COMP525: Reasoning about Action and Change
Lecture 25: Introduction to Propositional Dynamic Logic

Davide Grossi
Department of Computer Science
University of Liverpool
Liverpool, UK
d.grossi@liverpool.ac.uk
http://www.csc.liv.ac.uk/∼dgrossi

Lecture notes are based on material by Clare Dixon & Alexei Lisitsa

Introduction

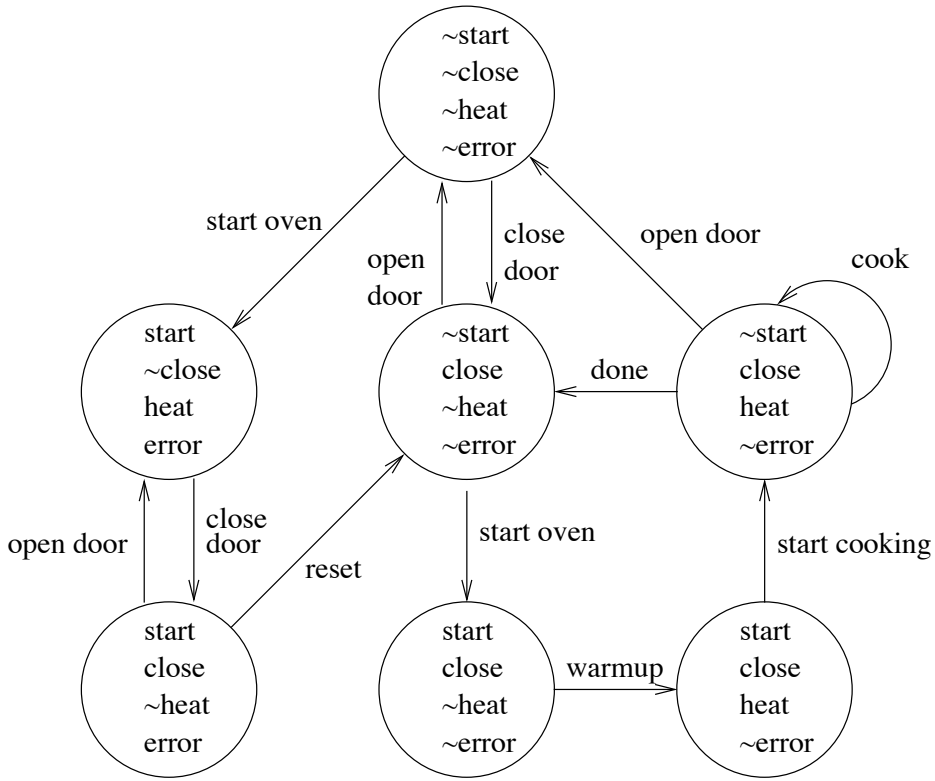
In Weeks 6 & 8:

- We have introduced the Situation Calculus as a formalism to specify and reason about action
- We have addressed typical problems concerning specification (e.g., the Frame Problem) and methods for reasoning (e.g., regression)

This and next week:

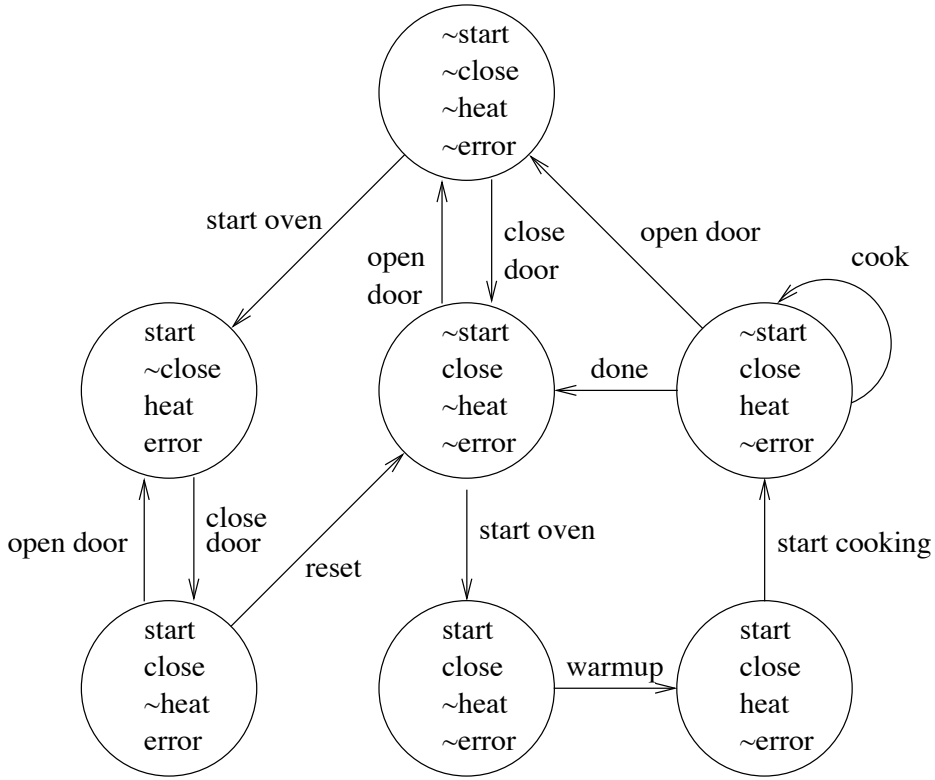
- We abandon the framework of FOL and go back to the one of Modal Logic which we used for LTL and CTL.
- We will study a logic (Propositional Dynamic Logic) which allows one to talk about actions (and not just time).
- Today we introduce the syntax and semantics of Propositional Dynamic Logic (PDL).

