|x_{N(i)}, a_i) \cdot \hat{V}^i_\delta(y_{N(k)}) \Big].$$

This is not enough since the ith component of the solution still depend on $x_{N(N(i))}$. So, we replace the transition probabilities p_j by appropriate mean values, in order to restrict dependencies, as follows

$$p_i(y_i|x_{N(i)}, a_i) \approx \frac{\sum_{x_{N(i)\setminus\{i\}}} p_i(y_i|x_{N(i)}, a_i)}{card(\mathcal{X}_{N(i)\setminus\{i\}})}$$

if $j \in N(i)$:

$$p_j(y_j|x_{N(j)}, \delta(x_{N(j)})) \approx \frac{\sum_{x_{N(j)\setminus\{j\}}} p_j(y_j|x_{N(j)}, \delta_j(x_{N(j)}))}{card(\mathcal{X}_{N(j)\setminus\{j\}})}$$

if $j \notin N(i)$:

$$p_j(y_j|x_{N(j)}, \delta(x_{N(j)})) \approx \frac{\sum_{x_j} p_j(y_j|x_{N(j)}, \delta_j(x_{N(j)}))}{card(\mathcal{X}_j)}$$

Proposition 3 (Complexity of Approximate Policy Update) *The Approximate Policy Update step is of complexity $O(n.\alpha.\nu^2.\sigma^{2\nu})$.*

Proof: The maximisation is performed for all sites, and is over α terms at most. The computation of the approximate version of expression (**??**) requires at most $\nu.\sigma^\nu$ summations and ν products. One element of the product is computed either in σ^ν or in σ operations. This leads to a complexity of at most $O(n.\alpha.\nu^2.\sigma^{2\nu})$.□

3.5 Experimental evaluation of API

We first tested the mean field approximation of the *Policy Iteration* (MF-PI) algorithm on problems of small size, where the exact solution of the PI algorithm for standard MDP is available, in order to assess the quality of the approximate policies returned by MF-PI. The first toy example used is the one presented in Section 2.2. The parameters were set to $p = 0.2, \varepsilon = 0.01, q = 0.9$. The second example is an extension to four states per node: the state 'healthy' and three states of increasing severity in the case of an infected field.

Example 1			
n	error	CPU MF	CPU PI
3	0%	1.44	10.11
4	6.5%	5.75	65.52
5	3.3%	6.78	454.54
6	10.1%	9.07	3485.6

Example 2			
n	error	CPU MF	CPU PI
3	0%	21.4	646
4	3.6%	574	21200
40		4380	

Table 2. Mean relative error (in percentage) between the optimal policy and the estimated one, and computation time.

In both cases, the graphs were generated randomly, with a maximum size of $N(i)$ of 3. For varying values of n, we observed a limited loss in the quality of the optimal policy (with a mean relative error no more than 10 %, see Table 2) and a significant gain in time complexity (see Figure 2). We observe a linear progression of the CPU time required by MF-PI with respect to the number of nodes (see Figure 2). Implementation was done in Scilab, on a quadriprocessor PC with four Intel 2.40 GHz processors with 512 Ko of cache.

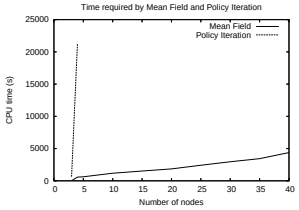

Figure 2. CPU times required by MF-PI and by exact PI

In a second set of experiments, we compared the relative performance of the MF-PI and the greedy (maximising the local immediate rewards) policies for graphs of larger size, on the problem of epidemiology control (version with three states of disease severity). Graph of dependencies were generated randomly and the MF-PI and the greedy policies were computed and their values were estimated using Monte Carlo simulations. Figure 3 illustrates the gain in the empirical value using MF-PI instead of the greedy policy, as a average over 5 problems, 10 initial states per problem, and simulation of 100 trajectories of length 20.

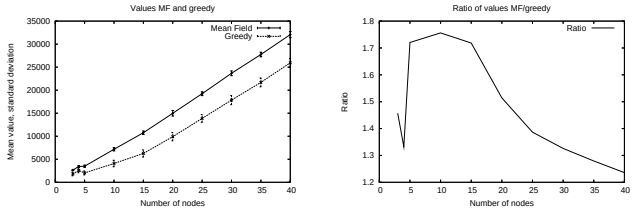

Figure 3. Left: estimated mean values of the MF-PI and the greedy policies, Right: estimated mean value of the ratio of the MF-PI policy value over the greedy policy value.

