# Exploration in Relational Worlds

Tobias Lang[1], Marc Toussaint[1], Kristian Kersting[2]

[1] Machine Learning and Robotics Group, Technische Universität Berlin, Germany
tobias.lang@tu-berlin.de, mtoussai@cs.tu-berlin.de
[2] Fraunhofer Institute IAIS, Sankt Augustin, Germany
kristian.kersting@iais.fraunhofer.de

**Abstract.** One of the key problems in model-based reinforcement learning is balancing exploration and exploitation. Another is learning and acting in large relational domains, in which there is a varying number of objects and relations between them. We provide a solution to exploring large relational Markov decision processes by developing relational extensions of the concepts of the Explicit Explore or Exploit ($E^3$) algorithm. A key insight is that the inherent generalization of learnt knowledge in the relational representation has profound implications also on the exploration strategy: what in a propositional setting would be considered a novel situation and worth exploration may in the relational setting be an instance of a well-known context in which exploitation is promising. Our experimental evaluation shows the effectiveness and benefit of relational exploration over several propositional benchmark approaches on noisy 3D simulated robot manipulation problems.

## 1 Introduction

Acting optimally under uncertainty is a central problem of artificial intelligence. In reinforcement learning, an agent's learning task is to find a policy for action selection that maximizes its reward over the long run. Model-based approaches learn models of the underlying Markov decision process from the agent's interactions with the environment, which can then be analyzed to compute optimal plans. One of the key problems in reinforcement learning is the exploration-exploitation tradeoff, which strives to balance two competing types of behavior of an autonomous agent in an unknown environment: the agent can either make use of its current knowledge about the environment to maximize its cumulative reward (i.e., to exploit), or sacrifice short-term rewards to gather information about the environment (i.e., to explore) in the hope of increasing future long-term return, for instance by improving its current world model. This exploration/exploitation tradeoff has received a lot of attention in propositional and continuous domains. Several powerful technique have been developed such as $E^3$ [14], $R^{\max}$ [3] and Bayesian reinforcement learning [19].

Another key problem in reinforcement learning is learning and acting in large relational domains, in which there is a varying number of objects and relations among them. Nowadays, relational approaches become more and more important [9]: information about one object can help the agent to reach conclusions

about other, related objects. Such relational domains are hard – or even impossible – to represent meaningfully using an enumerated state space. For instance, consider a hypothetical household robot which just needs to be taken out of the shipping box, turned on, and which then explores the environment to become able to attend its cleaning chores. Without a compact knowledge representation that supports abstraction and generalization of previous experiences to the current state and potential future states, it seems to be difficult – if not hopeless – for such a "robot-out-of-the-box" to explore one's home in reasonable time. There are too many objects such as doors, plates and water-taps. For instance, after having opened one or two water-taps in bathrooms, the priority for exploring further water-taps in bathrooms, and also in other rooms such as the kitchen, should be reduced. This is impossible to express in a propositional setting where we would simply encounter a new and therefore non-modelled situation.

So far, however, the important problem of exploration in stochastic relational worlds has received surprisingly little attention. This is exactly the problem we address in the current paper. Simply applying existing, propositional exploration techniques is likely to fail: what in a propositional setting would be considered a novel situation and worth exploration may in the relational setting be an instance of a well-known context in which exploitation is promising. This is the key insight of the current paper: *the inherent generalization of learnt knowledge in the relational representation has profound implications also on the exploration strategy.* Consequently, we develop relational exploration strategies in this paper. More specifically, our work is inspired by Kearns and Singh's seminal exploration technique $E^3$ (Explicit Explore or Exploit, discussed in detail below). By developing a similar family of strategies for the relational case and integrating it into the state-of-the-art model-based relational reinforcement learner PRADA [16], we provide a practical solution to the exploration problem in relational worlds. Based on actively generated training trajectories, the exploration strategy and the relational planner together produce in each round a learned world model and in turn a policy that either reduces uncertainty about the environment, i.e., improves the current model, or exploits the current knowledge to maximize utility of the agent. Our extensive experimental evaluation in a 3D simulated complex desktop environment with an articulated manipulator and realistic physics shows that our approaches can solve tasks in complex worlds where non-relational methods face severe efficiency problems.

We proceed as follows. After touching upon related work, we review background work. Then, we develop our relational exploration strategies. Before concluding, we present the results of our extensive experimental evaluation.

## 2 Related Work

Several exploration approaches such as $E^3$ [14], $R^{\max}$ [3] and extensions [13, 10] have been developed for propositional and continuous domains, i.e., assuming the environment to be representable as an enumerated or vector space. In recent years, there has been a growing interest in using rich representations such as relational languages for reinforcement learning (RL). While traditional

RL requires (in principle) explicit state and action enumeration, these symbolic approaches seek to avoid explicit state and action enumeration through a symbolic representation of states and actions. Most work in this context has focused on model-free approaches estimating a value function and has not developed relational exploration strategies. Essentially, a number of relational regression algorithms have been developed for use in these relational RL systems such as relational regression trees [8] or graph kernels and Gaussian processes [7]. Kersting and Driessens [15] have proposed a relational policy gradient approach. These approaches use some form of $\epsilon$-greedy strategy to handle explorations; no special attention has been paid to the exploration-exploitation problem as done in the current paper. Driessens and Džeroski [6] have proposed the use of "reasonable policies" to provide guidance, i.e., to increase the chance to discover sparse rewards in large relational state spaces. This is orthogonal to exploration. Ramon *et al.* [20] presented an incremental relational regression tree algorithm that is capable of dealing with concept drift and showed that it enables a relational Q-learner to transfer knowledge from one task to another. They, however, do not learn a model of the domain and, again, relational exploration strategies were not developed. Croonenborghs *et al.* [5] learn a relational world model online and additionally use lookahead trees to give the agent more informed Q-values by looking some steps into the future when selecting an action. Exploration is based on sampling random actions instead of informed exploration. Walsh [23] provides the first principled investigation into the exploration-exploitation trade-off in relational domains and establishes sample complexity bounds for specific relational MDP learning subproblems. In contrast, we learn full domain models and use them online to adapt our relational exploration strategies.

