Reasoning and Computation

Jan van Eijck

ESSLLI Logic in Action Course, Tuesday 2 August, 2011

Abstract

In this second part of today's lecture will give a lightning tour through some connections between systems of formal reasoning such as propositional logic, syllogistics and predicate logic, and computation. In particular, we will look at satisfiability checking, validity checking and model checking.

This lecture draws on material from Chapters 2, 3, and 10 from the Logic in Action course.

Some Key Notions of Logic: Truth, Satisfiability, Validity

ϕ is true in model M:

$$M \models \phi$$

ϕ is satisfiable:

there exists M with $M \models \phi$.

ϕ is valid:

for all models M it holds that $M \models \phi$.

Validity Checking in Syllogistics

See Section 3.5 of the book for a method.

Alternative method: reduction to propositional logic.

Key fact:

> A finite set of syllogistic forms Σ is unsatisfiable if and only if there exists an existential form ψ such that ψ taken together with the universal forms from Σ is unsatifiable.

This restricted form of satisfiability can easily be tested with propositional logic.

Syllogistics and Propositional Logic

We are talking about the properties of a single object x. Let proposition letter a express that object x has property A. Then a universal statement "all A are B" gets translated into $a \rightarrow b$: if x has property A then x also has property B. An existential statement "some A are B" gets translated into $a \wedge b$, expressing that x has both properties A and B. The universal negative statement "no A are B" gets translated into $a \rightarrow \neg b$, and the negative existential statement "some A are not B" gets translated as $a \wedge \neg b$.

This translation employs a single proposition letter for each property. No exponential blow-up here.

To test the satisfiability of a set of syllogistic statements containing n existential statements will need n tests: we have to check for each existential statement whether it is satisfiable when taken together with all universal statements. But this does not cause exponential blow-up if all these tests can be performed efficiently.

Literals, Clauses, Clause Sets

literals a literal is a proposition letter or its negation. If l is a literal, we use $\bar{l}$ for its negation: if l has the form p, then $\bar{l}$ equals $\neg p$, if l has the form $\neg p$, then $\bar{l}$ equals p. So if l is a literal, then $\bar{l}$ is also a literal, with opposite sign.

clause a clause is a set of literals.

clause sets a clause set is a set of clauses.

Read a clause as a disjunction of its literals, and a clause set as a conjunction of its clauses.

Here is an example: the clause form of

$$(p \to q) \land (q \to r)$$

is

$$\{\{\neg p, q\}, \{\neg q, r\}\}.$$

Unit Propagation

Unit Propagation If one member of a clause set is a singleton $\{l\}$ (a 'unit'), then:

1. remove every other clause containing l from the clause set (for since l has to be true, we know these other clauses have to be true as well, and no information gets lost by deleting them);
2. remove $\bar{l}$ from every clause in which it occurs (for since l has to be true, we know that $\bar{l}$ has to be false, so no information gets lost by deleting $\bar{l}$ from any disjunction in which it occurs).

The result of applying this rule is an equivalent clause set. Example: applying unit propagation using unit $\{p\}$ to

$$\{\{p\}, \{\neg p, q\}, \{\neg q, r\}, \{p, s\}\}.$$

yields:

$$\{\{p\}, \{q\}, \{\neg q, r\}\}.$$

Applying unit propagation to this, using unit $\{q\}$ yields

$$\{\{p\}, \{q\}, \{r\}\}.$$

HORNSAT

The Horn fragment of propositional logic consists of all clause sets where every clause has at most one positive literal. HORNSAT is the problem of checking Horn clause sets for satisfiability. This check can be performed in polynomial time (linear in the size of the formula, in fact).

If unit propagation yields a clause set in which units $\{l\}, \{\bar{l}\}$ occur, the original clause set is unsatisfiable, otherwise the units in the result determine a satisfying valuation. Recipe: for any units $\{l\}$ occurring in the final clause set, map their proposition letter to the truth value that makes l true; map all other proposition letters to false.

> The problem of testing satisfiability of syllogistic forms containing exactly one existential statement can be translated to the Horn fragment of propositional logic.

Check

To see that this is true, check the translations we gave above:

All A are B $\mapsto$ $a \to b$ or equivalently $\{\{\neg a, b\}\}$.

No A are B $\mapsto$ $a \to \neg b$ or equivalently $\{\{\neg a, \neg b\}\}$.

Some A are B $\mapsto$ $a \wedge b$ or equivalently $\{\{a\}, \{b\}\}$.

Not all A are B $\mapsto$ $a \wedge \neg b$ or equivalently $\{\{a\}, \{\neg b\}\}$.

These translations are all in the Horn fragment of propositional logic. We conclude that satisfiability of sets of syllogistic forms can be checked in time polynomial in the number of properties mentioned in the forms.

Computational Challenge

For those of you who are familiar with `http://www.haskell.org`.

Study the implementation of this method in Haskell, and complete the code given on the course website `Syllogistics.hs`.

Compare this implementation to: `http://www.computational-semantics.eu/InfEngine.hs`

Code Snippet

```
unitProp :: Lit -> [Clause] -> [Clause]
unitProp x cs = concat (map (unitP x) cs)

unitP :: Lit -> Clause -> [Clause]
unitP x ys = if elem x ys
              then []
              else
               if elem (neg x) ys
                then [delete (neg x) ys]
                else [ys]
```


CLAUDE
PRESBYTERIAN CHURCH
THERE ARE SOME
QUESTIONS THAT
CAN'T BE
ANSWERED BY
GOOGLE
Citytv
CityNews
CityNews.ca

Some New Billboards

There are some questions
that can’t be answered by logic

There are some questions
that can’t be answered
by computing machines

Formulas with Only Infinite Models

- Consider the conjunction of:
  - $\forall x \forall y (Rxy \to \neg Ryx)$ (R is asymmetric)
  - $\forall x \exists y Rxy$ (R is serial)
  - $\forall x \forall y \forall z ((Rxy \land Ryz) \to Rxz)$ (R is transitive).
- Suppose our domain is non-empty.
- Then every model of this conjunction is infinite. Why?
- The task of checking all relational structures (including infinite ones) in search for a model of a formula cannot be finished in a finite amount of time.

Consistency, Refutation of Consistency

- A first order formula is consistent if it has a model.
- The existence for formulas with only infinite models suggests that first order consistency is not decidable.
- In fact, we have a semi-decision method: if a formula is inconsistent the method will determine this after finitely many steps.
- The method consists of constructing a so-called semantic tableau. This boils down to a systematic search for an inconsistency.
- There are consistent formulas for which the method loops. The refutation method for consistency is not an algorithm.
- Note that nothing we have said above is a proof that a decision method for first order consistency cannot exist.

Undecidable Queries

- The deep reason behind the undecidability of first order logic is the fact that its expressive power is so great that it is possible to state undecidable queries.
- One of the famous undecidable queries is the halting problem.
- Here is what a halting algorithm would look like:
  - Input: a specification of a computational procedure P, and an input I for that procedure
  - Output: an answer 'halt' if P halts when applied to I, and 'loop' otherwise.

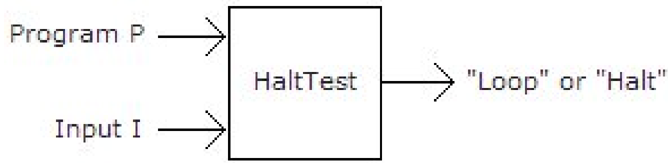
Undecidability of the Halting Problem

- Suppose there is an algorithm to solve the halting problem. Call this H.
- Then H takes a computational procedure P as input, together with an input I to that procedure, and decides. Note that H is itself a procedure; H takes two inputs, P and I.
- Let S be the procedure that is like H, except that it takes one input P, and then calls $H(P, P)$.
- Consider the following new procedure N for processing inputs P: If $S(P)$ says "halt", then loop,
  and if $S(P)$ says "loop", then print "halt" and terminate.

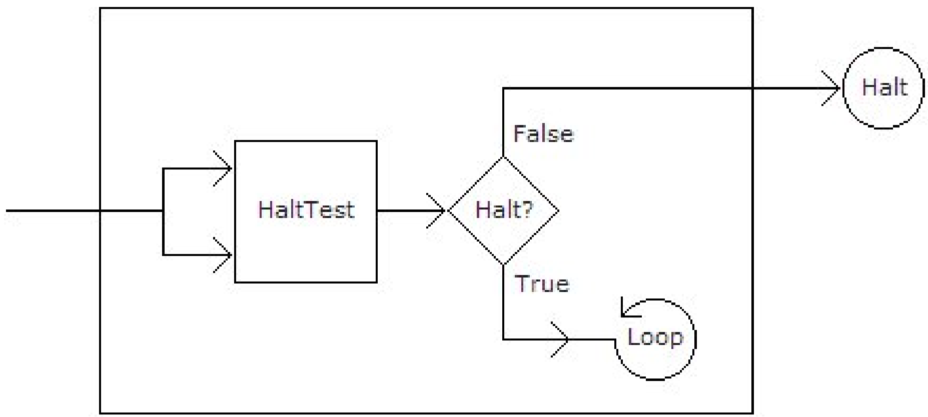
Undecidability of the Halting Problem (ctd)

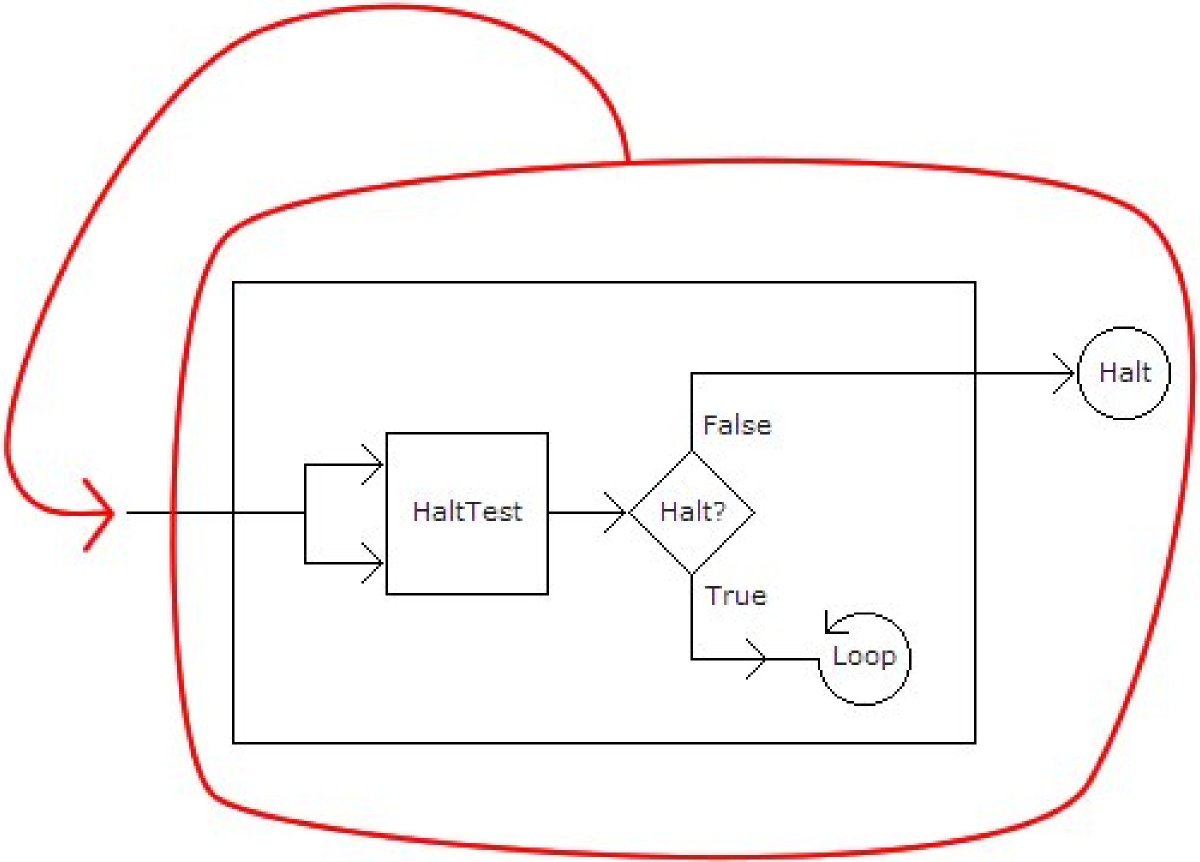
- What does N do when applied to N itself? In other words, what is the result of executing $N(N)$?
- Suppose N halts on input N. Then H should answer 'halt' when H is applied to N with input N, for H is supposed to be a correct halting algorithm. But then, by construction of the procedure, N loops. Contradiction.
- Suppose N loops on input N. Then H should answer 'loop' when H is applied to N with input N, for H is supposed to be a correct halting algorithm. But then, by construction of the procedure, N prints 'halt' and stops. Contradiction.
- We have a contradiction in both cases. Therefore a halting algorithm H cannot exist.

In Pictures . . .


Program P
Input I
HaltTest
"Loop" or "Halt"

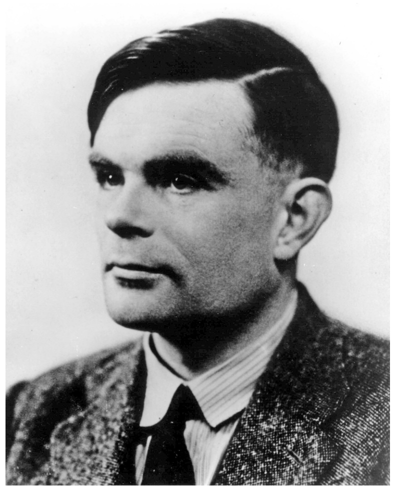

Program P
HaltTest
"Loop" or "Halt"


HaltTest
Halt?
False
True
Halt
Loop



Halt
False
HaltTest
Halt?
True
Loop

Alan Turing’s Insight

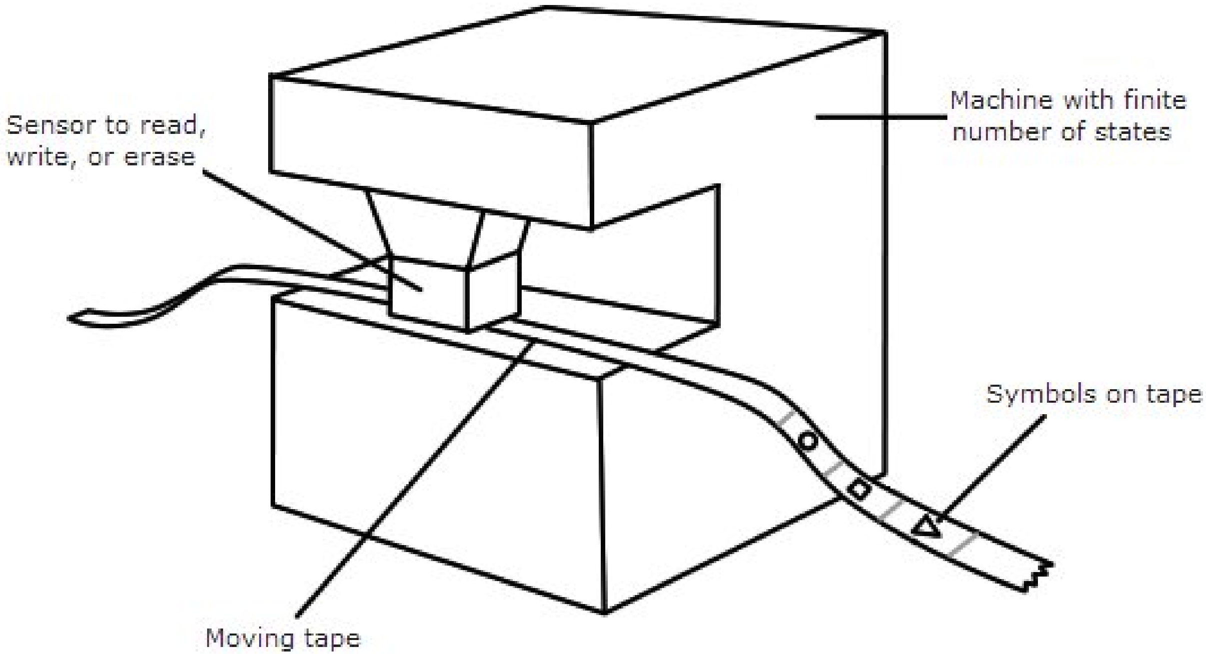


A language that allows the specification of ‘universal procedures’ such as H, S and N cannot be decidable.

But first order predicate logic is such a language . . .

Proof of Undecidability of First Order Logic

- The formal proof of the undecidability of first order logic consists of
  - A **very general** definition of computational procedures.
  - A demonstration of the fact that such computational procedures can be expressed in first order logic.
  - A demonstration of the fact that the halting problem for computational procedures is undecidable (see the above sketch).
  - A formulation of the halting problem in first order logic.
- This formal proof was provided by Alan Turing in [1]. The computational procedures he defined for this purpose were later called Turing machines.



A Turing Machine

References

[1] A.M. Turing. On computable real numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, 2(42):230–265, 1936.

