HAL
open science

Policies Generalization in Reinforcement Learning using Galois Partitions Lattices

Marc Ricordeau, Michel Liquière

To cite this version:

Marc Ricordeau, Michel Liquière. Policies Generalization in Reinforcement Learning using Galois Partitions Lattices. CLA: Concept Lattices and their Applications, Oct 2007, Montpellier, France. pp.286-297. lirmm-00187164

HAL Id: lirmm-00187164

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00187164v1

Submitted on 4 Sep 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Policies Generalization in Reinforcement Learning using Galois Partitions Lattices

Marc Ricordeau and Michel Liquière

mricorde@wanadoo.fr, liquiere@lirmm.fr
Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier
161 rue Ada 34392 Montpellier Cedex 5 France

Abstract. The generalization of policies in reinforcement learning is a main issue, both from the theoretical model point of view and for their applicability. However, generalizing from a set of examples or searching for regularities is a problem which has already been intensively studied in machine learning. Our work uses techniques in which generalizations are constrained by a language bias, in order to regroup similar states. Such generalizations are principally based on the properties of concept lattices. To guide the possible groupings of similar environment's states, we propose a general algebraic framework, considering the generalization of policies through a set partition of the states and using a language bias as an *a priori* knowledge. We give an application as an example of our approach by proposing and experimenting a bottom-up algorithm.

1 Introduction

The reinforcement learning domain gives the promising theoretical framework of an agent learning a behavior by interactions with an environment with the following properties: (1) The agent doesn't have any *a priori* knowledge, on the environment. (2) There is no separation between a learning phase and a phase of use of the learning. (3) The agent has to manage the temporal difference learning problem, that is to say considering that the effect of an action can be delayed or that an effect can be the consequence of several successive actions.
With the formalism used to realize and demonstrate the properties of most of the algorithms as well as in practice, the approach is severely more restrictive: environments are designed with a Markovian Decision Process (MDP), the succession of interactions and perceptions are discretized and the agent is supposed to perceive its environment completely.
The constant progress of reinforcement learning since the emergence of its most famous algorithm, *Q-Learning* [1], aims at adapting the algorithms to a wider class of problems than those that can be formalized with a MDP: extension of the formalism [2] or more recently [3], environment's states or continuous actions [4], non markovian environments, partially observable environments [5].
Presently, the question of generalization of policies for reinforcement learning and their corollaries: the use of *a priori* knowledge and the search for important description features to learn the task, has become an important research fields.

Several ways are intensively studied in this frame: generalization by function approximation, algebraical model generalization [6], [7], temporal abstraction generalization [8], link with planning and ILP methods [9] or search for relevant descriptions of the environment [3].
In our work, we consider the generalization of policies and task description problem from a double point of view: the learning of a behavior by reinforcement and the problem of grouping elements which are considered similar agreeing with a description language. We propose to apply to reinforcement learning, learning methods, based on the Galois lattice algebraical structure, used in classification and more recently in data mining.
In [10], methods in the same framework have been presented but in a different way. Notably, we present here an original search space: the descriptions partitions Galois lattice instead of the classical powerset Galois lattices. Moreover, we characterize algebraically the space search, instead of building the whole lattices structures in our algorithms.
Section 2 recapitulates briefly the basis of reinforcement learning, essentially to present formalisms used in this paper. In particular, we remind the reader how to formalize the generalization of policies as a partitioning of the environment's states. Then, section 3 shows how to use Galois lattices as a generalization space for a set constrained with a description language (possibly structural). Moreover, we present the descriptions partitions Galois lattice, which is an extension of this method to partitioning a set of objects. In section 4, we see how to use this algebraical structure for policies generalization in reinforcement learning. We study the existence of solutions for the learning problem as a function of the description language. Then, in section 5, we present an algorithm based on these methods and we apply it to an academic problem. We show notably the possibility to extract relevant description elements for the task to be learned by the agent.

2 Basis of Reinforcement Learning, policies generalization

2.1 Reinforcement Learning Problem

We only give here the basis and main formalisms of reinforcement learning. A complete introduction can be found in [11]. The reinforcement learning proposes a framework where an agent learns a task in a given environment, only by receiving rewards when the task is accomplished. A reinforcement learning is classically formalized as a succession of interactions. The agent receives at the step t a **state** s_t from the environment and a numerical value of **reinforcement** $r_t \in \mathbb{R}$, called **reward**, indicating the quality of the state in which the agent is. Then, the agent acts on its environment, selecting one of the eligible **actions** a_t for the state s_t, which modifies the environment's state s_{t+1}. This process continues indefinitely or until a terminal state is reached.
The goal of the agent is to learn to select the actions which maximize the flow of the rewards $r_i, i \geq t$ to come and not only the following reward r_{t+1}. However,

the closest rewards (in term of number of interactions) are considered as more important then the far ones. Thus, the goal of the agent is to maximize the following value called **cumulated delayed reward**: $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k},\ 0 < \gamma \leq 1$.

