Regression

- The formula \( \text{regr}_o(\phi) \) represents the set of states from which a state in \( \phi \) is reached by operator \( o \).
- Used in backward search in a transition system: from the goal states toward the initial states.
- Regression is powerful because it allows handling huge sets of states (progression: only one state at a time.)
- Handling formulae is more complicated than handling states: many questions about regression are NP-hard.

Regression for simple operators

1. Goals are conjunctions of literals.
2. Operator preconditions are conjunctions of literals.
3. Operators have no conditional effects.

Hence every operator is of the form

\[
(l_1 \land \ldots \land l_n, \ l'_1 \land \ldots \land l'_m)
\]

where \( l_i \) and \( l'_j \) are literals. Call this kinds of operators simple.

Regression for simple operators

1. The goal is \( l_1 \land \ldots \land l_n \).
2. Choose an operator that makes some of \( l_1, \ldots, l_n \) true and makes none of them false.
3. Form a new goal by removing the fulfilled goal literals and adding the preconditions of the operator.
Regression for simple operators

The regression \( \text{regr}_o(\phi) \) of \( \phi = l_1'' \land \cdots \land l_m'' \) with respect to

\[ o = (l_1 \land \cdots \land l_n, \ l'_1 \land \cdots \land l'_m) \]

such that \( \{l_1'', \ldots, l_m''\} \cap \{l_1', \ldots, l_m'\} = \emptyset \) is the following conjunction of literals.

\[ \bigwedge (\{l_1'', \ldots, l_m''\} \setminus \{l_1', \ldots, l_m'\}) \cup \{l_1, \ldots, l_n\} \]

Assume that \( s' \models \phi \). We show that \( s \models \text{regr}_o(\phi) \).

Because \( s' = \text{app}_o(s) \), the application of \( o \) in \( s \) is well-defined, and hence all preconditions of \( o \) are true in \( s \).

Let \( l \) be one of the remaining literals in \( \text{regr}_o(\phi) \), that is, one of the conjuncts of \( \phi \) that are not effects of \( o \).

Because \( o \) does not change the value of \( l \) and \( s' \models \phi \), also \( s \models l \).
Equivalences on effects

\[ c \triangleright (e_1 \land \cdots \land e_n) \equiv (c \triangleright e_1) \land \cdots \land (c \triangleright e_n) \]  
\[ c \triangleright (c' \triangleright e) \equiv (c \land c') \triangleright e \]  
\[ (e_1 \triangleright e) \land (e_2 \triangleright e) \equiv (e_1 \lor e_2) \triangleright e \]  
\[ e \land (c \triangleright e) \equiv e \]  
\[ e \equiv \top \triangleright e \]  
\[ e \equiv \top \land e \]  
\[ e \land e' \equiv e' \land e \]  
\[ (e_1 \land e_2) \land e_3 \equiv e_1 \land (e_2 \land e_3) \]

Normal form for operators and effects

DEFINITION: An operator \((c, e)\) is in **normal form** if for all occurrences of \(c' \triangleright e'\) in \(e\) the effect \(e'\) is either \(p\) or \(\neg p\) for some \(p \in P\), and \(e\) contains at most one occurrence of any atomic effect \(l\).

THEOREM: For every operator there is an equivalent one in normal form.

Proof is constructive: we can transform any operator into normal form by using the equivalences from the previous slide.

Normal form for effects: example

\[(A \triangleright (B \land (C \triangleright (\neg D \land E)))) \land (\neg B \triangleright E)\]

transformed to normal form is

\[(A \triangleright B) \land ((A \land C) \triangleright \neg D) \land ((\neg B \lor (A \land C)) \triangleright E)\]

Regression (for all operators)

1. When we have disjunction and conditional effects, things become more tricky. How to define regression e.g. for \(A \lor (B \land C)\) with \((Q, D \triangleright B)\)?

2. We present a general definition of how to do this.

3. Now we extensively use the idea of representing sets of states as formulae.

The story about goals and subgoals and fulfilling subgoals, as in the conjunctive case, does not work any more.
**Auxiliary definition: $EPC_l(e)$**

**DEFINITION** The condition $EPC_l(e)$ of literal $l$ becoming true when effect $e$ is applied is defined as follows.

\[
EPC_l(l) = \top
\]

\[
EPC_l(l') = \bot \text{ when } l \neq l' \text{ (for literals $l'$)}
\]

\[
EPC_l(\top) = \bot
\]

\[
EPC_l(e_1 \land \cdots \land e_n) = EPC_l(e_1) \lor \cdots \lor EPC_l(e_n)
\]

\[
EPC_l(c \triangleright e) = EPC_l(e) \land c
\]

**Auxiliary definition: $EPC_l(e)$, examples**

\[
EPC_A(B \land C) = \bot \lor \bot \equiv \bot
\]

\[
EPC_A(A \land (B \triangleright A)) = \top \lor (\top \land B) \equiv \top
\]

\[
EPC_A((C \triangleright A) \land (B \triangleright A)) = (\top \land C) \lor (\top \land B) \equiv C \lor B
\]

**LEMMA B:** Let $s$ be a state, $l$ a literal and $e$ an effect. Then $l \in [e]_s$ if and only if $s \models EPC_l(e)$.

**PROOF:** by induction on the structure of the effect $e$.

Base case 1, $e = \top$: By definition of $[\top]_s$, we have $l \notin [\top]_s = \emptyset$, and by definition of $EPC_l(\top)$ we have $s \not\models EPC_l(\top) = \bot$, so the equivalence holds.

Base case 2, $e = l$: $l \in [l]_s = \{l\}$ by definition, and $s \models EPC_l(l) = \top$ by definition.