Logic in Action

Chapter 5: Logic, Information and Knowledge

Jan van Eijck

http://www.logicinaction.org/

ESSLLI, 3-8-2011

Abstract

Today's lecture deals with the logic of knowledge as based on information, including changes in knowledge which result from observations of facts, or communication between agents knowing different things. This area is called epistemic logic, and its main difference with the earlier systems of Chapters 2, 3 and 4 is that we can also express facts about knowledge of one or more agents in the logical language itself. This 'social' perspective occurs in many settings: knowing what others do or do not know determines our actions. Another central theme of this chapter is "change": successive information processing steps change what agents know, and this, too, is essential to understanding the logic of language use and other cognitive tasks.

Observation, inference and communication

Someone is standing next to a room and sees a white object outside. Now another person tells her that there is an object inside the room of the same colour as the one outside.

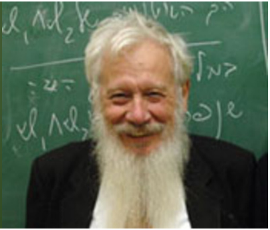
After all this, the first person reasons and get to know that there is a white object inside the room.

This is based on three actions: an **observation**, then an act of **communication**, and finally an **inference** putting things together.

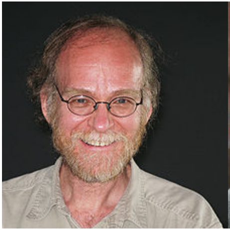
Very Brief History

David Lewis

Jaakko Hintikka



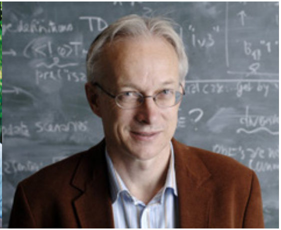
Robert Aumann



Joe Halpern

Jan Plaza

A. Baltag



Johan van Benthem

Muddy Children

picture by
Marco Swaen

The Muddy Children Puzzle

a clean, *b*, *c* and *d* muddy.

	a	b	c	d
at least one of you is muddy	○	●	●	●
who knows his state?	N	N	N	N
who knows his state now?	N	N	N	N
who knows his state now?	N	Y	Y	Y
who knows his state now?	Y			

The Muddy Children (2)

a, *b*, *c* clean, *d* muddy.

	a	b	c	d
at least one of you is muddy	∘	∘	∘	•
who knows his state?	N	N	N	Y
who knows his state now?	Y	Y	Y	

The Muddy Children (3)

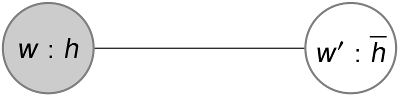
a, *b* clean, *c*, *d* muddy.

	a	b	c	d
at least one of you is muddy	○	○	●	●
who knows his state?	N	N	N	N
who knows his state now?	N	N	Y	Y
who knows his state now?	Y	Y		

Individual Ignorance

You have to finish a paper, and you are faced with a choice: do it today, or put it off until tomorrow.

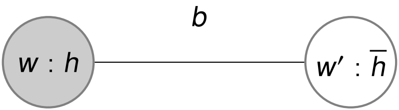
Result of coin flip under a cup:



Multi Agent Ignorance

Suppose Alice and Bob are present, and Alice tosses a coin under a cup.

The result of a hidden coin toss with the coin heads up:



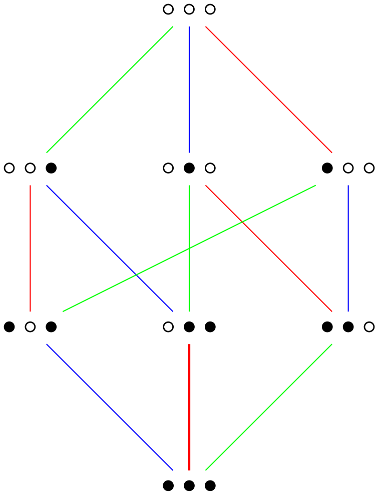
Alice is taking a look under the cup, while Bob is watching.

Now Alice knows the outcome.

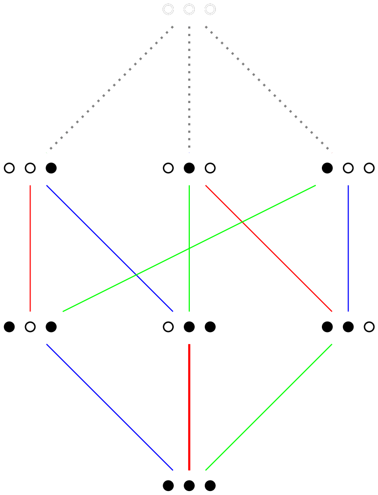
Bob knows that Alice knows the outcome.

Bob does not know the outcome himself.

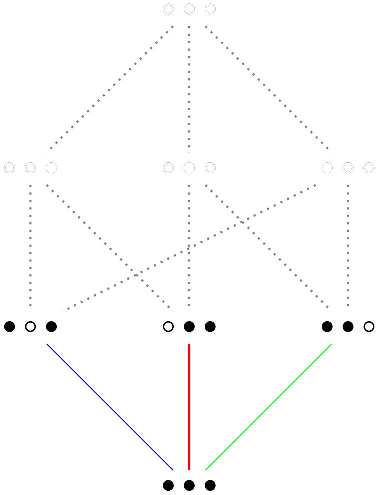
Back to the Children



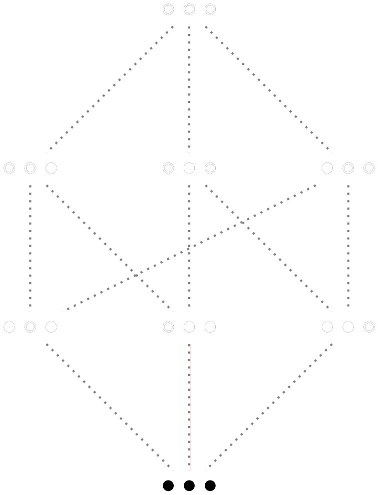
Back to the Children



Back to the Children

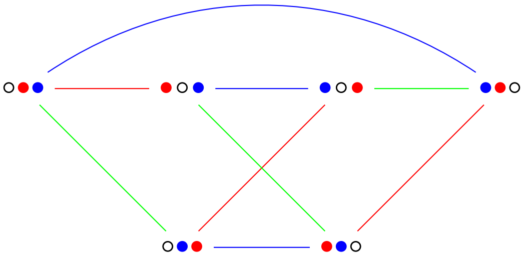


Back to the Children

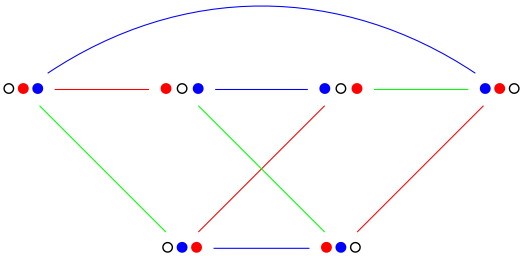


Epistemic Situations: Card Deals

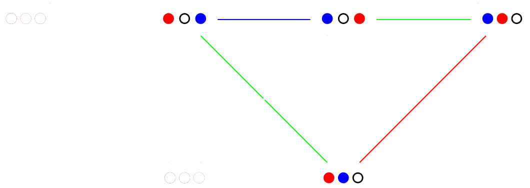
Alice, Bob and Carol, each draw a card from a stack of three cards. They know that the cards are red, white and blue. They cannot see the cards of the others.



Alice says: “I do not have white”



Alice says: “I do not have white”



The language

Let $\mathtt{P}$ be a set of atomic propositions and $\mathtt{N}$ a set of agents.

The **epistemic logic language** is built via the following rules.

1. Every basic proposition is in the language:

$$p, q, r, \ldots$$

2. If φ and ψ are formulas, then the following are formulas:

$$\neg\varphi, \quad \varphi \wedge \psi, \quad \varphi \vee \psi, \quad \varphi \rightarrow \psi, \quad \varphi \leftrightarrow \psi$$

3. If φ is a formula and i is an agent in $\mathtt{N}$, then the following are formulas:

$$\Box_i \varphi, \quad \Diamond_i \varphi.$$

Examples

- **A**lice knows that it is raining.

$$\Box_a r$$

- **B**ob knows whether it is raining.

$$\Box_b r \ \vee \ \Box_b \neg r$$

- **A**lice does not know whether it is raining.

$$\neg\Box_a r \ \wedge \ \neg\Box_a \neg r$$

- **A**lice does not know that it is raining, and actually it is not raining.

$$\neg\Box_a r \ \wedge \ \neg r$$

- **A**lice knows that **B**ob knows whether it is raining but she does not know it herself.

$$\Box_a(\Box_b r \ \vee \ \Box_b \neg r) \wedge (\neg\Box_a r \ \wedge \ \neg\Box_a \neg r)$$

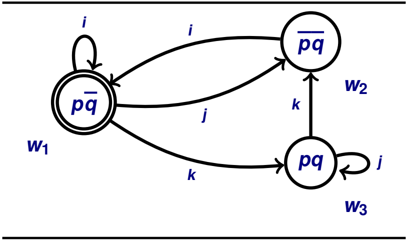
For Practice

1. Alice knows that it is raining.
2. Bob knows whether it is raining.
3. Alice knows that Bob knows whether it is raining, but she does not know it herself.
4. Bob considers raining possible.
5. Alice does not know that it is raining, and actually it is not raining.
6. Bob considers it possible that it is raining, but in fact it is not raining.
7. Alice knows that if it is raining, the floor will be wet.
8. If Alice knows that if it is raining the floor will be wet, and she also knows that it is raining, then she knows that the floor is wet.
9. Alice considers possible that Bob knows that it is raining.
10. Bob does not know that Alice knows that he knows whether it is raining.

The models

The structures in which we evaluate modal formulas, **relational structures**, have three components:

- a non-empty set W of **situations** or **worlds** (with a distinguished one),
- a **valuation function**, V, indicating which atomic propositions are true in each world $w \in W$, and
- an **accessibility** relation R_i for each agent i.



$$M = \langle W, R_i, V \rangle$$

Properties of the relation

Each **accessibility** relation R may have some special properties.

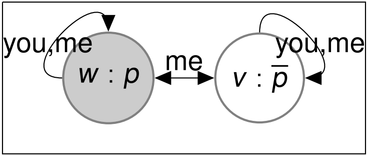
- **Reflexivity.** For all worlds w, Rww.
- **Symmetry.** For all worlds w and v, if Rwv then Rvw.
- **Transitivity.** For all worlds w, v and u, if Rwv and Rvu then Rwu.
- **Equivalence.** If it is reflexive, transitive and symmetric.
- **Euclidity.** For all worlds w, v and u, if Rwv and Rwu then Rvu.

Deciding truth-value of formulas

Take a relational structure $M = \langle W, R_i, V \rangle$, and pick a world $w \in W$:

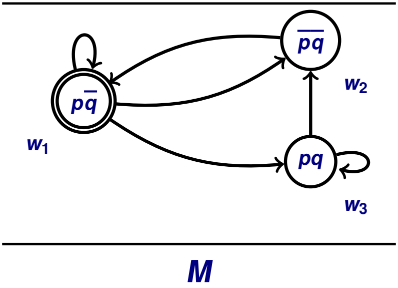
$(M, w) \models p$	if and only if	p is *true* at w
$(M, w) \models \neg\varphi$	if and only if	it is not the case that $(M, w) \models \varphi$
$(M, w) \models \varphi \vee \psi$	if and only if	$(M, w) \models \varphi$ or $(M, w) \models \psi$
...	if and only if	...
$(M, w) \models \Box_i \varphi$	if and only if	for all $u \in W$, if $R_i wu$ then $(M, u) \models \varphi$
$(M, w) \models \Diamond_i \varphi$	if and only if	for some $u \in W$ it holds that $R_i wu$ and $(M, u) \models \varphi$

Model Checking for Epistemic Logic



Formula	Worlds where true
p	w
$\neg p$	v
$\Box_{\text{you}} p$	w
$\Box_{\text{me}} p$	none
$\neg \Box_{\text{me}} p$	w, v
$\Box_{you} \neg p$	v
$\Box_{\text{me}} \neg p$	none
$\neg \Box_{\text{me}} \neg p$	w, v
$\neg \Box_{\text{me}} p \wedge \neg \Box_{\text{me}} \neg p$	w, v

For practice (1)

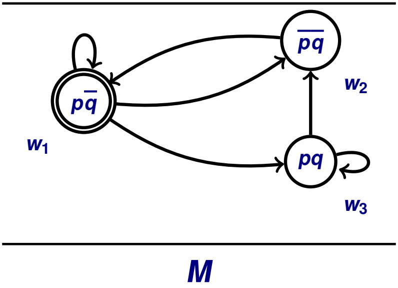


$(M, w_1) \models \Diamond\neg p$? $(M, w_2) \models \Diamond\neg p$? $(M, w_3) \models \Diamond\neg p$?

$(M, w_1) \models \Box(p \leftrightarrow q)$? $(M, w_2) \models \Box(p \leftrightarrow q)$? $(M, w_3) \models \Box(p \leftrightarrow q)$?

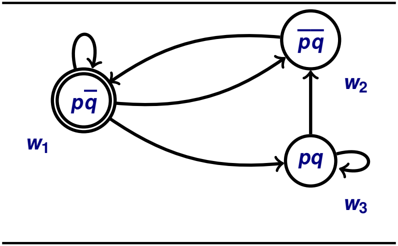
$(M, w_1) \models p \vee \Box p$? $(M, w_2) \models p \vee \Box p$? $(M, w_3) \models p \vee \Box p$?

For practice (1)



$(M, w_1) \models \Diamond \neg p$ √	$(M, w_2) \models \Diamond \neg p$?	$(M, w_3) \models \Diamond \neg p$?
$(M, w_1) \models \Box (p \leftrightarrow q)$?	$(M, w_2) \models \Box (p \leftrightarrow q)$?	$(M, w_3) \models \Box (p \leftrightarrow q)$?
$(M, w_1) \models p \vee \Box p$?	$(M, w_2) \models p \vee \Box p$?	$(M, w_3) \models p \vee \Box p$?

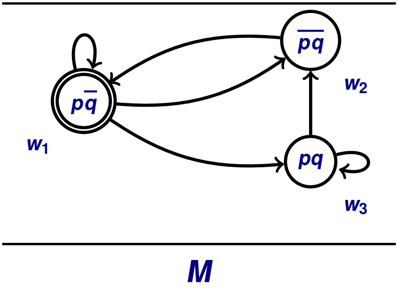
For practice (1)



M

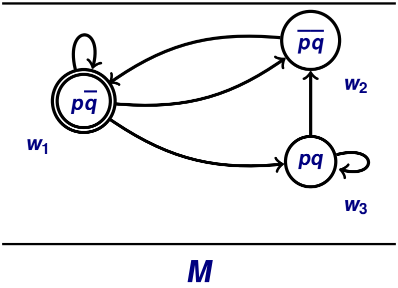
$(M, w_1) \models \Diamond \neg p$	√	$(M, w_2) \models \Diamond \neg p$	?	$(M, w_3) \models \Diamond \neg p$	?
$(M, w_1) \models \Box (p \leftrightarrow q)$	×	$(M, w_2) \models \Box (p \leftrightarrow q)$	?	$(M, w_3) \models \Box (p \leftrightarrow q)$	?
$(M, w_1) \models p \vee \Box p$	?	$(M, w_2) \models p \vee \Box p$	?	$(M, w_3) \models p \vee \Box p$	?

For practice (1)



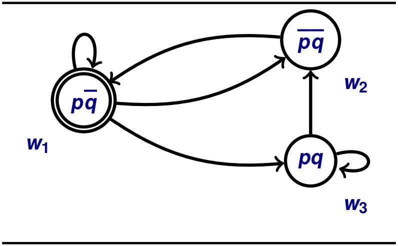
$(M, w_1) \models \Diamond \neg p$	$\surd$	$(M, w_2) \models \Diamond \neg p$	?	$(M, w_3) \models \Diamond \neg p$	?
$(M, w_1) \models \Box (p \leftrightarrow q)$	$\times$	$(M, w_2) \models \Box (p \leftrightarrow q)$	?	$(M, w_3) \models \Box (p \leftrightarrow q)$	?
$(M, w_1) \models p \vee \Box p$	$\surd$	$(M, w_2) \models p \vee \Box p$	?	$(M, w_3) \models p \vee \Box p$	?

For practice (1)



$(M, w_1) \models \Diamond \neg p$ √	$(M, w_2) \models \Diamond \neg p$ ×	$(M, w_3) \models \Diamond \neg p$?
$(M, w_1) \models \Box (p \leftrightarrow q)$ ×	$(M, w_2) \models \Box (p \leftrightarrow q)$?	$(M, w_3) \models \Box (p \leftrightarrow q)$?
$(M, w_1) \models p \vee \Box p$ √	$(M, w_2) \models p \vee \Box p$?	$(M, w_3) \models p \vee \Box p$?