PDL: basic semantic intuition



- In LTL and CTL we could talk about states and properties of states (what is true at each state);
- In PDL we can also talk about transitions, by assigning labels to them like we assign proposition to states.

PDL: basic semantic intuition



- Actions are *labels of transitions*;
- We usually denote generic actions by α, β, etc.

Introducing the Syntax: Actions

- For some action α and formula φ we can write:
  - $[\alpha]\varphi$: after the performance (execution) of the action (or program) α, φ necessarily holds; or
    after all executions of α, a state is reached where φ is true.
  - $\langle\alpha\rangle\varphi$: meaning after the performance (execution) of the action α, φ possibly holds; or
    there exists an execution of α that ends up in a state in which φ is true.
- Actions may be combined with operators to form complex actions or programs, like propositions are (although the operators are different!).

Introducing the Syntax (II): Examples of Formulae

- In the following examples *putdown_glass* and *drop_glass* are actions and *glass_on_table* and *glass_broken* are propositions:
  - $[putdown\_glass]glass\_on\_table$: the putdown glass action (necessarily) results in the glass being on the table;
  - $\langle drop\_glass \rangle glass\_broken$: the drop glass action possibly results in the glass being broken.
- Considering the Yale Shooting Problem with actions including *load* and propositions including *loaded* we can say:
  - $[load]loaded$: the load action (necessarily) results in the gun being loaded;
  - $\langle load \rangle \mathbf{true}$: its possible to do a load action.

Syntax of PDL: Alphabet

The alphabet of PDL consists of:

- A set PROP of propositional atoms;
- The Boolean connectives;
- A set ACT of atomic actions;
- The action operators: ; (sequencing), $\cup$ (choice), $*$ (iteration or *Kleene Star*), ? (test)
- The modal operators: [ ] and $\langle\ \rangle$.

NOTE: we have *two sorts* of expressions: actions and formulae.

We will now define the set of well-formed formulae WFF and well-formed actions WFA.

Syntax of PDL (II): WFFs

The set of well-formed formulae of PDL is so defined:

- PROP $\subseteq$ WFF
- $\{\textbf{true}, \textbf{false}\} \subseteq$ WFF
- If $\{\alpha, \beta\} \subseteq$ WFA and $\varphi \in$ WFF then $\{\alpha;\beta, \alpha \cup \beta, \alpha^*, ?\varphi\} \subseteq$ WFA
- If $\alpha \in$ WFA and $\{\varphi, \psi\} \subseteq$ WFF then:
  - $\{(\varphi),\ \neg\varphi,\ \varphi \vee \psi,\ \varphi \wedge \psi,\ \varphi \rightarrow \psi\} \subseteq$ WFF
  - $\{[\alpha]\varphi, \langle\alpha\rangle\varphi\} \subseteq$ WFF

QUESTION: is there anything interesting about this inductive definition?

Syntax of PDL (III): Examples

The following are well formed PDL formulae.

- $\langle \alpha^*; (\beta \cup \varphi?) \rangle \psi$
- $[(\alpha; \beta)^*](\varphi \rightarrow \psi)$

The following are not well formed PDL formulae.

