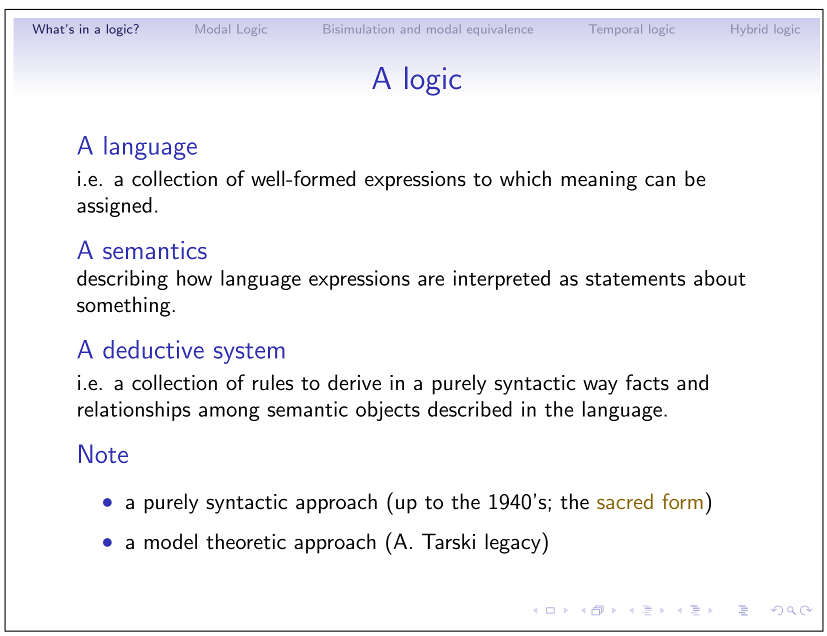
Lecture 3: Introduction to Modal Logic

Luís Soares Barbosa

Abstract

This lecture offers an introduction to modal logic, as part of the course background. As a tool to talk about relational or graph-like structures, modal logic is the 'lingua franca' to express and verify properties of transition systems which underly all the semantic models of reactive architectures discussed in the course. The emphasis is put on propositional modal logic, with a special focus on modal definability, bisimulation and the corresponding modal equivalence results. Several examples of modal logics are briefly introduced, as well as extensions also relevant to the course, namely temporal and hybrid logic.

1 What's in a logic?



1. Logic. If, with a certain philosophical flavour, Logic can be defined as the study of the principles of reasoning, in a Computer Science course we focus on an particular corner of that landscape. Our concern is the study of *logics*, *i.e.* of specific *languages* able to talk about specific *abstract structures* and equipped with *rules* for deducing one sentence from others and therefore properties from properties of the structures in which they are interpreted.

By the end of the 19th century such part of the landscape we are interested in, by then coined as *symbolic logic*, flourished with the aim to provide a foundation for Mathematics. A century after, again our programme has stricter limits: we seek for logics able to describe computational phenomena, state and verify their properties, as well as for computational mechanisms to automate reasoning within the former about the latter.

Lecture notes for *Arquitectura e Cálculo*, MEI profile in *Formal Methods in Software Engineering*, 2014-15.

What's in a logic? | Modal Logic | Bisimulation and modal equivalence | Temporal logic | Hybrid logic

Semantic reasoning: models

- sentences
- models & satisfaction: $\mathfrak{M} \models \phi$
- validity: $\models \phi$ (ϕ is satisfied in every possible structure)
- logical consequence: $\Phi \models \phi$ (ϕ is satisfied in every model of Φ)
- theory: $Th\,\Phi$ (set of logical consequences of a set of sentences Φ)

What's in a logic? | Modal Logic | Bisimulation and modal equivalence | Temporal logic | Hybrid logic

Syntactic reasoning: deductive systems

Deductive systems

- sequents
- Hilbert systems
- natural deduction
- tableaux systems
- resolution
- ...

- derivation and proof
- deductive consequence: $\Phi \vdash \phi$
- theorem: $\vdash \phi$

2. The triangle: Languages, models, proof systems. As van Benthem puts it *logical formalism starts with a language, a system of patterns behind some practice of communication and reasoning. These patterns are formal and austere, but that is precisely why they highlight basic features of the phenomenon described, while also suggesting analogies across different situations.* Then, *models*: algebraic structures; relational structures; topological structures. In each case *satisfaction* is a bridge connecting a language to its interpretation by means of models. Finally, *deductive systems* as an essentially syntactic way to build reasoning patterns, type them, derive new from old — proof theory has a major relationship with Computer Science (*cf.*, the Verification course in this same MIEI profile).