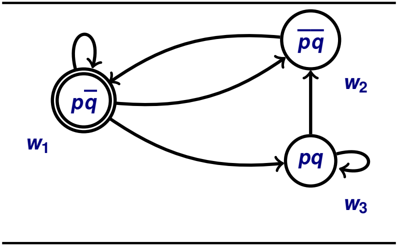
For practice (1)



M

$(M, w_1) \models \Diamond \neg p$	√	$(M, w_2) \models \Diamond \neg p$	×	$(M, w_3) \models \Diamond \neg p$	?
$(M, w_1) \models \Box (p \leftrightarrow q)$	×	$(M, w_2) \models \Box (p \leftrightarrow q)$	×	$(M, w_3) \models \Box (p \leftrightarrow q)$	?
$(M, w_1) \models p \vee \Box p$	√	$(M, w_2) \models p \vee \Box p$	?	$(M, w_3) \models p \vee \Box p$	?

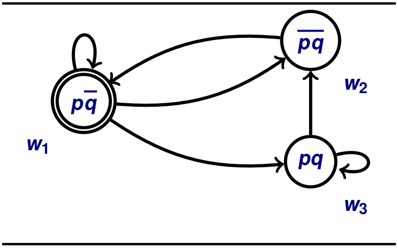
For practice (1)



M

$(M, w_1) \models \Diamond \neg p$	√	$(M, w_2) \models \Diamond \neg p$	×	$(M, w_3) \models \Diamond \neg p$	?
$(M, w_1) \models \Box (p \leftrightarrow q)$	×	$(M, w_2) \models \Box (p \leftrightarrow q)$	×	$(M, w_3) \models \Box (p \leftrightarrow q)$	?
$(M, w_1) \models p \vee \Box p$	√	$(M, w_2) \models p \vee \Box p$	√	$(M, w_3) \models p \vee \Box p$	?

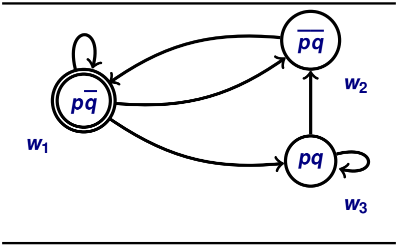
For practice (1)



M

$(M, w_1) \models \Diamond \neg p$ √	$(M, w_2) \models \Diamond \neg p$ ×	$(M, w_3) \models \Diamond \neg p$ √
$(M, w_1) \models \Box (p \leftrightarrow q)$ ×	$(M, w_2) \models \Box (p \leftrightarrow q)$ ×	$(M, w_3) \models \Box (p \leftrightarrow q)$?
$(M, w_1) \models p \vee \Box p$ √	$(M, w_2) \models p \vee \Box p$ √	$(M, w_3) \models p \vee \Box p$?

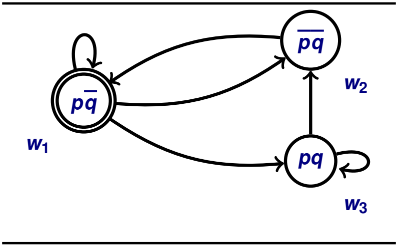
For practice (1)



M

$(M, w_1) \models \Diamond \neg p$	$\surd$	$(M, w_2) \models \Diamond \neg p$	$\times$	$(M, w_3) \models \Diamond \neg p$	$\surd$
$(M, w_1) \models \Box (p \leftrightarrow q)$	$\times$	$(M, w_2) \models \Box (p \leftrightarrow q)$	$\times$	$(M, w_3) \models \Box (p \leftrightarrow q)$	$\surd$
$(M, w_1) \models p \vee \Box p$	$\surd$	$(M, w_2) \models p \vee \Box p$	$\surd$	$(M, w_3) \models p \vee \Box p$	?

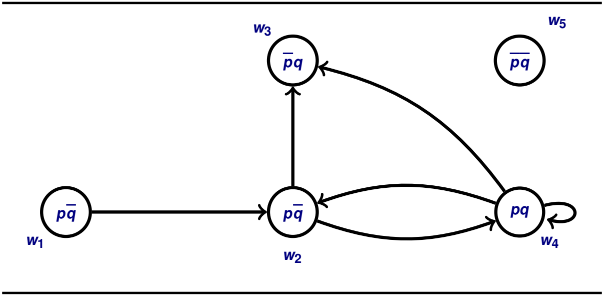
For practice (1)



M

$(M, w_1) \models \Diamond \neg p$	√	$(M, w_2) \models \Diamond \neg p$	×	$(M, w_3) \models \Diamond \neg p$	√
$(M, w_1) \models \Box (p \leftrightarrow q)$	×	$(M, w_2) \models \Box (p \leftrightarrow q)$	×	$(M, w_3) \models \Box (p \leftrightarrow q)$	√
$(M, w_1) \models p \lor \Box p$	√	$(M, w_2) \models p \lor \Box p$	√	$(M, w_3) \models p \lor \Box p$	√

For practice (2)

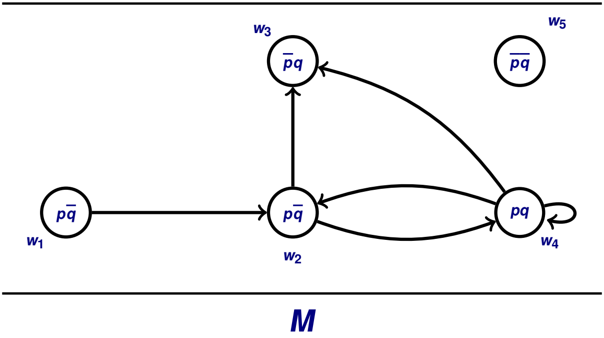


M

Indicate the worlds in which the following formulas are true.

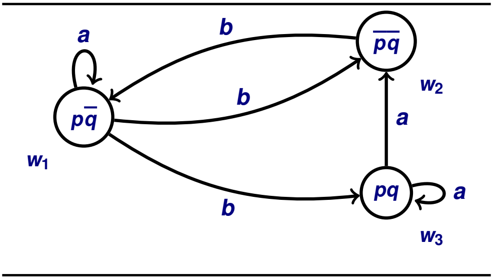
$\Diamond q$	$\{w_2, w_4\}$	$\Box p$	$\{w_1, w_3, w_5\}$
$\Box p \to p$	$\{w_1, w_2, w_4\}$	$\Diamond\Diamond p \to \Diamond p$	$\{w_1, w_2, w_3, w_4, w_5\}$
$q \to \Box\Diamond q$	$\{w_1, w_2, w_3, w_5\}$	$\Diamond\Box p \to \Box\Diamond p$	$\{w_1, w_3, w_5\}$
$\Diamond(p \to q)$	$\{w_2, w_4\}$	$\Diamond(\neg p \wedge \neg q)$	$\{\ \}$

For practice (3)



For each world in the model, provide a formula that is true only in that world and false in all the others.

Multiple relations



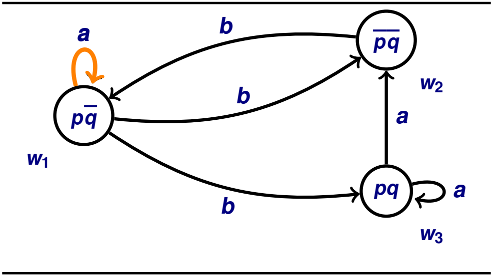
M

$(M, w_1) \models \Diamond_a \neg p$? $(M, w_2) \models \Diamond_a \neg p$? $(M, w_3) \models \Diamond_a \neg p$?

$(M, w_1) \models \Box_b (p \leftrightarrow q)$? $(M, w_2) \models \Box_b (p \leftrightarrow q)$? $(M, w_3) \models \Box_b (p \leftrightarrow q)$?

$(M, w_1) \models \Box_b\, p \vee \Diamond_a\, q$? $(M, w_2) \models \Box_b\, p \vee \Diamond_a\, q$? $(M, w_3) \models \Box_b\, p \vee \Diamond_a\, q$?

Multiple relations



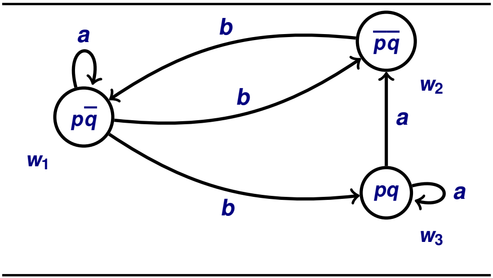
M

$(M, w_1) \models \Diamond_a \neg p$? $(M, w_2) \models \Diamond_a \neg p$? $(M, w_3) \models \Diamond_a \neg p$?

$(M, w_1) \models \Box_b (p \leftrightarrow q)$? $(M, w_2) \models \Box_b (p \leftrightarrow q)$? $(M, w_3) \models \Box_b (p \leftrightarrow q)$?

$(M, w_1) \models \Box_b p \vee \Diamond_a q$? $(M, w_2) \models \Box_b p \vee \Diamond_a q$? $(M, w_3) \models \Box_b p \vee \Diamond_a q$?

Multiple relations



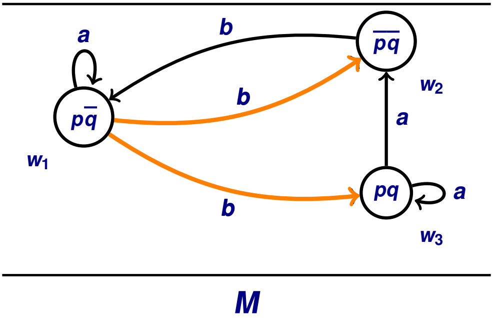
M

$(M, w_1) \models \Diamond_a \neg p$ × $(M, w_2) \models \Diamond_a \neg p$? $(M, w_3) \models \Diamond_a \neg p$?

$(M, w_1) \models \Box_b (p \leftrightarrow q)$? $(M, w_2) \models \Box_b (p \leftrightarrow q)$? $(M, w_3) \models \Box_b (p \leftrightarrow q)$?

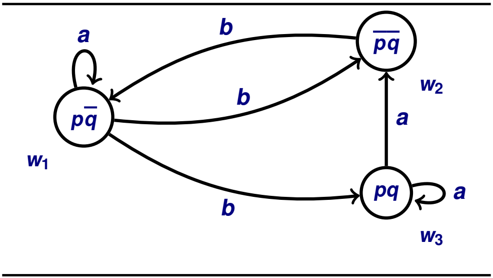
$(M, w_1) \models \Box_b\, p \vee \Diamond_a\, q$? $(M, w_2) \models \Box_b\, p \vee \Diamond_a\, q$? $(M, w_3) \models \Box_b\, p \vee \Diamond_a\, q$?

Multiple relations



$(M, w_1) \models \Diamond_a \neg p$ ×	$(M, w_2) \models \Diamond_a \neg p$?	$(M, w_3) \models \Diamond_a \neg p$?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$?	$(M, w_2) \models \Box_b (p \leftrightarrow q)$?	$(M, w_3) \models \Box_b (p \leftrightarrow q)$?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$?	$(M, w_2) \models \Box_b p \vee \Diamond_a q$?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$?

Multiple relations



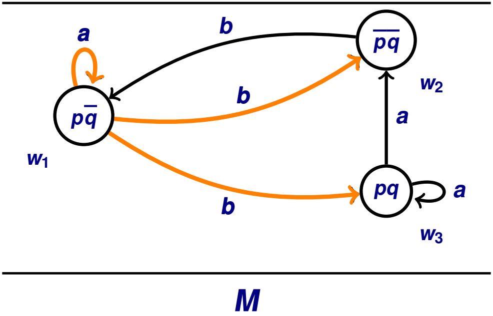
M

$(M, w_1) \models \Diamond_a \neg p$ × $(M, w_2) \models \Diamond_a \neg p$? $(M, w_3) \models \Diamond_a \neg p$?

$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √ $(M, w_2) \models \Box_b (p \leftrightarrow q)$? $(M, w_3) \models \Box_b (p \leftrightarrow q)$?

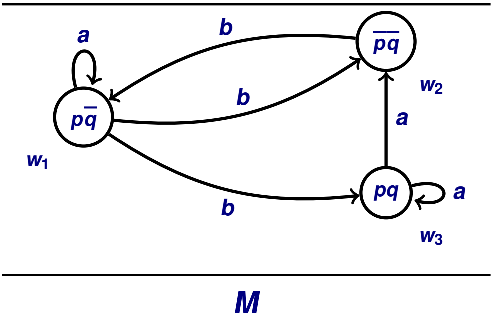
$(M, w_1) \models \Box_b p \vee \Diamond_a q$? $(M, w_2) \models \Box_b p \vee \Diamond_a q$? $(M, w_3) \models \Box_b p \vee \Diamond_a q$?

Multiple relations



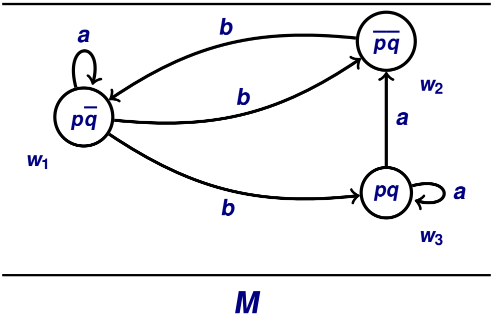
$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	?	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	?	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations



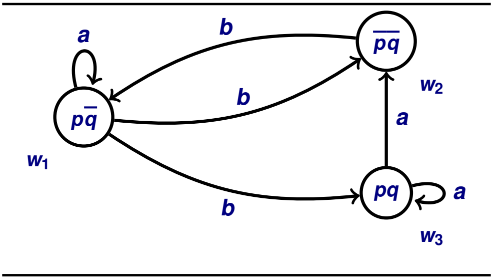
$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	?	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	?	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations



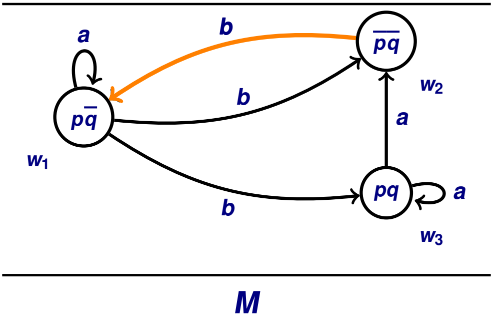
$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	?	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	?	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations



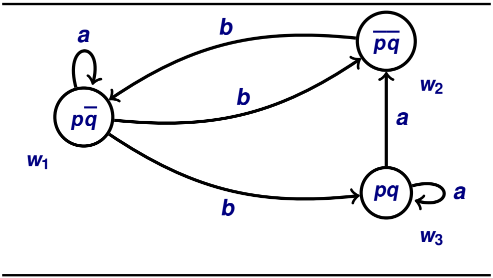
$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	×	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	?	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations



$(M, w_1) \models \Diamond_a \neg p$ ×	$(M, w_2) \models \Diamond_a \neg p$ ×	$(M, w_3) \models \Diamond_a \neg p$?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √	$(M, w_2) \models \Box_b (p \leftrightarrow q)$?	$(M, w_3) \models \Box_b (p \leftrightarrow q)$?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$ ×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$?

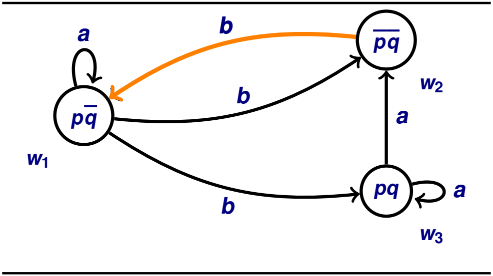
Multiple relations



M

$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	×	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	×	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

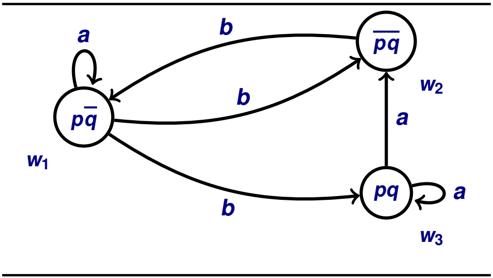
Multiple relations



M

$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	×	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	×	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	?	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations



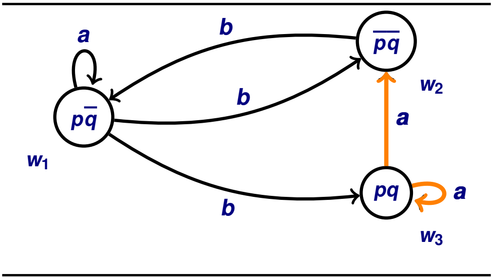
M

$(M, w_1) \models \Diamond_a \neg p$ × $(M, w_2) \models \Diamond_a \neg p$ × $(M, w_3) \models \Diamond_a \neg p$?

$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √ $(M, w_2) \models \Box_b (p \leftrightarrow q)$ × $(M, w_3) \models \Box_b (p \leftrightarrow q)$?

$(M, w_1) \models \Box_b p \vee \Diamond_a q$ × $(M, w_2) \models \Box_b p \vee \Diamond_a q$ √ $(M, w_3) \models \Box_b p \vee \Diamond_a q$?

