Action Learning: Recent Techniques and Properties

Michal Čertický

Department of Applied Informatics
Comenius University, Bratislava

August 29, 2011

Introduction - Action Learning

- **Action model** = some kind of representation of all the actions executable in our domain.
- Describes: *Effects* and *Preconditions*.
- We use AM for *planning* / goal-based behaviour.
- **Action learning**
  - **automatic creation and modification of action models**
  - discovering the causal rules of a domain
  - inductive learning, where observations of a form *(executed action, world state)* serve as examples

Motivation for Action Learning

- *Complexity:* Action models are usually hand-crafted by domain experts. If domains are complex enough, this task is overly tedious and time-consuming.
- *Sustainability:* When confronted with new information, we often need to revise our action models. We want to automate this process to save some time and avoid making mistakes.
- *Universality:* Automatic acquisition of action models is necessary for environmental universality (adaptation to different environments) of artificial agents.

Current Methods

- (Zettlemoyer-Pasula-Kaelbling, 2003) 3-layer **Greedy search** over the space of possible action models. Using many different operators, they modify a model and then evaluate it, based on how well it covers the training set.
- (Mourao-Petrick-Steedman, 2010) Learning reduced to a **binary classification problem**. One perceptron per fluent - input vector represents the observation, output determines if the fluent value changes. Perceptron algorithm for training.
- (Balduccini, 2007) Observations, action models, and learning semantics are encoded as **ASP logic program**. Its answer sets represent new action models. Declarative solution.
- (Amir-Chang, 2008) and
- (Yang et al., 2007) Build the set of propositional constraints after observations. Use external **SAT / MAX-SAT solvers** to interpret this knowledge as action models.

Properties (1-4)

7 important properties (or challenges) of action learning methods, studied in related literature:

- **Partially observable** domains (incomplete knowledge).
- **Probabilistic** action **models**.

$$\text{Deterministic effect: } \{\neg on(B, P_1), on(B, P_2)\}$$

$$\text{Probabilistic effect: } \begin{cases} 0.8 : & \neg on(B, P_1), on(B, P_2) \\ 0.1 : & \neg on(B, P_1), on(B, table) \\ 0.1 : & nochange \end{cases}$$

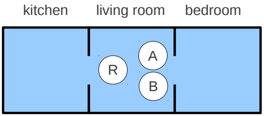
- Action **failures** and sensoric **noise**.
- Learning both **effects** and **preconditions**.
  (Some methods need to have preconditions in advance and learn only effects.)

Properties (5-7)

- **Conditional effects**.
  Consider an action $drink(P, B)$ with two effects:
  1. Person P ceases to be thirsty.
  2. If beverage B was poisoned, person P will get sick.

  :effect (not (thirsty P))
  :effect (when (poisonous B) (sick P))

- **Online** algorithms.
  Usually lower comp. complexity; Better suitable for autonomous agents.
- Probabilistic evaluation of **posible world states**.



Comparison of Methods

Paper	Method name	Partially observable domains	Probabilistic action models	Probabilistic world states	Dealing with action failures	Both preconditions and effects	Conditional effects	Online
[Amir-Chang, 2008]	SLAF	yes	no	no	only when failure is explicitly known	no	no	yes
[Yang-Wu-Jiang, 2007]	ARMS	yes	no	no	no	yes	no	no
[Balduccini, 2007]	A-Prolog with ASP semantics + Learning module	yes	no	no	no	yes	yes	no
[Mourao-Petrick-Steedman, 2010]	Perceptron Algorithm	yes	no	no	yes	no	no	yes
[Pasula-Zettlemoyer-Kaelbling, 2007]	Greedy Search	no	yes	no	yes	yes	yes	no

Representation Structures (1/3) - Transition Relation

Typical structure used for example in [Amir-Chang-2008].

Definition (Transition Relation)

Let $\mathcal{S}$ be a set of all the possible world states, and $\mathcal{A}$ a set of all the possible actions of our domain. Transition Relation $\mathcal{TR}$ is then:

$$\mathcal{TR} \subseteq \mathcal{S} \times \mathcal{A} \times \mathcal{S}$$

Intuitive meaning of every $(s, a, s') \in \mathcal{TR}$ is that *"execution of action* **a** *in a world state* **s** *causes a world state* **s'** *to hold in the next time step"*.

