Planning and Optimization

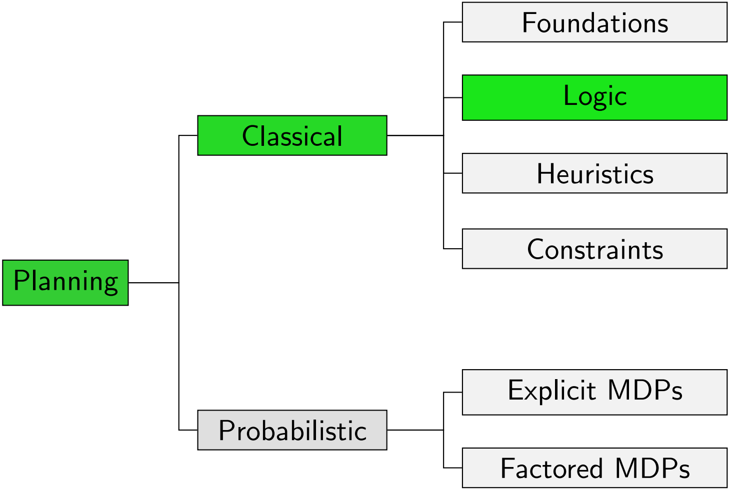
B3. General Regression

Malte Helmert and Thomas Keller

Universität Basel

October 7, 2019

Content of this Course



Regression for General Planning Tasks

- With disjunctions and conditional effects, things become more tricky. How to regress $a \lor (b \land c)$ with respect to $\langle q, d \rhd b \rangle$?
- In this chapter, we show how to regress general sets of states through general operators.
- We extensively use the idea of representing sets of states as formulas.

Regressing State Variables

Regressing State Variables: Motivation

Key question for general regression:

- Assume we are applying an operator with effect e.
- What must be true in the predecessor state for propositional state variable v to be true in the successor state?

If we can answer this question, a general definition of regression is only a small additional step.

Regressing State Variables: Key Idea

Assume we are in state s and apply effect e to obtain successor state s'.

Propositional state variable v is true in s' iff

- effect e makes it true, or
- it remains true, i.e., it is true in s and not made false by e.

Regressing a State Variable Through an Effect

Definition (Regressing a State Variable Through an Effect)

Let e be an effect of a propositional planning task, and let v be a propositional state variable.

The regression of v through e, written $regr(v, e)$, is defined as the following logical formula:

$$regr(v, e) = effcond(v, e) \vee (v \wedge \neg effcond(\neg v, e)).$$

Questions:

- Does this capture add-after-delete semantics correctly?
- How can we define regression for FDR tasks?

Regressing State Variables: Example

Example

Let $e = (b \rhd a) \land (c \rhd \neg a) \land b \land \neg d$.

v	$regr(v, e)$
a	$b \lor (a \land \neg c)$
b	$\top \lor (b \land \neg\bot) \equiv \top$
c	$\bot \lor (c \land \neg\bot) \equiv c$
d	$\bot \lor (d \land \neg\top) \equiv \bot$

Reminder: $regr(v, e) = effcond(v, e) \lor (v \land \neg effcond(\neg v, e))$

Regressing State Variables: Correctness (1)

Lemma (Correctness of $regr(v, e)$)

Let s be a state, e be an effect and v be a state variable of a propositional planning task.

Then $s \models regr(v, e)$ iff $s[\![e]\!] \models v$.

Regressing State Variables: Correctness (2)

Proof.

($\Rightarrow$): We know $s \models \mathit{regr}(v, e)$, and hence
$s \models \mathit{effcond}(v, e) \lor (v \land \neg \mathit{effcond}(\neg v, e))$.

Do a case analysis on the two disjuncts.

Regressing State Variables: Correctness (2)

Proof.

($\Rightarrow$): We know $s \models \mathit{regr}(v, e)$, and hence
$s \models \mathit{effcond}(v, e) \vee (v \wedge \neg \mathit{effcond}(\neg v, e))$.

Do a case analysis on the two disjuncts.

