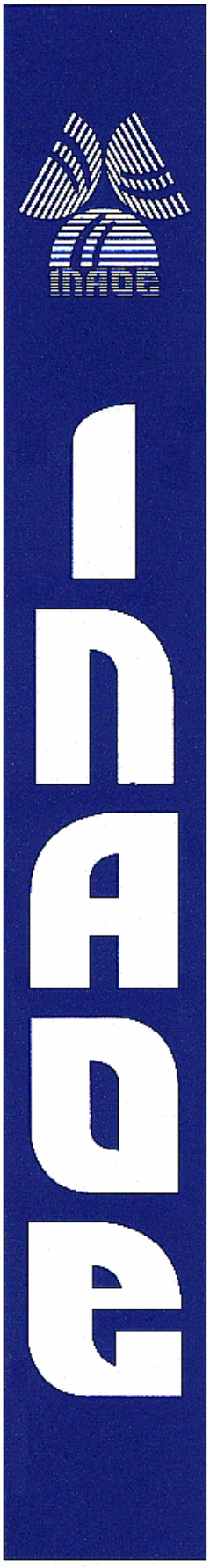

Automatic Discovery of Concepts for Dynamic Domains

Ana C. Tenorio González, Eduardo F. Morales M.

Reporte Técnico No. CCC-13-002
28 de mayo de 2013

Luis Enrique Erro 1
Sta. Ma. Tonantzintla,
72840, Puebla, México.

Automatic Discovery of Concepts for Dynamic Domains

Ana Cecilia Tenorio González, Eduardo F. Morales M.

Computer Science Department
National Institute of Astrophysics, Optics and Electronics
Luis Enrique Erro # 1, Santa María Tonantzintla, Puebla, 72840, México
E-mail: {catanace17}@inaoep.mx

Abstract

This thesis proposal explores how much an agent can autonomously learn about its environment just by interacting with it. This involves the use of machine learning techniques in novel ways. Machine learning outcomes strongly depend, besides the hypothesis language used, on how and what training data is given to the learning system. However, in dynamic or unknown domains, it is not always possible to previously prepare all the required data or predict the experimental conditions on which learning will take place. In these conditions an agent must acquire knowledge while interacting with the environment, learning concepts about the environment, learning actions to achieve goals, and use a motivation mechanism to promote and guide the learning process. Inductive Logic Programming (ILP) has been used to learn concepts from examples and can introduce new vocabulary when needed. however, little progress has been made to learn actions or behavior policies to perform tasks. On the other hand, Reinforcement Learning (RL) has been used to learn behavior policies in dynamic domains, however, the states and actions are normally predefined in advance and new representations can not be included in the learning process. Recent research on intrinsically motivated RL has been used to guide the learning process independently of the task, however it has been focused for learning behavior policies. In this research proposal, a novel algorithm for learning concepts about objects and actions in dynamic domains is proposed. Concepts are learned using ideas from demand driven predicate invention in ILP and a graph-based inductive algorithm to identify potential concepts. New actions are learned from the traces of the actions followed by the agent and from policies learn for particular goals. New goals are defined by an intrinsically motivated reward function which drives the learning process. Details of the proposed algorithm are given and preliminary results for concept learning are described.

Keywords: Concept learning, predicate invention, inductive logic programming, intrinsic motivation, reinforcement learning.

1 Introduction

Imagine an independent agent interacting with its environment and learning, through its interaction, definitions of relevant objects and their relationships, and new skills to achieve goals. This agent needs an

adequate representation framework, suitable machine learning techniques to induce concepts and skills autonomously, and a mechanism to drive continuously the learning process to explore and learn in an efficient way. This thesis proposal aims to advance the state-of-the-art of these issues.

In Machine Learning research, abstract concepts are induced from a set of instances, normally described by attribute-value pairs enumerating characteristics of objects and their respective values. However, in some domains such as robotics, it is more suitable to use a relational representation that can naturally express objects from the environment and relations among them and the robot. Successful learning, however, not only depends on the hypothesis language but also on the instances used for learning. Usually, the experimental conditions are carefully prepared by a user before the learning process takes place. There are domains, however, where all the conditions for successful learning can not be prepared in advance. We call these domains *dynamic*, since the agent does not know what to expect and the learning process occurs dynamically while the agent interacts with its environment. In this case, the agent must adapt and increase its knowledge from direct experience with unexpected situations and interactions with the environment.

Concept learning about objects and actions, under these conditions, has received little attention. Some areas have focused on concept learning of relations between object's, and other on learning sequences of actions or behavior policies. In particular, Inductive Logic Programming (ILP) [35, 38, 54, 36, 63, 55, 30, 29] has focused on concept learning and on the discovery of new vocabulary or predicates, but little work has been done on learning actions and behavior policies. In ILP, examples and suitable background knowledge definitions are normally carefully provided by the user, only one concept is learned at a time, and little work has been done on learning actions for achieving goals.

Learning from direct experience with the environment has been addressed by the RL community [56]. The idea is to learn how to perform a sequence of actions (behavior policy) to reach a goal obtaining the best expected accumulated reward. In general, the reward function is designed by the user for the task that is being learned. Recent research have introduced, what is called, intrinsically motivated reward function [51, 22, 45, 25, 47, 32], to provide the agent a sense of motivation (interest, curiosity) to guide the agent's behavior to reach both implicit and explicit goals. RL, however, assumes that the states and actions are suitably defined and they do not change during the learning process. Also intrinsically motivated reward functions have focused on learning behavior policies and have not been designed also for concept learning.

In this research work, a novel algorithm is proposed to learn both concepts and actions while interacting in a dynamic environment. Concepts are learned using ideas from demand driven predicate invention used in ILP. To define which concepts to learn a substructure discovery algorithm over graphs is used. The idea is to transform the data into a graph-based representation and discover common sub-graphs to use them as examples for an ILP algorithm (this process in explained in Section 4). Actions are learned from the traces followed by the agent, and policies, for particular goals, are learned using an RL algorithm. A novel intrinsically motivated reward function is used to drive the learning process. Among the main contributions of this research proposal are: (i) the concept learning algorithm using a graph-based approach and reinforcement learning, (ii) learning concepts about objects and actions while exploring a dynamic environment, and (iii) the design of a novel intrinsically motivated reward function to guide the learning process. The proposal is designed to reuse previously learned concepts and actions to learn new predicates and to reduce the user's intervention in the preparation of the experimental conditions for the agent.

The following section provides a general overview of ILP, RL and the required background information to understand this thesis proposal. It also analyses the most relevant work related with our research. Section 3 presents the main objectives of this research proposal. In Section 4, the methodology proposed in this work is described in detail. Some preliminary results are described in Section 5, and the work plan of this research is given in Section 6.

2 Background

In this section, a general overview of the basic theory used in this work is presented as well as a critical review of the most closely related work to our proposal.

2.1 Concept learning and Inductive Logic Programming

A concept is a category noun describing classes of objects, a subset of objects or events. The concepts can be formed by attributes (as relations) and objects [6]. In concept learning, a set of objects are grouped according to the similarity in a process named aggregation, after that, an intensional definition is induced for each group formed, this second stage is named characterization [63]. Objects are instances of concepts which are trying to be defined.

ILP [35] combines Machine Learning with Logic Programming to address concept learning. First-Order Logic (FOL) [3, 10] is used to represent objects and the relations between them. First, some definitions from logic are presented:

- Terms are logic expressions representing objects, can be constants, variables or functions.
- Constants are specific objects.
- Variables represent an specific object or another variable.
- Predicates represent relations, are formed by a symbol and a set of terms in parentheses.
- An atom or atomic sentence is a predicate applied to terms in parentheses.
- A literal is an positive or negative atomic sentence.
- A clause is a finite set of literals, when it is represented by a logical implication, the implied literal is named *head* and the other literals are named the *body* of the clause.
- Horn clauses are clauses with at most one positive literal.
- A logic program is a set of clauses.

In inductive learning the goal is to learn a complete and consistent hypothesis from a set of positive ($\varepsilon+$) and negative ($\varepsilon-$) examples. The hypothesis is complete if it covers all positive examples $\varepsilon+$ and it is consistent if it does not cover any negative example $\varepsilon-$. ILP can use background knowledge represented as a logic program for inducing concepts. In ILP, the goal is to induce a hypothesis H that with the background knowledge T produces a consistent and complete logic program, $T \cup H \vdash \varepsilon+$ and $T \cup H \nvdash \varepsilon-$.

The search for a hypothesis is performed in the set of all hypotheses which the algorithm can generate. The space of these hypotheses must be structured to perform an efficient search, usually with the use of a generalization model. A commonly used model is based on *θ-subsumption* [28]. Clause $C1$ subsumes another clause $C2$, if a substitution θ exists such that $C1\theta \subseteq C2$, where $C1$ is more general than $C2$, $C1 \leq C2$. The least general generalization (LGG) of two clauses $C1$ and $C2$ is the greatest lower bound of the lattice induced by the relation $\leq$. The LGG of two clauses relative to some background knowledge T is known as relative least general generalization (RLGG) [37].

In general, the hypotheses search space in ILP is huge and needs to be constrained by imposing some inductive bias. Sometimes this bias may constrain the system of finding a consistent hypothesis. One way to

tackle this problem is by introducing additional vocabulary or predicates, also known as predicate invention [54]. Formally, the new vocabulary introduced is expressed as $I = V(H) - (T \cup O)$, where I is the new vocabulary, $V(H)$ is the vocabulary of the hypothesis induced, T is the vocabulary of the background knowledge without the vocabulary of the examples, and O is the vocabulary of the examples [39]. With predicate invention it is possible to induce more compact and precise theories, and also, induce theories which can not be induced without the introduction of new vocabulary [54].