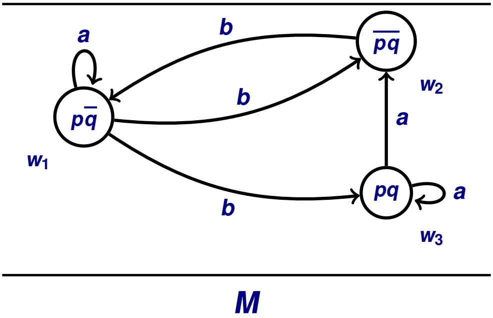
Multiple relations



M

$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	×	$(M, w_3) \models \Diamond_a \neg p$	?
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	×	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	√	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations

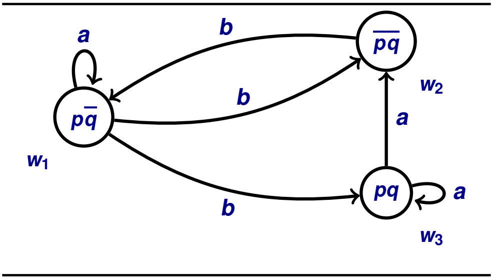


$(M, w_1) \models \Diamond_a \neg p$ × $(M, w_2) \models \Diamond_a \neg p$ × $(M, w_3) \models \Diamond_a \neg p$ √

$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √ $(M, w_2) \models \Box_b (p \leftrightarrow q)$ × $(M, w_3) \models \Box_b (p \leftrightarrow q)$?

$(M, w_1) \models \Box_b\, p \vee \Diamond_a\, q$ × $(M, w_2) \models \Box_b\, p \vee \Diamond_a\, q$ √ $(M, w_3) \models \Box_b\, p \vee \Diamond_a\, q$?

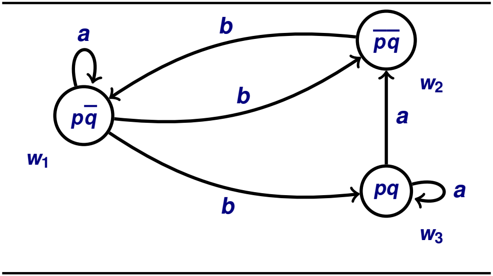
Multiple relations



M

$(M, w_1) \models \Diamond_a \neg p$ ×	$(M, w_2) \models \Diamond_a \neg p$ ×	$(M, w_3) \models \Diamond_a \neg p$ √
$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √	$(M, w_2) \models \Box_b (p \leftrightarrow q)$ ×	$(M, w_3) \models \Box_b (p \leftrightarrow q)$?
$(M, w_1) \models \Box_b p \vee \Diamond_a q$ ×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$ √	$(M, w_3) \models \Box_b p \vee \Diamond_a q$?

Multiple relations



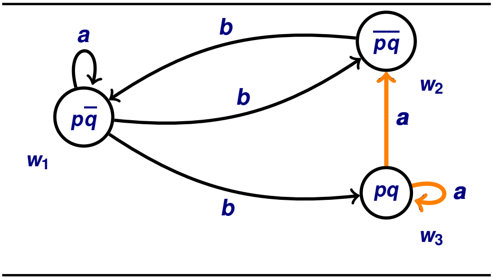
M

$(M, w_1) \models \Diamond_a \neg p$ × $(M, w_2) \models \Diamond_a \neg p$ × $(M, w_3) \models \Diamond_a \neg p$ √

$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √ $(M, w_2) \models \Box_b (p \leftrightarrow q)$ × $(M, w_3) \models \Box_b (p \leftrightarrow q)$ √

$(M, w_1) \models \Box_b\, p \vee \Diamond_a\, q$ × $(M, w_2) \models \Box_b\, p \vee \Diamond_a\, q$ √ $(M, w_3) \models \Box_b\, p \vee \Diamond_a\, q$?

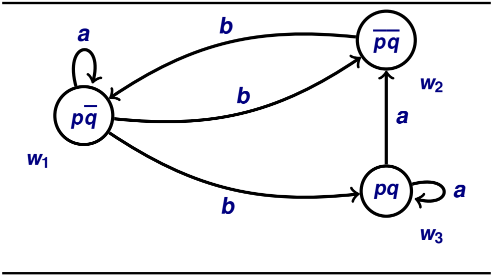
Multiple relations



M

$(M, w_1) \models \Diamond_a \neg p$	×	$(M, w_2) \models \Diamond_a \neg p$	×	$(M, w_3) \models \Diamond_a \neg p$	√
$(M, w_1) \models \Box_b (p \leftrightarrow q)$	√	$(M, w_2) \models \Box_b (p \leftrightarrow q)$	×	$(M, w_3) \models \Box_b (p \leftrightarrow q)$	√
$(M, w_1) \models \Box_b p \vee \Diamond_a q$	×	$(M, w_2) \models \Box_b p \vee \Diamond_a q$	√	$(M, w_3) \models \Box_b p \vee \Diamond_a q$	?

Multiple relations



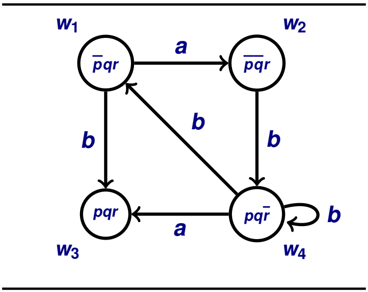
M

$(M, w_1) \models \Diamond_a \neg p$ × $(M, w_2) \models \Diamond_a \neg p$ × $(M, w_3) \models \Diamond_a \neg p$ √

$(M, w_1) \models \Box_b (p \leftrightarrow q)$ √ $(M, w_2) \models \Box_b (p \leftrightarrow q)$ × $(M, w_3) \models \Box_b (p \leftrightarrow q)$ √

$(M, w_1) \models \Box_b p \vee \Diamond_a q$ × $(M, w_2) \models \Box_b p \vee \Diamond_a q$ √ $(M, w_3) \models \Box_b p \vee \Diamond_a q$ √

For practice



M

Indicate the worlds in which the following formulas are true.

$\Diamond_a \Diamond_b p$	$\{w_1\}$	$\Box_a \Box_b r$	$\{w_2, w_3, w_4\}$
$p \land \Box_b (q \land \Box_a r)$	$\{w_3, w_4\}$	$r \rightarrow \Box_a q$	$\{w_2, w_3, w_4\}$
$\Box_a (q \rightarrow \Diamond_a r)$	$\{w_1, w_2, w_3\}$	$\Diamond_a p \leftrightarrow \Diamond_b q$	$\{w_3, w_4\}$
$\neg \Box_b r$	$\{w_2, w_4\}$	$\Diamond_b p \rightarrow \Box_a r$	$\{w_1, w_2, w_3, w_4\}$

Validities (1)

Some interesting validities:

$$\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$$

$$\Diamond\varphi \leftrightarrow \neg\Box\neg\varphi \qquad \Box\varphi \leftrightarrow \neg\Diamond\neg\varphi$$

$$\Diamond(\varphi \lor \psi) \leftrightarrow (\Diamond\varphi \lor \Diamond\psi) \qquad \Box(\varphi \land \psi) \leftrightarrow (\Box\varphi \land \Box\psi)$$

Validities (2)

Some validities with requirements:

- If we work only with models in which R is **reflexive**, then the following formula, the **veridicality** principle, is valid:

$$\Box\varphi \rightarrow \varphi$$

- If we work only with models in which R is **transitive**, then the following formula, the **positive introspection** principle, is valid:

$$\Box\varphi \rightarrow \Box\Box\varphi$$

- If we work only with models in which R is **symmetric**, then the following formula is valid:

$$\varphi \rightarrow \Box\Diamond\varphi$$

- If we work only with models in which R is **euclidean**, then the following formula, the **negative introspection** principle, is valid:

$$\neg\Box\varphi \rightarrow \Box\neg\Box\varphi$$

The K system

The valid formulas of epistemic logic can be derived from the following principles:

1. All propositional tautologies.
2. $\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$
3. **Modus ponens** (MP): from φ and $\varphi \to \psi$, infer ψ.
4. **Necessitation** (Nec): from φ infer $\Box\varphi$.

A formula that can be derived by following these principles in a *finite* number of steps is called a **theorem**.

Example

Prove that $\varphi \to \psi$ implies $\Box\varphi \to \Box\psi$

1.	$\varphi \to \psi$	Assumption
2.	$\Box(\varphi \to \psi)$	Nec from step 1
3.	$\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$	Axiom 2
4.	$\Box\varphi \to \Box\psi$	MP from steps 2 and 3

More systems

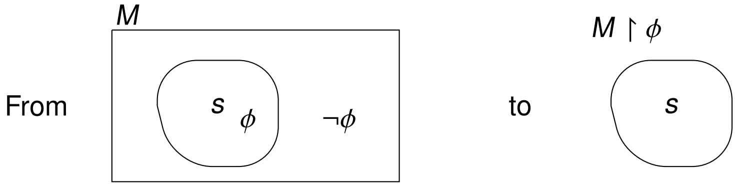
T := K + *veridicality* ($\Box\varphi \rightarrow \varphi$)

$S4$:= T + *positive introspection* ($\Box\varphi \rightarrow \Box\Box\varphi$)

$S5$:= $S4$ + $\varphi \rightarrow \Box\Diamond\varphi$

$S4$ + *negative introspection* ($\neg\Box\varphi \rightarrow \Box\neg\Box\varphi$)

In a Picture



Public Announcements

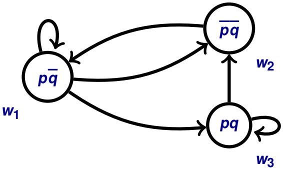
Jan Plaza

Effect of a public announcement ϕ: the domain gets restricted to situations where ϕ is true.

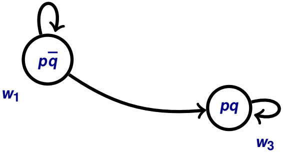
The intuition

An **update** with φ **eliminates** situations where φ is false.

If we have a model $M = \langle W, R, V \rangle$



then updating with p turns the model into $M|_p = \langle W', R', V' \rangle$



Formally,

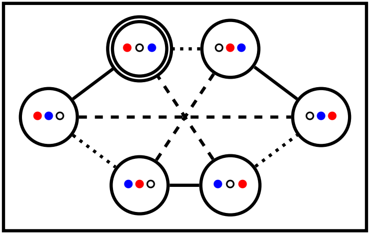
Take a model $\boldsymbol{M} = \langle \boldsymbol{W}, \boldsymbol{R_i}, \boldsymbol{V} \rangle$ and a formula $\boldsymbol{\varphi}$.

The model $\boldsymbol{M}|_{\varphi} = \langle \boldsymbol{W'}, \boldsymbol{R'_i}, \boldsymbol{V'} \rangle$, $\boldsymbol{M}$ **relativized to** $\boldsymbol{\varphi}$, is given by:

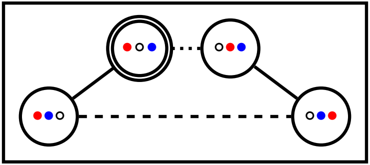
$$\begin{aligned} \boldsymbol{W'} &:= \{\boldsymbol{w} \in \boldsymbol{W} \mid (\boldsymbol{M}, \boldsymbol{w}) \models \boldsymbol{\varphi}\}. \\ \boldsymbol{R'_i} &:= \boldsymbol{R_i} \cap (\boldsymbol{W'} \times \boldsymbol{W'}). \\ \boldsymbol{V'}(\boldsymbol{w}) &:= \boldsymbol{V}(\boldsymbol{w}). \end{aligned}$$

Example

Everybody knows their own card:



Then **1** announces publicly: **"I do not have the blue card!"** ($\neg b_1$).



Syntax

We introduce new formulas to talk about the effect of public announcements:

$[!\varphi]\,\psi$ "If φ can be announced, then after doing it ψ is the case".

$\langle !\varphi\rangle\,\psi$ "φ can be announced, and after doing it ψ is the case".

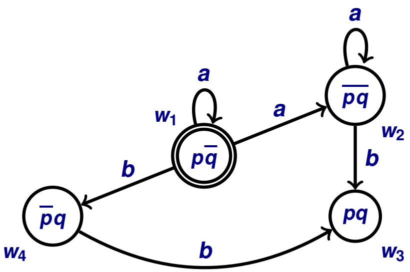
More precisely,

$$(M, w) \models [!\varphi]\,\psi \quad \text{iff} \quad (M, w) \models \varphi \ \text{ implies } \ (M|_\varphi, w) \models \psi$$
$$(M, w) \models \langle !\varphi\rangle\,\psi \quad \text{iff} \quad (M, w) \models \varphi \ \text{ and } \ (M|_\varphi, w) \models \psi$$

Card deal example:

$$[!\neg b_2][!\neg b_1]\Box_1(\neg\Box_3 w_1 \wedge \neg\Box_3 r_1).$$

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$?

$(M, w_1) \models [!q]\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !\neg q \rangle \Diamond_b\, q$?

$(M, w_1) \models [!\Diamond_b \neg p]\, \Box_a\, p$?

$(M, w_1) \models p \rightarrow [!p]\, p$?

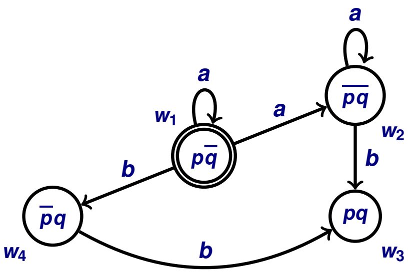
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !(p \vee q) \rangle\, \Box_a\, p$?

$(M, w_1) \models \langle !\Box_a \neg q \rangle\, \neg q$?

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !\neg q \rangle \Diamond_b\, q$?

$(M, w_1) \models [!\Diamond_b \neg p]\,\Box_a\, p$?

$(M, w_1) \models p \rightarrow [!p]\, p$?

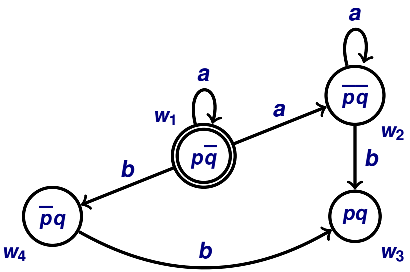
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !(p \vee q) \rangle\,\Box_a\, p$?

$(M, w_1) \models \langle !\Box_a \neg q \rangle\,\neg q$?

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !\neg q \rangle \Diamond_b\, q$?

$(M, w_1) \models [!\Diamond_b \neg p]\,\Box_a\, p$?

$(M, w_1) \models p \rightarrow [!p]\,p$?

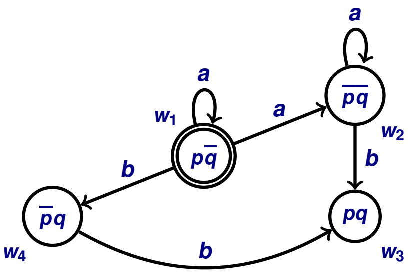
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !(p \vee q) \rangle\,\Box_a\, p$?

$(M, w_1) \models \langle !\Box_a \neg q \rangle\,\neg q$?

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$ √

$(M, w_1) \models \langle !\neg q \rangle \Diamond_b\, q$?

$(M, w_1) \models [!\Diamond_b \neg p]\, \Box_a\, p$?

$(M, w_1) \models p \rightarrow [!p]\, p$?

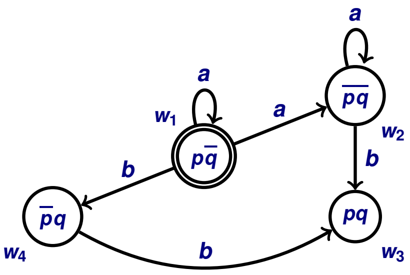
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$?

$(M, w_1) \models \langle !(p \vee q) \rangle\, \Box_a\, p$?

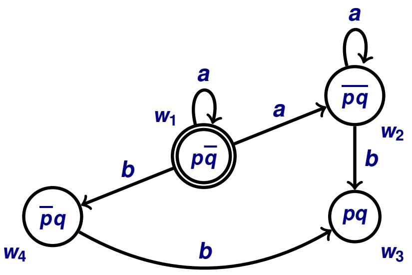
$(M, w_1) \models \langle !\Box_a \neg q \rangle\, \neg q$?

Examples



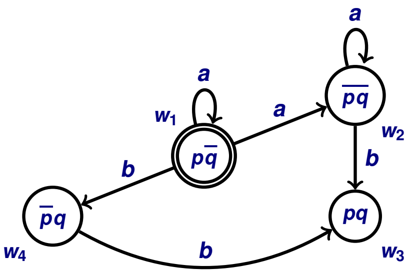
$(M, w_1) \models [!p]\,(q \wedge \neg q)$	×	$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$	×
$(M, w_1) \models [!q]\,(q \wedge \neg q)$	√	$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$	×
$(M, w_1) \models \langle !\neg q \rangle\, \Diamond_b\, q$	?	$(M, w_1) \models \langle !(p \vee q) \rangle\, \Box_a\, p$	?
$(M, w_1) \models [!\Diamond_b\, \neg p]\, \Box_a\, p$	?	$(M, w_1) \models \langle !\Box_a\, \neg q \rangle\, \neg q$	?
$(M, w_1) \models p \rightarrow [!p]\, p$	?		