Case 1: $s \models \mathit{effcond}(v, e)$.
Then $s[\![e]\!] \models v$ by the first case in the definition of $s[\![e]\!]$ (Ch. A4).

Regressing State Variables: Correctness (2)

Proof.

(⇒): We know $s \models regr(v, e)$, and hence
$s \models effcond(v, e) \lor (v \land \lnot effcond(\lnot v, e))$.

Do a case analysis on the two disjuncts.

Case 1: $s \models effcond(v, e)$.
Then $s[\![e]\!] \models v$ by the first case in the definition of $s[\![e]\!]$ (Ch. A4).

Case 2: $s \models (v \land \lnot effcond(\lnot v, e))$.
Then $s \models v$ and $s \not\models effcond(\lnot v, e)$.
We may additionally assume $s \not\models effcond(v, e)$
because otherwise we can apply Case 1 of this proof.
Then $s[\![e]\!] \models v$ by the third case in the definition of $s[\![e]\!]$. ...

Regressing State Variables: Correctness (3)

Proof (continued).

($\Leftarrow$): Proof by contraposition.

We show that if $regr(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

Regressing State Variables: Correctness (3)

Proof (continued).

($\Leftarrow$): Proof by contraposition.

We show that if $regr(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

- By prerequisite, $s \not\models effcond(v, e) \lor (v \land \neg effcond(\neg v, e))$.

Regressing State Variables: Correctness (3)

Proof (continued).

($\Leftarrow$): Proof by contraposition.

We show that if $regr(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

- By prerequisite, $s \not\models effcond(v, e) \lor (v \land \neg effcond(\neg v, e))$.
- Hence $s \models \neg effcond(v, e) \land (\neg v \lor effcond(\neg v, e))$.

Regressing State Variables: Correctness (3)

Proof (continued).

($\Leftarrow$): Proof by contraposition.

We show that if $\mathit{regr}(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

- By prerequisite, $s \not\models \mathit{effcond}(v, e) \lor (v \land \lnot \mathit{effcond}(\lnot v, e))$.
- Hence $s \models \lnot \mathit{effcond}(v, e) \land (\lnot v \lor \mathit{effcond}(\lnot v, e))$.
- From the first conjunct, we get $s \models \lnot \mathit{effcond}(v, e)$ and hence $s \not\models \mathit{effcond}(v, e)$.

Regressing State Variables: Correctness (3)

Proof (continued).

(⇐): Proof by contraposition.

We show that if $regr(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

- By prerequisite, $s \not\models effcond(v, e) \lor (v \land \neg effcond(\neg v, e))$.
- Hence $s \models \neg effcond(v, e) \land (\neg v \lor effcond(\neg v, e))$.
- From the first conjunct, we get $s \models \neg effcond(v, e)$ and hence $s \not\models effcond(v, e)$.
- From the second conjunct, we get $s \models \neg v \lor effcond(\neg v, e)$.

Regressing State Variables: Correctness (3)

Proof (continued).

(⇐): Proof by contraposition.

We show that if $regr(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

- By prerequisite, $s \not\models effcond(v, e) \lor (v \land \neg effcond(\neg v, e))$.
- Hence $s \models \neg effcond(v, e) \land (\neg v \lor effcond(\neg v, e))$.
- From the first conjunct, we get $s \models \neg effcond(v, e)$ and hence $s \not\models effcond(v, e)$.
- From the second conjunct, we get $s \models \neg v \lor effcond(\neg v, e)$.
- Case 1: $s \models \neg v$. Then v is false before applying e and remains false, so $s[\![e]\!] \not\models v$.

Regressing State Variables: Correctness (3)

Proof (continued).

($\Leftarrow$): Proof by contraposition.

We show that if $regr(v, e)$ is false in s, then v is false in $s[\![e]\!]$.