Much can be said on the vertices of this triangle; as undergrads all of us went up and down, along its edges, at least for propositional logic.

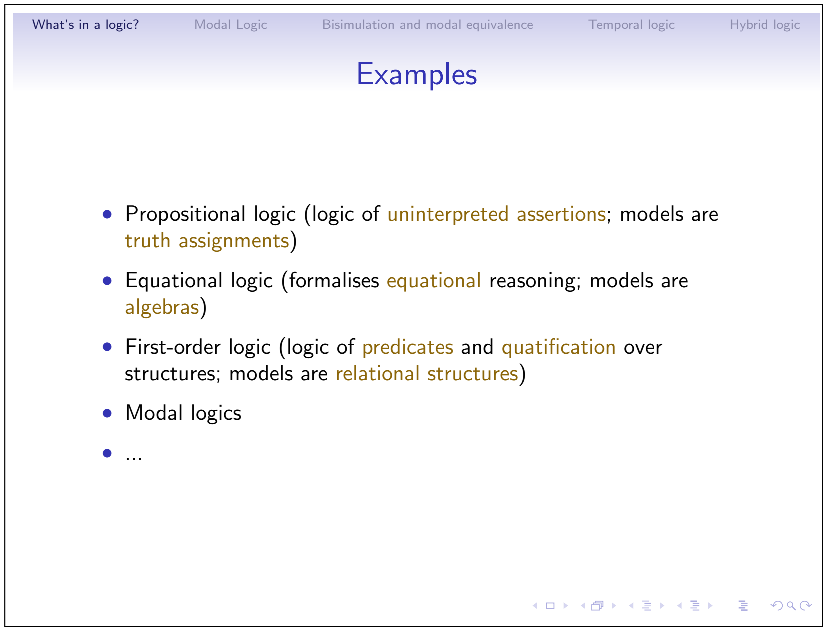
Soundness & completeness

- A deductive system $\vdash$ is sound wrt a semantics $\models$ if for all sentences ϕ
$$\vdash \phi \implies \models \phi$$
(every theorem is valid)
- $\cdots$ complete ...
$$\models \phi \implies \vdash \phi$$
(every valid sentence is a theorem)

Consistency & refutability

For logics with negation and a conjunction operator

- A sentence ϕ is refutable if $\neg\phi$ is a theorem (i.e. $\vdash \neg\phi$)
- A set of sentences Φ is refutable if some finite conjunction of elements in Φ is refutable
- ϕ or Φ is consistent if it is not refutable.



3. Further reading. Although not strictly necessary for this course, students may like to revisit some introductory textbook on Logic, probably the one they have already studied as undergraduates. From the plethora of introductory texts, we single out references [**?**, **?**, **?**, **?**]. The first is very pleasant introduction to propositional and first-order logic, co-authored by Wilfried Hogdes, the author of a reference book on model theory in the early nineties [**?**]. A classical textbook is van Dalen's *Logic and Structure* [**?**], which also covers other relevant topics from a Computer Science perspective, *e.g.*, intuitionistic logic and Gödel incompleteness theorem. Hedman book [**?**] covers in addition basic notions of complexity and its relation to logic; chapter 9 provides some motivation and pointers to what lies beyond first-order, in particular second-order and infinitary logics. The last reference [**?**], closer to a philosophic perspective, is a lively discussion on the nature of meaning and logic, relating the model and proof oriented views.

2 Modal logic

4. Motivation (from J. van Benthem [**?**]) *Some truths seem merely* contingent*, such as the fact what clothes you are wearing today. But other truths seem* necessary*, such as the fact that, like it or not, you are not someone else. Modal notions of* necessity*,* possibility*, and* contingency *were standard fare in traditional logic up to the 19th century. All these notions went out the door in the work of the founding fathers of modern logic, like Boole and Frege (...) who claims that some proposition is necessarily true just means that it is true, plus some autobiographical information about how strongly you believe in it. (...) The result are the familiar logical systems like propositional and predicate logic, which describe properties and relations of objects in* fixed situations*, represented by models. (...) Even so, while extensional logics might be adequate for analyzing mathematical proof and truth in an eternal realm of abstraction, modality made a fast come-back. (...) . And then, one finds that there is a host of notions of a* modal *character going far beyond mere truth:* necessity, knowledge, belief, obligation, temporal change, action*, and so on. Indeed, it is hard to think of any use of language which is purely informative: every sentence we utter resonates in a web of communication, expectations, goals, and emotions.* Modal logic *tries to analyze this structure with techniques taken from the mathematical turn in modern logic.*