There is also an increasing number of (approximate) dynamic programming approaches for solving relational MDPs, see e.g. [2, 21]. In contrast to the current paper, however, they assume a given model of the world. Recently, Lang and Toussaint [17] and Joshi *et al.* [12] have shown that successful planning typically involves only a small subset of relevant objects respectively states and how to make use of this fact to speed up symbolic dynamic programming significantly. A principled approach to exploration, however, has not been developed.

## 3 Background on MDPs, Exploration, and Relational Worlds

A Markov decision process (MDP) is a discrete time stochastic control process used to model the interaction of an agent with its environment. At each time-step, the process is in one of a fixed set of discrete states $S$ and the agent can choose an action from a set $A$. The conditional transition probabilities $P(s'|a, s)$ specify the distribution over successor states when executing an action in a given state. The agent receives rewards in states according to a function $R : S \to \mathbb{R}$. The goal is to find a policy $\pi : S \to A$ specifying which action to take in a given state in order to maximize the future rewards. For a discount factor $0 < \gamma < 1$, the value of a policy $\pi$ for a state $s$ is defined as the sum of discounted rewards $V^\pi(s) = E[\sum_t \gamma^t R(s_t) \,|\, s_0 = s, \pi]$. In our context, we do

not know the transition probabilities $P(s'|a,s)$ so that we face the problem of reinforcement learning (RL). We pursue a model-based approach: we estimate $P(s'|a,s)$ from our experiences and compute (approximately) optimal policies based on the estimated model. The quality of these policies depends on the accuracy of this estimation. We need to ensure that we learn enough about the environment in order to be able to plan for high-value states (explore). At the same time, we have to ensure not to spend too much time in low-value parts of the state space (exploit). This is known as the exploitation/exploration-tradeoff.

Kearns and Singh's $E^3$ (Explicit Explore or Exploit) algorithm [14] provides a near-optimal model-based solution to the exploitation/exploration problem. It distinguishes explicitly between exploitation and exploration phases. The central concept are *known states* where all actions have been observed sufficiently often. If $E^3$ enters an *unknown* state, it takes the action it has tried the fewest times there ("**direct exploration**"). If it enters a *known* state, it tries to calculate a high-value policy within an MDP built from all known states (where its model estimates are sufficiently accurate). If it finds such a policy which stays with high probability in the set of known states, this policy is executed ("**exploitation**"). Otherwise, $E^3$ plans in a different MDP in which the unknown states are assumed to have very high value ("optimism in the face of uncertainty"), ensuring that the agent explores unknown states efficiently ("**planned exploration**").

One can prove that with high probability $E^3$ performs near optimally for all but a polynomial number of time-steps. The theoretical guarantees of $E^3$ and similar algorithms such as $R^{\max}$ are strong. In practice, however, the number of exploratory actions becomes huge so that in case of large state spaces, such as in relational worlds, it is unrealistic to meet the theoretical thresholds of state visits. To address this drawback, variants of $E^3$ for factored but propositional MDP representations have been explored [13, 10]. Our evaluations will include variants of factored exploration strategies (PEX and OPT-PEX) where the factorization is based on the grounded relational formulas. However, such factored MDPs still do not generalize over objects. Relational worlds can be represented more compactly using relational MDPs. The state space $S$ of a relational MDP (RMDP) has a relational structure defined by predicates $\mathcal{P}$ and functions $\mathcal{F}$, which yield the set of ground atoms with arguments taken from the set of domain objects $\mathcal{O}$. The action space $A$ is defined by atoms $\mathcal{A}$ with arguments from $\mathcal{O}$. In contrast to ground atoms, abstract atoms contain logical variables as arguments. We will speak of grounding an abstract formula $\psi$ if we apply a substitution $\sigma$ that maps all of the variables appearing in $\psi$ to objects in $\mathcal{O}$.

A compact relational transition model $P(s'|a,s)$ uses formulas to abstract from concrete situations and object identities. The principle ideas of relational exploration we develop in this paper work with any type of relational model. In this paper, however, we employ *noisy indeterministic deictic (NID) rules* [18] to illustrate and empirically evaluate our ideas. A NID rule $r$ is given as

$$a_r(\mathcal{X}):\ \phi_r(\mathcal{X}) \quad \rightarrow \quad \begin{cases} p_{r,1} & : \Omega_{r,1}(\mathcal{X}) \\ & \vdots \\ p_{r,m_r} & : \Omega_{r,m_r}(\mathcal{X}) \\ p_{r,0} & : \Omega_{r,0} \end{cases}, \tag{1}$$

where $\mathcal{X}$ is a set of logic variables in the rule (which represent a (sub-)set of abstract objects). The rule $r$ consists of preconditions, namely that action $a_r$ is applied on $\mathcal{X}$ and that the abstract state context $\phi_r$ is fulfilled, and $m_r+1$ different abstract outcomes with associated probabilities $p_{r,i} > 0$, $\sum_{i=0} p_{r,i} = 1$. Each outcome $\Omega_{r,i}(\mathcal{X})$ describes which atoms "change" when the rule is applied. The context $\phi_r(\mathcal{X})$ and outcomes $\Omega_{r,i}(\mathcal{X})$ are conjunctions of literals constructed from the literals in $\mathcal{P}$ as well as equality statements comparing functions from $\mathcal{F}$ to constant values. The so-called *noise outcome* $\Omega_{r,0}$ subsumes all possible action outcomes which are not explicitly specified by one of the other $\Omega_{r,i}$. The arguments of the action $a(\mathcal{X}_a)$ may be a true subset $\mathcal{X}_a \subset \mathcal{X}$ of the variables $\mathcal{X}$ of the rule. The remaining variables are called deictic references $DR = \mathcal{X} \setminus \mathcal{X}_a$ and denote objects relative to the agent or action being performed.