Predicate invention can be required for the following reasons [5, 54]:

- *Bias shift*/Demand driven: a new predicate is invented to produce a consistent hypothesis. Demand driven predicate invention is used when the existing vocabulary is not enough to form a concept, so it is necessary to add a new predicate.

- Reformulation: predicates are modified to produce more compact hypotheses. The new predicates are a combination of others or a restructured predicate. When predicates are modified to improve the efficiency of a theory, this process is called *transformation*.

Regarding the approaches including interaction, MIS [49] is one of the first approaches based on ILP to induce models from examples with interaction with an oracle. MIS is based on a top-down algorithm with an algorithm for backtracking contradictions (when the hypothesis implies a negative example) and refinement operators (when the hypothesis does not imply a positive example). In this system an oracle provides a set of examples (facts) to the system as input and sets the truth of a statement relative to a model (in a kind of supervised learning). A theory covering all positive examples and none negative examples is obtained as output. MIS was initially tested in arithmetic problems, in axiomatization for dense partial order with end points and synthesizing logic programs for operations of lists, satisfiability of boolean formulas, binary tree inclusion and binary tree isomorphism among others [48].

Respecting the approaches using predicate invention with or without interaction, first we have those based on reformulation with inverse resolution, such as CIGOL [36], LFP2 [59], ITOU [43], RINCON [61], Banerji's System [2] and INDEX [16]. In particular, CIGOL [36], builds upon the operators define in DUCE [34] for FOL based on inverse resolution. The system defines the "V" operators (absorption and identification) and the "W" (intra- and inter-construction) operators. The "V" operator generalizes clauses without introducing new predicates, it can be seen as a single inverse resolution step; the "W" operator introduces new auxiliary predicates and it is a combination of two "V" operators. It represents single inverse resolution steps with clauses which have common literals and is used to introduce new predicates. CIGOL used an oracle to name and test the new predicates. It was tested in the arch problem, definition of lists, finding the minimum of a list, definition of a inverse list, and sorting lists.

Among the approaches based on reformulation are the scheme-driven systems [54]. In these approaches new predicates are introduced with combinations of literals of existing predicates, using schemes or predefined structures to describe the combinations of literals. Among the most representative approaches of this kind are the synthesis of inductive programming proposed in [18], CIA [9] and FOCL [50].

On the other hand, demand driven predicate invention approaches [54], introduce new predicates when the vocabulary is not enough to produce a consistent theory or hypothesis about a concept. Examples using this approach include MENDEL [31], STERES [60], DBC [26], MOBAL/CLT [33, 62], Non-Monotonic Learning [1], MERLIN [5], LUCID [14], SOLAR [23], IGOR 2 [46], *Statistical Predicate Invention* [55], research with teleo-reactive programs [30] and Hyper [29]. In particular, MOBAL [63] introduces new predicates using demand driven reformulation. It can work with the intervention of an oracle, who names the new predicates and evaluates the learned concepts, or in an automatic mode. For the formation of new

concepts the system introduces three operators based on specialization: minimal specialization (the most general), localization (search for restrictions on unique variables to exclude exceptions), and introduction of an existing predicate (as new condition in a rule). The system was tested on databases of traffic violations and security for telecommunications.

In *Statistical Predicate Invention* (SPI) [55] the authors address the discovery of new concepts and propose a generalization of predicate invention, known as statistical learning for hidden variable discovery. The algorithm, *Multiple Relational Clustering* (MRC), is presented to cluster objects, attributes and their relations using Markov logic (an extension of FOL). The process of clustering is done automatically and it is equivalent to the introduction of new predicates (predicate invention), where each cluster represents a unit predicate. The algorithm was tested on databases of animals, nations, tribes, and medicine.

In [30] an algorithm based on teleo-reactive programs is proposed for learning new concepts and skills of different hierarchies from existing knowledge. The teleo-reactive programs represent hierarchical procedural knowledge, and they are based in the reactive execution of goal-oriented skills. In the manner of a STRIPS planner, each skill consists of a goal, an initial state or precondition, an action or method to reach the goal, and a final state or post-condition. First a bottom-up inference mechanism is performed to identify the current state using the perceptions of the agent and the background knowledge, then it searches for the first high-level goal that is not satisfied, and tries to form a path in the hierarchy of skills from the current state of the agent to that goal. When a sequence of skills to achieve the goal can not be found, the algorithm introduces new skills. The new skills form their preconditions and goals using preconditions of concepts and skills closest to the goal. The proposed algorithm was tested on card games.

Hyper is applied to the discovery of concepts using robots [29]. This system is one of the latest systems using demand driven predicate invention. The system works automatically once the experimental setup has been properly prepared as the original Hyper algorithm, but implementing a placeholder predicate in order to perform predicate invention. The robot starts with background knowledge, including a placeholder predicate for the predicate that will be learned. The system reuses previously learned concepts for learning new concepts. Positive and negative examples are provided to the system, the positive examples are obtained by the robot while it explores the environment guided by a human (through commands) and the negative examples are synthetically generated. The arguments of the predicates and their type are manually defined. The system was tested for learning the concepts of movable, not movable, and obstacle.

In most of the predicate invention systems, the user carefully designs all the required elements for learning to take place, by providing, beforehand, suitable background knowledge, the name and arguments of the target predicate, and all the training examples. The success of the system depends on the human skills to provide the right conditions and the systems are not used on dynamic domains with incremental and changing data over time. Also, current systems tend to learn only one predicate at a time, so re-using learned predicates requires the definition of new experimental conditions.

Our research work proposes a demand driven predicate invention algorithm on dynamic environments, where data will be collected and predicates will be learned as the robot explores its environment. We will use graphs to represent the incoming data from the robot and search for similar sub-graphs as input for learning predicates using an *rlgg*-based algorithm. In the following section we describe the relevant work on substructure discovery on graphs.

2.2 Substructure discovery in graphs

In graph-based machine learning algorithms an essential component is the search for frequent sub-graphs representing interesting concepts. In particular, Subdue [13] searches for common sub-graphs within a

graph and replace them with a new node to compress the original graph. Subdue's algorithm is based on the Minimum Description Length (MDL) principle, uses contextual information to guide the sub-graph search, forms hierarchical representations through the replacement of frequent sub-graphs, and uses an algorithm for inexact matching of graphs to identify similar structures [13]. The discovery of frequent sub-graphs starts with a labeled graph which represents structured data. The nodes represent objects of the domain and the edges (directed or undirected) relations between objects.

The search of these substructures in a graph is the identification of groups of nodes connected by edges that are instances of a common substructure (identical or similar instances) [13]. Subdue performs a restricted beam-search.

Using the MDL principle, the best substructure is the one that best compresses the input graph. The measure for inexact matching between two graphs calculate the number of transformations that must be performed to match the two graphs. The transformations can be adding or removing of nodes and edges, changing labels of the edges or changing the direction of edges [24].

In this research proposal, the robot will explore its environment and incrementally build a graph representing relations between objects using its background knowledge. We will use Subdue to find common sub-graphs representing potential concept predicates. The rationale is that common sub-graphs, representing relations between objects in the environment and the robot, can form useful concepts for the robot. Due to possible noisy information and the inexact matching used by Subdue, a large number of very similar sub-graphs can be produced. Similar sub-graphs will be then transformed into sets of predicates and an *rlgg*-based algorithm will be used to induce concepts.

One advantage of using graphs is that it allows to create, in a fairly straightforward way, hierarchies of concepts. In particular, Subdue replaces the common substructures identified by nodes in the graph and then, a new search of substructures on the reduced graph is performed.

2.3 Intrinsically Motivated Reinforcement Learning

Reinforcement Learning (RL) [56] deals with learning optimal or near optimal policies while interacting with the environment. An agent receives rewards while exploring the environment and learns which action to perform on each state to obtain the maximum expected accumulated reward. The reward function, which drives the learning process, is normally defined by the user. Recent research on RL has focused on the definition of a reward function that can be independent of the goal, and that recognizes unexpected events or novelty. RL with intrinsic motivation aims to design a general method of motivation to help an agent to develop skills autonomously, regardless of the traditional external rewards which may or may not exist simultaneously with the intrinsic motivation. This kind of learning was introduced using a representation of skills based on *options*. An *option* is similar to a subroutine, formed by a set of possible start-up states, a behavior policy which determines the set of actions that can be performed in the states belonging to the *option* and a set of ending conditions whereby the *option* ends. The concept of intrinsic motivation is introduced as an element to learn skills at different levels (hierarchies). The agent learns a set of skills through intrinsic rewards based on novelty. The intrinsic reward is calculated according to how well the learned model can predict an outgoing event (interesting, novel) [51].

Some authors have based their approaches on intrinsic motivation functions guided by novelty [22, 45]. In [22] the authors defined a function of novelty which measures the level of agreement between the expected and the obtained observations. The expected observations are calculated using an incremental hierarchical discriminant regression tree (IHDR), more predictable observations have less novelty. The model uses intrinsic and extrinsic rewards, and it was tested in a robotic vision system. Another approach based on

novelty is presented in [45], where a HSOM or habituated self-organizing map was used as model of novelty for the observed states. In this approach, novelty is used directly as reward, however, this type of model may have problems when they are applied to real domains, because random events can occur and be recognized as highly novel [32].