What's in a logic? | Modal Logic | Bisimulation and modal equivalence | Temporal logic | Hybrid logic

Modal logic (from P. Blackburn, 2007)

Over the years modal logic has been applied in many different ways. It has been used as a tool for reasoning about time, beliefs, computational systems, necessity and possibility, and much else besides.

These applications, though diverse, have something important in common: the key ideas they employ (flows of time, relations between epistemic alternatives, transitions between computational states, networks of possible worlds) can all be represented as simple graph-like structures.

Modal logics are

- tools to talk about relational, or graph-like structures.
- fragments of classical ones, with restricted forms of quantification ...
- ... which tend to be decidable and described in a pointfree notations.

What's in a logic? | Modal Logic | Bisimulation and modal equivalence | Temporal logic | Hybrid logic

The language

Syntax

$$\phi \ ::= \ p \mid \mathsf{true} \mid \mathsf{false} \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \rightarrow \phi_2 \mid \langle m \rangle\, \phi \mid [m]\, \phi$$

where $p \in \mathrm{PROP}$ and $m \in \mathrm{MOD}$

Disjunction ($\vee$) and equivalence ($\leftrightarrow$) are defined by abbreviation. The signature of the basic modal language is determined by sets PROP of propositional symbols (typically assumed to be denumerably infinite) and MOD of modality symbols.

5. Modalities The intuition behind a modality symbol is that is represents a particular perspective over the world, or, more precisely, over the dynamics, the evolution of the universe of discourse. Modal operators, cf, *boxes* and *diamonds* are a sort of quantifiers, but with a *local* flavour: they only refer to states accessible (in the evolution map specified by the interpretation of the relevant modality symbol) from the current one, *i.e.* the one from where observations are taking place.

The language

Notes

- if there is only one modality in the signature (i.e., MOD is a singleton), write simply $\Diamond\phi$ and $\Box\phi$
- the language has some redundancy: in particular modal connectives are dual (as quantifiers are in first-order logic): $[m]\,\phi$ is equivalent to $\neg\langle m\rangle\,\neg\phi$
- define modal depth in a formula ϕ, denoted by $\mathrm{md}\,\phi$ as the maximum level of nesting of modalities in ϕ

Semantics

Semantics

A model for the language is a pair $\mathfrak{M} = \langle \mathbb{F}, V\rangle$, where

- $\mathfrak{F} = \langle W, \{R_m\}_{m \in \mathrm{MOD}}\rangle$
  is a Kripke frame, ie, a non empty set W and a family of binary relations over W, one for each modality symbol $m \in \mathrm{MOD}$.
  Elements of W are called points, states, worlds or simply vertices in directed graphs.
- $V : \mathrm{PROP} \longrightarrow \mathcal{P}(W)$ is a valuation.

Semantics

Satisfaction: for a model $\mathfrak{M}$ and a point w

$\mathfrak{M}, w \models \mathsf{true}$		
$\mathfrak{M}, w \not\models \mathsf{false}$		
$\mathfrak{M}, w \models p$	iff	$w \in V(p)$
$\mathfrak{M}, w \models \neg\phi$	iff	$\mathfrak{M}, w \not\models \phi$
$\mathfrak{M}, w \models \phi_1 \wedge \phi_2$	iff	$\mathfrak{M}, w \models \phi_1$ and $\mathfrak{M}, w \models \phi_2$
$\mathfrak{M}, w \models \phi_1 \rightarrow \phi_2$	iff	$\mathfrak{M}, w \not\models \phi_1$ or $\mathfrak{M}, w \models \phi_2$
$\mathfrak{M}, w \models \langle m \rangle \phi$	iff	there exists $v \in W$ st $vR_m w$ and $\mathfrak{M}, v \models \phi$
$\mathfrak{M}, w \models [m]\, \phi$	iff	for all $v \in W$ $vR_m w$ implies $\mathfrak{M}, v \models \phi$