- $[\alpha \wedge \beta]\varphi$
- $\langle \varphi \rangle \psi$
- $[\,]\varphi$
- $\langle \varphi? \rangle \alpha$
- $\langle (\alpha \cup \beta)^* \rangle \varphi; \psi$

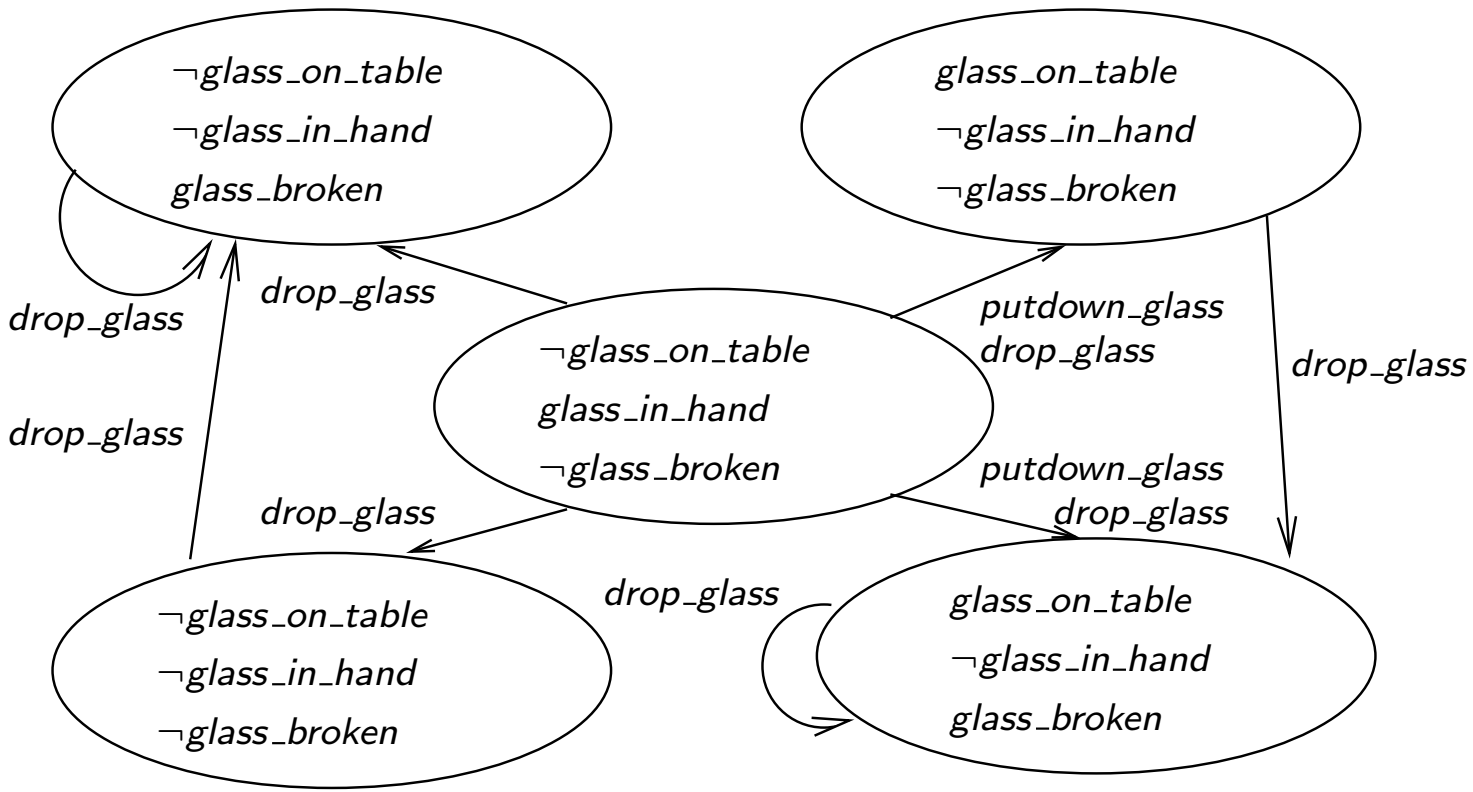
Semantics: Introduction

- We interpret the formulae of PDL on state transition systems.
- The semantics is therefore of the same type we have seen for LTL and CTL: Kripke Semantics
- We have states (where formulae can be evaluated as true or false), and transition between states which are now labeled by atomic actions.
- For formulae of the form $[\alpha]\varphi$ to hold in a state all α edges must lead to a state where φ holds.
- For formulae of the form $\langle\alpha\rangle\varphi$ to hold in a state there must exist some α edge which leads to a state where φ holds.