- By prerequisite, $s \not\models effcond(v, e) \lor (v \land \neg effcond(\neg v, e))$.
- Hence $s \models \neg effcond(v, e) \land (\neg v \lor effcond(\neg v, e))$.
- From the first conjunct, we get $s \models \neg effcond(v, e)$ and hence $s \not\models effcond(v, e)$.
- From the second conjunct, we get $s \models \neg v \lor effcond(\neg v, e)$.
- Case 1: $s \models \neg v$. Then v is false before applying e and remains false, so $s[\![e]\!] \not\models v$.
- Case 2: $s \models effcond(\neg v, e)$. Then v is deleted by e and not simultaneously added, so $s[\![e]\!] \not\models v$.

□

Regressing Formulas Through Effects

Regressing Formulas Through Effects: Idea

- We can now generalize regression from state variables to general formulas over state variables.
- The basic idea is to replace every occurrence of every state variable v by $regr(v, e)$ as defined in the previous section.
- The following definition makes this more formal.

Regressing Formulas Through Effects: Definition

Definition (Regressing a Formula Through an Effect)

In a propositional planning task, let e be an effect, and let φ be a formula over propositional state variables.

The regression of φ through e, written $\mathit{regr}(\varphi, e)$, is defined as the following logical formula:

$$
\begin{aligned}
\mathit{regr}(\top, e) &= \top \\
\mathit{regr}(\bot, e) &= \bot \\
\mathit{regr}(v, e) &= \mathit{effcond}(v, e) \lor (v \land \neg \mathit{effcond}(\neg v, e)) \\
\mathit{regr}(\neg\psi, e) &= \neg \mathit{regr}(\psi, e) \\
\mathit{regr}(\psi \lor \chi, e) &= \mathit{regr}(\psi, e) \lor \mathit{regr}(\chi, e) \\
\mathit{regr}(\psi \land \chi, e) &= \mathit{regr}(\psi, e) \land \mathit{regr}(\chi, e).
\end{aligned}
$$

Question: definition for FDR tasks?

Regressing Formulas Through Effects: Example

Example

Let $e = (b \rhd a) \land (c \rhd \neg a) \land b \land \neg d$.

Recall:

- $regr(a, e) \equiv b \lor (a \land \neg c)$
- $regr(b, e) \equiv \top$
- $regr(c, e) \equiv c$
- $regr(d, e) \equiv \bot$

We get:

$$\begin{aligned} regr((a \lor d) \land (c \lor d), e) &\equiv ((b \lor (a \land \neg c)) \lor \bot) \land (c \lor \bot) \\ &\equiv (b \lor (a \land \neg c)) \land c \\ &\equiv b \land c \end{aligned}$$

Regressing Formulas Through Effects: Correctness (1)

Lemma (Correctness of $regr(\varphi, e)$)

Let φ be a logical formula, e an effect and s a state of a propositional planning task.

Then $s \models regr(\varphi, e)$ iff $s[\![e]\!] \models \varphi$.

Regressing Formulas Through Effects: Correctness (2)

Proof.

The proof is by structural induction on φ.

Regressing Formulas Through Effects: Correctness (2)

Proof.

The proof is by structural induction on φ.

Induction hypothesis: $s \models \mathit{regr}(\psi, e)$ iff $s[\![e]\!] \models \psi$
for all proper subformulas ψ of φ.

Regressing Formulas Through Effects: Correctness (2)

Proof.

The proof is by structural induction on φ.

Induction hypothesis: $s \models \mathit{regr}(\psi, e)$ iff $s[\![e]\!] \models \psi$
for all proper subformulas ψ of φ.

Base case $\varphi = \top$:

We have $\mathit{regr}(\top, e) = \top$, and $s \models \top$ iff $s[\![e]\!] \models \top$ is correct.

Regressing Formulas Through Effects: Correctness (2)

Proof.

The proof is by structural induction on φ.

Induction hypothesis: $s \models \mathit{regr}(\psi, e)$ iff $s[\![e]\!] \models \psi$ for all proper subformulas ψ of φ.

Base case $\varphi = \top$:

We have $\mathit{regr}(\top, e) = \top$, and $s \models \top$ iff $s[\![e]\!] \models \top$ is correct.