Semantics

Satisfaction

A formula ϕ is

- satisfiable in a model $\mathfrak{M}$ if it is satisfied at some point of $\mathfrak{M}$
- globally satisfied in $\mathfrak{M}$ ($\mathfrak{M} \models \phi$) if it is satisfied at all points in $\mathfrak{M}$
- valid ($\models \phi$) if it is globally satisfied in all models
- a semantic consequence of a set of formulas Γ ($\Gamma \models \phi$) if for all models $\mathfrak{M}$ and all points w, if $\mathfrak{M}, w \models \Gamma$ then $\mathfrak{M}, w \models \phi$

6. A RELATIONAL INTERLUDE. Recall the relational calculus studied before. Let R_m stand for the accessibility relation associated to modality m. Then,

$$w \models \langle m \rangle \varphi \;\Leftrightarrow\; \varphi\,(\models^\circ \cdot R_m)\, w$$
$$w \models [m]\, \varphi \;\Leftrightarrow\; \varphi\,(R_m^\circ / \models)\, w$$

where $c(R \setminus S)a \equiv \langle \forall\ b\ ::\ bRa \Rightarrow bRc \rangle$.

To do. Express relationally, through similar constructions, the remaining clauses of the satisfaction relation.

Proof system **K**

Minimal modal logic

- all formulas with the form of a propositional tautology (including formulas which contain modalities but are truth-functionally tautologous)
- all instances of the axiom schema:

$$\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$$

- two proof rules:

$$\text{if } \vdash \phi \text{ and } \vdash \phi \to \psi \text{ then } \vdash \psi \quad \text{(modus ponens)}$$
$$\text{if } \vdash \phi \text{ then } \vdash \Box\phi \quad \text{(generalization)}$$

What's in a logic? Modal Logic Bisimulation and modal equivalence Temporal logic Hybrid logic

Variants

Normal modal logics are axiomatic extensions to **K**

- different applications of modal logic typically validate different modal axioms;
- a normal modal logic is identified with the set of formulas it generates; it is said to be consistent if it does not contain all formulas. This identification immediately induces a lattice structure on the set of all such logics.

What's in a logic? Modal Logic Bisimulation and modal equivalence Temporal logic Hybrid logic

Variants

Modal axioms reflect properties of accessibility relations:

- transitive frames: $\Box\phi \to \Box\,\Box\,\phi$
- simple frames: $\Diamond\phi \to \Box\phi$
- frames consisting of isolated reflexive points: $\phi \leftrightarrow \Box\phi$
- frames consisting of isolated irreflexive points: $\Box$false

But there are classes of frames which are not modally definable, eg, connected, irreflexive, containing a isolated irreflexive point

7. FRAME DEFINABILITY. Richer variants to the minimal modal logic can be characterised at the level of frames (a *frame* being the pair formed by a set of states and an accessibility relation). We say that a formula is *valid on a frame* $\mathfrak{F} = \langle W, R\rangle$ if it is valid at any point $w \in W$ for each valuation of its propositional symbols. For example, the axiom scheme $\Box\phi \Rightarrow \Box\,\Box\,\phi$ is valid in any model whose accessibility relation is transitive, *i.e.* is valid for all *transitive frames*, which are the ones suitable to express, e.g., the flow of time.

8. EXERCISE. Resorting to the semantics definition, prove the following are valid formulas in propositional modal logic:

$$\begin{aligned}
\Box(\varphi \wedge \psi) &\Leftrightarrow \Box\varphi \wedge \Box\psi \\
\Diamond(\varphi \vee \psi) &\Leftrightarrow \Diamond\varphi \vee \Diamond\psi \\
\Diamond\varphi \wedge \Box\psi &\Rightarrow \Diamond(\varphi \wedge \psi) \\
\Diamond(\varphi \wedge \psi) &\Rightarrow \Diamond\varphi \wedge \Diamond\psi \\
\Box\varphi \vee \Box\psi &\Rightarrow \Box(\varphi \vee \psi) \\
\Box(\varphi \to \psi) &\Rightarrow \Box\varphi \to \Box\psi
\end{aligned}$$

9. EXERCISE. Verify the soundness of the following inference rules for propositional model logic:

$$\frac{\varphi}{\Box\varphi}\,(GEN) \qquad \frac{\varphi \to \psi}{\Diamond\varphi \to \Diamond\psi}\,(MON1) \qquad \frac{\varphi \to \psi}{\Box\varphi \to \Box\psi}\,(MON2)$$