Examples



$(M, w_1) \models [!p]\,(q \land \neg q)$ ×	$(M, w_1) \models \langle !p \rangle\,(q \land \neg q)$ ×
$(M, w_1) \models [!q]\,(q \land \neg q)$ √	$(M, w_1) \models \langle !q \rangle\,(q \land \neg q)$ ×
$(M, w_1) \models \langle !\neg q \rangle\, \Diamond_b\, q$ ×	$(M, w_1) \models \langle !(p \lor q) \rangle\, \Box_a\, p$?
$(M, w_1) \models [!\Diamond_b\, \neg p]\, \Box_a\, p$?	$(M, w_1) \models \langle !\Box_a\, \neg q \rangle\, \neg q$?
$(M, w_1) \models p \rightarrow [!p]\, p$?	

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$ √

$(M, w_1) \models \langle !\neg q \rangle\, \Diamond_b\, q$ ×

$(M, w_1) \models [!\Diamond_b \neg p]\, \Box_a\, p$?

$(M, w_1) \models p \rightarrow [!p]\, p$?

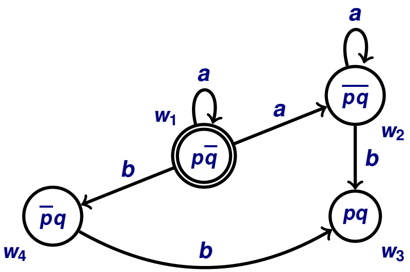
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !(p \vee q) \rangle\, \Box_a\, p$ √

$(M, w_1) \models \langle !\Box_a \neg q \rangle\, \neg q$?

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$ √

$(M, w_1) \models \langle !\neg q \rangle \Diamond_b\, q$ ×

$(M, w_1) \models [!\Diamond_b \neg p]\,\Box_a\, p$ √

$(M, w_1) \models p \rightarrow [!p]\, p$?

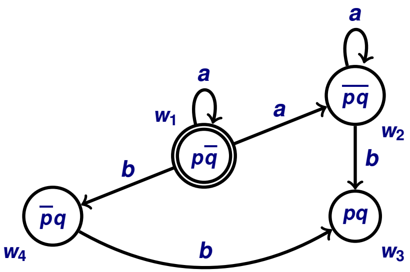
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !(p \vee q) \rangle\,\Box_a\, p$ √

$(M, w_1) \models \langle !\Box_a \neg q \rangle\,\neg q$?

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$ √

$(M, w_1) \models \langle !\neg q\rangle\, \Diamond_b\, q$ ×

$(M, w_1) \models [!\Diamond_b \neg p]\, \Box_a\, p$ √

$(M, w_1) \models p \rightarrow [!p]\, p$?

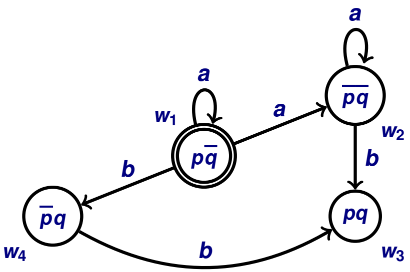
$(M, w_1) \models \langle !p\rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !q\rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !(p \vee q)\rangle\, \Box_a\, p$ √

$(M, w_1) \models \langle !\Box_a \neg q\rangle\, \neg q$ √

Examples



$(M, w_1) \models [!p]\,(q \wedge \neg q)$ ×

$(M, w_1) \models [!q]\,(q \wedge \neg q)$ √

$(M, w_1) \models \langle !\neg q \rangle \Diamond_b\, q$ ×

$(M, w_1) \models [!\Diamond_b \neg p]\, \Box_a\, p$ √

$(M, w_1) \models p \rightarrow [!p]\, p$ √

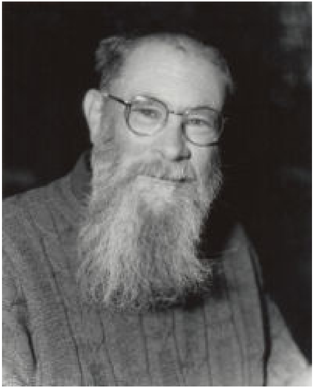
$(M, w_1) \models \langle !p \rangle\,(q \wedge \neg q)$ ×

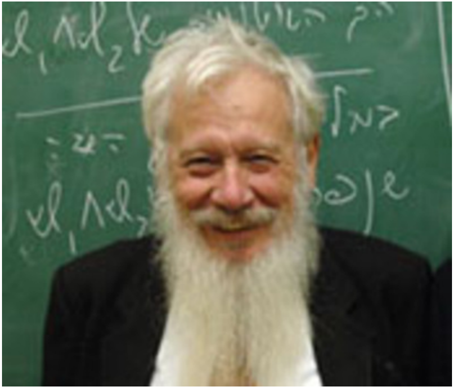
$(M, w_1) \models \langle !q \rangle\,(q \wedge \neg q)$ ×

$(M, w_1) \models \langle !(p \vee q) \rangle\, \Box_a\, p$ √

$(M, w_1) \models \langle !\Box_a \neg q \rangle\, \neg q$ √

The Emergence of Common Knowledge



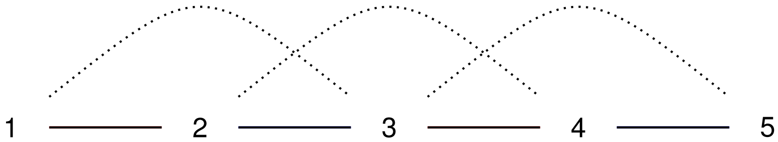


David Lewis   Robert Aumann

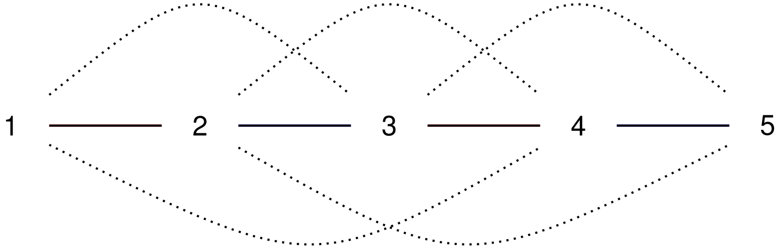
Computing the Common Knowledge Relation

Computing the Common Knowledge Relation

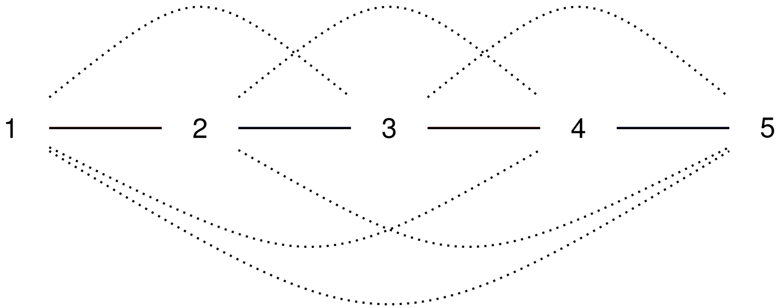
Computing the Common Knowledge Relation



Computing the Common Knowledge Relation



Computing the Common Knowledge Relation



Common Knowledge: Definition

ϕ is common knowledge if everyone knows that ϕ and, moreover, everyone knows that ϕ is common knowledge.

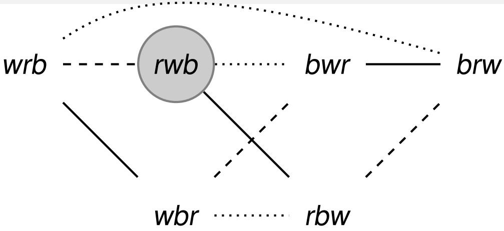
$$C\varphi \leftrightarrow (E\varphi \wedge EC\varphi).$$

Compare:

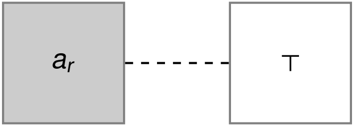
zeros = 0 : zeros

Cashiers, ATMs, and the Creation of Common Knowledge

Effect of Private Messages



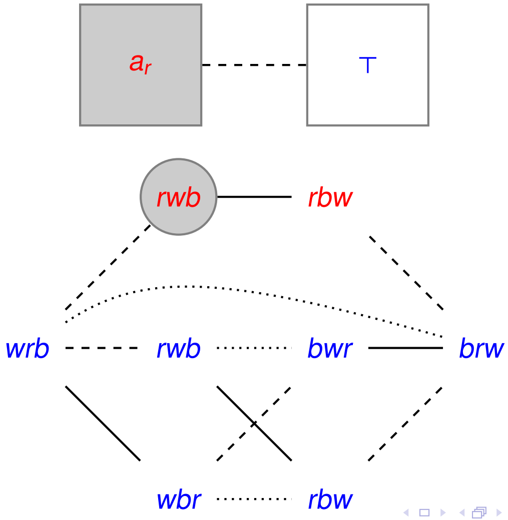
Alice says “I hold the red card” privately to Bob.



Carol cannot distinguish this from the action where nothing happens.

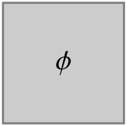
Effect of This

Compute the result with a model product construction (Baltag cs., [1]):

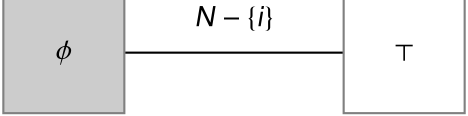


Sending Email Messages

“Wouter Bos email”: message where all can see the recipient list. This is like a public announcement.



Private message ϕ to agent i: all other agents cannot distinguish this from the action where nothing happens:



Epistemic Model Checking of Muddy Children

- mu0: model where the children cannot see each other.
- mu1: model where the children can see each other.
- mu2: model after public announcement “at least one of you is muddy.”
- mu3: model after public announcement “no-one knows their state.”
- mu4: model after public announcement “no-one knows their state.”
- mu5: model after public announcement “b, c, d know their state.”

`http://homepages.cwi.nl/~jve/software/demolight`

A. Baltag, L.S. Moss, and S. Solecki.
The logic of public announcements, common knowledge, and private suspicions.
In I. Bilboa, editor, Proceedings of TARK'98, pages 43–56, 1998.

Logic in Action

Chapter 6: Logic and Action

Jan van Eijck

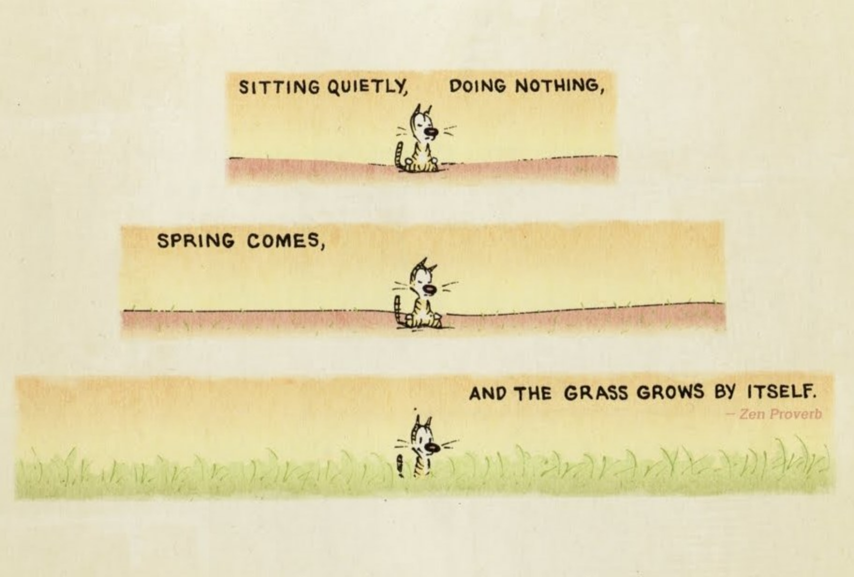
http://www.logicinaction.org/

ESSLLI, 4-8-2011

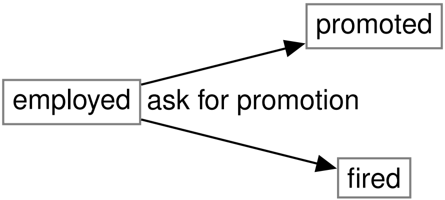
Abstract

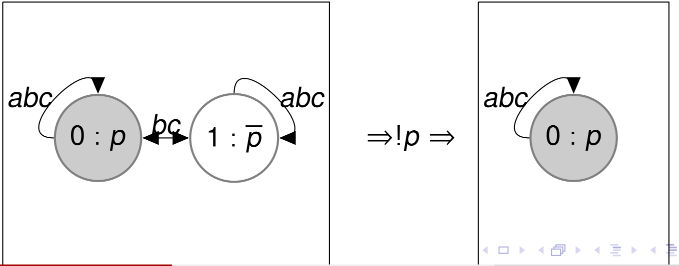
An action is something that takes place in the world, and that makes a difference to what the world looks like. Thus, actions are maps from states of the world to new states of the world. Actions can be of various kinds. The action of spilling coffee changes the state of your trousers. The action of telling a lie to your friend changes your friend's state of mind (and maybe the state of your soul). The action of multiplying two numbers changes the state of certain registers in your computer. Despite the differences between these various kinds of actions, we will see that they can all be covered under the same logical umbrella.

Sitting Quietly . . .



Actions in General



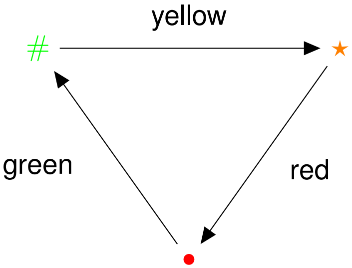


Computation as Action

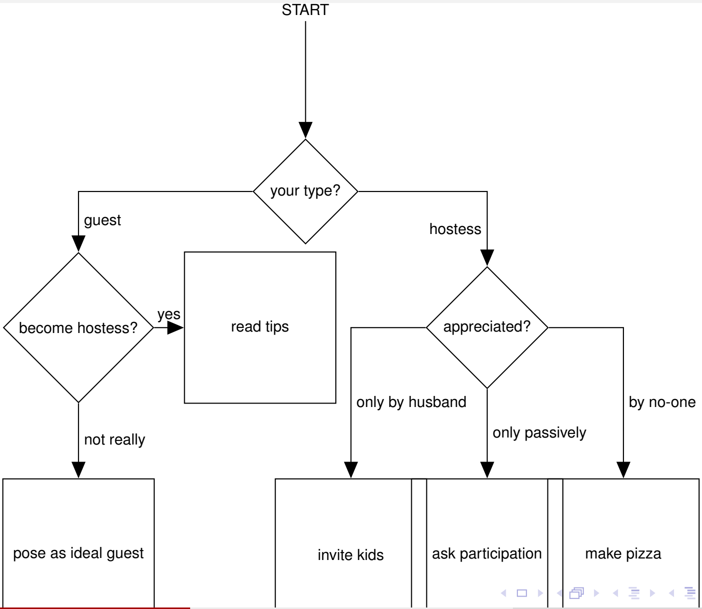
x	3
y	2
z	4

$x := y$ →

x	2
y	2
z	4



Happy Xmas Procedure



Operations over actions

Actions can be combined in several ways:

- **Sequence.** Execute one action after another:

  Pour the mixture over the potatoes, and then cover pan with foil.

- **Choice.** Choose between actions:

  Pick one of the boxes.

- **Repetition.** Perform the same action several times:

  Press the door until you hear a 'click'.

- **Test.** Verify whether a given condition holds:

  Check if the bulb is broken.

- **Converse.** Undo an executed action:

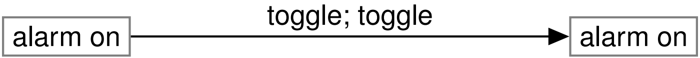
  Close the window you just opened.