Base case $\varphi = \bot$:

We have $\mathit{regr}(\bot, e) = \bot$, and $s \models \bot$ iff $s[\![e]\!] \models \bot$ is correct.

Regressing Formulas Through Effects: Correctness (2)

Proof.

The proof is by structural induction on φ.

Induction hypothesis: $s \models regr(\psi, e)$ iff $s[\![e]\!] \models \psi$
for all proper subformulas ψ of φ.

Base case $\varphi = \top$:

We have $regr(\top, e) = \top$, and $s \models \top$ iff $s[\![e]\!] \models \top$ is correct.

Base case $\varphi = \bot$:

We have $regr(\bot, e) = \bot$, and $s \models \bot$ iff $s[\![e]\!] \models \bot$ is correct.

Base case $\varphi = v$:

We have $s \models regr(v, e)$ iff $s[\![e]\!] \models v$ from the previous lemma. ...

Regressing Formulas Through Effects: Correctness (3)

Proof (continued).

Inductive case $\varphi = \neg\psi$:

$$\begin{aligned} s \models \mathit{regr}(\neg\psi, e) &\text{ iff } s \models \neg\mathit{regr}(\psi, e) \\ &\text{ iff } s \not\models \mathit{regr}(\psi, e) \\ &\text{ iff } s[\![e]\!] \not\models \psi \\ &\text{ iff } s[\![e]\!] \models \neg\psi \end{aligned}$$

Regressing Formulas Through Effects: Correctness (3)

Proof (continued).

Inductive case $\varphi = \neg\psi$:

$$\begin{aligned} s \models \mathit{regr}(\neg\psi, e) &\text{ iff } s \models \neg \mathit{regr}(\psi, e) \\ &\text{ iff } s \not\models \mathit{regr}(\psi, e) \\ &\text{ iff } s[\![e]\!] \not\models \psi \\ &\text{ iff } s[\![e]\!] \models \neg\psi \end{aligned}$$

Inductive case $\varphi = \psi \vee \chi$:

$$\begin{aligned} s \models \mathit{regr}(\psi \vee \chi, e) &\text{ iff } s \models \mathit{regr}(\psi, e) \vee \mathit{regr}(\chi, e) \\ &\text{ iff } s \models \mathit{regr}(\psi, e) \text{ or } s \models \mathit{regr}(\chi, e) \\ &\text{ iff } s[\![e]\!] \models \psi \text{ or } s[\![e]\!] \models \chi \\ &\text{ iff } s[\![e]\!] \models \psi \vee \chi \end{aligned}$$

Regressing Formulas Through Effects: Correctness (3)

Proof (continued).

Inductive case $\varphi = \neg\psi$:

$$\begin{aligned} s \models \mathit{regr}(\neg\psi, e) &\text{ iff } s \models \neg\mathit{regr}(\psi, e) \\ &\text{ iff } s \not\models \mathit{regr}(\psi, e) \\ &\text{ iff } s[\![e]\!] \not\models \psi \\ &\text{ iff } s[\![e]\!] \models \neg\psi \end{aligned}$$

Inductive case $\varphi = \psi \vee \chi$:

$$\begin{aligned} s \models \mathit{regr}(\psi \vee \chi, e) &\text{ iff } s \models \mathit{regr}(\psi, e) \vee \mathit{regr}(\chi, e) \\ &\text{ iff } s \models \mathit{regr}(\psi, e) \text{ or } s \models \mathit{regr}(\chi, e) \\ &\text{ iff } s[\![e]\!] \models \psi \text{ or } s[\![e]\!] \models \chi \\ &\text{ iff } s[\![e]\!] \models \psi \vee \chi \end{aligned}$$

Inductive case $\varphi = \psi \wedge \chi$:

Like previous case, replacing "$\vee$" by "$\wedge$"
and replacing "or" by "and". □

Regressing Formulas Through Operators

Regressing Formulas Through Operators: Idea

- We can now regress arbitrary formulas through arbitrary effects.
- The last missing piece is a definition of regression through operators, describing exactly in which states s applying a given operator o leads to a state satisfying a given formula φ.
- There are two requirements:
  - The operator o must be applicable in the state s.
  - The resulting state $s[\![o]\!]$ must satisfy φ.