The environment is formalized with a MDP. That is to say that the state s_{t+1} depends only on the state s_t, on the selected action a_t and on a probability transition $P(s_t, a_t, s_{t+1})$. In particular, the probability doesn't depend on the previous states-actions sequence $s_0, a_0, \ldots, s_{t-1}, a_{t-1}$.
Formally, an environment is defined with a tuple $\langle S, A, \psi, P, R \rangle$; $S = \{s_1,\ s_2,\ \ldots,\ s_n\}$ is a set of states; $A = \{a_1, a_2, \ldots, a_n\}$ is a set of actions; $\psi \subseteq S \times A$ is a set of eligible couples $(state, action)$. We note $A(s)$ the set of eligible actions for the state s ($(s, a_i) \in \psi$); $P : \psi \times S \rightarrow [0, 1]$ is the set of transition probabilities from the state s to the state s' selecting the action a; $R : \psi \rightarrow \mathbb{R}$ is the reward function giving a numerical value for each eligible couple (s, a).
The function $\pi : \psi \rightarrow [0, 1]$ gives the probability for the agent to select the action a being in the state s. The function π is called the **policy** of the agent.
The function $Q^{\pi} : \psi \rightarrow \mathbb{R}$, called **quality** function, defines the expected cumulated delayed reward (expected because the process is stochastic and not deterministic) the agent would obtain following the policy π from the state s and taking the action a.
The set of all policies can have a partial order relation, based the cumulated delayed reward. A policy π_1 is greater than a π_2 if $\forall (s, a) \in \psi,\ Q^{\pi_1}(s, a) \geq Q^{\pi_2}(s, a)$. The policies allowing the agent to collect the maximum possible rewards (R_t) are called **optimal policies**. All the optimal policies share the same values for the function $Q(s, a)$ which is noted $Q^*(s, a)$. Thus, we note π^* to refer to an optimal policy.
The knowledge of $Q^*(s, a)$ allows to find greedily an optimal policy π^* by selecting in each state s, one of the actions a with a maximal expected cumulated delayed reward. We note $A^*(s)$ the set of the optimal actions for the state s.
The reinforcement learning algorithms are based on the approximation of the $Q^*(s, a)$ function, during the interactions of the agent with the environment. The hypothesis on the $Q(s, a)$ function is modified with an update rule. For example, this is the update rule for the *Q-learning* algorithm [1]:

$$Q(s, a) \leftarrow (1-\alpha)Q(s, a) + \alpha\big(r_t + \gamma \underset{a \in A(s_{t+1})}{Max} Q(s_{t+1}, a)\big),\ 0 < \alpha \leq, 0 \leq \gamma \leq 1 \quad (1)$$

Some main issues characterize the reinforcement learning. Firstly, the temporal difference learning problem, that is to say the repartition of an acquired experience (a value r_t) over the previously selected actions. Secondly, the *exploration-exploitation* problem, that is to say the balance between using the actual knowledge of $Q(s, a)$ function to maximize the flow of rewards (exploitation) and increasing its precision by selecting assumed non-optimal actions (exploration). One of the corollaries is the balance between keeping the actual theory and incorporating the newly acquired knowledge (managed with the γ parameter).
Finally, the question we propose to give elements on: the question of the generalizing of policies. It can be viewed as: "How a behavior learned in an environment

can be used in a similar environment", or "How a learned knowledge on a set of states can be generalized on a greater set of states". This problem is related to a main question of machine learning: "How to find a relevant description for a given task ?"

2.2 Generalization by partitioning the environment's states

In [7], [6], the generalization for MDPs and consequently for reinforcement learning is presented as the search for an equivalence relation between the environment's states. Two similar states can be merged if they are equivalent, i.e. if the merging doesn't change the dynamic of the environment. It's actually a generalization, because a learning on a state s is applied to the set of the states equivalent with s. However, in these articles, the equivalence relation between the states is computed directly from the MDP, that is to say from the model of the environment. In our work, we want to apply generalization methods without this information, preserving the *agent* point of view.
The equivalence relation on S, the set of the environment's states, we propose is not based on the compliance with the environment's dynamic, but on the equivalence of their optimal actions.