Putting Actions Together

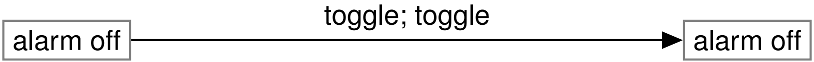
Sequence

Perform one action after another

Writing the sequence of two actions *a* and *b* as $a; b$, we get:

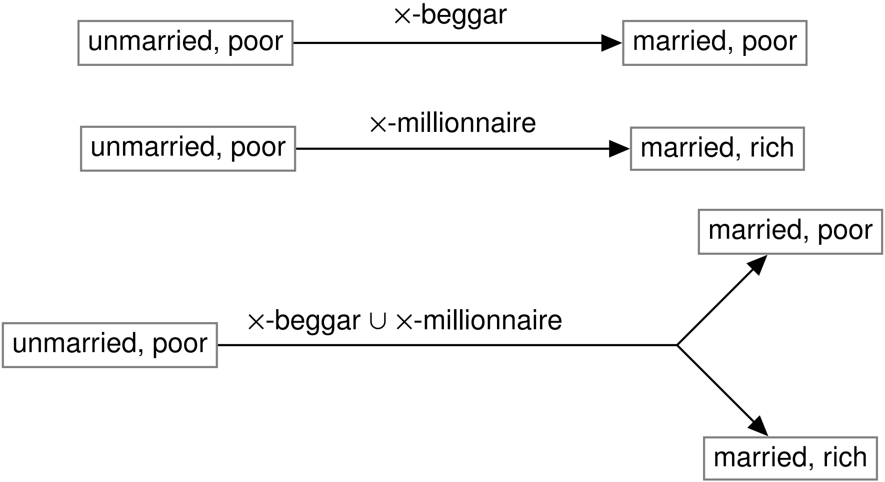


Starting out from the situation where the alarm is off, we would get:



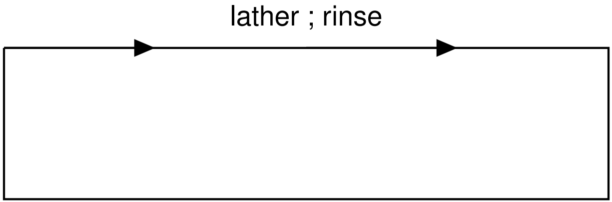
Choice

A complex action may consist of a choice between simpler actions:

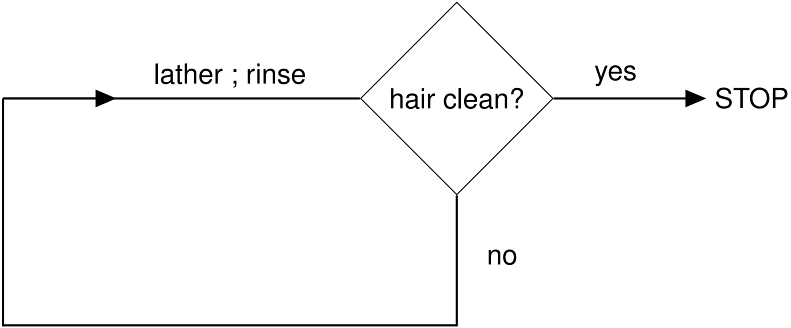


Repetition

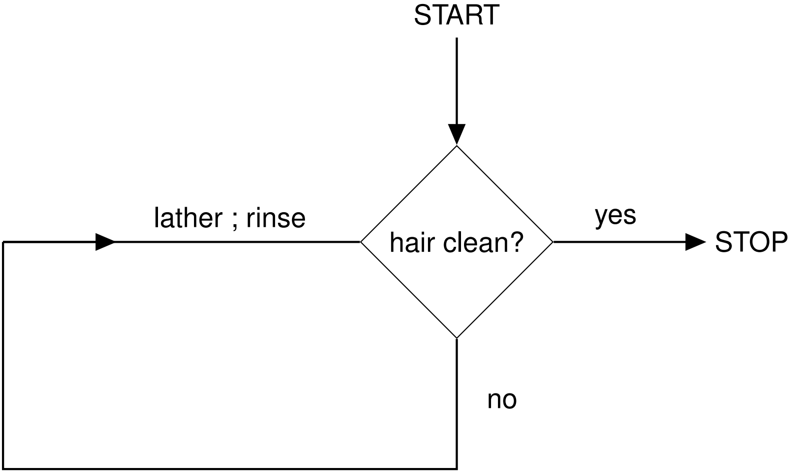
Lather, rinse, repeat:



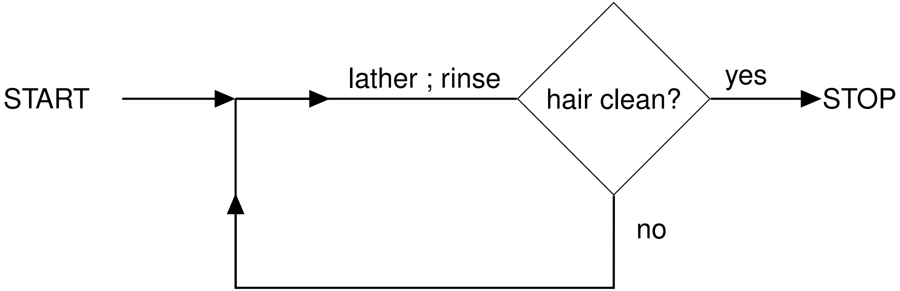
With Stop Condition



Where Do We Start?



With Another Starting Point



In Programming: Difference Between While and Repeat Loops

while not hair_clean **do** { lather; rinse }

repeat { lather ; rinse } **until** hair_clean

Test

The ‘condition’ in a condition-controlled loop (the condition ‘hair_clean’, for example) can itself be viewed as an action: a test whether a certain fact holds.

$?\phi$

Using test, sequence and choice we can express the familiar ‘if then else’ from many programming languages.

```
if hair_clean then skip else { lather ; rinse }
```

```
?hair_clean ∪ { ?¬hair_clean ; lather ; rinse }
```

Generally:

if ϕ **then** α_1 **else** α_2

$?\phi; \alpha_1 \;\cup\; ?\neg\phi; \alpha_2.$

Example: programming languages

Consider three famous control structures:

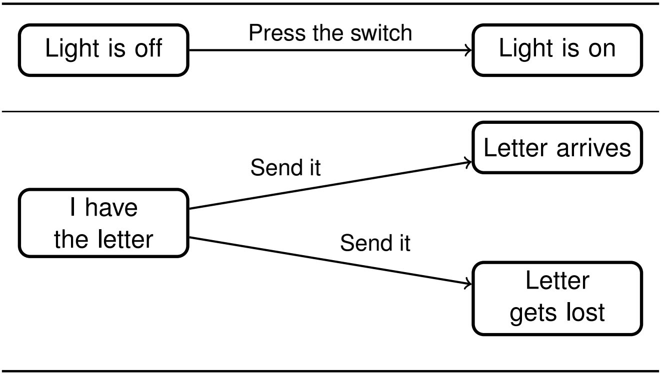
1. **WHILE P do A**
   This can be defined as the repetition of a test for '*P*' and the execution of '*A*', followed by a test for 'not *A*'.

2. **REPEAT A UNTIL P**
   This can be defined as the sequence of '*A*' and then **WHILE (not P) do A.**

3. **IF P THEN A ELSE B**
   This can be defined as a choice between a test for '*P*' and then '*A*', or a test for 'not *P*' and then '*B*'.

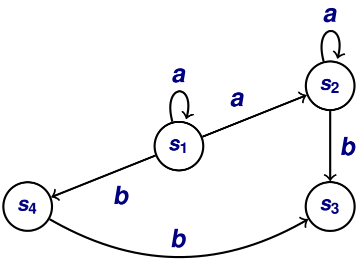
Representing actions abstractly (1)

We can see actions as transitions between states:



Representing actions abstractly (2)

More precisely, if we consider a set of states $S = \{s_1, s_2, \ldots\}$, then we can represent **actions as binary relations on S**.



$$R_a := \{(s_1, s_1), (s_1, s_2), (s_2, s_2)\}$$

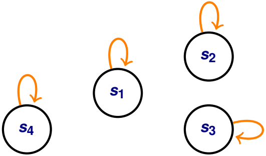
$$R_b := \{(s_1, s_4), (s_2, s_3), (s_4, s_3)\}$$

Operations on relations (1)

Let S be a domain $\{s_1, s_2, \ldots\}$.

- **Identity relation.**

$$I := \{(s, s) \mid s \in S\}$$



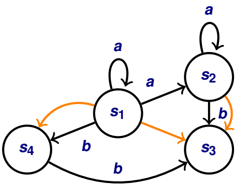
$$I = \{(s_1, s_1), (s_2, s_2), (s_3, s_3), (s_4, s_4)\}$$

Operations on relations (2)

Let S be a domain $\{s_1, s_2, \ldots\}$, and R_a, R_b be binary relations on S.

- **Composition.**

$$R_a \circ R_b := \{(s, s') \mid \text{there is } s'' \in S \text{ such that } R_a s s'' \text{ and } R_b s'' s'\}$$



$$R_a := \{(s_1, s_1), (s_1, s_2), (s_2, s_2)\}$$
$$R_b := \{(s_1, s_4), (s_2, s_3), (s_4, s_3)\}$$
$$R_a \circ R_b = \{(s_1, s_4), (s_1, s_3), (s_2, s_3)\}$$

In particular, for any relation R_a, we have

$$R_a^0 := I, \qquad R_a^1 := R_a \circ R_a^0, \qquad R_a^2 := R_a \circ R_a^1, \qquad R_a^3 := R_a \circ R_a^2,$$

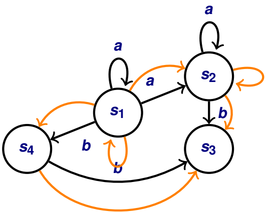
and so on.

Operations on relations (3)

Let S be a domain $\{s_1, s_2, \ldots\}$, and R_a, R_b be binary relations on S.

- **Union.**

$$R_a \cup R_b := \{(s, s') \mid R_a ss' \text{ or } R_b ss'\}$$



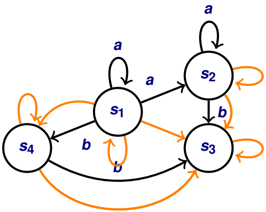
$$R_a := \{(s_1, s_1), (s_1, s_2), (s_2, s_2)\}$$
$$R_b := \{(s_1, s_4), (s_2, s_3), (s_4, s_3)\}$$
$$R_a \cup R_b = \{(s_1, s_1), (s_1, s_2), (s_2, s_2)$$
$$(s_1, s_4), (s_2, s_3), (s_4, s_3)\}$$

Operations on relations (4)

Let S be a domain $\{s_1, s_2, \ldots\}$, and R_a, R_b be binary relations on S.

- **Repetition zero or more times.**

$$R_a^* := \{(s, s') \mid R_a^n ss' \text{ for some } n \in \mathbb{N}\}$$



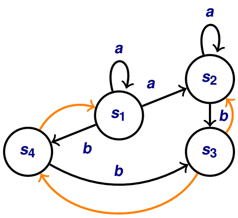
$$\begin{aligned}
R_b &:= \{(s_1, s_4), (s_2, s_3), (s_4, s_3)\} \\
R_b^0 &= \{(s_1, s_1), (s_2, s_2), (s_3, s_3), (s_4, s_4)\} \\
R_b^1 &= \{(s_1, s_4), (s_2, s_3), (s_4, s_3)\} \\
R_b^2 &= \{(s_1, s_4), (s_2, s_3), (s_4, s_3), (s_1, s_3)\} \\
R_b^3 &= \{(s_1, s_4), (s_2, s_3), (s_4, s_3), (s_1, s_3)\} \\
&\vdots \\
R_b^* &= \{(s_1, s_1), (s_2, s_2), (s_3, s_3), (s_4, s_4), \\
&\quad (s_1, s_4), (s_2, s_3), (s_4, s_3), (s_1, s_3)\}
\end{aligned}$$

Operations on relations (5)

Let S be a domain $\{s_1, s_2, \ldots\}$, and R_a, R_b be binary relations on S.

- **Converse.**

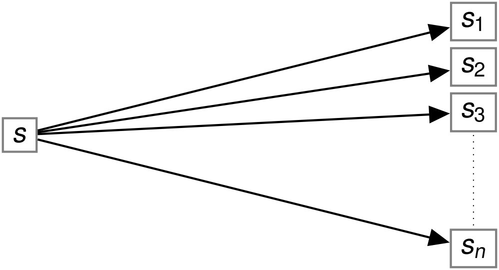
$$R_a^{\smile} := \{(s', s) \mid R_a s s'\}$$



$$R_b := \{(s_1, s_4), (s_2, s_3), (s_4, s_3)\}$$
$$R_b^{\smile} = \{(s_4, s_1), (s_3, s_2), (s_3, s_4)\}$$

Viewing Actions as Relations

Suppose we are in some state s in S. Then performing some action a will result in a new state that is a member of some set of new states $\{s_1, \dots, s_n\}$.

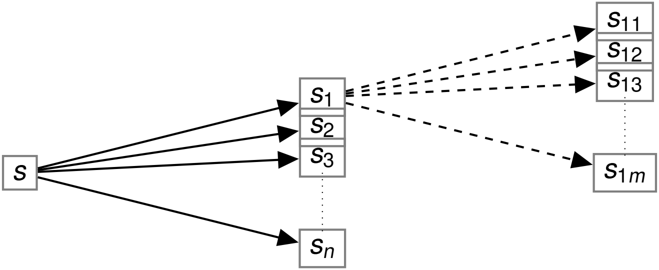


Operations on Relations : Composition

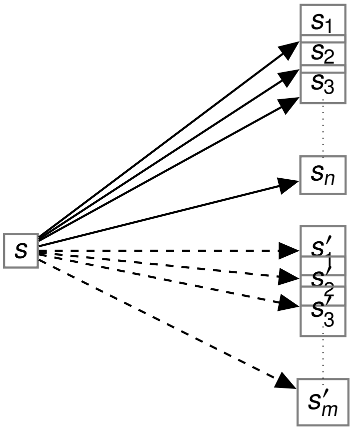
relational composition. Let R_a and R_b be binary relations on the same set S

$R_a \circ R_b$ is the binary relation on S given by:

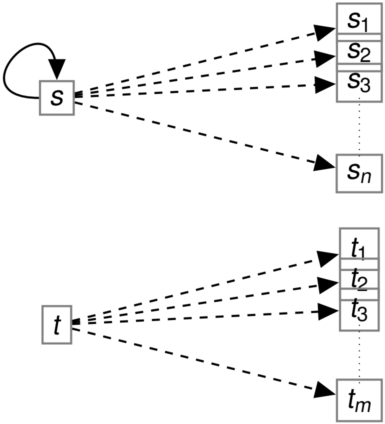
$$R_a \circ R_b = \{(s, s') \mid \text{ there is some } s_0 \in S : (s, s_0) \in R_a \text{and } (s_0, s') \in R_b\}.$$



Operations on Relations: Choice

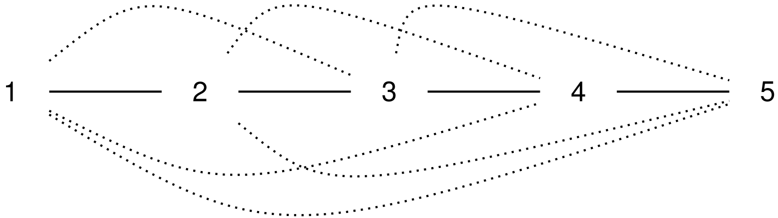


Operations on Relations: Guarded Action



The solid arrow represents a test that succeeds in state s but fails in state t.

Operations on Relations: Transitive Closure



TC of R is the smallest transitive relation S that contains R.

S is the transitive closure of R if

1. $R \subseteq S$,
2. $S \circ S \subseteq S$,
3. if $R \subseteq T$ and $T \circ T \subseteq T$ then $S \subseteq T$.

Reflexive Transitive Closure; While Loop

Notation for transitive closure of R: R^+.

Reflexive transitive closure of R is $I \cup R^+$. Notation: R^*.

'while ϕ do a'

$$(R_{?\phi} \circ R_a)^* \circ R_{?\neg\phi}.$$

Propositional Dynamic Logic or PDL: Brief History

Vaughan Pratt   Krister Segerberg   Dexter Kozen   Rohit Parikh

Syntax (1)

The language of **propositional dynamic logic (*PDL*)** has two components, **formulas** φ and **actions** α.

- **Formulas** are built via the following rules.
  - Every basic proposition is a formula

$$p, \quad q, \quad r, \quad \ldots$$

  - If φ and ψ are formulas, then the following are formulas:

$$\neg\varphi, \quad \varphi \land \psi, \quad \varphi \lor \psi, \quad \varphi \rightarrow \psi, \quad \varphi \leftrightarrow \psi$$

  - If φ is a formula and α an action, then the following are formulas:

$$\langle\alpha\rangle\,\varphi, \quad [\alpha]\,\varphi$$

Syntax (2)

The language of **propositional dynamic logic (*PDL*)** has two components, **formulas** φ and **actions** α.

- **Actions** are built via the following rules.
  - Every basic action is a action

$$\boldsymbol{a}, \quad \boldsymbol{b}, \quad \boldsymbol{c}, \quad \ldots$$

  - If α and β are actions, then the following are actions:

$$\alpha;\beta, \quad \alpha \cup \beta, \quad \alpha^*$$

  - If φ is a formula, then the following is an action:

$$?\varphi$$

Intuitions and abbreviations

$\alpha;\beta$	**sequential composition**: execute α and then β.
$\alpha \cup \beta$	**non-deterministic choice**: execute α or β.
α^*	**repetition**: execute α zero, one, or any *finite* number of times.
$?\varphi$	**test**: check whether φ is true or not.
$\langle\alpha\rangle\,\varphi$	α can be executed in such a way that, after doing it, φ is the case.
$[\alpha]\,\varphi$	After any execution of α, φ is the case.