Regressing Formulas Through Operators: Definition

Definition (Regressing a Formula Through an Operator)

In a propositional planning task, let o be an operator, and let φ be a formula over state variables.

The regression of φ through o, written $regr(\varphi, o)$, is defined as the following logical formula:

$$regr(\varphi, o) = pre(o) \wedge regr(\varphi, eff(o)).$$

Question: definition for FDR tasks?

Regressing Formulas Through Operators: Correctness (1)

Theorem (Correctness of $regr(\varphi, o)$)

Let φ be a logical formula, o an operator and s a state of a propositional planning task.

Then $s \models regr(\varphi, o)$ iff o is applicable in s and $s[\![o]\!] \models \varphi$.

Regressing Formulas Through Operators: Correctness (2)

Reminder: $regr(\varphi, o) = pre(o) \wedge regr(\varphi, eff(o))$

Proof.

Case 1: $s \models pre(o)$.

Then o is applicable in s and the statement we must prove simplifies to: $s \models regr(\varphi, eff(o))$ iff $s[\![e]\!] \models \varphi$.
This was proved in the previous lemma.

Regressing Formulas Through Operators: Correctness (2)

Reminder: $\mathit{regr}(\varphi, o) = \mathit{pre}(o) \land \mathit{regr}(\varphi, \mathit{eff}(o))$

Proof.

Case 1: $s \models \mathit{pre}(o)$.

Then o is applicable in s and the statement we must prove simplifies to: $s \models \mathit{regr}(\varphi, \mathit{eff}(o))$ iff $s[\![e]\!] \models \varphi$.
This was proved in the previous lemma.

Case 2: $s \not\models \mathit{pre}(o)$.

Then $s \not\models \mathit{regr}(\varphi, o)$ and o is not applicable in s.
Hence both statements are false and therefore equivalent. □

Regression Examples (1)

Examples: compute regression and simplify to DNF

- $\mathit{regr}(b, \langle a, b\rangle)$
  $\equiv a \wedge (\top \vee (b \wedge \neg\bot))$
  $\equiv a$
- $\mathit{regr}(b \wedge c \wedge d, \langle a, b\rangle)$
  $\equiv a \wedge (\top \vee (b \wedge \neg\bot)) \wedge (\bot \vee (c \wedge \neg\bot)) \wedge (\bot \vee (d \wedge \neg\bot))$
  $\equiv a \wedge c \wedge d$
- $\mathit{regr}(b \wedge \neg c, \langle a, b \wedge c\rangle)$
  $\equiv a \wedge (\top \vee (b \wedge \neg\bot)) \wedge \neg(\top \vee (c \wedge \neg\bot))$
  $\equiv a \wedge \top \wedge \bot$
  $\equiv \bot$

Regression Examples (2)

Examples: compute regression and simplify to DNF

- $regr(b, \langle a, c \rhd b \rangle)$
  $\equiv a \land (c \lor (b \land \neg\bot))$
  $\equiv a \land (c \lor b)$
  $\equiv (a \land c) \lor (a \land b)$
- $regr(b, \langle a, (c \rhd b) \land ((d \land \neg c) \rhd \neg b) \rangle)$
  $\equiv a \land (c \lor (b \land \neg(d \land \neg c)))$
  $\equiv a \land (c \lor (b \land (\neg d \lor c)))$
  $\equiv a \land (c \lor (b \land \neg d) \lor (b \land c))$
  $\equiv a \land (c \lor (b \land \neg d))$
  $\equiv (a \land c) \lor (a \land b \land \neg d)$

Summary

Summary

- Regressing a propositional state variable through an (arbitrary) operator must consider two cases:
  - state variables made true (by add effects)
  - state variables remaining true (by absence of delete effects)
- Regression of propositional state variables can be generalized to arbitrary formulas φ by replacing each occurrence of a state variable in φ by its regression.
- Regressing a formula φ through an operator involves regressing φ through the effect and enforcing the precondition.