Definition 1. *Let $s_1, s_2 \in S$, be two environment's states. s_1 and s_2 are equivalent, which is noted $s_1 \equiv^* s_2$ if and only if $A^*(s_1) = A^*(s_2)$. The equivalence relation $\equiv^*$ implies a partitioning of the set S, noted $\mathcal{P}^*$.*

This means that all the optimal actions of s_1 are the sames (considering their label) as the optimal actions of s_2. It's important to note that we have introduced a first *a priori* knowledge since this definition implies that two actions can be equivalent according to their description.
The partition $\mathcal{P}^*$ allows to express rules such as "in the set of states S_1, we must select an action among $A^*(S_1)$ in order to act optimally". The knowledge of this set of rules, gives the optimal policies and enables us to present them in a more intelligible and general way than an classical probabilities function. Thus, the search for $\mathcal{P}^*$ becomes in our framework, the goal of the learning. The space search becomes the set partitions of S.
However, grouping the states according to their optimal actions doesn't give a direct generalization method. To do it, we need to be able to compare the states. In the next section, we show how the Galois Lattice algebraical structure can be used to generalize a set of objects with a generalization language.

3 Galois lattice as generalization space

The definitions and theorems used here about Galois lattices can be found in [12]. First, we show how this structure is used in machine learning to allow a generalization over a set of elements biased with a language. Secondly, we give an extend of Galois lattice by presenting the partitions Galois lattice.

3.1 Galois lattice and generalization languages

The Galois lattice, or more generally the lattice structure is commonly used in machine learning, directly or in an underlying way. The first well known use of such mathematical objects is certainly the Version Space [13]. More recently, the formal concept analysis theory [12] uses directly Galois lattices as structure used for generalization over a set of objects. We give here a general algebraical approach that can be found in [14].
A Galois lattice can be classically summed up as a structure defined by a double closure relation between two join semi-lattices. We note it $\mathcal{GL}(A, \leq_A, \otimes_A, B, \leq_B, \otimes_B, f, g)$, $(A, \leq_A, \otimes_A)$ and $(B, \leq_B, \otimes_B)$ being the two join semi-lattices defined respectively by a set, a partial order relation and a product operator producing the supremum of two elements and f and g being the two closure operators. In the frame of machine learning, the Galois Lattice produced from the powerset of objects and a generalization langage $(L, \leq_L, \otimes_L)$ defines the set of all the subsets authorized by a given description language and their associated description.
More formally, to use a Galois lattice as a generalization space for a set of objects E constrained with a generalization language L, we need to identify the elements of the Galois lattice definition: (1) The set A with $\mathcal{P}(E)$, the powerset of the elements to classify. (2) The partial order relation $\leq_A$ with the set inclusion $\subseteq$. (3) The product operator $\otimes_A$ with the set intersection $\cap$. (4) The set B, $\leq_B$ and $\otimes_B$ can be identified with any description language L with a partial order relation $\leq_L$ and a product operator $\otimes_L$ such that exists a description function d and an instantiation function i described below and agreeing with the Galois connection conditions. Such languages are called **generalization languages**. The relation $\leq_L$ defines the relation *more general than* or *more specific than* and $\otimes_L$ is called *generalization operator*. (5) The function f with the following description function $d : \mathcal{P}(E) \rightarrow L$, $d(P) = \bigotimes_L \{d(e), e \in E\}$. The description $d(P)$ of a subset $P \subseteq E$ is the least general generalization of the descriptions of the elements $e \in P$. Note that only the description $d(e)$, $e \in E$ must be known in order to compute the description of all subset $P \subseteq E$. (6) The function g with the following instantiation function $i : L \rightarrow \mathcal{P}(E)$: $\forall l \in L,\ i(l) = \{e \in E \mid d(e) \leq_L l\}$. The instantiation of an element of the language $l \in L$ is the set of all the elements $e \in E$ which description is more specific than l.
Finally, the set $\mathcal{C}$ of the couples $(c_a, c_b) \in C_{i \circ d}(\mathcal{P}(E)) \times C_{d \circ i}(L)$ is called the set of **concepts** of $\mathcal{GL}(\mathcal{P}, \subseteq, \cap, L, \leq_L, \otimes_L, d, i)$. Let a concept $c = (P, l)$, $P \in C_{i \circ d}(\mathcal{P}(E))$, $l \in C_{d \circ i}(L)$, P is the **extension** of c and l the **intention** of c.
We have three main properties about this structure. First, the set $C_{i \circ d}(\mathcal{P}(E))$ is the powerset of E that can be expressed with the language L. Thus, L implies a selection between all the possible generalizations of the elements of E. Roughly speaking, we can also say that the language L implies rules like "if we decide to put together the elements e_1 and e_2, then we must also regroup them with e_3". Secondly, the set $C_{d \circ i}(L)$ is the set of the descriptions of the elements of $C_{i \circ d}(\mathcal{P}(E))$. If a new, unknown element e_n occurs, we can use its description $d(e_n)$ to compare and eventually classify it in a previously found group. Finally, the set C of concepts has a lattice structure, which can be used to compare and