Examples

An automaton

$$A \;=\; \begin{tikzcd} 1 \arrow[r, "a"] & 2 \arrow[r, "b"] \arrow[loop above, "a"] & 3 \arrow[loop below, "b"] \end{tikzcd}$$

- two modalities $\langle a \rangle$ and $\langle b \rangle$ to explore the corresponding classes of transitions
- note that
$$1 \models \langle a \rangle \cdots \langle a \rangle \langle b \rangle \cdots \langle b \rangle\, t$$
where t is a proposition valid only at the (terminal) state 3.
- all modal formulas of this form correspond to the strings accepted by the automaton, i.e. in language $\mathcal{L} = \{a^m b^n \mid m, n > 0\}$

Examples

$(P, <)$ a strict partial order with infimum 0

- $P, x \models \Box$false if x is a maximal element of P
- $P, 0 \models \Diamond \Box\, \mathit{false}$ iff ...
- $P, 0 \models \Box \Diamond \Box\, \mathit{false}$ iff ...

10. Answer. The second assertion captures the fact that P has a maximal element; the third that every element is below a maximal element.

Examples

Process logic (Hennessy-Milner logic)

- PROP $= \emptyset$
- $W = \mathbb{P}$ is a set of states, typically process terms, in a labelled transition system
- each subset $K \subseteq Act$ of actions generates a modality corresponding to transitions labelled by an element of K

Assuming the underlying LTS $\mathfrak{F} = \langle \mathbb{P}, \{p \xrightarrow{K} p' \mid K \subseteq Act\}\rangle$ as the modal frame, satisfaction is abbreviated as

$$p \models \langle K \rangle\, \phi \quad \text{iff} \quad \exists_{q \in \{p' \mid p \xrightarrow{a} p' \,\wedge\, a \in K\}} \,.\, q \models \phi$$

$$p \models [K]\, \phi \quad \text{iff} \quad \forall_{q \in \{p' \mid p \xrightarrow{a} p' \,\wedge\, a \in K\}} \,.\, q \models \phi$$

11. Exercise. Hennessy-Milner logic, often in extended, temporal variants, is used to express properties of processes specified in a process algebra (see *e.g.* [?] or [?]). Typical properties include:

- *inevitability of* a: $\langle - \rangle\, \mathsf{true} \wedge [-a]\, \mathsf{false}$
- *progress*: $\langle - \rangle\, \mathsf{true}$
- *deadlock or termination*: $[-]\, \mathsf{false}$

where Act is abbreviated to $-$, and $Act \setminus K$ to $-K$.
What does express $\langle - \rangle\, \mathsf{false}$ and $[-]\, \mathsf{true}$?

12. Exercise Consider the following requirements concerning a management system for a taxi network, and translate into Hennessy-Milner logic.

- $\phi_0 =$ *In a taxi network, a car can collect a passenger or be allocated by the Central to a pending service*
- $\phi_1 =$ *This applies only to cars already on service*
- $\phi_2 =$ *If a car is allocated to a service, it must first collect the passenger and then plan the route*
- $\phi_3 =$ *On detecting an emergence the taxi becomes inactive*
- $\phi_4 =$ *A car on service is not inactive*

Solution.

- $\phi_0 = \langle rec, alo \rangle\, \mathsf{true}$
- $\phi_1 = [onservice]\, \phi_0$
- $\phi_2 = [alo]\, \langle rec \rangle\, \langle plan \rangle\, \mathsf{true}$
- $\phi_3 = [sos]\, [-]\, \mathsf{false}$
- $\phi_4 = [onservice]\, \langle - \rangle\, \mathsf{true}$

Examples

Temporal logic

- $\langle T, < \rangle$ where T is a set of time points (instants, execution states , ...) and $<$ is the earlier than relation on T.
- Thus, $\Box\varphi$ (respectively, $\Diamond\varphi$) means that φ holds in all (respectively, some) time points.
- origin: Arthur Prior, an attempt to *deal with temporal information from the inside, capturing the situated nature of our experience and the context-dependent way we talk about it*

Examples

$\langle T, < \rangle$

The structure of time is a strict partial order
(i.e., a transitive and asymmetric relation)