Semantics (II): Introduction

Here is a model for:

$\neg glass\_on\_table$, $\neg glass\_broken$, $glass\_in\_hand$,
$[putdown\_glass]glass\_on\_table$, $\langle drop\_glass\rangle glass\_broken$,
$[putdown\_glass]\neg glass\_in\_hand$, $[drop\_glass]\neg glass\_in\_hand$



Operators on Actions: Intuitions

Just as we can combine formulae with operators such as $\wedge$, $\vee$, $\rightarrow$, $\neg$ we can also apply operators to actions (or in one case the formulae) to produce more complex actions.

- ; denotes sequential composition: i.e. $\alpha;\beta$ denotes first do α and then do β. E.g.: *open_door*; *move_through_door*;
- $\cup$ denotes (non-deterministic) choice: i.e. $\alpha \cup \beta$ denotes performing either α or β. E.g.: *wash_up* $\cup$ *read_book*;
- $*$ denotes arbitrary repetition: i.e. *add_one** denotes performing the *add_one* action zero or more times (note the connection with loop commands in programming!).
- ? denotes the test action: i.e. $\varphi?$ denotes the action of checking whether φ is true. E.g., denotes the action that tests whether *is_raining* holds.

Semantics: Models

PDL model

A PDL model is a tuple $\mathcal{M} = (S, R, \pi)$ where:

- S is a set of states;
- $R : \text{ACT} \to \wp(S \times S)$ is a function assigning to each atomic action the set of transitions labeled by the atomic action;
- $\pi : \text{PROP} \to \wp(S)$ is a valuation function assigning to each atomic proposition the set of states where the proposition is true.

That is:

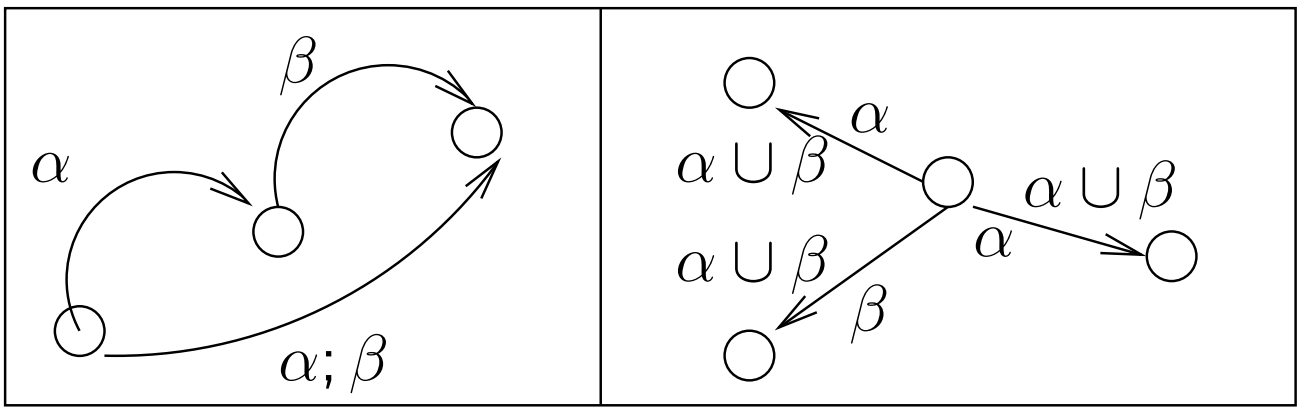
- $R(a) \subseteq S \times S$ such that $a \in \text{ACT}$;
- $\pi(p) \subseteq S$ such that $p \in \text{PROP}$.

QUESTION: Each $R(a)$ is a binary relation associated to atomic action a. What relations do we assign to complex actions like $a; b$, $?p$, $a \cup b$, etc.?

Semantics (II): Extension of R

For $\alpha, \beta \in \text{WFA}$ and $\varphi \in \text{WFF}$:

$$\begin{aligned} R(\alpha;\beta) &= \{(u,v) \mid \exists w \in S \text{ s.t. } (u,w) \in R(\alpha) \\ &\qquad \text{and } (w,v) \in R(\beta)\} \\ R(\alpha \cup \beta) &= \{(u,v) \mid (u,v) \in R(\alpha) \text{ or } (u,v) \in R(\beta)\} \end{aligned}$$



$$\begin{aligned} R(\varphi?) &= \{(u,v) \mid u = v \text{ and } \mathcal{M}, v \models \varphi\} \\ R(\alpha^*) &= \{(u,v) \mid \exists n \geqslant 0 \text{ and } u_0, \ldots u_n, \text{ s.t. } u = u_0, v = u_n \\ &\qquad \text{and } (u_i, u_{i+1}) \in R(\alpha), \forall 0 \leqslant i \leqslant n-1\} \end{aligned}$$