- Robust in terms of space requirements. Space complexity of $\mathcal{TR}$ is $O(|\mathcal{A}| \cdot |\mathcal{S}|^2)$.
- **Note:** Cardinality of $\mathcal{S}$ can be expressed as $|\mathcal{S}| = 2^{|\mathcal{F}|}$ where $\mathcal{F}$ is the set of all the fluent literals. $O(|\mathcal{A}| \cdot |\mathcal{S}|^2)$ is then equal to $O(|\mathcal{A}| \cdot (2^{|\mathcal{F}|})^2)$.

Representation Structures (2/3) - Effect Relation

Our first improvement over $\mathcal{TR}$ in terms of space complexity.

Definition (Effect Relation)

Let $\mathcal{S}$ be a set of world states, $\mathcal{F}$ a set of fluent literals, and $\mathcal{A}$ the set of actions of our domain. Effect Relation $\mathcal{ER}$ is then:

$$\mathcal{ER} \subseteq \mathcal{S} \times \mathcal{A} \times \mathcal{F}$$

The meaning of triple $(s, a, f) \in \mathcal{ER}$ is that *"execution of action* **a** *in a world state* **s** *causes a fluent* **f** *be true in the next time step"*.

- **Space complexity** of $\mathcal{ER}$ is $O(2^{|\mathcal{F}|} \cdot |\mathcal{A}| \cdot |\mathcal{F}|)$ which is **lower** than in previous case.
- Anything that can be expressed in $\mathcal{TR}$ can also be expressed in $\mathcal{ER}$ and vice versa. This means, that **expressive power** of those two structures is **equal**.
- In case of $\mathcal{ER}$, some information is expressed implicitly by the **absence** of elements in the relation (this saves space).

Representation Structures (3/3) - Effect Formula

Our new structure used by $3SG$ algorithm. Not a relation this time.

Definition (Effect Formula)

Effect Formula $\mathcal{EF}$ is any finite set of **propositional atoms** over a vocabulary $\mathcal{L}_{\mathcal{EF}} = \{a^f \mid a \in \mathcal{A} \wedge f \in \mathcal{F}\} \cup \{a^f_c \mid a \in \mathcal{A} \wedge f, c \in \mathcal{F}\}$.

The meaning of atoms from $\mathcal{EF}$ follows:

a^f: "action **a** causes **f**"

a^f_c: "**c** must hold in order for **a** to cause **f**" (c is a condition of a^f)

- Again, the **space complexity** of $\mathcal{EF}$ is **lower** than in previous cases, only $O(|\mathcal{A}| \cdot (|\mathcal{F}| + |\mathcal{F}|^2))$, while the **expressive power** remains the **same**.
- Space is saved by assigning implicit meaning to the combination of absence and presence of some of atoms in $\mathcal{EF}$. For example: $(s, a, f) \in \mathcal{ER}$ is expressed in $\mathcal{EF}$ by the presence of a^f together with the absence of all the a^f_c, such that $c \in s$.

3SG Algorithm

- **3SG algoritmhm** (Simultaneous Specification, Simplification, and Generalization), is merely the first candidate method. More approaches will probably come in future.
- Comparison based on previously mentioned properties:

Method name	Partially observable domains	Probabilistic action models	Probabilistic world states	Dealing with action failures	Both precondition s and effects	Conditional effects	Online
3SG	yes	yes	?	yes	?	yes	yes

- Probabilistic action model here is a double $\langle \mathcal{EF}, \mathcal{P} \rangle$, where $\mathcal{EF}$ is an Effect Formula expressing the **conditional effects** of actions, and $\mathcal{P}$ is a probabilistic function over the elements of $\mathcal{EF}$.

3*SG* Algorithm

- 3*SG* runs **once after every** executed action.
- Its input is a triple (o, a, o'), where o and o' are incomplete **observations** from two most recent time steps, and a is the **action** executed between them.
- Algorithm always:
  - **specifies** our knowledge by **adding** some elements to $\mathcal{EF}$,
  - modifies the value of prob. function $\mathcal{P}$ for each of previously added elements (if recent observations *confirms* or *denies* them),
  - and **simplifies** our model by **removing** very improbable elements from $\mathcal{EF}$.
- Is **polynomial** in the size of observation.
- Is **online**. This means, that we always have (increasingly accurate) action model at our disposal.

Further Research

- First, we need to formalize the **translation** from $\langle \mathcal{EF}, \mathcal{P} \rangle$ to some of the **planning languages** (such as PDDL, STRIPS, $\mathcal{A}$ or $\mathcal{K}$, etc.).
- Then we will be able to decide all the properties of $3SG$.
- Finally, we need to **test** it in various kinds of **domains**, using benchmarks and/or games.