For any such structure, a new modality, $\bigcirc$, can be defined based on the cover relation $\lessdot$ for $<$ (*i.e.*, the smallest relation whose transitive closure is $<$). Thus,

$$t \models \bigcirc\phi \quad \text{iff} \quad \forall_{t' \in \{p' | t \lessdot t'\}} \,.\, t' \models \phi$$

$$t \models \Box\phi \quad \text{iff} \quad \forall_{t' \in \{p' | t < t'\}} \,.\, t' \models \phi$$
$$t \models \Diamond\phi \quad \text{iff} \quad \exists_{t' \in \{p' | t < t'\}} \,.\, t' \models \phi$$

Examples

... but typical structures, however, are

Linear time structures

- linear: $\langle \forall\, x, y \;:\; x, y \in T :\; x = y \wedge x < y \wedge y < x \rangle$.
- discrete: for each $t \in T$, i) if there is a $u > t$ there is a first such u; ii) if there is a $u < t$ there is a last such u.
- dense: if for all $t, x \in T$, if $x < t$ there is a $v \in T$ such that $x < v < t$.
- Dedekind complete: if for all $S \subseteq T$ non-empty and bounded above, there is a lest upper bound in T.
- continuous: if it is both dense and Dedekind complete

13. THE NEXT INSTANT. For a linear temporal structure $\bigcirc\phi$ refers to the validity of ϕ in *the* (unique) next time point. Of course in an arbitrary discrete structure, more than one next time point may exist. In any case, however, the formula $\bigcirc\phi$ does not imply the existence of a next instant: the dual operator $\neg \bigcirc \neg$ should be used when this is wanted. In a sense, temporal logic over a discrete structure introduces two modalities: one corresponding to $<$, upon which $\Diamond$ and $\Box$ are defined; another to its cover $\lessdot$. In the language of § **??** one should write $\langle < \rangle\, \phi$, $[<]\, \phi$ and $[\lessdot]\, \phi$, for $\Diamond\, \phi$, $\Box\, \phi$ and $\bigcirc\phi$, respectively.

Examples

Epistemic logic (J. Hintikka, 1962)

- W is a set of agents
- $\alpha \models i$ means i is the current knowledge of agent i
- $\alpha \models \Box j$ means the agent knows that j (in the sense that at each alternative epistemic situation information j is known)
- $\alpha \models \Diamond j$ means the agent knows that knowledge j is consistent with what the agent knows (is an epistemically acceptable alternative)

The first order connection

Boxes and diamonds are essentially a macro notation to encode quantification over accessible states in a point free way.

The standard translation

... to first-order logic expands these macros:

$$
\begin{aligned}
ST_x(p) &= P\,x \\
ST_x(\text{true}) &= \text{true} \\
ST_x(\text{false}) &= \text{false} \\
ST_x(\neg\phi) &= \neg ST_x(\phi) \\
ST_x(\phi_1 \wedge \phi_2) &= ST_x(\phi_1) \wedge ST_x(\phi_1) \\
ST_x(\phi_1 \rightarrow \phi_2) &= ST_x(\phi_1) \rightarrow ST_x(\phi_1) \\
ST_x(\langle m\rangle\,\phi) &= \langle \exists\ y\ ::\ (yR_m x\ \wedge\ ST_y(\phi))\rangle \\
ST_x([m]\,\phi) &= \langle \forall\ y\ ::\ (yR_m x\ \rightarrow\ ST_y(\phi))\rangle
\end{aligned}
$$

The first order connection

Lemma

For any ϕ, $\mathfrak{M}$ and point w in $\mathfrak{M}$,

$$\mathfrak{M}, w \models \phi \quad \text{iff} \quad \mathfrak{M} \models ST_x(\phi)[x \leftarrow w]$$

Note

Note how the (unique) free variable x in ST_x mirrors in first-order the internal perspective: assigning a value to x corresponds to evaluating the modal formula at a certain state.

The first order connection

The standard translation provides a bridge between modal logic and classical logic which makes possible to transfer results from one side to the other. For example,

Compactness

If Φ is a set of basic modal formulas and every finite subset of Φ is satisfiable, then Φ itself is satisfiable.

Löwenheim-Skolem

If Φ is a set of basic modal formulas satisfiable in at least one infinite model, then it is satisfiable in models of every infinite cardinality.

14. A BALANCE. The standard translation discussed here is the tool to formalise the idea that modal logics correspond to particularly well-behaved fragments of first order logic. This always entails a balance between computational complexity and expressive power. A similar balance arise between first and second-order logic, the latter loosing in axiomatizability wrt the former, but having an increased expressive power.