In [25] the authors identified three motivators: predictability, familiarity, and stability. Predictability is measured by calculating the prediction error of an observed state in a situation characterized by a set of observations and actions. Familiarity is measured by calculating how often there is a transition between a given situation (set of observations and actions) and an observed state. Stability is measured by calculating the distance between an observation of a state and its average value over a period of time. These three components are used to calculate the final intrinsic reward, being this larger when the stability is maximized and the familiarity and predictability are increasing. This mechanism avoids given high rewards to random events. The model was tested on sensing and controlling a motor of a robot. However, when the goal is to perform a task rather than staying in certain states, the stability can be a problem. Other motivation measures have used notions of cognitive concepts such as interest, boring and curiosity [32]. One of these approaches is presented in [47]. In this approach, a model of curiosity based on predictability using a neural network is proposed. When the prediction error calculated in a state is large, the reward for visiting again that state is large too. In [32] the authors introduce a model of intrinsic rewards based on the occurrences of events. The model represents the environment through context-free grammars and potential tasks which must be learned through events (changes in the environment). A computational model of interest is introduced as intrinsic motivation. The model was tested on role-play games for computers.

Besides the aforementioned approaches of RL with intrinsic motivation, a vast number of approaches have been developed using traditional RL for learning behavior policies that allow an agent to perform tasks. However, there is little work on learning other elements, which may assist an agent to identify and explore the environment, such as concepts about objects, attributes, relations, and new actions. This thesis proposal poses the use of reinforcement learning with intrinsic motivation to guide the learning process, where new concepts can be used to represent new states in the environment.

The use of an intrinsic motivation function, following the tendency of motivators based on novelty, curiosity and boredom, is also planned, used on its own or simultaneously with a traditional reward function for specific tasks.

2.4 Additional related work

Other closely related approaches have been used for learning concepts with robots. One of these approaches is presented in [27], where an algorithm for concept learning from sensors' data of a mobile robot is presented. The approach learns a hierarchy of basic concepts and concepts based on actions in the form of rules using the relational learning algorithm GRDT. In [58] the author presents a system for learning teleo-reactive programs for mobile robots using the ILP system Aleph and a representation of the environment based on the identification of natural marks. In [20] the author describes an algorithm for learning conjunctive concepts that are consistent with scenes applied to the blocks world. These approaches, however, do not deal with incremental knowledge acquisition in dynamic domains.

Finally, regarding the learning of hierarchies of concepts, this issue has been addressed with different techniques as clustering, relational learning and ILP, teleo-reactive programs, rules of expert systems, neural networks, genetic programming and boolean functions, among others. Among the approaches based on hierarchical concept learning using clustering are LUCID [14] based on clustering and FOL, the system of discovery of subestructures Subdue [21, 24], the system for learning of fuzzy concepts using agglomerative

clusters [11]. Among the systems to learn hierarchical skills using relational learning are the approach for learning concepts from sensors data [27] and learning of skills for games using teleo-reactive programs [30]. Among the approaches based on other techniques are the systems based on genetic algorithms [42], systems inspired by boolean functions and design of circuits ([41], HINT (Hierarchical Induction Tool) [64]), systems using neural networks [57] and approaches based on expert systems [19]. These approaches for hierarchical concept learning usually work with databases (clustering, relational learning and ILP, boolean functions, expert systems, genetic algorithms) or in prepared environments of experimentation (neural networks, teleo-reactive programs, relational learning and ILP), they learn concepts about a specific type or domain (concepts of objects, mainly based on attributes, or concepts of skills), also some of them learn a target concept and/or the intermediate concepts of the hierarchy requiring sets of instances of those concepts as examples for learning. Unlike these systems in this thesis an algorithm to learn concepts about objects and their relations, and also about actions, is proposed building hierarchies of concepts based on data obtained incrementally from the direct experience of an agent with an unknown environment.

3 Research objectives

In this section we describe the main technical challenges, the objectives of this research proposal and its contributions.

3.1 Relevance

In this section is summarized the importance of the research topic addressed in this thesis and also are presented the main characteristics of the techniques that will be used in the algorithm proposed in this thesis. Regarding an agent learning on dynamic domains:

- Agents must learn to perform tasks in unknown situations and face unexpected events [44].
- Autonomous robots must be endowed with skills to improve their performing of tasks through practice. An evaluation method and feedback is necessary to guide the learning [4].
  - Learning could increase the speed and the accuracy to perform a task, or expand the ability of a robot to handle different tasks.
  - Learning is important for robots to work in real world environments.
  - In [12] the problem of what must be learned by robots is addressed, they identified three main issues to be learned:
    - Hard to program knowledge (new skills).
    - Unknown information (like environments).
    - Changing environments (world is dynamic and changes constantly, a robot must learn about changes).
- One way to learn about unknown environments during the exploration is to identify their components and develop new skills. A representation based on abstract concepts can capture these components and skills as part of a learning system for an agent.
  - Abstract concepts can simplify the representation of the world [8].

  - Abstract concepts can make easier the learning of new knowledge and the reasoning about the environment. Also it is possible to design general methods to learn and improve the theory and the understanding of the world instead of improving the performance of learning a specific task [8].

The algorithm of concept learning proposed in this thesis is based on ILP and Reinforcement Learning; next the main characteristics of why these techniques are suitable for the proposed algorithm are summarized. Regarding ILP as an inductive technique for learning concepts:

- Unlike other inductive systems (propositional), ILP can represent relational knowledge and include background knowledge [35].
- It is not restricted by the number and features used in the representation as Attribute-Value representations and fixed features vectors are [15].
- It is based on FOL which have strong theoretical foundations, and can represent objects, attributes and relations between them. Some of the most important features of this logic as can be seen in [44], [15] are:
  - It manages a variable number of objects and relations between them.
  - Almost any assertion of the world can be represented in terms of objects, features and relations.
  - This logic is declarative, it can express incomplete information and it is independent of the context. Therefore, FOL is one of the most natural ways for knowledge discovery, because its language allows to form representations comprehensible by humans (nominal and verbal sentences can be represented).
  - Inference methods allow to obtain knowledge from other previously known knowledge.
- It can learn new vocabulary/concepts (predicate invention) when background knowledge is insufficient [35].

About the main advantages of using Reinforcement Learning to learn policies [44]:

- An agent can learn in unknown environments.
- An agent learns to perform actions from its own direct experience through exploration.
- It is not necessary a teacher to train the agent.
- These characteristics made RL suitable for learning in domains as games and robotics where is not always possible to provide all knowledge previously to the learning.

According to the features described above, about the domain and the techniques, summarizing, the main advantages of the algorithm proposed will be:

- Discovering and learning concepts about objects and actions in unknown environments.
- Induction and use of an incremental and hierarchical representation of the environment based on abstract concepts which can facilitate learning.
- Induction and use of actions learned through direct experience in the environment.
- A self-motivated guide for an agent using an intrinsic motivation function during exploration of unknown environments.

3.2 Research challenges

Autonomously learning from interactions with the environment pose several technical challenges. Learning can involve concept learning useful for describing and interacting with the environment, learning actions to perform new tasks and achieve goals, learning policies for planning sequences of actions, and a strategy to guide and maintain the learning process.

ILP provides a powerful tool for learning relational concepts and can introduce new vocabulary using predicate invention, however:

- It normally work with static databases and environments.
- The learning set-up is carefully predefined by the users for learning specific concepts.
- Usually, only one concept is learned at a time.
- Little work has been done for learning actions and behavior policies.

Besides these problems related with dynamic domains, more generally, predicate invention has some issues which must be addressed independently of the application domain:

- A method to identify when it is necessary to introduce a new predicate.
- How new predicates affect or benefit the learning process.

Learning policies have been the main topic of RL and recently functions have been designed to drive and motivate continuous learning in the area of intrinsically motivated RL, however:

- It is assumed that representations for states and actions are given by the user, they are suitable for the task, and do not change during the learning process.
- Intrinsically motivated reward functions normally work without explicit goals and do not consider changes in the states or actions during the learning process.

In this thesis proposal we plan to address some of these issues. In particular, we plan to learn concepts while the robot is exploring its environment. The learning process will be guided by an intrinsically motivated reward function using the current actions of the robot. It is expected to achieve continuous learning, and consequently to be able to learn more than one concept at a time. Traces of the actions used to achieve goals will be used to learn (near) optimal policies that will be translated into new actions.

3.3 Main objective

Develop an algorithm to incrementally learn concepts about objects and actions while exploring the environment.

3.4 Specific objectives

- Identify and characterize when to learn new concepts about objects and actions while exploring an environment.
- Design the algorithm for incrementally learning concepts about objects (static concepts).

- Design the algorithm for incrementally learning concepts about actions.
- Integrate both approaches re-using previously learned concepts about objects and actions in a coherent way.
- Define a motivation function to drive and keep the learning process.
- Design an evaluation method for the concepts learned by the proposed algorithm.
- Test and evaluate the proposed algorithm on at least two different tasks on dynamic domains.

3.5 Contributions

The main contributions of this proposal are:

- Continuous learning of concepts with demand driven predicate invention, about objects and their relations (static concepts), and also based on actions, on dynamic domains.
- An intrinsically motivated reward function suitable for the learning process.
- Hierarchical learning of static concepts and concepts based on actions.
- Reduction in the utilization of predicates and situations of learning prepared by a human.

4 Methodology

This section describes the methodology that will be followed during our research work. In general terms, we can divide it into three stages:

- Research on the main related work, select an initial experimental design, and familiarize with the main techniques related with the thesis.
- Develop the proposed algorithm.
- Evaluate at different stages the progress of the developed algorithm.

Figure 1 depicts the proposed algorithm. In general terms, an agent explores its environment using an intrinsically motivated reward function and learns a relational representation of its environment. When it achieves a goal, it uses RL to obtain a (near) optimal policy that is then translated into a more abstracted action. Then, the agent explores the environment using the relational concepts and actions learned and uses this more abstracted representation to learn new concepts and actions.

4.1 Selection of the application domain and initial experiments with predicate invention

As part of the methodology, in addition to reviewing state of the art work, we performed some initial experiments with two ILP systems (Aleph, Hyper) and exploratory experiments in concept learning and predicate invention. In these systems, algorithms of generalization/specialization of clauses were analyzed, the effect of adding predicates to the background knowledge on the induction of an hypothesis was observed and experiments were done using predicate invention based on *Generalized Closed-World Specialization*. Also, robotics was chosen as an application domain to test our main ideas.