merge generalizations of elements of E.
This framework is very large, every generalization language L provided with $\leq_L$ and $\otimes_L$ is usable. For example, the formal concept analysis theory studies particularly the attributes-values description case with: L is the powerset $\mathcal{P}(A)$ of a set of attributes A, $\leq_L$ is the set inclusion $\subseteq$, $\otimes_L$ is the set intersection $\cap$. Notably, we can also use structural languages (see [14] for examples). Thus, the descriptions of the generalizations keep their structural properties.

3.2 Descriptions Partitions Galois lattice

We present a special case of Galois lattice: the descriptions partitions Galois lattice as generalization space for the set partitions. It is built in a similar way than the powerset Galois lattice. It will be used to constrain the possible set partitions with a generalization language describing the elements of E, with the same properties. Figure 1 presents the general scheme.

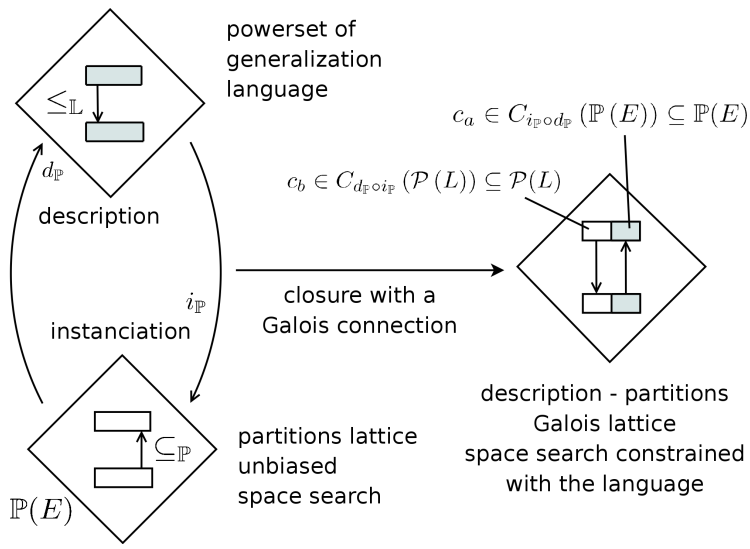

Fig. 1. Scheme of partitions Galois lattice constrained with a generalization language

The main principle of the Galois lattices for partitions is to use the partitions lattice of E instead of powerset lattice and to use the powerset lattice of the generalization language L instead of L. We also have to redefine the description and instantiation functions to fit the definition. Finally, we show that the extents of the generated concepts are the partitions of E for which each element is a concept of the classical Galois lattice.
Let's define the partial order relation $\subseteq_{\mathbb{P}}$ for partitions, the product $\otimes_{\mathbb{P}}$ and the sum $\oplus_{\mathbb{P}}$ of two partitions.

Definition 2 ($\subseteq_{\mathbb{P}}, \otimes_{\mathbb{P}}, \oplus_{\mathbb{P}}$). *Let E be a set and $\mathbb{P}(E)$ be set partitions of E. Let $\mathcal{P}_1$, $\mathcal{P}_2 \in \mathbb{P}(E)$ be two partitions of E. $\mathcal{P}_1 \subseteq_{\mathbb{P}} \mathcal{P}_2 \Leftrightarrow \forall P_1 \in \mathcal{P}_1, \exists P_2 \in \mathcal{P}_2$ such that $P_1 \subseteq P_2$.*
Let $R_\otimes$ and $R_\oplus$ be two binary relations such that $\forall e_i, e_j \in E$, $e_i R_\otimes e_j \Leftrightarrow e_i$ and e_j are in the same part in $\mathcal{P}_1$ ***or*** *in $\mathcal{P}_2$; $e_i R_\oplus e_j \Leftrightarrow e_i$ and e_j are in the same part in $\mathcal{P}_1$* ***and*** *in $\mathcal{P}_2$. The transitive closure of $R_\otimes$ defines a partition which is $\mathcal{P}_1 \otimes_{\mathbb{P}} \mathcal{P}_2$ and $R_\oplus$ defines a partition which is $\mathcal{P}_1 \oplus_{\mathbb{P}} \mathcal{P}_2$.*