4 Concluding remarks

In this paper, we described a specific form of factored MDP (FMDP) which uses a graph-based representation of local dependencies: the GMDP framework. The specificities are: first, each decision node is the parent of exactly one state and no two decision nodes are attached to the same state node. second, the scope of each reward node is included in the set of parents of one state variable and no two reward nodes are attached to the same state variable.

An important feature of GMDP is that they admit problem and approximately optimal solution representations in space linear in the size of the underlying graph. However, classical Dynamic Programming algorithms use space and time exponential in the size of the graph to compute optimal solutions. We have proposed an approximate *Policy Iteration* algorithm which requires only linear space and time provided that the graph width remains small. The quality loss associated with this gain in time complexity is not known yet. However, experiments showed that it is limited. In particular, the approximate algorithm greatly improves the greedy short-term solution which is usually the only one available in problems of high dimension.

Our approach should be contrasted to [2, 7, 5] which are also devoted to FMDP. [2] uses *algebraic decision diagrams* (ADD) to model and solve FMDP and [7, 5] use a *linear programming* approach. [5] propose a method to tackle FMDP with multiple decision variables based on the idea of sampling the (exponential size) set of constraints generated by a linear programming (LP) modelling of a FMDP. However, to keep acceptable probabilistic bounds on the quality of the returned policy, the sample size should remain proportional to the number of possible actions, i.e. exponential in the number of decision variables. [7] instead consider the exact LP, but propose a specific *variable elimination* method which take the structure of the LP into account in order to solve it exactly. [7] points out experiments in which his method outperforms that of [5] when the same computation time is allowed to both methods.

[2, 7, 5] can only handle problems with a small number of decision variables and are thus inadequate to solve GMDP. In [8], *collaborative multi-agent factored MDP*, which are more general than GMDP, are studied and approximately solved by a dedicated algorithm. The empirical comparison between this algorithm and ours has not been performed yet, but the complexity of this algorithm is quadratic in the number of decision variables, while ours is linear.

Finally, [6] propose a LP-based method for solving GMDP, derived from the method of [7]. Experiments led so far show that even if this approach differs from ours, the returned policies have similar values, and the CPU times are linearly related. We are currently investigating the benefits of each method, in particular their ability to face the following generalisation of GMDP. An important assumption of the GMDP framework is that the action space $\mathcal{A}$ is a cross product of *independent* local action spaces $\mathcal{A}_i$. However, this assumption should be relaxed ($\mathcal{A}$ becoming a strict subset of $\prod_i \mathcal{A}_i$), as soon as we want to model resource allocation or local constraints between the actions. Relaxing this independence assumption clearly affects the approximation method suggested in this paper and we are currently exploring the effects of this relaxation.

REFERENCES

[1] E. Altman, *Constrained Markov Decision Processes*, Chapman & Hall / CRC, 1999.

[2] C. Boutilier, R. Dearden, and M. Goldszmidt, 'Stochastic dynamic programming with factored representations', *Artificial Intelligence*, **121**(1), 49–107, (2000).

[3] D. Chandler, *Introduction to Modern Statistical Mechanics*, Oxford University Press, 1987.

[4] R. K. Chornei, H. Daduna, and P. S. Knopov, 'Controlled markov fields with finite state space on graphs', *Stochastic Models*, (21), 847–874, (2005).

[5] D. P. de Farias and B. Van Roy, 'On constraint sampling in the linear programming approach to approximate dynamic programming', *Math. of Op. Research*, **29**(3), 462–478, (2004).

[6] N. Forsell and R. Sabbadin, 'Approximate linear-programming algorithms for gmdp', in *Proc' ECAI'06.*, (2006).

[7] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman, 'Efficient solution algorithms for factored MDPs', *Journal of Artificial Intelligence Research*, **19**, 399–468, (2003).

[8] C. E. Guestrin, *Planning under uncertainty in complex structured environments*, Ph.D. dissertation, Stanford University, 2003.

[9] M. L. Puterman, *Markov Decision Processes*, John Wiley and Sons, New York, 1994.