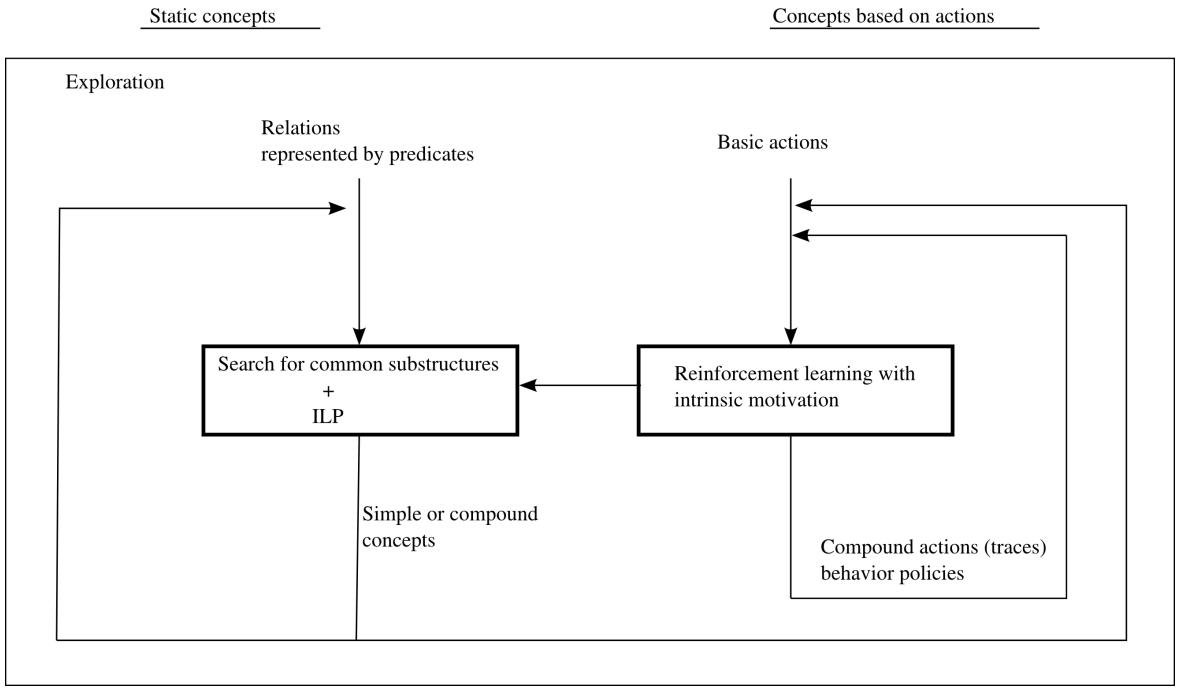

Figure 1: This figure illustrates in a block diagram the main components of the proposed algorithm and how they are related. An agent uses basic actions to explore its environment while discovering objects and their relations among them. With objects and their relations, through the algorithm of substructure discovery and ILP, the static concepts are formed. RL with intrinsic motivation guides the formation of static concepts and concepts based on actions. From the static concepts, which represent the states in the environment, and basic/compound actions, sequences representing more complex actions are formed. The compound concepts can be used during exploration for the formation of new concepts producing a hierarchy of concepts.

4.2 Design of the algorithm for concept discovery

In the learning process, we will learn, what we call, static concepts and concepts based on actions.

4.2.1 Static concepts

In this subsection the steps for learning static concepts are described. Static concepts are defined by Horn clauses with literals describing, properties of and relations between, objects. For example, a definition for a room could be expressed as:

$$\begin{aligned} room(State) :- \quad & wall(Wall1, State), wall(Wall2, State), \\ & wall(Wall3, State), wall(Wall4, State), \\ & touches(Wall1, Wall2), touches(Wall2, Wall3), \\ & touches(Wall3, Wall4), touches(Wall4, Wall1). \end{aligned}$$

where *wall/2* identifies a "wall" in the current state of the environment, and *touches/2* describes that two walls are connected in some way. Figure 2 shows the main components of how to learn this type of concepts.

1. It is assumed that the robot starts with an initial set of basic actions that allow the robot to explore its environment (e.g., move-forward, turn-right, etc.), and with initial background knowledge definitions to identify basic properties of its environment (e.g., wall, touches, etc.).

2. The robot explores its environment guided by its intrinsically motivated function.

3. While exploring its environment the applicable known static concepts are collected and translated into a graph-based representation.

   (a) Where objects and attribute values from the triggered predicates are represented by nodes and binary relations between objects and attribute names are represented by arcs. When the arity of a relation is greater than two, conceptual graphs [52] can be used to represent objects and their relations as concepts and conceptual relations (both, as nodes of a graph).

4. This graph-construction process continues until the reward function identifies an intrinsic or an extrinsic goal, after which a new graph is constructed with further interactions with the environment.

5. Once the graphs are constructed they are given to Subdue to discover common sub-graphs. Subdue has a threshold of tolerance for accounting differences between sub-graphs, and can produce several, very similar, sub-graphs of essentially the same concept. All the similar sub-graphs (measured with Subdue metrics) are translated back into predicates and given to an ILP system. We are currently using an algorithm based on *rlgg*. To avoid, at least partially, over-generalizations, we will compare our hypotheses with other existing concepts and if they are covered, the hypothesis is rejected and a new disjunct definition is induced.

6. The number of arguments for the new predicate will depend on the arguments used in its definition, while the name will be given by the user.

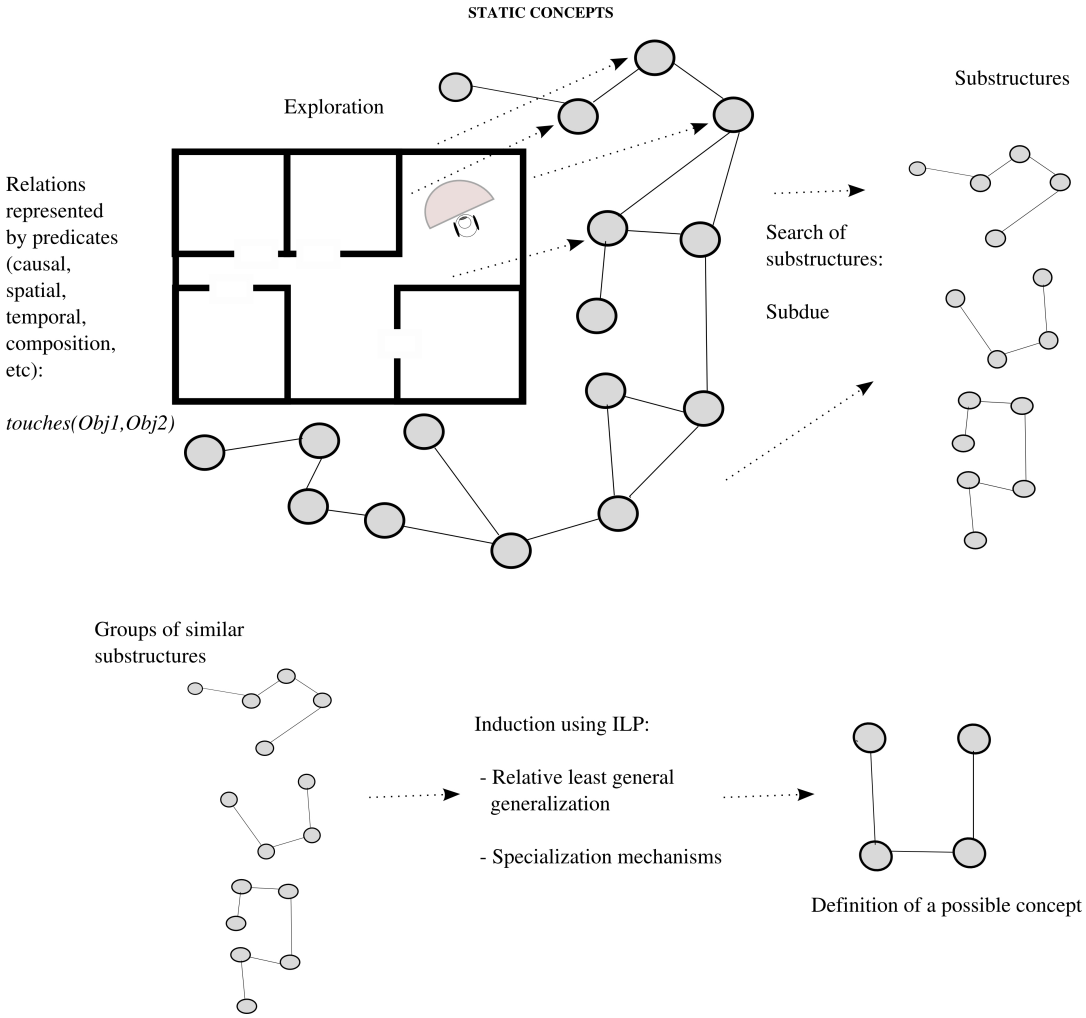

Figure 2: In this figure the main steps for the discovery of static concepts using the proposed algorithm are illustrated. An agent uses basic actions to explore an environment while discovering objects in it. With the basic relations, provided as background knowledge, the agent discovers the relations between the identified objects in the environment. Objects and their relations are represented with a graph. The algorithm of substructure discovery is applied on the graph. The substructures found are used as examples for an algorithm based on ILP and definitions are induced for potential concepts. The potential simple and compound (consists of simple concepts) concepts are used to identify concepts during the exploration and form new concepts from them.