We abbreviate $p \lor \neg p$ as $\top$.

We abbreviate $\neg\top$ as $\bot$.

Note that $\neg\langle\alpha\rangle\,\neg\varphi$ is equivalent to $[\alpha]\,\varphi$.

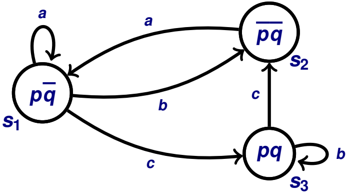
Some examples of formulas

$\langle \alpha \rangle \top$	α can be executed.
$[\alpha] \bot$	α cannot be executed.
$\langle \alpha \rangle \varphi \wedge \neg [\alpha] \varphi$	α can be executed it at least two different ways.

The models (1)

The structures in which we evaluate PDL formulas, **labelled transition systems (*LTS*)**, have three components:

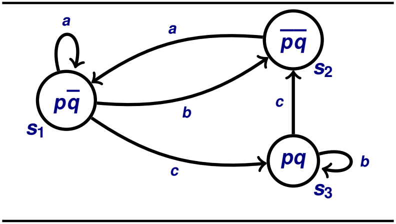
- a non-empty set S of **states**,
- a **valuation function**, V, indicating which atomic propositions are true in each state $s \in S$, and
- an **binary relation** R_a for each basic action a.



$$M = \langle S, R_a, V \rangle$$

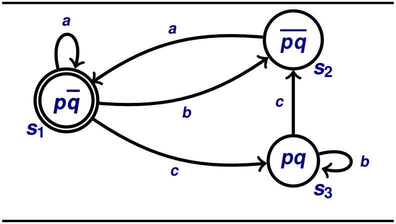
The models (2)

A **labelled transition system** with a designate state (the *root* state) is called a **pointed labelled transition system** or a **process graph**.



The models (2)

A **labelled transition system** with a designate state (the *root* state) is called a **pointed labelled transition system** or a **process graph**.



Deciding truth-value of formulas

Take a pointed labelled transition system (M, s) with $M = \langle S, R_a, V \rangle$:

$(M, s) \models p$	iff	$p \in V(s)$
$(M, s) \models \neg\varphi$	iff	it is not the case that $(M, s) \models \varphi$
$(M, s) \models \varphi \vee \psi$	iff	$(M, s) \models \varphi$ or $(M, s) \models \psi$
...	iff	...
$(M, s) \models \langle\alpha\rangle\,\varphi$	iff	there is a $t \in S$ such that $R_\alpha st$ and $(M, t) \models \varphi$
$(M, s) \models [\alpha]\,\varphi$	iff	for all $t \in S$ such that $R_\alpha st$ it holds that $(M, t) \models \varphi$

where the relation R_α is given, in case α is not a basic action, by

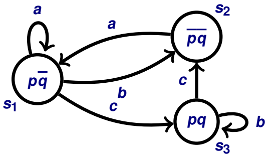
$$R_{\alpha;\beta} := R_\alpha \circ R_\beta$$

$$R_{\alpha\cup\beta} := R_\alpha \cup R_\beta$$

$$R_{\alpha^*} := (R_\alpha)^*$$

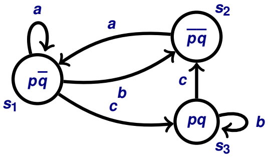
$$R_{?\varphi} := \{(s, s) \in S \times S \mid (M, s) \models \varphi\}$$

Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

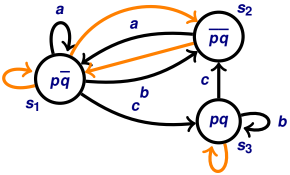
Example: building complex relations



$R_a := \{(s_1, s_1), (s_2, s_1)\}$
$R_b := \{(s_1, s_2), (s_3, s_3)\}$
$R_c := \{(s_1, s_3), (s_3, s_2)\}$

$R_{a \cup b} =$

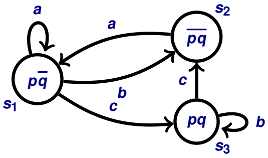
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$

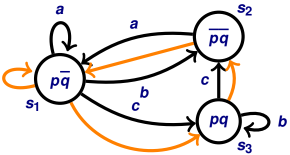
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} =$$

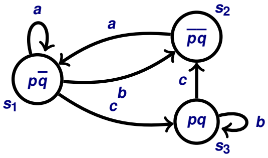
Example: building complex relations



$R_a := \{(s_1, s_1), (s_2, s_1)\}$
$R_b := \{(s_1, s_2), (s_3, s_3)\}$
$R_c := \{(s_1, s_3), (s_3, s_2)\}$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$

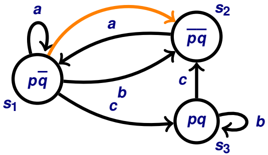
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} =$$

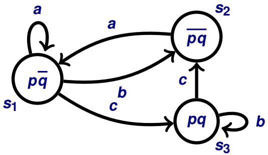
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$

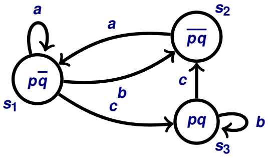
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} =$$

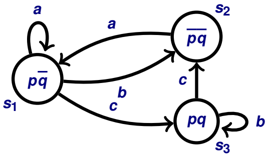
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$

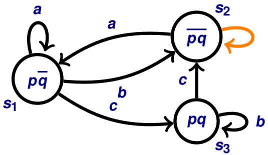
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p \vee q)} =$$

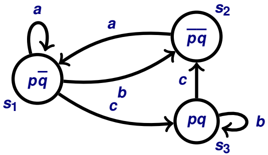
Example: building complex relations



$R_a := \{(s_1, s_1), (s_2, s_1)\}$
$R_b := \{(s_1, s_2), (s_3, s_3)\}$
$R_c := \{(s_1, s_3), (s_3, s_2)\}$

$$
\begin{aligned}
R_{a \cup b} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\} \\
R_{a \cup c} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\} \\
R_{c;c} &= \{(s_1, s_2)\} \\
R_{b;b} &= \{\ \} \\
R_{?\neg(p \vee q)} &= \{(s_2, s_2)\}
\end{aligned}
$$

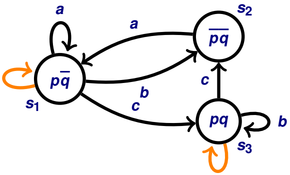
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p \vee q)} = \{(s_2, s_2)\}$$
$$R_{?(p \vee q)} =$$

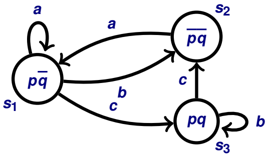
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p \vee q)} = \{(s_2, s_2)\}$$
$$R_{?(p \vee q)} = \{(s_1, s_1), (s_3, s_3)\}$$

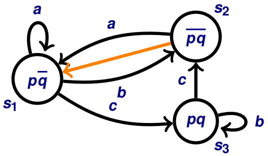
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p \vee q)} = \{(s_2, s_2)\}$$
$$R_{?(p \vee q)} = \{(s_1, s_1), (s_3, s_3)\}$$
$$R_{?\neg(p \vee q);a;?(p \vee q)} =$$

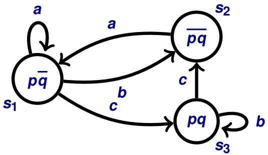
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p \vee q)} = \{(s_2, s_2)\}$$
$$R_{?(p \vee q)} = \{(s_1, s_1), (s_3, s_3)\}$$
$$R_{?\neg(p \vee q);a;?(p \vee q)} = \{(s_2, s_1)\}$$

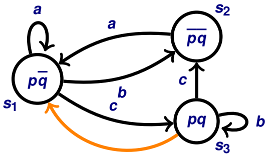
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a \cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a \cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p \vee q)} = \{(s_2, s_2)\}$$
$$R_{?(p \vee q)} = \{(s_1, s_1), (s_3, s_3)\}$$
$$R_{?\neg(p \vee q);a;?(p \vee q)} = \{(s_2, s_1)\}$$
$$R_{c;a} =$$

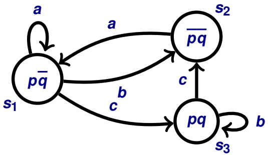
Example: building complex relations



$R_a := \{(s_1, s_1), (s_2, s_1)\}$
$R_b := \{(s_1, s_2), (s_3, s_3)\}$
$R_c := \{(s_1, s_3), (s_3, s_2)\}$

$$
\begin{aligned}
R_{a\cup b} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\} \\
R_{a\cup c} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\} \\
R_{c;c} &= \{(s_1, s_2)\} \\
R_{b;b} &= \{\ \} \\
R_{?\neg(p\vee q)} &= \{(s_2, s_2)\} \\
R_{?(p\vee q)} &= \{(s_1, s_1), (s_3, s_3)\} \\
R_{?\neg(p\vee q);a;?(p\vee q)} &= \{(s_2, s_1)\} \\
R_{c;a} &= \{(s_3, s_1)\}
\end{aligned}
$$

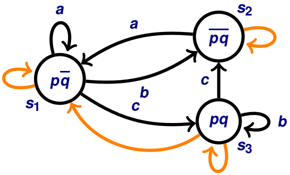
Example: building complex relations



$R_a := \{(s_1, s_1), (s_2, s_1)\}$
$R_b := \{(s_1, s_2), (s_3, s_3)\}$
$R_c := \{(s_1, s_3), (s_3, s_2)\}$

$$
\begin{aligned}
R_{a \cup b} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\} \\
R_{a \cup c} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\} \\
R_{c;c} &= \{(s_1, s_2)\} \\
R_{b;b} &= \{\ \} \\
R_{?\neg(p \vee q)} &= \{(s_2, s_2)\} \\
R_{?(p \vee q)} &= \{(s_1, s_1), (s_3, s_3)\} \\
R_{?\neg(p \vee q);a;?(p \vee q)} &= \{(s_2, s_1)\} \\
R_{c;a} &= \{(s_3, s_1)\} \\
R_{(c;a)^*} &=
\end{aligned}
$$

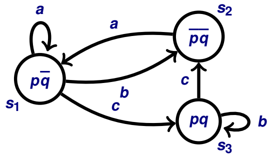
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$R_{a\cup b} = \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\}$$
$$R_{a\cup c} = \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\}$$
$$R_{c;c} = \{(s_1, s_2)\}$$
$$R_{b;b} = \{\ \}$$
$$R_{?\neg(p\vee q)} = \{(s_2, s_2)\}$$
$$R_{?(p\vee q)} = \{(s_1, s_1), (s_3, s_3)\}$$
$$R_{?\neg(p\vee q);a;?(p\vee q)} = \{(s_2, s_1)\}$$
$$R_{c;a} = \{(s_3, s_1)\}$$
$$R_{(c;a)^*} = \{(s_3, s_1), (s_1, s_1), (s_2, s_2), (s_3, s_3)\}$$

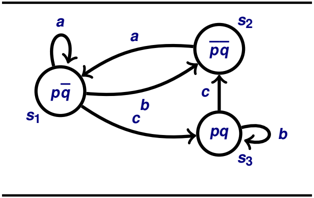
Example: building complex relations



$$R_a := \{(s_1, s_1), (s_2, s_1)\}$$
$$R_b := \{(s_1, s_2), (s_3, s_3)\}$$
$$R_c := \{(s_1, s_3), (s_3, s_2)\}$$

$$
\begin{aligned}
R_{a \cup b} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_2), (s_3, s_3)\} \\
R_{a \cup c} &= \{(s_1, s_1), (s_2, s_1), (s_1, s_3), (s_3, s_2)\} \\
R_{c;c} &= \{(s_1, s_2)\} \\
R_{b;b} &= \{\ \} \\
R_{?\neg(p \vee q)} &= \{(s_2, s_2)\} \\
R_{?(p \vee q)} &= \{(s_1, s_1), (s_3, s_3)\} \\
R_{?\neg(p \vee q);a;?(p \vee q)} &= \{(s_2, s_1)\} \\
R_{c;a} &= \{(s_3, s_1)\} \\
R_{(c;a)^*} &= \{(s_3, s_1), (s_1, s_1), (s_2, s_2), (s_3, s_3)\}
\end{aligned}
$$

Example: evaluating formulas



M

$(M, s_1) \models \langle a \cup b \rangle p \wedge \neg [a \cup b] p$?

$(M, s_1) \models [b] \perp$?

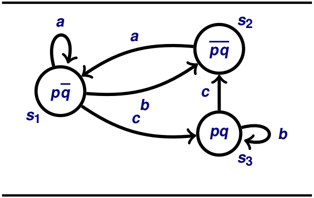
$(M, s_2) \models \langle a \rangle \top \rightarrow \langle b \rangle \top$?

$(M, s_2) \models \langle c^* \rangle \top$?

$(M, s_3) \models [(c; a)^*] p$?

$(M, s_3) \models [?p] p$?

Example: evaluating formulas



$\boldsymbol{M}$

$(M, s_1) \models \langle a \cup b \rangle p \wedge \neg [a \cup b] p$ √

$(M, s_1) \models [b] \bot$?

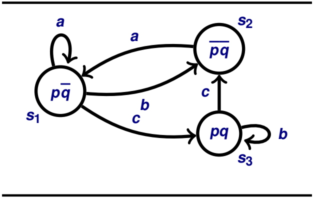
$(M, s_2) \models \langle a \rangle \top \rightarrow \langle b \rangle \top$?

$(M, s_2) \models \langle c^* \rangle \top$?

$(M, s_3) \models [(c; a)^*] p$?

$(M, s_3) \models [?p] p$?

Example: evaluating formulas



M

$(M, s_1) \models \langle a \cup b \rangle\, p \wedge \neg [a \cup b]\, p$ √

$(M, s_1) \models [b]\, \bot$ ×

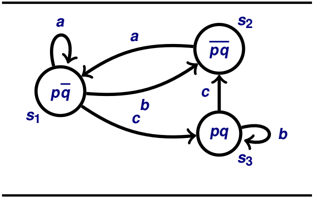
$(M, s_2) \models \langle a \rangle \top \rightarrow \langle b \rangle \top$?

$(M, s_2) \models \langle c^* \rangle \top$?

$(M, s_3) \models [(c; a)^*]\, p$?

$(M, s_3) \models [?p]\, p$?

Example: evaluating formulas



M

$(M, s_1) \models \langle a \cup b \rangle p \wedge \neg [a \cup b] p$ √

$(M, s_1) \models [b] \bot$ ×

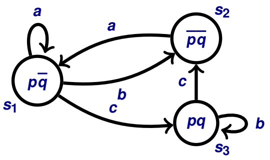
$(M, s_2) \models \langle a \rangle \top \rightarrow \langle b \rangle \top$ ×

$(M, s_2) \models \langle c^* \rangle \top$?

$(M, s_3) \models [(c; a)^*] p$?

$(M, s_3) \models [?p] p$?

Example: evaluating formulas



M

$(M, s_1) \models \langle a \cup b \rangle p \wedge \neg [a \cup b] p$ √

$(M, s_1) \models [b] \bot$ ×

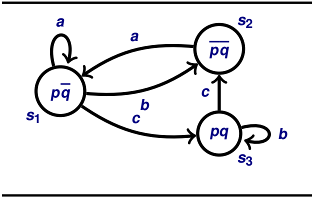
$(M, s_2) \models \langle a \rangle \top \rightarrow \langle b \rangle \top$ ×

$(M, s_2) \models \langle c^* \rangle \top$ √

$(M, s_3) \models [(c; a)^*] p$?

$(M, s_3) \models [?p] p$?

Example: evaluating formulas



M

$(M, s_1) \models \langle a \cup b \rangle\, p \wedge \neg [a \cup b]\, p$ √

$(M, s_1) \models [b]\, \bot$ ×

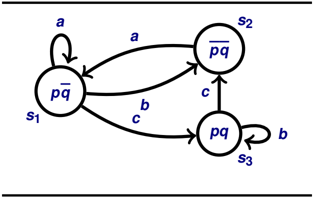
$(M, s_2) \models \langle a \rangle \top \rightarrow \langle b \rangle \top$ ×

$(M, s_2) \models \langle c^* \rangle \top$ √

$(M, s_3) \models [(c; a)^*]\, p$ √

$(M, s_3) \models [?p]\, p$?

Example: evaluating formulas



M

$(M, s_1) \models \langle a \cup b \rangle p \wedge \neg [a \cup b] p$ √

$(M, s_1) \models [b] \bot$ ×

$(M, s_2) \models \langle a \rangle \top \to \langle b \rangle \top$ ×

$(M, s_2) \models \langle c^* \rangle \top$ √

$(M, s_3) \models [(c; a)^*] p$ √

$(M, s_3) \models [?p] p$ √

Axiom system (1)

The valid formulas of ***PDL*** can be derived from the following principles:

1. All propositional tautologies.
2. $[\alpha](\varphi \to \psi) \to ([\alpha]\varphi \to [\alpha]\psi)$ for any action α.
3. **Modus ponens** (MP): from φ and $\varphi \to \psi$, infer ψ.
4. **Necessitation** (Nec): from φ infer $[\alpha]\varphi$ for any action α.