15. EXERCISE.

- Explain how propositional symbols and modalities are translated to first-order logic?
- In what sense can modal logic be regarded as a *pointfree* version of a FOL fragment?
- Compute $ST_x(p \Rightarrow \langle m \rangle\, p)$

Summing up

- Propositional modal languages are syntactically simple languages that offer a pointfree notation for talking about relational structures
- They do this from the inside, using the modal operators to look for information at accessible states
- Regarded as a tool for talking about models, any basic modal language can be seen as a fragment of first-order language
- The standard translation systematically maps modal formulas to first-order formulas (in one free variable) and makes the quantification over accessible states explicit

3 Bisimulation and modal equivalence

Bisimulation

Definition

Given two models $\mathfrak{M} = \langle\langle W, R\rangle, V\rangle$ and $\mathfrak{M}' = \langle\langle W', R'\rangle, V'\rangle$, a bisimulation is a non-empty binary relation $S \subseteq W \times W'$ st whenever wSw' one has that

- points w and w' satisfy the same propositional symbols
- if vRw, then there is a point v' in $\mathfrak{M}'$ st $v'Rw'$ and vSv' (zig)
- if $v'R'w'$, then there is a point v in $\mathfrak{M}$ st vRw and vSv' (zag)

Bisimulation

Definition

- Bisimulations can be used to expand or contract models (cf via tree unraveling and contraction)
- Bisimulation vs model constructions (disjoint union, generated submodels and bounded morphisms)

Note

Note the relation to the notion of bisimulation in transition systems, independently discovered by Park (1982) in Computer Science.

What's in a logic? Modal Logic **Bisimulation and modal equivalence** Temporal logic Hybrid logic

Invariance and definability

Lemma (invariance: bisimulation implies modal equivalence)

Given two models $\mathfrak{M} = \langle\langle W, R\rangle, V\rangle$ and $\mathfrak{M}' = \langle\langle W', R'\rangle, V'\rangle$, and a bisimulation $S \subseteq W \times W'$, if two points w, w' are related by S, i.e. wSw', then w, w' satisfy the same basic modal formulas.

Applications

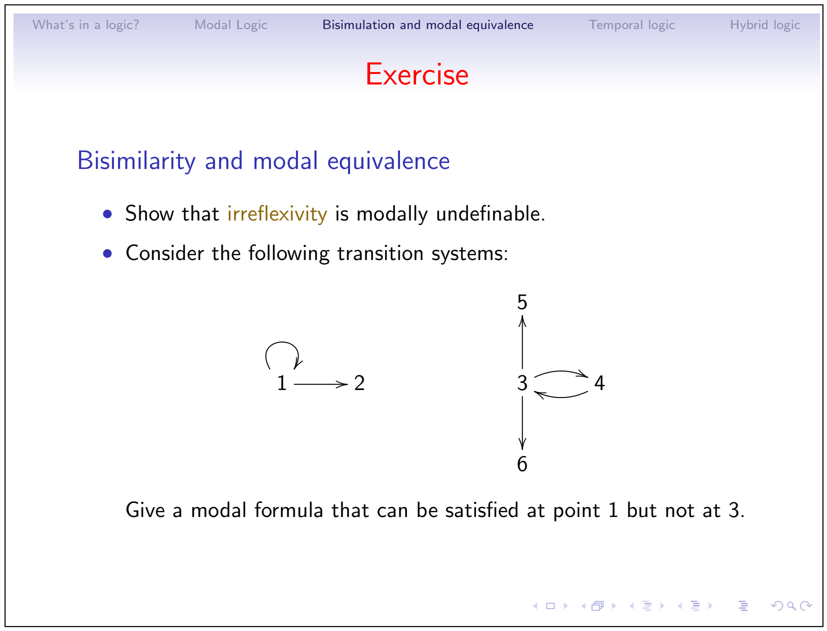
- to prove bisimulation failures
- to show the undefinability of some structural notions, e.g. irreflexivity is modally undefinable
- to show that typical model constructions are satisfaction preserving
- ...