So, how do we apply NID rules? Let $\sigma$ denote a substitution that maps variables to constant objects, $\sigma : \mathcal{X} \to \mathcal{O}$. Applying $\sigma$ to an abstract rule $r(\mathcal{X})$ yields a *grounded rule* $r(\sigma(\mathcal{X}))$. We say a grounded rule $r$ *covers* a state $s$ and a ground action $a$ if $s \models \phi_r$ and $a = a_r$. Let $\Gamma$ be our set of rules and $\Gamma(s,a) \subset \Gamma$ the set of rules covering $(s,a)$. If there is a unique covering rule $r_{(s,a)} \in \Gamma(s,a)$, we use it to model the effects of action $a$ in state $s$. If no such rule exists (including the case that more one rule covers the state-action pair), we use a noisy default rule $r_\nu$ which predicts all effects as noise. The semantics of NID rules allow one to efficiently plan in relational domains, i.e. to find a "satisficing" action sequence that will lead with high probability to states with large rewards. In this paper, we use the PRADA algorithm [16] for planning in grounded relational domains. PRADA converts NID rules into dynamic Bayesian networks, predicts the effects of action sequences on states and rewards by means of approximate inference and samples action sequences in an informed way. PRADA copes with different types of reward structures, such as partially abstract formulas or maximizing derived functions. We learn NID rules from the experiences $\mathcal{E} = \{(s_t, a_t, s_{t+1})_{t=0}^{T-1}\}$ of an actively exploring agent, using a batch algorithm that trades off the likelihood of these triples with the complexity of the learned rule-set. $\mathcal{E}(r) = \{(s,a,s') \in \mathcal{E} \,|\, r = r_{(s,a)}\}$ are the experiences which are uniquely covered by a learned rule $r$. For more details, we refer the reader to Pasula et al. [18].

## 4 Exploration in Relational Domains

We first discuss the implications of a relational knowledge representation for exploration on a conceptual level. We adopt a density estimation view to pinpoint the differences between propositional and relational exploration (Sec. 4.1). This conceptual discussion opens the door to a large variety of possible exploration strategies – we cannot test all such approaches within this paper. Thus, we focus on specific choices to estimate novelty and hence of the respective exploration strategies (Sec. 4.2), which we found effective as a first proof of concept.

### 4.1 A Density Estimation View on Known States and Actions

The theoretical derivations of the non-relational near-optimal exploration algorithms $E^3$ and $R^{\max}$ show that the concept of known states is crucial. On the

one hand, the confidence in estimates in known states drives exploitation. On the other hand, exploration is guided by seeking for novel (yet unknown) states and actions. For instance, the direct exploration phase in $E^3$ chooses novel actions, which have been tried the fewest; the planned exploration phase seeks to visit novel states, which are labeled as yet unknown.

In the case of the original $E^3$ algorithm (and $R^{\max}$ and similar methods) operating in an enumerated state space, states and actions are considered known based directly on the number of times they have been visited. In relational domains, there are two reasons for why we should go beyond simply counting state-action visits to estimate the novelty of states and actions:

1. The size of the state space is exponential in the number of objects. If we base our notion of known states directly on visitation counts, then the overwhelming majority of all states will be labeled yet-unknown and the exploration time required to meet the criteria for known states of $E^3$ even for a small relevant fraction of the state space becomes exponential in large domains.
2. The key benefit of relational learning is the ability to generalize over yet unobserved instances of the world based on relational abstractions. This implies a fundamentally different perspective on what is novel and what is known and permits qualitatively different exploration strategies compared to the propositional view.

A constructive approach to pinpoint the differences between propositional and relational notions of exploration, novelty and known states is to focus on a density estimation view. This is also inspired by the work on active learning which typically selects points that, according to some density model of previously seen points, are novel (see, e.g., [4] where the density model is an implicit mixture of Gaussians). In the following we first discuss different approaches to model a distribution of known states and actions in a relational setting. These methods estimate which relational states are considered known with some useful confidence measures according to our experiences $\mathcal{E}$ and world model $\mathcal{M}$.

**Propositional:** Let us first consider briefly the propositional setting from a density estimation point of view. We have a finite enumerated state space $S$ and action space $A$. Assume our agent has so far observed the set of state transitions $\mathcal{E} = \{(s_t, a_t, s_{t+1})\}_{t=1}^{T-1}$. This translates directly to a density estimate

$$P(s) \propto c_{\mathcal{E}}(s) \ , \quad \text{with } c_{\mathcal{E}}(s) = \sum\nolimits_{(s_e, a_e, s'_e) \in \mathcal{E}} I(s_e = s) \ , \tag{2}$$

where $c_{\mathcal{E}}(s)$ counts the number of occasions state $s$ has been visited in $\mathcal{E}$ (in the spirit of [22]) and $I(\cdot)$ is the indicator function which is 1 if the argument evaluates to true and 0 otherwise. This density implies that all states with low $P(s)$ are considered novel and should be explored, as in $E^3$. There is no generalization in this notion of known states. Similar arguments can be applied on the level of state-action counts and the joint density $P(s, a)$.

**Predicate-based:** Given a relational structure with the set of logical predicates $\mathcal{P}$, an alternative approach to describe what are known states is based on counting how often a ground or abstract predicate has been observed true or false in the experiences $\mathcal{E}$ (all statements equally apply to functions $\mathcal{F}$, but

we neglect this case here). First, we consider grounded predicates $p \in \mathcal{P}^G$ with arguments taken from the domain objects $\mathcal{O}$. This leads to a density estimate

$$P_p(s) \;\propto\; c_p(s)\; I(s \models p) \;+\; c_{\neg p}(s)\; I(s \models \neg p) \qquad (3)$$
$$\text{with} \quad c_p(s) := \textstyle\sum_{(s_e, a_e, s'_e) \in \mathcal{E}} I(s_e \models p).$$

Each $p$ implies a density $P_p(s)$ which counts how often $p$ has the same truth values in $s$ and in experienced states. We take the product to combine all $P_p(s)$. This implies that a state is considered familiar (with non-zero $P(s)$) if each predicate that is true (false) in this state has been observed true (false) before. We will use this approach for our planned exploration strategy (Sec. 4.2). We can follow the same approach for partially grounded predicates $\mathcal{P}^{PG}$. For $p \in \mathcal{P}^{PG}$ and a state $s$, we examine whether there are groundings of the logical variables in $p$ such that $s$ covers $p$. More formally, we replace $s \models p$ by $\exists \sigma : s \models \sigma(p)$. E.g., we may count how often the blue ball was on top of *some* other object. If this was rarely the case this implies a notion of novelty which guides exploration.