Axiom system (2)

5. **Principles for action operations**:
   - **Test**:
   $$[?\psi]\,\varphi \leftrightarrow (\psi \to \varphi)$$
   - **Sequence**:
   $$[\alpha;\beta]\,\varphi \leftrightarrow [\alpha]\,[\beta]\,\varphi$$
   - **Choice**:
   $$[\alpha \cup \beta]\,\varphi \leftrightarrow ([\alpha]\,\varphi \wedge [\beta]\,\varphi)$$
   - **Repetition**:
     - **Mix**:
     $$[\alpha^*]\,\varphi \leftrightarrow (\varphi \wedge [\alpha]\,[\alpha^*]\,\varphi)$$
     - **Induction**:
     $$\big(\varphi \wedge [\alpha^*]\,(\varphi \to [\alpha]\,\varphi)\big) \to [\alpha^*]\,\varphi$$

A formula that can be derived by following these principles in a *finite* number of steps is called a **theorem**.

Example

Prove that $[(\alpha \cup \beta);\gamma]\,\varphi \leftrightarrow ([\alpha;\gamma]\,\varphi \wedge [\beta;\gamma]\,\varphi)$ is valid.

From left to right:

1.	$[(\alpha \cup \beta);\gamma]\,\varphi$	Assumption
2.	$[\alpha \cup \beta]\,[\gamma]\,\varphi$	Sequence from step 1
3.	$[\alpha]\,[\gamma]\,\varphi \wedge [\beta]\,[\gamma]\,\varphi$	Choice from step 2
4.	$[\alpha;\gamma]\,\varphi \wedge [\beta;\gamma]\,\varphi$	Sequence from step 3

The right to left direction is similar.

PDL as a programming language

With ***PDL*** we can define actions representing program control structures.

1. **WHILE** φ **do** α:
$$(?\varphi; \alpha)^*; ?\neg\varphi$$
2. **REPEAT** α **UNTIL** φ:
$$\alpha; (?\neg\varphi; \alpha)^*; ?\varphi$$
3. **IF** φ **THEN** α **ELSE** β:
$$(?\varphi; \alpha) \cup (?\neg\varphi; \beta)$$

Programs and Computation

$V = \{v_1, \ldots, v_n\}$ is a set of storage locations for integer numbers.

A V-memory:

v_1	v_2	v_3	v_4	v_5	v_6	v_7	
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

A V-state s is a function $V \to \mathbb{Z}$.

v_1	v_2	v_3	v_4	v_5	v_6	v_7	
2	−3	334	0	2	1	102	$\cdots$

Assignment Actions

Let i range over integer names, such as 0, −234 or 53635 and let v range over V. Then the following defines arithmetical expressions:

$$a \quad ::= \quad i \mid v \mid a_1 + a_2 \mid a_1 * a_2 \mid a_1 - a_2.$$

$[\![a]\!]_s$ gives the value of arithmetical expression a in V-state s.

Basic propositions: $a_1 \leq a_2$

$a_1 \leq a_2$ is true in s iff $[\![a_1]\!]_s \leq [\![a_2]\!]_s$.

Basic actions: $v := a$.

$[\![v := a]\!] = \{(s, s') \mid s \sim_v s' \text{ and } s'(v) = [\![a]\!]_s\}$.

Reasoning about Computation

- Determinism:

$$\langle\alpha\rangle\phi \rightarrow [\alpha]\phi.$$

- The basic programming actions $v := a$ are deterministic.
- Termination:

$$\langle\alpha\rangle\top.$$

- the basic programming actions $v := a$ always terminate.
- Non-Termination (for deterministic programs):

$$[\alpha]\bot.$$

- Example: while $\top$ do $v := v + 1$.
- Hoare correctness reasoning: $\{P\}\ \alpha\ \{Q\}$.
- PDL version: $P \rightarrow [\alpha]Q$.

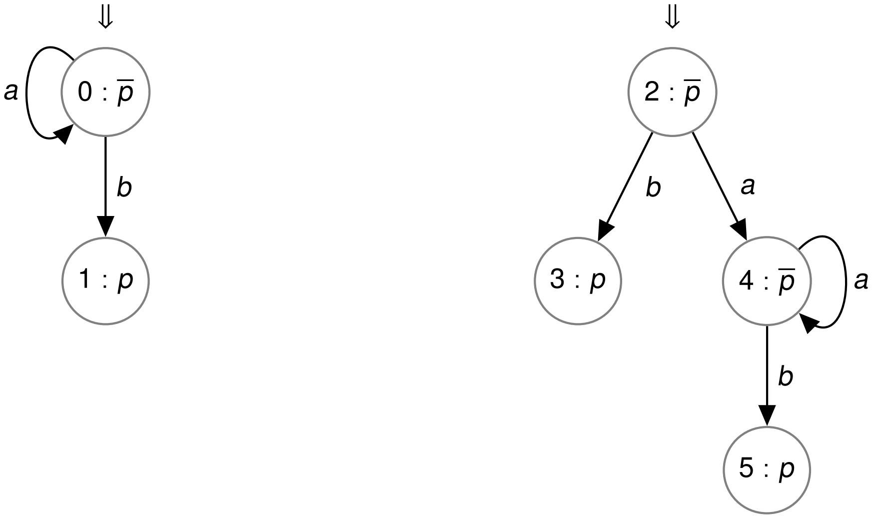
Example of Hoare Correctness Reasoning

A vase contains 35 white pebbles and 35 black pebbles. Proceed as follows to draw pebbles from the vase, as long as this is possible. Every round, draw two pebbles from the vase. If they have the same colour, then put a black pebble back into the vase, if they have different colours, then put the white pebble back. You may assume that there are enough additional black pebbles. In every round one pebble is removed from the vase, so after 69 rounds there is a single pebble left. What is the colour of this pebble?

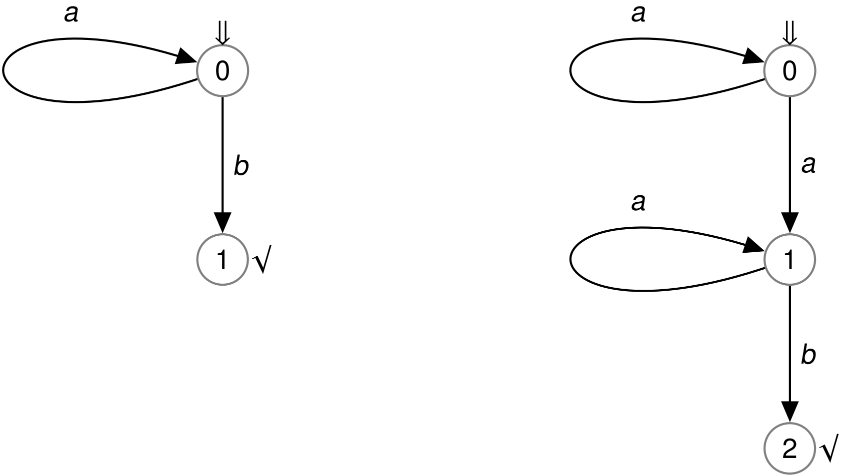
Epistemic PDL, LCC

- Interpret the *a* relations as knowledge relations.
- $[a]\phi$ expresses that agent *a* knows ϕ.
- $[(a \cup b)^*]\phi$ expresses that *a* and *b* have common knowledge of ϕ.
- Logic of Communication and Change (LCC): add $[A]\phi$ where *A* is an action model in the sense of Baltag & Moss [1].
- Theorem (Van Benthem, vE, Kooi [2]): LCC has the same expressive power as epistemic PDL.

Equivalence of programs and bisimulation



Not Bisimilar (at the top level)



Collective Rational Action: Is it Possible?

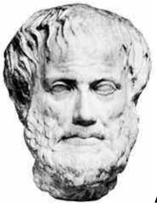
A famous quotation . . .

> *"For that which is common to the greatest number has the least care bestowed upon it. Every one thinks chiefly of his own, hardly at all of the common interest; and only when he is himself concerned as an individual. For besides other considerations, everybody is more inclined to neglect the duty which he expects another to fulfill; as in families many attendants are often less useful than a few."*

Collective Rational Action: Is it Possible?

A famous quotation ...

> *"For that which is common to the greatest number has the least care bestowed upon it. Every one thinks chiefly of his own, hardly at all of the common interest; and only when he is himself concerned as an individual. For besides other considerations, everybody is more inclined to neglect the duty which he expects another to fulfill; as in families many attendants are often less useful than a few."*



Aristotle, Politics, Book Two (about 350 B.C.)

Rational for the one, disastrous for the many

- Farmers grazing their goats on a common meadows.
- Citizens who want free parking in their inner cities.
- Prosperous families wanting to drive bigger and bigger SUVs.
- Airport hubs wanting to attract ever more air traffic.
- Fishermen roaming the oceans in ever bigger fishing trawlers
- Logging companies cutting down ever more tropical forest.
- Developed countries exporting their industrial waste to developing countries.
- US citizens defending the Second Amendment right to keep and bear firearms. (“NRA: The largest civil-rights group ever”)

The Tragedy of the Commons

Garrett Hardin

Garrett Hardin (1915–2003) was a microbiologist and ecologist. ‘The Tragedy of the Commons’ is his most well-known essay.

‘I try to convince people that
what sounds like bad news is better
than “good news” that’s wrong.’

Interview in Skeptic magazine, 1996.

A Problem with Inconvenient Truths

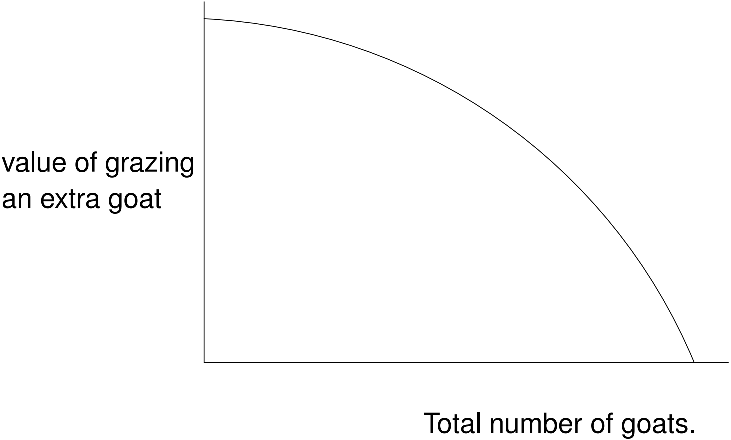
The Tragedy of the Commons

The tragedy of the commons develops in this way. Picture a pasture open to all. It is to be expected that each herdsman will try to keep as many cattle as possible on the commons. Such an arrangement may work reasonably satisfactorily for centuries because tribal wars, poaching, and disease keep the numbers of both man and beast well below the carrying capacity of the land. Finally, however, comes the day of reckoning, that is, the day when the long-desired goal of social stability becomes a reality. At this point, the inherent logic of the commons remorselessly generates tragedy.

Garrett Hardin, 'The Tragedy of the Commons', in Science, 1968 [6].

See http://www.garretthardinsociety.org/articles/art_tragedy_of_the_commons.html.

The Tragedy in a Picture, from [5]



The Tragedy in a Picture

picture by
Marco Swaen

From the Fourth IPCC Assessment Report

"The climate system tends to be overused (excessive GHG concentrations) because of its natural availability as a resource whose access is open to all free of charge. In contrast, climate protection tends to be underprovided. In general, the benefits of avoided climate change are spatially indivisible, freely available to all (non-excludability), irrespective of whether one is contributing to the regime costs or not. As regime benefits by one individual (nation) do not diminish their availability to others (non-rivalry), it is difficult to enforce binding commitments on the use of the climate system [8, 7]. This may result in "free riding", a situation in which mitigation costs are borne by some individuals (nations) while others (the "free riders") succeed in evading them but still enjoy the benefits of the mitigation commitments of the former[12, page 102]."

Punishment of Free-riders

Binary reciprocity is the simplest kind of helping others. Societies thrive if there is generalized reciprocity, or “paying it forward”.

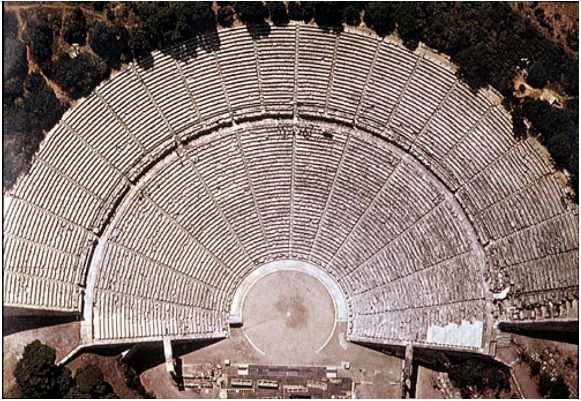
Two things help generalized reciprocity to emerge [10]:

- shared information/common experience
- a reputation mechanism by which an agent’s social score depends on whether they are free riders or are paying it forward.

Agents who are known to be free riders, are not helped anymore.

In a public goods game with 1,000 participants on the Internet, participants dealt out many more altruistic punishments when it was cheap to do so and had high impact on the free-riders, than when it was expensive or had low impact [3].

Common Experience—The Theatre at Epidauros



Collective Action

- Effective collective action can never be the sum of individual actions.
- Needed for successful collective action:
  - common knowledge of the moral stature of those influencing the group,
  - common knowledge of what is the interest of the group as a whole,
  - common knowledge of the collective willingness to take action.
- Social structures for this are all structures that foster the sense of community (Epidauros is a paradigm).

Knowledge Based Obligation

See ‘The Logic of Knowledge Based Obligation’ [11]. Presuppositions of First Order Obligation:

- Opportunity (a medical doctor happens to be present when Suzie gets a heart attack has an opportunity to help)
- Ability (a medical doctor may have obligations that a layperson has not)
- Knowledge (a doctor who does not know that Suzie is ill may have no obligation to help her)

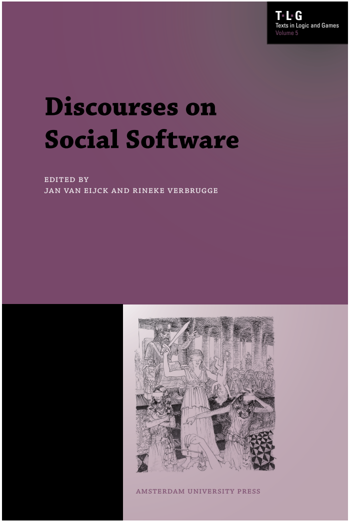
There are also Second Order Oligations:

- Duty to maintain/acquire an ability (can you think of examples?)
- Duty to get/keep informed (can you think of examples?)

Knowledge Based Obligation for Scientists [4]

Martin Rees [the president of Royal Society] has urged scientists to get more involved in public debate, to speak out against minority “maverick” views [9]. Only those who understand how science works—and I suppose that includes all of us—can appreciate the difference between peer-reviewed papers in top-ranking scientific journals and mere pamphlets on the internet. We are skilled in distinguishing false from true in scientific matters, and I believe that this skill comes with responsibilities. We can see that there is a consensus on climate change. The scientific consensus is that there is global warming, that it is to a large extent anthropogenic, and that it is dangerous.

More on this: http://homepages.cwi.nl/~jve/dss



A. Baltag and L.S. Moss.
Logics for epistemic programs.
Synthese, 139(2):165–224, 2004.

J. van Benthem, J. van Eijck, and B. Kooi.
Logics of communication and change.
Information and Computation, 204(11):1620–1662, 2006.

Martijn Egas and Arno Riedl.
The economics of altruistic punishment and the demise of cooperation.
Tinbergen Institute Discussion Papers 05-065/1, Tinbergen Institute, June 2005.

J. van Eijck and R. Verbrugge, editors.
Discourses on Social Software, volume 5 of Texts in Logic and Games.
Amsterdam University Press, Amsterdam, 2009.

Robert Gibbons.

A Primer in Game Theory.
Prentice Hall, 1992.

Garrett Hardin.
The tragedy of the commons.
Science, 162:1243–48, 1968.

I. Kaul, P. Conceiąo, K. Le Gouven, and R.U. Mendoz.
Providing Global Public Goods.
Oxford University Press, 2003.

I. Kaul, I. Grunberg, and M.A. Stern.
Global Public Goods.
Oxford University Press, 1999.

BBC News.
'Maverick' risk to science debate, November 30 2006.
http://news.bbc.co.uk/2/hi/science/nature/6159371.stm.

M.A. Nowak and K. Sigmund.
Evolution of indirect reciprocity.

Nature, 437:1291–1298, 2005.

Eric Pacuit, Rohit Parikh, and Eva Cogan.
The logic of knowledge based obligation.
Synthese, 31:311–341, 2006.

H.-H. Rogner, R. Zhou, R. Bradley, P. Crabbé, O. Edenhofer, B. Hare, L. Kuijpers, and M. Yamaguchi.
Introduction.
In B. Metz et al., editor, Climate Change 2007: Mitigation. Contribution of Working Group III to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge University Press, Cambridge, 2007.