For example, $E = \{a, b, c, d, e, f, g\}$, $\mathcal{P}_1 = \{a, b\}, \{c\}, \{d, e\}, \{f, g\}$ and $\mathcal{P}_2 = \{a\}, \{b\}, \{c\}, \{d, \}\{e, f\}, \{g\}$. $\mathcal{P}_1 \otimes_{\mathbb{P}} \mathcal{P}_2 = \{a, b\}, \{c\}, \{d, e, f, g\}$.
Remind that a partition defines an equivalence relation. Thus $\otimes_{\mathbb{P}}$ can be viewed as the merging of the equivalence classes defined by $\mathcal{P}_1$ and $\mathcal{P}_2$.
Let's identify the generalization language $(\mathbb{L}, \leq_{\mathbb{L}}, \otimes_{\mathbb{L}})$ to describe the partitions of E, which is based on the generalization language $(L, \leq_L, \otimes_L)$ for E.
A partition of E is composed of subsets of E. Each of these subsets can be described with an element $l \in L$ with the description function d. Consequently, the description of a partition of E is described with a set $\mathcal{L}$ of elements of L. Thus, the description language for the partitions set is the powerset of L. Nevertheless, we only consider a subset of $\mathcal{P}(L)$. Indeed, we only need the set, noted $\mathcal{P}(L)_{\mathbb{P}(E)}$ of the subsets of L such that their instantiations are partitions of E. In the general case, the instantiation is only a coverset of E.

Definition 3 ($\leq_{\mathbb{L}}, \otimes_{\mathbb{L}}$). *Let $(L, \leq_L, \otimes_L)$ be a generalization language. Let $\mathcal{L}_1$, $\mathcal{L}_2 \in \mathcal{P}(L)_{\mathbb{P}(E)}$. $\mathcal{L}_1 \leq_{\mathbb{L}} \mathcal{L}_2 \Leftrightarrow \forall l_1 \in \mathcal{L}_1$, $\exists l_2 \in \mathcal{L}_2$ such that $l_1 \leq l_2$.*
$\mathcal{L}_1 \otimes_{\mathbb{L}} \mathcal{L}_2$ is the set $\mathcal{L}_\otimes$ such that: $\mathcal{L}_\otimes = \big\{l \in L \mid l = d(P_1), P_1 \in merge(\{i(l_1) \cup i(l_2), l_1 \in \mathcal{L}_1, l_2 \in \mathcal{L}_2\})\big\}$

The definitions of $\leq_{\mathbb{L}}$ and $\otimes_{\mathbb{L}}$ are presented to keep mathematical formalism and are not used in practice. The description function $d_{\mathbb{P}}$ and the instantiation function $i_{\mathbb{P}}$ illustrate the method in a better way.

Definition 4 (description of a partition $d_{\mathbb{P}}$). *We define description function $d_{\mathbb{P}} : \mathbb{P}(E) \to \mathcal{P}(\mathbb{L})$ with: $\forall \mathcal{P} \in \mathbb{P}(E), d_{\mathbb{P}}(\mathcal{P}) = \big\{l \in L \mid l = d(P_1), P_1 \in merge(\{i \circ d(P), P \in \mathcal{P}\})\big\}$*

To find the description of a partition $\mathcal{P} \in \mathbb{P}(E)$, we first close the subsets P of E composing the partition with the closure $i \circ d$. This gives a coverset of E. Then, we merge transitively the subsets of this coverset which have at least one element e in common with the operator $\otimes_{\mathbb{P}}$. This gives a partition of E for which all subsets are closed elements of $i \circ d$, i.e. extents of the classical Galois lattice. Finally, we redescribe those elements with d.

Definition 5 (instantiation of a subset of L $i_{\mathbb{P}}$). *We define the instantiation function $i_{\mathbb{P}}(E)_L \to \mathbb{P}(E)$ with:$\forall \mathcal{P}_L \in \mathcal{P}(L)_{\mathbb{P}(E)}, i_{\mathbb{P}}(\mathcal{P}_L) = \big\{i(l),\ l \in \mathcal{P}_L\big\}$.*

The instantiation of a subset $\mathcal{P}_L \in \mathcal{P}(L)_{\mathbb{P}(E)}$, is simply the set composed with the instantiations of each element $l \in \mathcal{P}_L$. We can finally produce the partitions Galois lattice.