**Context-based:** Assume that we are given a finite set $\Phi$ of contexts, which are formulas of abstract predicates and functions. While many relational knowledge representations have some notion of context or rule precondition, in our case these may correspond to the set of NID rule contexts $\{\phi_r\}$. These are learnt from the experiences $\mathcal{E}$, which have specifically been optimized to be a compact context representation that covers the experiences and allows for the prediction of action effects (cf. Sec. 3). Analogous to the above, given a set of such formulas we may consider the density

$$P_\phi(s) \;\propto\; \textstyle\sum_{\phi \in \Phi} c_{\mathcal{E}}(\phi)\; I(\exists \sigma : s \models \sigma(\phi)) \qquad (4)$$
$$\text{with} \quad c_{\mathcal{E}}(\phi) \;=\; \textstyle\sum_{(s_e, a_e, s'_e) \in \mathcal{E}}\; I(\exists \sigma : s_e \models \sigma(\phi)).$$

$c_{\mathcal{E}}(\phi)$ counts in how many experiences $\mathcal{E}$ the context $\phi$ was covered with arbitrary groundings. Intuitively, the context of the NID rules may be understood as describing situation classes based on whether the same predictive rules can be applied. Taking this approach, states are considered novel if they are not covered by any existing context ($P_\phi(s) = 0$) or covered by a context that has rarely occurred in $\mathcal{E}$ ($P_\phi(s)$ is low). That is, the description of novelty which drives exploration is lifted to the level of abstraction of these relational contexts. Similarly, we formulate a density estimation over states and actions based on the set of NID rules, where each rule defines a state-action context,

$$P_r(s, a) \;\propto\; \textstyle\sum_{r \in \Gamma} c_{\mathcal{E}}(r)\; I(r = r_{s,a}), \quad \text{with} \quad c_{\mathcal{E}}(r) \;:=\; |\mathcal{E}(r)|, \qquad (5)$$

which is based on counting how many experiences are covered by the unique covering rule $r_{s,a}$ for $a$ in $s$. Recall that $\mathcal{E}(r)$ are the experiences which are covered by $r$. Thus, the more experiences the corresponding unique covering rule $r_{(s,a)}$ covers the larger is $P_r(s, a)$ and it can be seen as a measure of confidence in $r$. We will use $P_r(s, a)$ to guide direct exploration below.

**Distance-based:** As mentioned in the related work discussion, different methods to estimate the similarity of relational states exist. These can be used

for relational density estimation (in the sense of 1-class SVMs) which, when applied in our context, would readily imply alternative notions of novelty and thereby exploration strategies. To give an example, [7] and [11] present relational reinforcement learning approaches which use relational graph kernels to estimate the similarity of relational states. Applying such a method to model $P(s)$ from $\mathcal{E}$ would imply that states are considered novel (with low $P(s)$) if they have a low kernel value (high "distance") to previous explored states. For a given state $s$, we directly define a measure of distance to all observed data, $d(s) = \min_{(s_e,a_e,s'_e)\in\mathcal{E}} d(s, s_e)$, and set

$$P_d(s) \propto \frac{1}{d(s)+1} \,. \tag{6}$$

Here, $d(s, s')$ can be any distance measure, for instance based on relational graph kernels. We will use a similar but simplified approach as part of a specific direct exploration strategy on the level of $P_d(s, a)$, as described in detail in Sec. 4.2. In our experiments, we use a simple distance based on least general unifiers.

All three relational density estimation techniques emphasize different aspects and we combine them in our algorithms.

### 4.2 Relational Exploration Algorithms

The density estimation approaches discussed above open a large variety of possibilities for concrete exploration strategies. In the following, we derive model-based relational reinforcement learning algorithms which explicitly distinguish between exploration and exploitation phases in the sense of $E^3$. Our methods are based on simple, but empirically effective relational density estimators. We are certain that more elaborate and efficient exploration strategies can be derived from the above principles in the future. Our algorithms perform the following general steps: *(i)* In each step they first adapt the relational model $\mathcal{M}$ with the set of experiences $\mathcal{E}$. *(ii)* Based on $\mathcal{M}$, $s$ and $\mathcal{E}$, they select an action $a$ – we focus on this below. *(iii)* The action $a$ is executed, the resulting state $s'$ observed and added to the experiences $\mathcal{E}$, and the process repeated.

Our first algorithm transfers the general $E^3$ approach (distinguishing between exploration and exploitation based on whether the current state is fully known) to the relational domain to compute actions. The second tries to exploit more optimistically, even when the state is not known or only partially known. Both algorithms are based on a set of subroutines which instantiate the ideas mentioned above and which we describe first:

plan(world model $\mathcal{M}$, reward function $\tau$, state $s_0$): Returns the first action of a plan of actions that maximizes $\tau$. Typically, $\tau$ is expressed in terms of logical formulas, describing goal situations to which a reward of 1 is associated. If the planner estimates a maximum expected reward close to zero (i.e., no good plan is found), it returns a 0 instead of the first action. In this paper, we employ NID rules as $\mathcal{M}$ and use the PRADA algorithm for planning.

isKnown(world model $\mathcal{M}$, state $s$): $s$ is known if the estimated probabilities $P(s, a)$ of all actions $a$ are larger than some threshold. We employ the rule-context based density estimate $P_r(s, a)$ (Eq. 5).