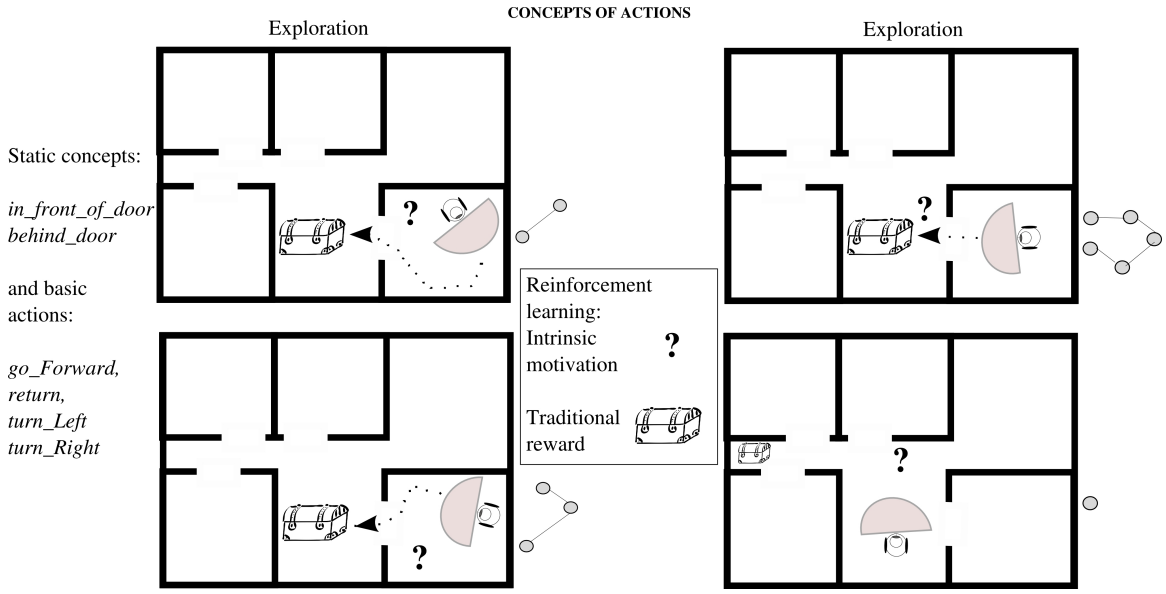

Figure 3: In this figure the main steps for the discovery of concepts of actions are illustrated. An agent explores the environment using a set of basic actions and using static concepts to identify states in the environment. In this example the agent learns 'how to leave a room'. During the exploration it uses reinforcement learning with intrinsic motivation, illustrated with a "**?**", to guide the agent to experience new situations, and also traditional rewards, illustrated as a chest, to mark goals in the environment. The vertices of the graph shown to the right to the maps represent the formation of the relational representation of the environment during exploration (these graphs are used to discover new static concepts). In this way, the agent learns sequences of actions and behavior policies to represent concepts based on actions. After learning new concepts based on actions, they can be used to explore the environment in order to learn new tasks. This fact is depicted in the figure with the setting of a new goal (chest on other site) and resetting the formation of a new graph.

4.2.2 Concepts based on actions

Concepts based on actions are also defined by Horn clauses with literals describing the pre-conditions for the action to be applied, a predicate that executes the action, and literals describing the post-conditions after the action has been applied. For example, the action "leaving a room" could be defined as:

```
leaving_room(Initial_State,goForward,Final_State) :-
        in_front_of_door(Initial_State),
        action(Initial_State,goForward,Final_State),
        behind_door(Final_State).
```

Figure 3 shows the main steps of this part of the algorithm.

1. It is assumed that the robot starts with an initial set of basic actions that allow the robot to explore its environment (e.g., move-forward, turn-right, etc.), and with initial background knowledge definitions to identify basic properties of its environment (e.g., wall, touches, etc.).
2. The robot explores its environment guided by its intrinsically motivated function.
3. While exploring its environment after each action a concept based on actions is, if new, induced, with the applicable known static concepts, the performed action (or sequence of actions), and the applicable known static concepts after the action.
4. The number of arguments for the new predicate will depend on the arguments used in its definition, while the name will be given by the user.
5. This process continues until the reward function identifies an intrinsic or an extrinsic goal, and the system is re-started. Also at this point the state-action pairs build during this process are rewarded using a standard Q-learning algorithm.
6. This process is repeated until an (near) optimal policy is learned.

4.2.3 Reward function

The reward function, that will drive the learning process, must promote the discovery of new concepts and the learning of new skills. We will use ideas from intrinsically motivated reward functions as a starting point. One commonly used reward function is:

$$f(predictedState, currentState) = \begin{cases} 0 & \textit{predictedState} = \textit{currentState} \\ 1 & \textit{predictedState} \neq \textit{currentState} \end{cases}$$

where novelty or inability to predict a state is highly rewarded. However, noisy events can produce false motivations. Trying to avoid this, another popular function is:

$$f(predictedState, currentState) = \begin{cases} > & \textit{predictedState} \approx \textit{currentState} \\ 0 & \textit{predictedState} = \textit{currentState} \\ < & \textit{predictedState} \neq \textit{currentState} \end{cases}$$

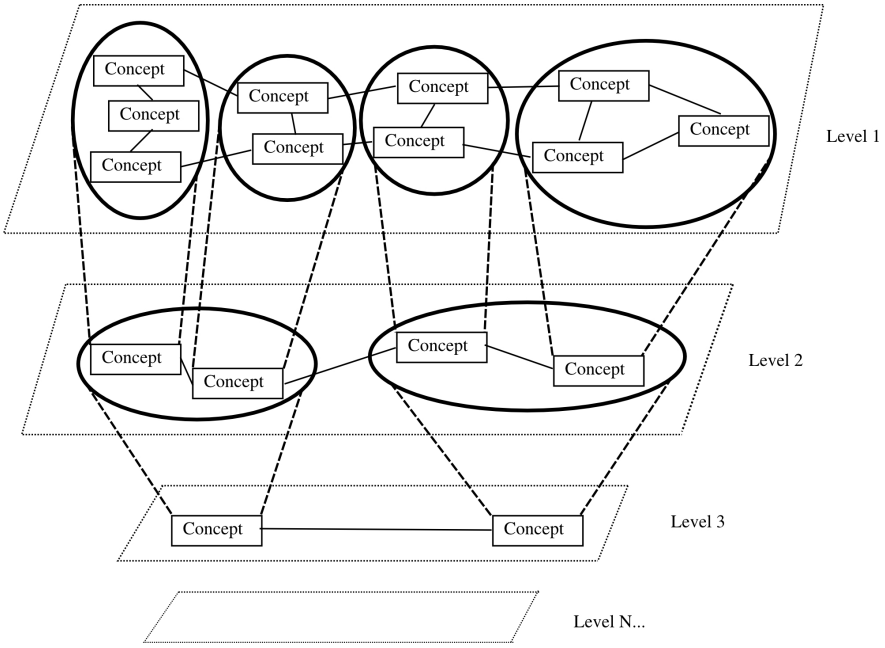

Figure 4: In this figure the formation of concepts at different levels of a hierarchy is illustrated. Each level will have a set of concepts related between them. For each level, groups of concepts will be formed, each group will encompass the components of a potential concept at a higher level. This process can be repeated in order to form conceptual levels as relations between concepts are discovered during exploration in the environment.

where unpredicted states are penalized, states with some imperfect prediction are highly rewarded, and completely predicted states are ignored. This function, however, is penalizing novelty and does not take into account other aspects, like learning new concepts. During the development of this research we will define a suitable reward function for our learning system.

4.2.4 Hierarchy of concepts

Finally, this last section describes how the static concepts and the concepts based on actions can be part of a hierarchy of concepts. Two hierarchies will be formed, one for static concepts and other for concepts of actions, however, as seen above, to the formation of the hierarchy of concepts based on actions, static concepts of different hierarchical levels will be related with actions according to the relations discovered during the exploration of the environment. Figure 4 shows a schematic representation of the general idea, where groups of concepts at some level in the hierarchy represent a single concept at a higher level.

- For static concepts, the hierarchy is build almost in a straightforward way, since Subdue automatically learns a hierarchical structure. With the graph-based representation, several concepts can be learned at the same time at the same or at different hierarchical levels. Once a new concept is learned, as soon as it it recognized while the robot is exploring its environment, it is used as a new node in the current graph-based representation.

- For concepts based on actions, the actions used for achieving goals (either intrinsically or extrinsically) will receive rewards, and a policy will be learned for that task. A hierarchical goal will be build the same way as for basic actions, however, instead on using after the preconditions a basic action, it will use the learned policy. Details of this stage will become clearer as the research progresses.
- To avoid the proliferation of irrelevant knowledge, static concepts and concepts based on actions that are rarely used will be discarded.

4.3 Evaluation of the proposed algorithm

The proposed framework will be evaluated in robotics, and possibly in other domains. It is difficult to make direct comparisons with existing work or use benchmark problems, so we will evaluate the general performance of the method and its individual components. In particular:

- Evaluation of the algorithm for each kind of concept:
  - For static concepts we will consider the number of concepts, their accuracy and their usefulness in the performance of the robot.
    - Comparison with other ILP systems: Hyper [29].
  - For concepts based on actions we will consider the number of induced actions, the convergence times of the policies, cumulative reward and the usefulness of the actions for performing tasks.
    - Comparison of the concepts of actions (learned by the proposed system) with the actions learned using only RL (Q-Learning or Sarsa).
- We will measure the overall performance of the system in terms of achieving goals.
  - Learning of hierarchical concepts on the domains:
    - Testing the learning of static concepts: Exploring and learning about a structural domain (floors of buildings).
    - Testing the learning of actions: A navigation/manipulation task (Pionner, robotic arm/gripper).
    - Testing the complete algorithm: A navigation task on a structural domain, going to a goal site from other.
  - Learning of similar concepts as those learned by systems of related work:
    - *Move through the door* [27].
    - *Obstacle* and *movability* [29].
    - Learning of concepts with and without the designed intrinsic motivation function. And the designed intrinsic motivation function against other existent intrisic motivation functions.