Base case 3, $e = l'$ for some literal $l' \neq l$: $l \notin [l']_s = \{l\}$ by definition, and $s \not\models EPC_l(l') = \bot$ by definition.

Inductive case 1, $e = e_1 \land \cdots \land e_n$:

\[
EPC_l(e) \text{ if and only if } l \in [e']_s \text{ for some } e' \in \{e_1, \ldots, e_n\}
\]

\[
EPC_l(e) \text{ if and only if } s \models EPC_l(e') \text{ for some } e' \in \{e_1, \ldots, e_n\}
\]

\[
EPC_l(e_1) \lor \cdots \lor EPC_l(e_n)
\]

\[
EPC_l(e_1 \land \cdots \land e_n)
\]

Inductive case 2, $e = c \triangleright e'$:

\[
l \in [c \triangleright e']_s \text{ if and only if } l \in [e']_s \text{ and } s \models c
\]

\[
l \in [c \triangleright e']_s \text{ if and only if } s \models EPC_l(e') \text{ and } s \models c
\]

\[
l \in [c \triangleright e']_s \text{ if and only if } s \models EPC_l(c \triangleright e')
\]

Q.E.D.
**Auxiliary definition: EPC\(e\) vs. normal form**

Notice that in terms of \(EPC_p(e)\) any operator \((c, e)\) can be expressed in normal form as

\[
\left\langle c, \bigwedge_{p \in P} (EPC_p(e) \triangleright p) \land (EPC_{\neg p}(e) \triangleright \neg p) \right\rangle.
\]

**Regression: definition for literals**

The formula \((p \land \neg EPC_{\neg p}(e)) \lor EPC_p(e)\) expresses the truth-value of \(p \in P\) after applying \(o\) in terms of truth-values of formulae before applying \(o\): either

- \(p\) was true before and did not become false, or
- \(p\) became true.

**Regression: definition for literals, examples**

Let \(e = (B \triangleright A) \land (C \triangleright \neg A) \land B \land \neg D\).

\[
p \quad \begin{array}{c}
A \quad (A \land \neg C) \lor B \\
B \quad (B \land \bot) \lor \top \equiv \top \\
C \quad (C \land \bot) \lor \bot \equiv C \\
D \quad (D \land \neg \top) \lor \bot \equiv \bot
\end{array}
\]

**Regression: definition for literals**

**LEMMA C:** Let \(p\) be a state variable and \(o = (c, e) \in O\) and operator. Let \(s\) be a state and \(s' = \text{app}_o(s)\). Then \(s \models (p \land \neg EPC_{\neg p}(e)) \lor EPC_p(e)\) if and only if \(s' \models p\).

**PROOF:** Assume \(s \models (p \land \neg EPC_{\neg p}(e)) \lor EPC_p(e)\). Do a case analysis on the two disjuncts.

**Case 1:** Assume that \(s \models p \land \neg EPC_{\neg p}(e)\). By Lemma B \(\neg p \notin [e]_s\). Hence \(p\) remains true in \(s'\).

**Case 2:** Assume that \(s \models EPC_p(e)\). By Lemma B \(p \in [e]_s\), and hence \(s' \models p\).
For the other half of the equivalence, assume that \( s \not\models (p \land \neg EPC_{\neg p}(e)) \lor EPC_p(e) \).

Hence \( s \models (\neg p \lor EPC_{\neg p}(e)) \land \neg EPC_p(e) \).

Assume that \( s \models EPC_{\neg p}(e) \) because \( s \models \neg p \lor EPC_{\neg p}(e) \), and hence by Lemma B \( \neg p \in [e]_s \) and hence \( s' \not\models p \).

Assume that \( s \not\models p \). Because \( s \models \neg EPC_p(e) \), by Lemma B \( p \not\in [e]_s \) and hence \( s' \not\models p \).

Therefore it must be that \( s' \not\models p \). Q.E.D.

Regression: definition for formulae

The formula \( EPC_l(o) \) can now be used in defining regression for operators \( o \).

**DEFINITION** Let \( \phi \) be a propositional formula. Let \( (z, e) \) be an operator. The **regression** of \( \phi \) with respect to \( (z, e) \) is

\[
\text{reg}_{(z, e)}(\phi) = \phi_r \land z \land f
\]

where \( \phi_r \) is obtained from \( \phi \) by replacing \( p \in P \) by \( (p \land \neg EPC_{\neg p}(e)) \lor EPC_p(e) \), and \( f = \bigwedge_{p \in P} (\neg EPC_p(e) \land EPC_{\neg p}(e)) \).

The conjuncts of \( f \) say that no state variable may become both true and false.

Regression: examples

\[
\text{reg}_{(a,b)}(b) = (((b \land \neg \bot) \lor \top) \land a) \equiv a
\]

\[
\text{reg}_{(a,b)}(b \land c \land d) = (((b \land \neg \bot) \lor \top) \land ((c \land \neg \bot) \lor \bot) \land ((d \land \neg \bot) \lor \bot) \land a) \equiv c \land d \land a
\]

\[
\text{reg}_{(a,c,b)}(b) = (((b \land \neg \bot) \lor c) \land a) \equiv (b \lor c) \land a
\]

\[
\text{reg}_{(a,c,d,b)}(b) = (((b \land \neg b) \lor c) \land a \land \neg (c \land b)) \equiv c \land a \land \neg b
\]

\[
\text{reg}_{(a,c,b,d)}(b) = (((b \land \neg d) \lor c) \land a \land \neg (c \land d)) \equiv (b \lor c) \land (\neg d \lor c) \land a \land (\neg c \lor d)
\]