**Algorithm 1** REX – Action Computation

**Input:** World model $\mathcal{M}$, Reward function $\tau$, State $s_0$, Experiences $\mathcal{E}$
**Output:** Action $a$

```
if isKnown(M, s_0) then
    a = plan(M, τ, s_0)                          ▷ Try to exploit
    if a ≠ 0 then
        return a                                 ▷ Exploit succeeded
    end if
    τ_explore = getPlannedExplorationReward(M, E)
    a = plan(M, τ_explore, s_0)                  ▷ Try planned exploration
    if a ≠ 0 then
        return a                                 ▷ Planned exploration succeeded
    end if
end if
w = getDirectExplorationWeights(M, E, s_0)       ▷ Sampling weights for actions
a = sample(w)                                    ▷ Direct exploration (without planning)
return a
```

isPartiallyKnown(world model $\mathcal{M}$, reward function $\tau$, state $s$): In contrast to before, we only consider relevant actions. These refer to objects which appear explicitly in the reward description or are related to them in $s$ by some binary predicate.

getPlannedExplorationReward(world model $\mathcal{M}$, experiences $\mathcal{E}$): Returns a reward function for planned exploration, expressed in terms of logical formulas as for plan, describing goal situations worth for exploration. We follow the predicate-based density estimation view (Eq. (3)) and set the reward function to $\frac{1}{P_p(s)}$.

getDirectExplorationWeights(world model $\mathcal{M}$, experiences $\mathcal{E}$, state $s$): Returns weights according to which an action is sampled for direct exploration. Here, the two algorithms use different heuristics:
*(i)* REX sets the weights for actions $a$ with minimum value $|\mathcal{E}(r_{s,a})|$ to 1 and for all others to 0, thereby employing $P_r(s,a)$. This combines $E^3$ (choosing the action with the fewest "visits") with relational generalization (defining "visits" by means of confidence in abstract rules).
*(ii)* OPT-REX combines three scores to decide on direct exploration weights. The first score is inverse proportional to $P_r(s,a)$. The second is inverse proportional to the distance-based density estimation $P_d(s,a)$ (Eq. 6). The third score is an additional heuristic to increase the probability of relevant actions (with the same idea as in partially known states, that we care more about the supposedly relevant parts of the action space).

These subroutines are the basic building blocks for the two relational exploration algorithms REX and OPT-REX that we discuss now in turn.

REX **(Relational Explicit Explore or Exploit)** (Algorithm 1) REX lifts the $E^3$ planner to relational exploration and uses the same phase order as $E^3$. If the current state is known, it tries to exploit $\mathcal{M}$. In contrast to $E^3$, REX also plans through unknown states as it is unclear how to efficiently build and

**Algorithm 2** OPT-REX – Action Computation

**Input:** World model $\mathcal{M}$, Reward function $\tau$, State $s_0$, Experiences $\mathcal{E}$
**Output:** Action $a$

$a = \mathsf{plan}(\mathcal{M}, \tau, s_0)$ ▷ Try to exploit
**if** $a \neq 0$ **then**
  return $a$ ▷ Exploit succeeded
**end if**
**if** $\mathsf{isPartiallyKnown}(\mathcal{M}, \tau, s_0)$ **then**
  $\tau_{explore} = \mathsf{getPlannedExplorationReward}(\mathcal{M}, \mathcal{E})$
  $a = \mathsf{plan}(\mathcal{M}, \tau_{explore}, s_0)$ ▷ Try planned exploration
  **if** $a \neq 0$ **then**
    return $a$ ▷ Planned exploration succeeded
  **end if**
**end if**
$\mathbf{w} = \mathsf{getDirectExplorationWeights}(\mathcal{M}, \mathcal{E}, s_0)$ ▷ Sampling weights for actions
$a = \mathrm{sample}(\mathbf{w})$ ▷ Direct exploration (without planning)
return $a$

exclusively use an MDP of known relational states. However, in every state only sufficiently known actions are taken into account. In our experiments, for instance, our planner PRADA achieves this by only considering actions with unique covering rules in a given state. If exploitation fails, an exploration goal is set up for planned exploration. In case planned exploration fails as well or the current state is unknown, the action with the lowest confidence is carried out (similarly, as $E^3$ chooses the action which was performed the least often in the current state).

OPT-REX **(Optimistic** REX) (Algorithm 2) OPT-REX modifies REX according to the intuition that there is no need to understand the world dynamics to full extent: rather it makes sense to focus on the relevant parts of the state and action space. OPT-REX exploits the current knowledge optimistically to plan for the goal. For a given state $s_0$, it tries immediately to come up with an exploitation plan. If this fails, it checks whether $s_0$ is partially known, i.e., whether the world model $\mathcal{M}$ can predict the actions which are relevant for the reward $\tau$. If the state $s_0$ is partially known, planned exploration is tried. If this fails or $s_0$ is partially unknown, direct exploration is undertaken, with action sampling weights as described above.

## 5 Evaluation

Our intention here is to compare propositional and relational techniques for exploring relational worlds. More precisely, we investigate the following questions:

- $Q1$: Can relational knowledge improve exploration performance?
- $Q2$: How do propositional and relational explorers scale with the number of domain objects?
- $Q3$: Can relational explorers transfer knowledge to new situations, objects and tasks?

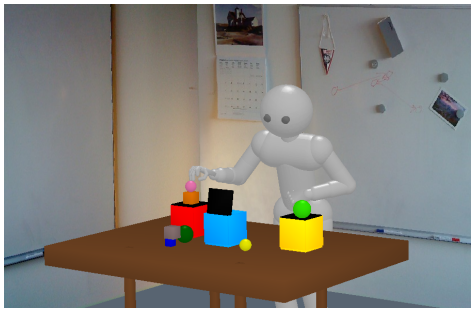

**Fig. 1.** In our experiments, a robot has to explore a 3D simulated desktop environment with cubes, balls and boxes of different sizes and colors to master various tasks.