5 Preliminary results

We performed some exploratory experiments with two ILP systems (Aleph [53] and Hyper [7]) and explore some ideas of predicate invention using *Generalized Closed-World Specialization* in Aleph. Subsequently, the proposed algorithm of this research was designed. Initially, an initial algorithm was designed using a representation of concepts similar to the one presented in [27], using Aleph to induce definitions of concepts, performing random exploration in robotics. This first algorithm designed specifically for the task, served to lay the foundations and identify features that were not initially considered. In the following sections we describe some results for learning static concepts with the proposed methodology.

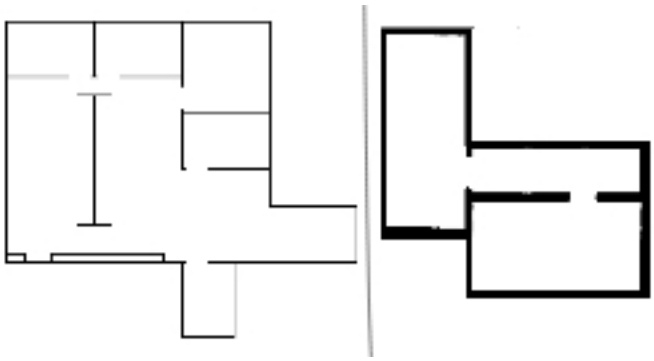

Figure 5: In this figure the two maps used in the experiments are illustrated.

5.1 Static concepts: Initial experiments and results

Initial experiments were performed to discover concepts in structural domains (maps of buildings with rooms connected by hallways) using a robot. A *Pioneer 2* robot equipped with a laser sensor (with a range of 180°) was used. The experiments were simulated using the software *Player/Stage* and repeated 20 times. The objective of these experiments was to test all the main elements of the proposed algorithm in the formation of static concepts. For these experiments, two maps were used (shown in Figure 5) and the robot was manually driven by the user to collect data from each map. It is expected that the exploration process in the final algorithm will be guided by intrinsic motivation.

An incremental algorithm for identification of lines was used in the experiments. This algorithm was chosen for its ability to abstract features of the environment (basically formed by walls) from laser sensor data and for its superior performance in comparison with other algorithms of its kind [40]. Our initial background knowledge consisted then on two predicates: "wall/2", representing an object (wall) of the world, and "touches/2", representing when two walls form a corner. From the exploration, a graph-based representation was build, with each "wall" represented by a node and the binary relation "touches" by an edge. Two graphs were built, one per map. The graph of the smallest map, formed by 24 nodes, is illustrated in Figure 6, the largest graph was formed by 38 nodes.

Subdue was used to find the best three substructures for compressing the graph (evaluated with the minimum description length principle, MDL), with no directed edges and a threshold of 0.3 (the fraction of the size of an instance by which the instance can be different). Examples of the substructures found in both maps are shown in Figure 7. There were used as examples to induce a definition excluding the last one (because it was the least similar to the others). The similarity between substructures in these experiments was determined by a user, however, during the research work, we will determine a suitable similarity measure to use.

The substructures were transformed into clauses and given to a relative least general generalization algorithm, as described in [17]. The induced definition is illustrated in Figure 8. From this result, it is possible to identify the concept induced as "room with a door". The definition has two extra elements that were included because in the algorithm to detect walls, artificial walls were created between the corner of the door and the wall outside the room. We are fixing this bug. Also no additional methods were used to refine the definition and we used a small number of examples.

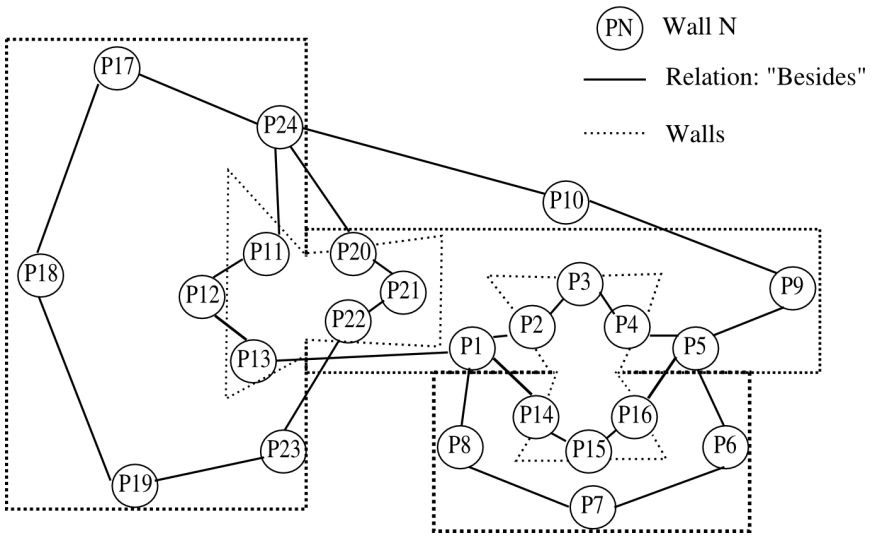

Figure 6: This figure shows the graph of the second map (the smallest) as example of the relational representation used, superimposed on the map. The walls are represented by nodes, each node is identified by a letter **P** plus a number, the edges correspond to the relation "touches" and are showed with a continuous line. The walls of the map are showed as a dotted line. The edges that correspond to each wall of the map are the nearest to them.

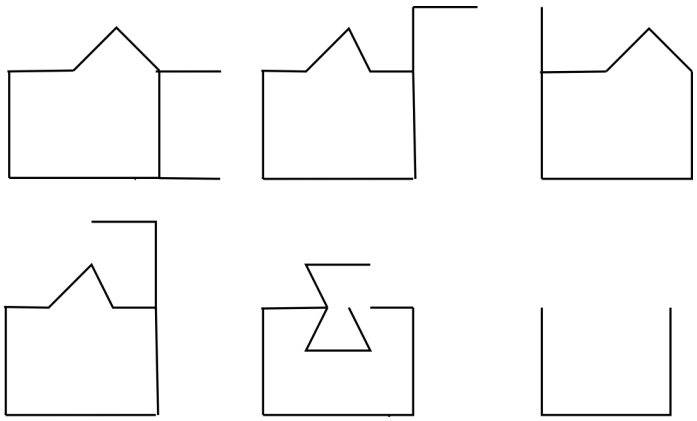

Figure 7: This figure shows the substructures identified by Subdue. The first row corresponds to the largest map, while the second row corresponds to the smallest map.

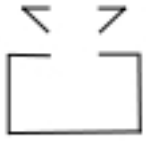

Figure 8: In this figure the definition induced for the concept "room" is illustrated.

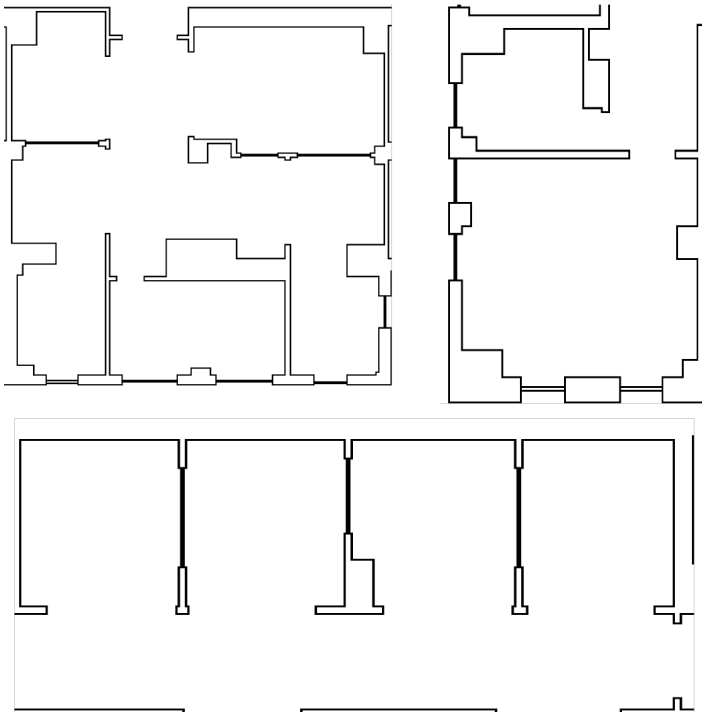

Figure 9: In this figure the three additional maps used in the experiments are illustrated.

These first experiments were done in order to test the main elements of the proposed algorithm for learning static concepts. The results show that, in principle, it is possible to learn this type of concepts using a mobile robot. It is important to point out that the robot did not know what concept would be learned.

The static concepts will be used to represent the states which will be part of the definition of the concepts based on actions. Even though the initial experiments were done to learn a concept at the bottom level of the hierarchy of concepts, the algorithm is designed to be used at different levels without additional changes. It is expected that the coding of the algorithm for intrinsically motivated RL will be done with greater ease since we have previous experience from our master's thesis in RL algorithms with *reward shaping*.

5.2 Static concepts: Refinement of the algorithm, experiments and results

In order to continue with the developing and testing of the first part of the proposed algorithm corresponding to the learning of static concepts, some additional refinement of the algorithm and experiments were done. The objectives of these new experiments were continue testing the proposed algorithm, and show its utility as method of concept discovery in comparison with the algorithms of discovery of substructures, as Subdue.