Theorem 1 (descriptions partitions Galois lattice).
$\mathcal{GL}_{\mathbb{P}}(\mathbb{P}(E),\subseteq_{\mathbb{P}},\otimes_{\mathbb{P}},\mathbb{P}(E)_L,\ \subseteq_{\mathbb{P}},\otimes_L,d_{\mathbb{P}},i_{\mathbb{P}})$ is a lattice called ***descriptions partitions Galois lattice****. Let $\mathbb{C}$ be the set of concepts of $\mathcal{GL}_{\mathbb{P}}$. Let $\mathcal{C} = (\mathtt{ext}, \mathtt{int}) \in \mathbb{C}$. The extension $\mathtt{ext}$ is a partition of E and the intention $\mathtt{int}$ is its description.*

With all the elements described above, the proof constits in showing that $(d_{\mathbb{P}} \circ i_{\mathbb{P}}, i_{\mathbb{P}} \circ d_{\mathbb{P}})$ defines a Galois connexion. Note that the same elements are needed to define the partitions Galois lattice compared to the classical Galois lattice: a set E and a generalization language $(L, \leq_L, \otimes_L)$. The classical properties of classical Galois lattices as a generalization space constrained with a language can be applied to the partitions Galois lattice: rules concerning regrouping objects, the possibility to deal with unknown objects, and a comparison between generalizations. The difference is that instead of considering the set of the regroupings of elements of E which can be expressed with the language L, we consider the set partitions of E which can be expressed with L.

Proposition 1 (partition agreeing with a language).
Let E be a set and $(L, \leq_L, \otimes_L)$ be a generalization language for E. A partition $\mathcal{P}(E)$ of E agreeing with the language L is a partition whose all the subsets can be expressed with an element $l \in L$. That is to say whose all the subsets are closed elements of $\mathcal{P}(E)$ with the closure $i \circ d$. The set of the extensions of the concepts of the description partitions Galois Lattice $\mathcal{GL}_{\mathbb{P}}$ is the set partitions of E agreeing with the language L.

Size of $\mathcal{GL}_{\mathbb{P}}$ The size of the $\mathcal{GL}_{\mathbb{P}}$ is equal to the number of partitions agreeing with L. Consequently, the size of the $\mathcal{GL}_{\mathbb{P}}$ is lower than the $|E|^{th}$ Bell number. The exact number of partitions agreeing with L is an open question. However, we can say that it's a function of the size of the classical $\mathcal{GL}$ size. In the worst case, the size of the $\mathcal{GL}$ is $2^{|E|}$. Then, the size of $\mathcal{GL}_{\mathbb{P}}$ is the $|E|^{th}$ Bell number, but in this case, L doesn't give any bias concerning the possible ways of regrouping the elements of E. This case is useless in our context. We can note that for each subset of E which can't be described with an element of L, the size of $\mathcal{GL}_{\mathbb{P}}$ is falling as a function of the number of elements of this subset.

4 Using the partitions Galois lattice for reinforcement learning

The set partition $\mathcal{P}^*$ of the environment's states induced by the quality function of optimal policies $Q^*(s, a)$ can be considered as observed section 2.2, as the goal of a reinforcement learning. We have just defined the descriptions partitions Galois lattice as space search for partitioning a set of objects, here S, the set of the environment's states. The space search being constrained with a language L_S, we have first to study the relationship between $\mathcal{P}^*$ and the possibility to express it with L_S. Then, we propose an algorithm to build $\mathcal{P}^*$.

4.1 Relationship between $\mathcal{P}^*$ and a generalization language L_S

Let an agent evolving in an environment formalized with a MDP $\mathcal{M}$ $\langle S,A,\psi,P,R\rangle$. The set S is described with the generalization language $(L_S, \leq_{L_S}, \otimes_{L_S})$.
Let's consider three different cases concerning the agreement of $\mathcal{P}^*$ with L_S. (1) *The partition $\mathcal{P}^*$ agrees with L_S*. In this case, each of the sets regrouping the states according to their optimal actions can be expressed with an element of the language $l \in L_S$. (2) *It exists at least one partition $\mathcal{P}$ agreeing with L_S* such that $\mathcal{P} \subseteq \mathcal{P}^*$. In this case, if we add the condition which is that each state $s \in S$ has a different description (the agent has a complete vision of its environment), it always occurs. Indeed, in the worst case, it's the partition of size $|S|$. Moreover, if the description language doesn't have any generalization properties that is to say the product of l_1, $l_2 \in L_S$ always gives the same element $l_\top$, then we have the classical case of reinforcement learning. Note that there can be more than one such partitions. (3) *There is no partition $\mathcal{P}$ agreeing with L_S such that $\mathcal{P} \subseteq \mathcal{P}^*$*. The agent is unable to distinguish states which should be considered different from the task point of view. We are in the case of Partially Observable MDP. We need for example, to use a memory based system [5] to separate the states.

4.2 A bottom-up Algorithm

The general framework we proposed: search for the partition $\mathcal{P}^*$, using the set partitions of S, can be implemented in several ways, depending on the chosen space exploration (bottom-up, top-down, breadth-first,...) and on the heuristics managing the imprecision of the current $Q(s,a)$ function.
We propose the bottom-up algorithm 1 which partitions the set of the states at a given step of the learning. The built partition is the most general, agreeing with L_S. The equivalence relation $\equiv^*$ doesn't always give a unique partition. Thus, the algorithm regroups in priority the states with the same value for their optimal actions.