To do so, we compare five different methods inspired by $E^3$ based on propositional or abstract symbolic world models. In particular, we learn (propositional or abstract) NID rules after each new observation from scratch using the algorithm of Pasula et al. [18] and employ PRADA [16] for exploitation or planned exploration. All methods deem an action to be known in a state if the confidence in its covering rule is above a threshold $\varsigma$. Instead of deriving $\varsigma$ from the $E^3$ equations which is not straightforward and will lead to overly large thresholds (see [10]), we set it heuristically such that the confidence is high while still being able to explore the environments of our experiments within a reasonable number of actions ($< 100$).

PEX (propositional $E^3$) is a variant of $E^3$ based on propositional NID rules (with ground predicates and functions). While it abstracts over states using the factorization of rules, it cannot transfer knowledge to unseen objects. OPT-PEX (optimistic PEX) is similar, but always tries to exploit first, independently of whether the current state is known or not. REX and OPT-REX (cf. Sec. 4.2) use abstract relational NID rules for exploration and exploitation. In addition, we investigate a relational baseline method RAND-REX (Relational exploit or random) which tries to exploit first (being as optimistic as OPT-REX) and if this is impossible produces a random action.

Our test domain is a simulated complex desktop environment where a robot manipulates cubes, balls and boxes scattered on a table (Fig. 1). We use a 3D rigid-body dynamics simulator (ODE) that enables a realistic behavior of the objects. For instance, piles of objects may topple over or objects may even fall off the table (in which case they become out of reach for the robot). Depending on their type, objects show different characteristics. For example, it is almost impossible to successfully put an object on top of a ball, and building piles with small objects is more difficult. The robot can grab objects, try to put them on top of other objects, in a box or on the table. Boxes have a lid; special actions may open or close the lid; taking an object out of a box or putting it into it is possible only when the box is opened. The actions of the robot are affected by noise so that resulting object piles are not straight-aligned. We assume full observability of triples $(s, a, s')$ that specify how the world changed when an action was executed in a certain state. We represent the data with predicates $cube(X)$, $ball(X)$, $box(X)$, $table(X)$, $on(X, Y)$, $contains(X, Y)$, $out(X)$,

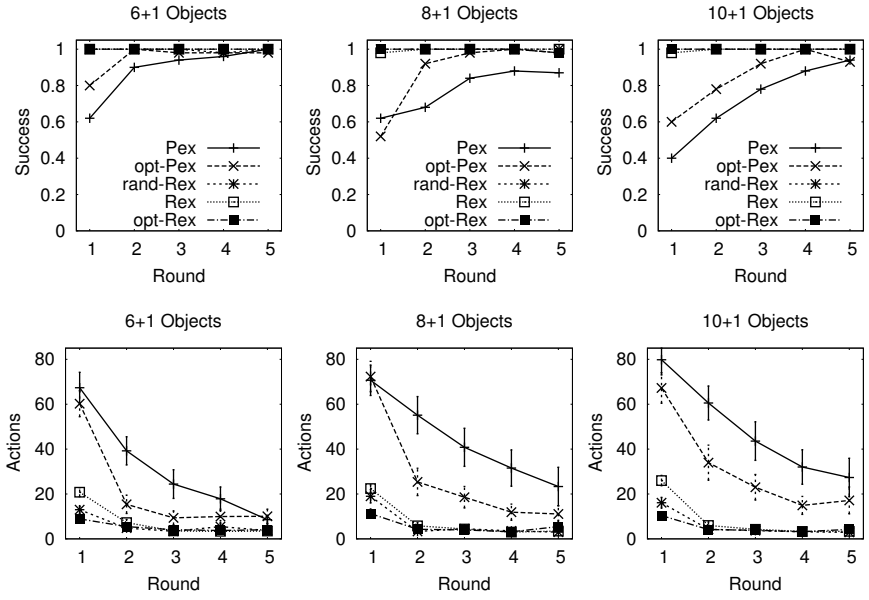


**Fig. 2.** *Experiment 1: Unchanging Worlds of Cubes and Balls.* A run consists of 5 subsequent rounds with the same start situations and goal objects. The robot starts with no knowledge in the first round. The success rate and the mean estimators of the action numbers with standard deviations over 50 runs are shown (5 start situations, 10 seeds).

$inhand(X)$, $upright(X)$, $closed(X)$, $clear(X) \equiv \forall Y.\neg on(Y,X)$, $inhandNil() \equiv \neg\exists X.inhand(X)$ and functions $size(X)$, $color(X)$ for state descriptions and $grab(X)$, $puton(X)$, $openBox(X)$, $closeBox(X)$ and $doNothing()$ for actions. If there are $o$ objects and $f$ different object sizes and colors in a world, the state space is huge with $f^{2o}2^{2o^2+7o}$ different states (not excluding states one would classify as "impossible" given some intuition about real world physics). This points at the potential of using abstract relational knowledge for exploration.

We perform four increasingly complex series of experiments[1] where we pursue the same or similar tasks over multiple rounds. In all experiments *the robot starts from zero knowledge ($\mathcal{E} = \emptyset$) in the first round* and carries over experiences to the next rounds. In each round, we execute a maximum of 100 actions. If the task is still not solved by then, the round fails. We report the success rates and the action numbers to which failed trials contribute with the maximum number.

**Unchanging Worlds of Cubes and Balls:** The goal in each round is to pile two specific objects, $on(obj1, obj2)$. To collect statistics we investigate worlds of varying object numbers and for each object number, we create five worlds with different objects. For each such world, we perform 10 independent runs with different random seeds. Each run consists of 5 rounds with the same goal instance

[1] The website `http://www.user.tu-berlin.de/lang/explore/` provides videos of exemplary rounds as well as pointers to the code of our simulator, the learning algorithm of NID rules and PRADA.