Extending the initial experiments, three maps were added to the initial two presented above, see Figure 9. The same robot equipped with the laser sensor, as was described in the previous section, explored and collected the data from the structural environments represented in the maps. The data collected were used to induce concepts following the proposed algorithm for learning static concepts, summarizing the main steps refined:

1. A graph for each map must be done following the proposed algorithm as seen above.
2. After that, use Subdue to find the substructure which best compress the input graph.
3. Using this discovered substructure, set a definition of potential concept to compress the graph.
   (a) If this discovered substructure is covered by an existent definition or it is similar to others previoulsy seen:
      i. If the substructure is covered by a definition previously induced by ILP for a group of substructures, use this definition to compress the graph.
      ii. If the substructure is not covered by an existent definition induced by ILP, but it is similar to a group of substructures previoulsy seen, add the substructure to this group, and adjust the definition for this set with the new substructure using ILP. Use this new definition to compress the graph.
         A. The measures of similarity used to group the substructures representing potential examples of concepts are:
            - The inexact matching measure provided by Subdue.
            - The number of common attributes/relations between the bodies of clauses of potential concepts compared.
      iii. If there are predicates representing attributes/relations in the definition induced by ILP with variables as arguments, choose a value from the domain of each variable arbitrarily and instantiate them. Use this definition with their variables instantiated to compress the graph. But, keep the definition with its original arguments (variables) as the definition of the potential concept.
   (b) If the substructure is not covered by an existent definition or it is not similar to others substructures previously seen, use this substructure (representing the initial definition for a new potential concept) to compress the graph.
4. Once chosen or induced a definition of the potential concept, use it to compress the graph.
5. Back to the first step, but use the new compressed graph as input, and repeat the process until the graph is fully compressed.
6. At the end, additional definitions of potential concepts can be induced by ILP using definitions of concepts from different levels of the hierarchy as examples of hierarchical potential concepts.

The proposed algorithm was compared with Subdue, which was used to discover concepts on the data collected by the robot too. The results for the five maps explored by the robot using the proposed algorithm and Subdue are shown in the Tables 1 and 2.

In the Table 1 are showed results of experiments using only the first two maps (presented in the section 5.1), G1 representing the graph of the small map and G2 representing the graph of the larger map. The proposed algorithm for learning static concepts is labeled as "Subdue+ILP" and the label "Subdue" is used to specify that only Subdue was used to discover concepts. The experiments labeled with "G2 after G1" indicate that first G1 was analyzed to find potential concepts, and then G2 was processed to find other new potential concepts or, when it was possible, it was processed using the concepts previously discovered in G1. This experiment was repeated using a higher threshold of inexact matching of Subdue as can be seen in the

Table 1: Results

Experiment 1: Structural environment	No. Simple Concepts (#G1/#G2)	No. Hierarchical Concepts (#G1/#G2)	No. ILP Concepts (#s+#h+#hv)	No. Concepts
Subdue +ILP: G2 after G1	2/0	2/7	1s+2hv	14
Subdue +ILP: G2 after G1 (higher threshold)	2/0	2/7	1s	12
Subdue +ILP: G1 and G2 simultaneously	0/1	1/4	1s+2h+1hv	10
Subdue: G1+G2	2	13	-	15
Subdue: G2 after G1	3/0	3/2	-	8
Subdue: G1 and G2 independently	3/2	3/6	-	14

Table. The experiment labeled with “G1 and G2 simultaneously” indicate that both graphs were processed at the same time, using subgraphs from both graphs as examples of potential concepts. The experiment labeled with “G1+G2" indicate that the graph G2 was attached to the graph G1 to create a single graph, in this way, only the resulting graph was processed to discover potential concepts. The experiment labeled with “G1 and G2 independently" indicate that the discovery of potential concepts was done independently for each graph without share definitions of concepts or examples as in other experiments.

In the Table 1, the first column indicate which experiment was done, the second and third column contain the number of simple (without include any other concept in their definition) and hierarchical (including other concepts previously discovered in their definitions) concepts discovered, without definitions induced by ILP. The fourth column contains the number of concepts discovered whose definition was induced by ILP, indicating with “s” the simple concepts, “h” the hierarchical concepts and “hv” the ones induced using as examples the concepts of all the different hierarchical levels. The last column indicate the total number of concepts that were induced in each experiment. In the Table 2 are showed the results of the experiments done with the five maps, in the same way as the previous Table. The experiments are labeled to indicate if the proposed algorithm was used or only was used Subdue, and how the graphs were processed to discover potential concepts (attached, simultaneously, independently). The experiments done with the five maps were only those whose conditions would be more like the presented in dynamic domains.

In the Table 1, the number of concepts induced using the proposed algorithm is similar to the number of concepts discovered using only Subdue. In fact, the smaller number of concepts is obtained in the experiment “G2 after G1” using Subdue. However, the similar concepts or “repeated” concepts decrease using the proposed algorithm as expected. This fact can be seen more clearly in the Table 2 where were used the five maps and where a larger number of similar substructures could be found. In these experiments, the number of concepts decreases by half using the proposed algorithm. These results support the proposed algorithm as method for discovery of concepts significantly different from each other. The experiment using a single graph (“G1+...+G5”) has the same number of concepts that the experiment using ILP. But, in real situations of dynamic domains, graphs can not be attached in this way. This experiment was done only to evaluate Subdue. Further, using the proposed algorithm is possible to induce hierarchical concepts beyond a structural similarity measure which can not be discovered using only a system of substructure discovery. Among the concepts learned a user could identify concepts as “room”, “hallway”, “corner” and “floor” (set of rooms, hallways and others) among others. However, as the next step is necessary to evaluate the utility and relevance of these concepts in the domain where were learned.

Table 2: Results

Experiment 2: Structural environment	No. Simple Concepts	No. Hierarchical Concepts	No. ILP Concepts (#s+#h+#hv)	No. Concepts
Subdue +ILP: G1 to G5 simultaneously	1	12	3s+3h+4hv	23
Subdue: G1+...+G5	3	20	-	23
Subdue: G1 to G5 independently	12	30	-	42

5.3 Conclusions and future work

In domains like robotics or games, it is desirable for an agent to learn autonomously components of its environment, such as objects, attributes and the relations between them, and actions to explore the environment and solve tasks. Approaches based on ILP with predicate invention have proposed techniques to learn concepts about objects and attributes. However, these approaches have been applied to databases assembled in advance, one concept is learned per learning session, concepts of actions are rarely learned, and the sessions of learning are prepared by a human. Other approaches based on RL learn behavior policies in dynamic domains. Nonetheless, these approaches assume that the states and actions representations are suitable for the task which is being learned, and generally, they learn behavior policies and do not learn concepts about objects. Considering these problems, this thesis proposal suggest an algorithm for learning concepts for objects and actions guided by an intrinsically motivated function applied to dynamic domains.

The first algorithm for demand driven predicate invention applied to a dynamic domain was presented. This part of the algorithm was tested on data obtained by a robot exploring its environment. It is important to point out that the robot did not know what concept to learn. The initial experiments using the proposed algorithm showed that it is possible to induce static concepts, which could be part of a hierarchy of concepts to model the knowledge acquired while an agent is interacting directly with the environment. These concepts induced can be used to represent states of the environment used in the formation of concepts based on actions. Also, additional experiments to support the proposed algorithm of learning static concepts as method of discovery of concepts were presented.

As future work we must continue working on our initial algorithm for learning static concepts. We believe that learning several concepts in a hierarchy can be achieved relatively fast, however, we still need to deal with probable over-generalizations and the design of an intrinsic motivated function to guide the process. We need to implement the learning algorithm for concepts based on actions for which we will build on our previous experience in the development of RL systems. The whole system needs to be integrated and rigorously evaluated.

6 Work plan

Table **??** presents the activities of the methodology and the task assignment by year.

References

[1] M. Bain and S. Muggleton. Non-monotonic learning. In *Inductive Logic Programming*, pages 145–161. Academic Press, 1992.

[2] R. Banerji. Learning theoretical terms. In *Inductive Logic Programming*, pages 94–112. Academic Press, 1992.

Table 3: Work plan summarizing the main activities of the thesis by year

ACTIVITY	YEAR 1	YEAR 2	YEAR 3	YEAR 4
Selection of tasks and application domains, initial experimentation with predicate invention.	X			
Design of the algorithm for concept discovery.				
Design the strategy for learning of concepts with demand driven predicate invention.	X			
* Identify situations in which new predicates must be introduced.	X	X	X	
- Build static concepts selecting a subset of basic elements related between them, potential elements of a concept, from the set of elements in the environment.	X	X		
- Build concepts based on actions using the static concepts to characterize initial and final states that precedes and follows an action, and a predicate representing the action.		X	X	
* Induce definitions for the potential concepts.	X	X	X	
* Build the incremental hierarchy of concepts reusing predicates previously learned.		X	X	
- Evaluate the utility and the quality of the new predicates introduced.			X	X
Evaluation of the proposed algorithm.			X	X

[3] J. Barwise. An introduction to first-order logic. *Handbook of Mathematical Logic. Studies in Logic and the Foundations of Mathematics*, 1982.

[4] G. Bekey. *Autonomous Robots: From Biological Inspiration to Implementation and Control*. MIT Press, 2005.

[5] H. Bostrom. Predicate invention and learning from positive examples only. In *In Proceedings of the Tenth European Conference on Machine Learning*, volume 1398 of *Lecture Notes in Computer Science*, pages 226–237. Springer, 1998.

[6] R. Branchman and H. Levesque. *Knowledge representation and reasoning*. Morgan Kaufmann, 2004.

[7] I. Bratko. Refining complete hypotheses in ilp. In *Proceedings of the 9th International Workshop on Inductive Logic Programming*, pages 44–55, London, UK, UK, 1999. Springer-Verlag.