Proof elements for the algorithm All the states are considered successively using a FIFO structure (variable `States` line 1). In the worst case, each state is added as a singleton to the resulting partition (line 7). This ensures that the final result is a cover of S. The lines 2 and 5 ensure that the elements are added to only one subset, ensuring that the result is a partition of S.
Line 3, the algorithm constructs a subset of S from a state and according to the equivalence relation $\equiv^*$, using te closure operator $i \circ d$. This ensures that the produced subset agrees with the language L. Then, we verify line 4 that the closure has only added equivalent elements, ensuring that the resulting partition is lower (using $\subseteq_{\mathbb{P}}$) than $\mathcal{P}^*$.

5 Experiment

We present an experiment of our algorithm on an academic problem to illustrate our method and to validate our theoretical approach with an implementation.

Data
• A generalization language $(L_S, \leq_{L_S}, \otimes_{L_S})$ for the states S the current estimation of $Q(s, a)$
Result
• A partition $\mathcal{P}$, based on $\equiv^*$, agreeing with the language L_S, such that $\mathcal{P}$ is the most general possible, with grouping uppermost the states of S with the same optimal actions values.

$Partition \leftarrow \emptyset$;
1 $States \leftarrow$ Sort S according to the decreasing values of $Q(s, a^*)$;
while $States \neq \emptyset$ **do**
  $s \leftarrow$ remove($States$);
  $Equivalents \leftarrow s_1 \in States \mid s_1 \equiv^* s$; $NewSubset \leftarrow \{s\}$;
  while $Equivalents \neq \emptyset$ **do**
2 $s_2 \leftarrow$ remove($Equivalents$);
3 $add \leftarrow i \circ d(NewSubset \cup \{s_2\}) - (NewSubset \cup \{s_2\})$;
4 **if** $\forall s_3 \in add$, $s_3 \equiv^* s$ **then**
    $NewSubset \leftarrow NewSubset \cup \{s_2\} \cup add$;
5 remove from $Equivalents(add \cup \{s_2\})$; remove from $States(add \cup \{s_2\})$;
7 $Partition \leftarrow Partition \cup \{NewSubset\}$;
return $Partition$;

Algorithme 1 : Partitioning the set S according to $\equiv^*$ and agreeing with a generalization language $(L_S, \leq_{L_S}, \otimes_{L_S})$

The example consists in a grid-world of size (3×3) containing an *agent*, a *reward* and a *wall*. All the other tiles are *empty*. The walls can be moved, if there is no other behind, with the action *push*.
The task to be learned is to move as fast as possible to the reward tile. Each episode lasts until the agent reaches the reward tile or until it destroys it with a wall. The agent receives an reward of +1.0 if it reaches the reward tile. All other actions receive -0.1. There are 504 different states with 4 actions in each. *push* and *move* are different actions, $|\psi| = 2016$. The consequences of actions are deterministic and the environment is stationary.
The actions are described with: {*move west, move east, move north, move south, push west, push east, push north, push south*}. The language $(L_S, \leq_{L_S}, \otimes_{L_S})$ used to describe the states is an attribute-values language. The description of a state is composed with one attribute for each tile that the agent can see (see Figure 2). The partial order relation $\leq_{L_S}$ is defined by: $d(s_1) \leq_{L_S} d(s_2) \Leftrightarrow$ all the attributes of $d(s_1)$ are less than or equal to the corresponding one in $d(s_2)$ according to the join semi-lattice given Figure 2. The product $\otimes_{L_S}$ of two descriptions is the generalization of each attribute. The selection of actions and learning has been made with a classical *Q-Learning* algorithm.

Our algorithm allows the extraction of relevant description elements according to the task. Figure 3, shows an extract of the partition obtained after the convergence of the learning. Quantitatively, we need only 92 description elements