Semantics (III): Satisfaction

$\mathcal{M}, u \models \textbf{true}$ $\qquad$ $\mathcal{M}, u \not\models \textbf{false}$

$\mathcal{M}, u \models p$ iff $u \in \pi(p)$ $\qquad$ $\mathcal{M}, u \models \neg\varphi$ iff $\mathcal{M}, u \not\models \varphi$

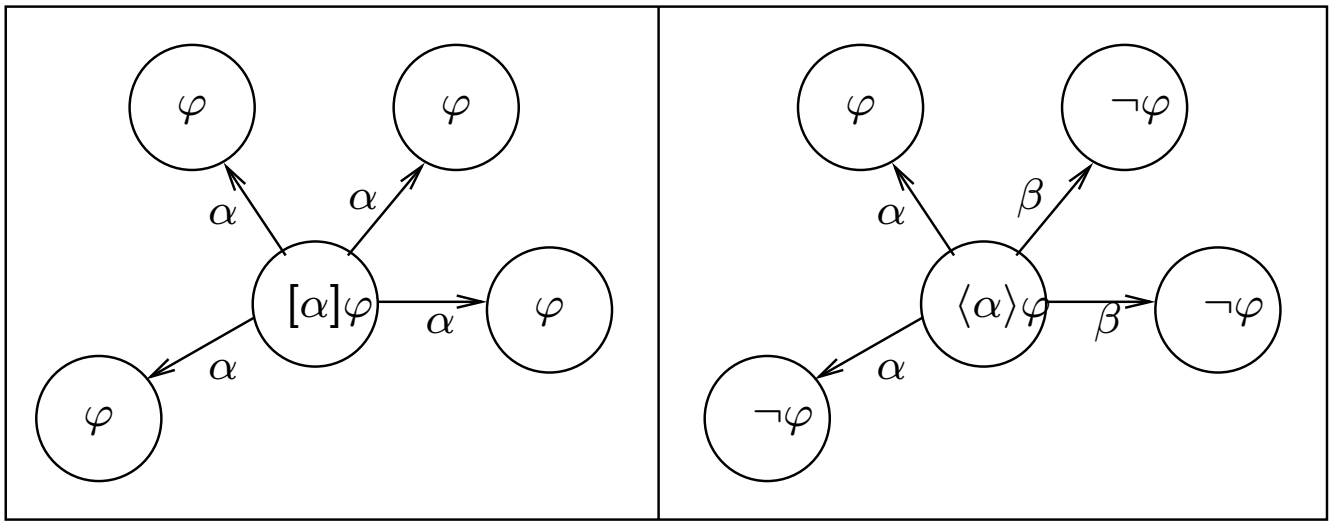
$\mathcal{M}, u \models \varphi \wedge \psi$ iff $\mathcal{M}, u \models \varphi$ and $\mathcal{M}, u \models \psi$

$\mathcal{M}, u \models \varphi \vee \psi$ iff $\mathcal{M}, u \models \varphi$ or $\mathcal{M}, u \models \psi$

$\mathcal{M}, u \models \varphi \rightarrow \psi$ iff $\mathcal{M}, u \not\models \varphi$ or $\mathcal{M}, u \models \psi$

$\mathcal{M}, u \models [\alpha]\varphi$ iff for all $v \in S$ if $(u, v) \in R(\alpha)$ then $\mathcal{M}, v \models \varphi$

$\mathcal{M}, u \models \langle\alpha\rangle\varphi$ iff for some $v \in S, (u, v) \in R(\alpha)$ and $\mathcal{M}, v \models \varphi$



PDL and the Situation Calculus

- Both PDL and the situation calculus are concerned with the changes brought about by performing actions in given situations.
- For example to represent the statement "loading the gun results in the gun being loaded" we write:
  - $loaded(do(load, s))$ in the situation calculus; and
  - $[load]loaded$ in dynamic logic.
- Similarly to represent "it is possible to load the gun":
  - $Poss(load, s)$ in the situation calculus; and
  - $\langle load \rangle \mathbf{true}$ in dynamic logic.
- QUESTION: any ideas about what the key differences of the two formalisms are?

Summary

- We have introduced the main motivating ideas behind PDL.
- We have given the syntax and semantics of PDL.
- The semantics is closely related to the formalisms of LTL and CTL we already know.