[8] I. Bratko. Discovery of abstract concepts by a robot. In *Proceedings of the 13th international conference on Discovery science*, volume 6332, pages 372–379, Berlin, Heidelberg, 2010. Springer-Verlag.

[9] M. Bruynooghe and K. Morik. Interactive concept-learning and constructive induction by analogy. In *Machine Learning*, volume 8.

[10] L. Cauman. *First-Order Logic: An Introduction*. De Gruyter, 1998.

[11] B. Chien, C. Hu, and M. Ju. Learning fuzzy concept hierarchy and measurement with node labeling. *Information Systems Frontiers*, 11:551–559, 2009.

[12] J. Connell and S. Mahadevan. In *Robot Learning*, volume 233. 1993.

[13] D. Cook and L. Holder. Substructure discovery using minimum description length and background knowledge. *J. Artif. Int. Res.*, 1(1):231–255, 1994.

[14] J. Davis, E. Berg, D. Page, V. S. Costa, P. Peissig, and M. Caldwell. Discovering latent structure in clinical databases. In *Proceedings of NIPS 2011 Workhop: From Statistical Genetics to Predictive Models in Personalized Medicine*, Granada, Spain, 2011.

[15] L. De Raedt. *Logical and relational learning*. Springer-Verlag Berlin Heilderberg, 2008.

[16] P. Flach. Predicate invention in inductive data engineering. In *Proceedings of the European Conference on Machine Learning*, volume 667, pages 83–94, London, UK, UK, 1993. Springer-Verlag.

[17] P. Flach. *Simply Logical*. John Wiley, 1994.

[18] P. Flener. Predicate invention in inductive program synthesis. Technical report, 1995.

[19] L. Fu and B. Buchanan. Learning intermediate concepts in constructing a hierarchical knowledge base. In *Proceedings of the 9th international joint conference on Artificial intelligence - Volume 1*, IJCAI'85, pages 659–666, San Francisco, CA, USA, 1985. Morgan Kaufmann Publishers Inc.

[20] D. Haussler. Learning conjunctive concepts in structural domains. *Machine Learning*, 4(1):7–40, Oct. 1989.

[21] L. B. Holder, D. J. Cook, and S. Djoko. Substructure discovery in the subdue system. In *AAAI Workshop on Knowledge Discovery in Databases*, pages 169–180, 1994.

[22] X. Huang and J. Weng. Novelty and reinforcement learning in the value system of developmental robots. In *Lund University Cognitive Studies*, pages 47–55, 2002.

[23] K. Inoue, K. Furukawa, and I. Kobayashi. Abducing rules with predicate invention. In *Proceedings of the 19th International Conference on Inductive Logic Programming*, 2009.

[24] I. Jonyer, D. J. Cook, and L. B. Holder. Graph-based hierarchical conceptual clustering. *Journal of Machine Learning Research*, 2(2):10–1, 2001.

[25] F. Kaplan and P. Oudeyer. Motivational principles for visual know-how development. In *Proceedings of the 3rd international workshop on Epigenetic Robotics : Modeling cognitive development in robotic systems*, pages 73–80, 2003.

[26] B. Kijsirikul, M. Numao, and M. Shimura. Efficient learning of logic programs with non-determinant, non-discriminating literals. In *Inductive Logic Programming*, pages 417–421. Academic Press, 1992.

[27] V. Klingspor, K. Morik, and A. Rieger. Learning concepts from sensor data of a mobile robot. In *Machine Learning*, pages 305–332, 1996.

[28] N. Lavrac and S. Dzeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, 1994.

[29] G. Leban, J. Zabkar, and I. Bratko. An experiment in robot discovery with ILP. In F. Zelezný and N. Lavrac, editors, *Inductive Logic Programming*, volume 5194 of *Lecture Notes in Computer Science*, pages 77–90, Berlin, Heidelberg, 2008. Springer.

[30] N. Li, D. Stracuzzi, and P. Langley. Learning conceptual predicates for teleoreactive logic programs. In *Proceedings of the Late-Breaking Papers Track at the Eighteenth International Conference on Inductive Logic Programming*, Prague, Czech Republic, 2008.

[31] C. Ling. *Inventing necessary theoretical terms: in scientific discovery and inductive logic programming*. Reporte técnico (University of Western Ontario. Dept. of Computer Science). Dept. of Computer Science, University of Western Ontario, 1991.

[32] K. Merrick and M. Mahler. Modelling motivation as an intrinsic reward signal for reinforcement learning agents. *Machine Learning*, 2006.

[33] K. Morik. Balanced cooperative modeling. In *First International Workshop on MultiStrategy Learning*, pages 65–80. Morgan Kaufmann, 1991.

[34] S. Muggleton. Duce, an oracle-based approach to constructive induction. In *Proceedings of the 10th international joint conference on Artificial intelligence*, volume 1, pages 287–292, San Francisco, CA, USA, 1987. Morgan Kaufmann Publishers Inc.

[35] S. Muggleton. Inductive logic programming. In *New Generation Computing*. Academic Press, 1992.

[36] S. Muggleton and W. Buntine. Machine invention of first-order predicates by inverting resolution. In *Proceedings of the 5th International Conference on Machine Learning (ICML 88)*, pages 339–352. Morgan Kaufmann, 1988.

[37] S. Muggleton and C. Feng. Efficient induction of logic programs. In *New Generation Computing*. Academic Press, 1990.

[38] S. Muggleton, L. Raedt, D. Poole, I. Bratko, P. Flach, K. Inoue, and A. Srinivasan. *Machine Learning*, 86, 2012.

[39] S. Muggleton and K. Road. Predicate invention and utilisation. *Journal of Experimental and Theoretical Artificial Intelligence*, 6:6–1, 1994.

[40] V. Nguyen, S. Gächter, A. Martinelli, N. Tomatis, and R. Siegwart. A comparison of line extraction algorithms using 2d range data for indoor mobile robotics. *Autonomous Robots*, 23(2):97–111, 2007.

[41] R. Rivest and R. Sloan. A formal model of hierarchical concept learning. *Inf. Comput.*, 114:88–114, 1994.

[42] J. Rosca. *Hierarchical Learning with Procedural Abstraction Mechanisms*. PhD thesis, Department of Computer Science, The College of Arts and Sciences, University of Rochester, Rochester, NY 14627, USA, 1997.

[43] C. Rouveirol. Extensions of inversion of resolution applied to theory completion. In *Inductive Logic Programming*, pages 63–92, London, 1992. Academic Press.

[44] S. Russell and P. Norvig. *Inteligencia Artificial. Un enfoque moderno*. Pearson. Prentice Hall, 2008.

[45] R. Saunders and J. Gero. Designing for interest and novelty - motivating design agents. In *Proceedings of the ninth international conference on Computer aided architectural design futures*, pages 725–738. Kluwer, 2001.

[46] U. Schmid, M. Hofmann, and E. Kitzelmann. Inductive programming: Example-driven construction of functional programs. *Kúnstliche Intelligenz*, 23(2):38–41, 2009.

[47] J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats*, pages 222–227, Cambridge, MA, USA, 1990. MIT Press.

[48] E. Shapiro. *Inductive inference of theories from facts*. Research report, Yale University. Dept. of Computer Science. Yale University, Department of Computer Science, 1981.

[49] E. Y. Shapiro. The model inference system. In *Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2*, IJCAI'81, pages 1064–1064. Morgan Kaufmann Publishers Inc., 1981.

[50] G. Silverstein and M. Pazzani. Relational cliches: Constraining constructive induction during relational learning. In *In Proceedings of the Eighth International Workshop on Machine Learning*, pages 203–207. Morgan Kaufmann, 1991.

[51] S. Singh, A. Barto, and N. Chentanez. *Intrinsically Motivated Reinforcement Learning*, pages 1281–1288. MIT Press, Cambridge, MA, 2005.

[52] J. Sowa. *Handbook of knowledge representation (Conceptual Graphs, Ch. 5)*. Elsevier B. V., 2008.

[53] A. Srinivasan. A learning engine for proposing hypotheses, 1993.

[54] I. Stahl. Predicate invention in inductive logic programming. In L. D. Raedt, editor, *Advances in Inductive Logic Programming*, pages 34–47. IOS Press, 1996.

[55] K. Stanley and P. Domingos. Statistical predicate invention. In *Proceedings of the 24th annual international conference on machine learning (ICML-2007*, pages 433–440, 2007.

[56] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning Series. Bradford Book, 1998.

[57] J. Tani and S. Nolfi. Learning to perceive the world as articulated: An approach for hierarchical learning in sensory-motor systems. In *NEURAL NETWORKS*, pages 1131–1141. MIT Press, 1999.

[58] B. Vargas. *Aprendizaje de Programas Teleo-Reactivos para Robótica Móvil*. Ph.d. dissertation, Computer Science Department, National Institute of Astrophysics, Optics and Electronics, 2009.

[59] R. Wirth. Learning by failure to prove. In *EWSL*, pages 237–251, 1988.

[60] R. Wirth and P. ORorke. Constraints on predicate invention. In *8th International workshop on Machine Learning*, pages 457–461. Morgan Kaufmann, 1991.

[61] J. Wogulis and P. Langley. Improving efficiency by learning intermediate concepts. In *Proceedings of the 11th international joint conference on Artificial intelligence*, volume 1, pages 657–662, 1989.

[62] S. Wrobel. Exploiting a problem-solving context to focus concept formation. *Machine Learning*.

[63] S. Wrobel. Concept formation during interactive theory revision. *Machine Learning*, 14(2):169–191, 1994.

[64] B. Zupan, M. Bohanec, I. Bratko, and J. Demšar. Learning by discovering concept hierarchies. *Artificial Intelligence*, pages 211–242, 1999.