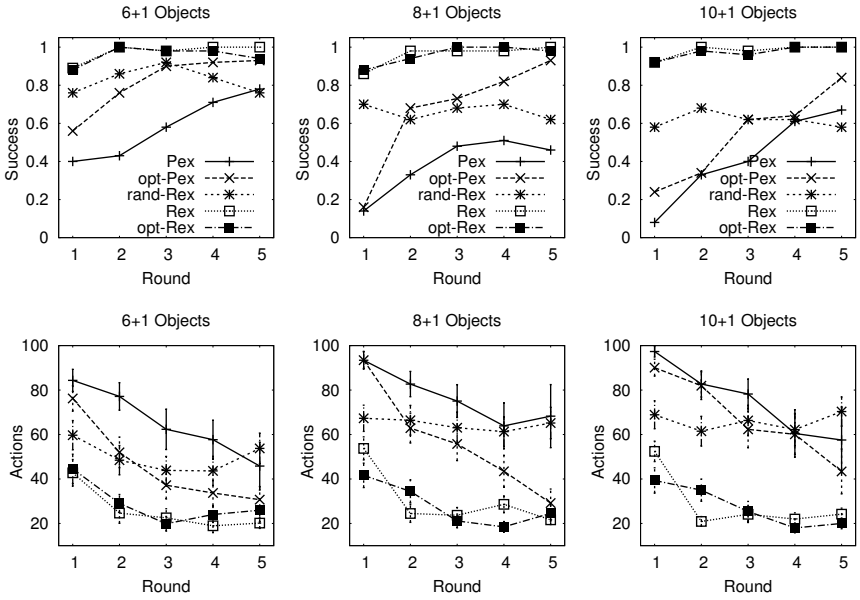


**Fig. 3.** *Experiment 2: Unchanging Worlds of Boxes.* A run consists of 5 subsequent rounds with the same start situations and goal objects. The robot starts with no knowledge in the first round. The success rate and the mean estimators of the action numbers with standard deviations over 50 runs are shown (5 start situations, 10 seeds).

and the same start situation. The results presented in Fig. 2 show that already in the first round the relational explorers solve the task with significantly higher success rates and require up to 8 times fewer actions than the propositional explorers. OPT-REX is the fastest approach which we attribute to its optimistic exploitation bias. In subsequent rounds, the relational methods use previous experiences much better, solving those in almost minimal time. In contrast, the action numbers of the propositional explorers fall only slowly.

**Unchanging Worlds with Boxes:** We keep the task and the experimental setup as before, but in addition the worlds contain boxes, resulting in more complex action dynamics. In particular, some goal objects are put in boxes in the beginning, necessitating more intense exploration to learn how to deal with boxes. Fig. 3 shows that again the relational explorers have superior success rates, require significantly fewer actions and reuse their knowledge effectively in subsequent rounds. While the performance of the propositional planners deteriorates with increasing numbers of objects, OPT-REX and REX scale well. In worlds with many objects, the cautious exploration of REX has the effect that it requires about one third more actions than OPT-REX in the first round, but performs better in subsequent rounds due to the previous thorough exploration.

After the first two experiments we conclude that the usage of relational knowledge improves exploration (question $Q1$) and relational explorers scale better with the number of objects than propositional explorers (question $Q2$).

**Generalization to New Worlds:** In this series of experiments, the objects, their total numbers and the specific goal instances are different in each round

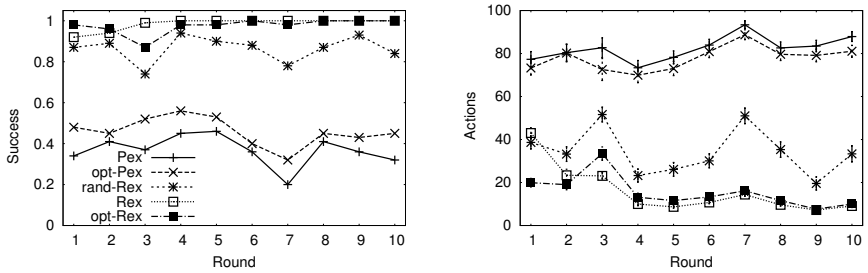


**Fig. 4.** *Experiment 3: Generalization to New Worlds.* A run consists of a problem sequence of 10 subsequent rounds with different objects, numbers of objects (6 - 10 cubes/balls/boxes + table) and start situations in each round. The robot starts with no knowledge in the first round. The success rate and the mean estimators of the action numbers with standard deviations over 100 runs are shown (10 sequences, 10 seeds).

(worlds of 7, 9 and 11 objects). We create 10 problem sequences (each with 10 rounds) and perform 10 trials for each sequence with different random seeds. As Fig.4 shows the performance of the relational explorers is good from the beginning and becomes stable at a near-optimal level after 3 rounds. This answers the first part of question $Q3$: relational explorers can transfer their knowledge to new situations and objects. In contrast, the propositional explorers cannot transfer the knowledge to different worlds and thus neither their success rates nor their action numbers improve in subsequent rounds. Similarly as before, OPT-REX requires less than half of the actions of REX in the first round due to its optimistic exploitation strategy; in subsequent rounds, REX is on par as it has sufficiently explored the system dynamics before.

**Generalization to New Tasks:** In our final series of experiments, we perform in succession three tasks of increasing difficulty: piling two specific objects in simple worlds with cubes and balls (as in Exp. 1), in worlds extended by boxes (as in Exp. 2 and 3) and building a tower on top of a box where the required objects are partially contained in boxes in the beginning. Each task is performed for three rounds in different worlds with different goal objects. The results presented in Fig. 5 confirm the previous results: the relational explorers are able to generalize over different worlds for a fixed task, while the propositional explorers fail. Beyond that, again in contrast to the propositional explorers, the relational explorers are able to transfer the learned knowledge from simple to difficult tasks in the sense of curriculum learning [1], answering the second part of question $Q3$. To see that, one has to compare the results of round 4 (where the second task of piling two objects in worlds of boxes is given the first time) with the results of round 1 in Experiments 2 and 3. In the latter, no experience from previous tasks is available and REX requires $43.0 - 53.8 \pm 2.5$ actions. In contrast, here it can reuse the knowledge of the simple task (rounds 1-3) and needs about $29.9 \pm 2.3$ actions. It is instructive to compare this with OPT-REX which performs about the same or even slightly better in the first rounds of Exp. 2 and 3: here, it can fall victim to its optimistic bias which is not appropriate given the changed world dynamics due to the boxes. As a final remark, the third task (rounds 7-9) was deliberately chosen to be very difficult to test the limits of the different