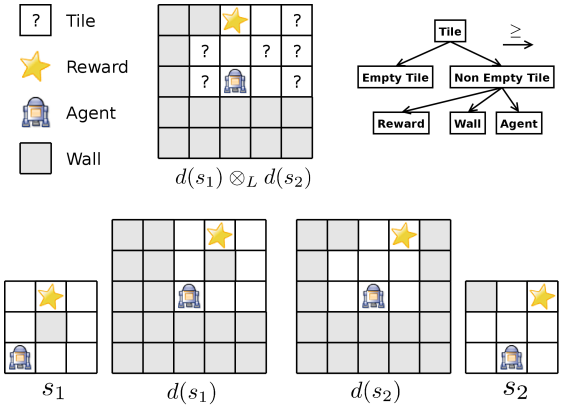

Fig. 2. A graphical example of two states, their descriptions and their product

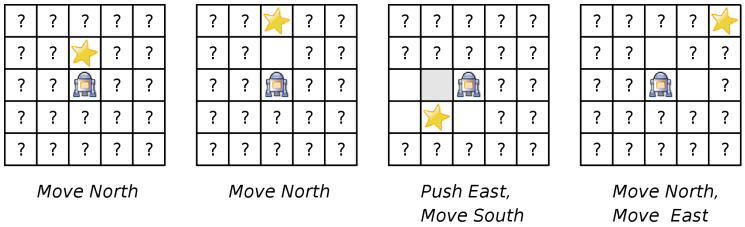

Fig. 3. Descriptions and optimal actions of 4 subsets among the 92 produced by partitionning the 504 environnement's states after convergence of the reinforcement learning

to describe the task instead of 504 initially. There are expected elements such as *being at one tile at the south of the reward* implies *move north* (example 1), the other tiles don't matter. We also find non intuitive rules (example 4).
Note that there are descriptions with the same optimal actions which are not grouped (example 1 and 2). This is because there are no language elements allowing the grouping of these descriptions without regrouping others, with different optimal actions. We could add, for example, the notion of relative position with the reward, either directly or using a knowledge data base.

6 Conclusion

First, we reminded the reader of the general principles in the use of a Galois lattice as a generalization space for a set of objects described with a generalization language. Then, we extended these results to define an original generalization space for the partitions instead of powerset, in particular by introducing the notion of partition agreeing with a language. We showed that, considering generalization as a partitioning, this structure can be used to generalize learning in reinforcement learning. Finally, we proposed a bottom-up algorithm which

through a reinforcement learning and a generalization language made it possible to produce general rules linking description elements of the environment and optimal actions.
This work considered principally the algebraical aspects. Consequently, we didn't treat two remaining important issues: "How to use the rules on unknown examples" and "How to deal with knowledge uncertainty".
Our future work will focus on three directions. First, a more formal link between our work and the relational reinforcement learning [9]. Secondly, an algorithmic improvement, studying the different ways of exploring the space search and the classical balancing between spatial and temporal complexity. Thirdly, we will consider the *exploration-exploitation* problem, i.e. using previously acquired knowledge or acquiring new knowledge and as a corollary, managing knowledge uncertainty. It remains a difficult problem in machine learning and in reinforcement learning in particular. In our case, it's transposed in a particular way as the concept reliability, already presented in [10]. We think that a relevant approach can consists in the explicit management of second order elements (states distribution, maximal reward,. . .) directly through the algorithms.

References

1. Watkins, C., Dayan, P.: Q-learning. Machine Learning **8** (1992) 279–292
2. Sutton, R.S., Precup, D., Singh, S.: Between mdps and semi-mdps : A framework for temporal abstraction in reinforcement learning. Artificial Intelligence **112**(1-2) (1999) 181–211
3. James, M.R., Singh, S.: Learning and discovery of predictive state representations in dynamical systems with reset. In: ICML 2004. (2004) 417–424
4. Munos, R.: Error bounds for approximate policy iteration. In: International Conference on Machine Learning ICML 2003. (2003) 560–567
5. McCallum, A.: Reinforcement Learning with Selective Perception and Hidden State. PhD thesis (1996)
6. Ravindran, B., Barto, A.G.: Smdp homomorphisms: An algebraic approach to abstraction in semi markov decision processes. In: IJCAI 2003. (2003) 1011–1016
7. Givan, R., Dean, T., Greig, M.: Equivalence notions and model minimization in markov decision processes. Artificial Intelligence **147**(1-2) (7 2003) 163–223
8. Dietterich, T.G.: State abstraction in maxq hierachical reinforcement learning. Artificial Intelligence Research (13) (2000) 227–303
9. Dzeroski, S., Raedt, L.D., Driessens, K.: Relational reinforcement learning. Machine Learning (43) (2001) 7–52
10. Ricordeau, M.: Q-concept-learning: Generalization with concept lattice representation in reinforcement learning. In Society, I.C., ed.: ICTAI 03. (2003) 316–323
11. Sutton, R.S., Barto, A.G.: Reinforcement Learning : An Introduction. MIT press (1998)
12. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer (1999)
13. Mitchell, T.M.: Machine Learning. McGraw-Hill (1997)
14. Liquière, M., Sallantin, J.: Structural machine learning with galois lattice and graphs. In Ed, M.K., ed.: ICML 1998. (1998) 305–313