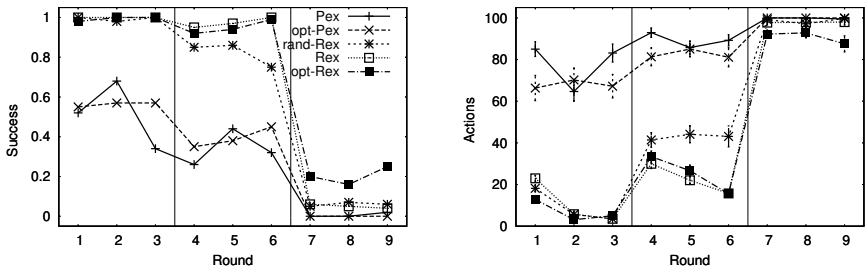


**Fig. 5.** *Experiment 4: Generalization to New Tasks.* A run consists of a problem sequence of 9 subsequent rounds with different objects, numbers of objects (6 - 10 cubes/balls/boxes + table) and start situations in each round. The tasks are changed between round 3 and 4 and round 6 and 7 to more difficult tasks. The robot starts with no knowledge in the first round. The success rate and the mean estimators of the action numbers with standard deviations over 100 runs are shown (10 sequences, 10 seeds).

approaches. While the propositional planners almost always fail to solve it, the relational planners achieve 5 to 25 times higher success rates.

## 6 Conclusions

Efficient exploration in relational worlds is an interesting problem that is fundamental to many real-life decision-theoretic planning problems, but has only received little attention so far. We have approached this problem by proposing relational exploration strategies that borrow ideas from efficient techniques for propositional and continuous MDPs. A few principled and practical issues of relational exploration have been discussed, and insights are drawn by relating it to its propositional counterpart. The experimental results show a significant improvement over established results for solving difficult, highly stochastic planning tasks in a 3D simulated complex desktop environment, even in a curriculum learning setting where different problems have to be solved one after the other.

There are several interesting avenues for future work. One is to investigate incremental learning of rule-sets. Another is to explore the connection between relational exploration and transfer learning. Finally, one should start to explore statistical relational reasoning and learning techniques for the relational density estimation problem implicit in exploring relational worlds.

**Acknowledgements:** TL and MT were supported by the German Research Foundation (DFG), Emmy Noether fellowship TO 409/1-3. KK was supported by the European Commission under contract number FP7-248258-First-MM and the Fraunhofer ATTRACT Fellowship STREAM.

## References

1. Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 41–48, 2009.

2. C. Boutilier, R. Reiter, and B. Price. Symbolic dynamic programming for first-order MDPs. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 690–700, 2001.
3. R. I. Brafman and M. Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, 3:213–231, 2002.
4. D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4(1):129–145, 1996.
5. T. Croonenborghs, J. Ramon, H. Blockeel, and M. Bruynooghe. Online learning and exploiting relational models in reinforcement learning. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 726–731, 2007.
6. K. Driessens and S. Džeroski. Integrating guidance into relational reinforcement learning. *Machine Learning*, 57(3):271–304, 2004.
7. K. Driessens, J. Ramon, and T. Gärtner. Graph kernels and Gaussian processes for relational reinforcement learning. *Machine Learning*, 2006.
8. S. Džeroski, L. de Raedt, and K. Driessens. Relational reinforcement learning. *Machine Learning*, 43:7–52, 2001.
9. L. Getoor and B. Taskar, editors. *A Introduction to Statistical Relational Learning*. MIT Press, 2007.
10. C. Guestrin, R. Patrascu, and D. Schuurmans. Algorithm-directed exploration for model-based reinforcement learning in factored MDPs. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 235–242, 2002.
11. F. Halbritter and P. Geibel. Learning models of relational MDPs using graph kernels. In *Proc. of the Mexican Conf. on A.I. (MICAI)*, pages 409–419, 2007.
12. S. Joshi, K. Kersting, and R. Khardon. Self-taught decision theoretic planning with first order decision diagrams. In *Proceedings of ICAPS-10*, 2010.
13. M. Kearns and D. Koller. Efficient reinforcement learning in factored MDPs. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 740–747, 1999.
14. M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.
15. K. Kersting and K. Driessens. Non–parametric policy gradients: A unified treatment of propositional and relational domains. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, July 5–9 2008.
16. T. Lang and M. Toussaint. Approximate inference for planning in stochastic relational worlds. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2009.
17. T. Lang and M. Toussaint. Relevance grounding for planning in relational domains. In *Proc. of the European Conf. on Machine Learning (ECML)*, September 2009.
18. H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling. Learning symbolic models of stochastic domains. *Artificial Intelligence Research*, 29:309–352, 2007.
19. P. Poupart, N. Vlassis, J. Hoey, and K. Regan. An analytic solution to discrete bayesian reinforcement learning. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, pages 697–704, 2006.
20. J. Ramon, K. Driessens, and T. Croonenborghs. Transfer learning in reinforcement learning problems through partial policy recycling. In *Proc. of the European Conf. on Machine Learning (ECML)*, pages 699–707, 2007.
21. S. Sanner and C. Boutilier. Practical solution techniques for first order MDPs. *Artificial Intelligence Journal*, 173:748–788, 2009.
22. S. Thrun. The role of exploration in learning control. In D. White and D. Sofge, editors, *Handbook for Intelligent Control: Neural, Fuzzy and Adaptive Approaches*. Van Nostrand Reinhold, Florence, Kentucky 41022, 1992.
23. T. J. Walsh. *Efficient learning of relational models for sequential decision making*. PhD thesis, Rutgers, The State University of New Jersey, 2010.