16. Proof (invariance). The proof is by induction on the structure of modal formulas. The base case, for propositional symbols, is immediate from the the definition of bisimulation. Similarly the inductive arguments of the Boolean connectives are straightforward. Consider, thus, the case $\langle m\rangle\,\phi$. We want to show that if $\mathfrak{M}, w \models \langle m\rangle\,\phi$ and wSw', then $\mathfrak{M}', w' \models \langle m\rangle\,\phi$. Clearly,

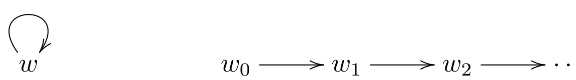
$$
\begin{array}{ll}
 & \mathfrak{M}, w \models \langle m\rangle\,\phi \\
\Leftrightarrow & \qquad \{ \text{ satisfaction } \} \\
 & \text{there exists } v \in W \text{ st } vR_m w \text{ and } \mathfrak{M}, v \models \phi \\
\Leftrightarrow & \qquad \{ \; wSw' \text{ (zig condition) } \} \\
 & \text{there exists } v' \in \mathfrak{M}' \text{ st } v'R_m w' \text{ and } vSv' \\
\Leftrightarrow & \qquad \{ \; \mathfrak{M}', v' \models \phi \text{ because } vSv' \text{ and IH } \} \\
 & \mathfrak{M}', w' \models \langle m\rangle\,\phi
\end{array}
$$

Now, suppose $\mathfrak{M}', w' \models \langle m\rangle\,\phi$. To conclude, $\mathfrak{M}, w \models \langle m\rangle\,\phi$ the argument is similar to the one used above, now resorting to the bisimulation (zag) condition.

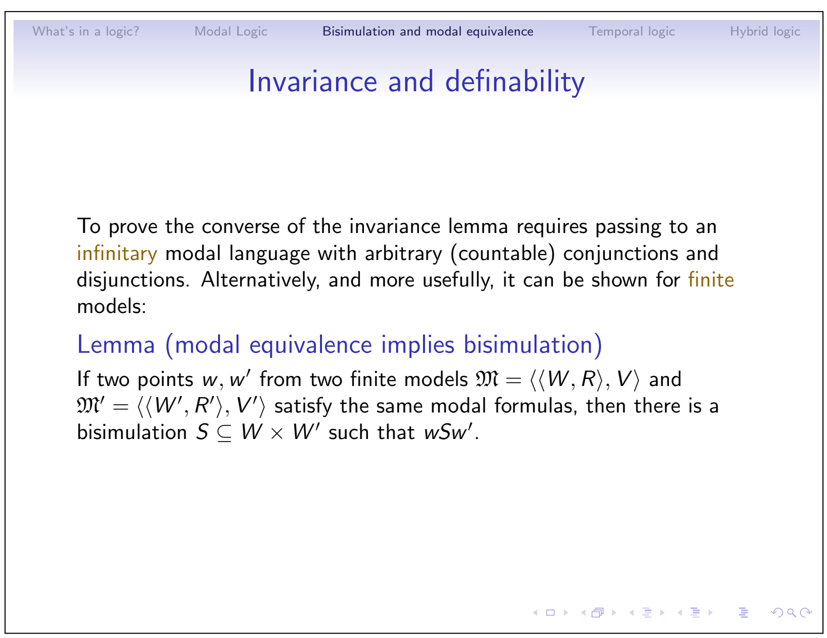
$\square$



17. ANSWER. For the first question consider states w and w_0 in the following transition systems, with and without a reflexive arrow. Clearly, both states are bisimilar (*i.e.* relation $S = \{\langle w, w_i\rangle \mid i \geq 0\}$ is a bisimulation) and then, by the invariance lemma, there is no modal formula able to distinguish between them.



For the second question, formula $\Box(\Box\mathsf{false} \vee \Diamond\Box\,\mathsf{false})$ is satisfied at state 1 but not at state 3.



18. PROOF. Without loss of generality we shall consider models with a single relation R (and thus, restrict our attention to a language with a single modality $\Diamond$). Define a relation S on the states of $\mathfrak{M}$ and $\mathfrak{M}'$ as follows

$$wSw' \;\Leftrightarrow\; w \text{ and } w' \text{ satisfy the same modal formulas.}$$

Clearly, S is modal equivalence. We want to prove that it is also a bisimulation. Let wSw'. Obviously, they satisfy the same propositions, which is the first condition for S to be a bisimulation. We shall consider now the (zig) condition. Let vRw and suppose there is no v' in $\mathfrak{M}'$ such that $v'R'w'$ and vSv'. Consider the set $T = \{u \mid uR'w'\}$ of R'-successors of w'. This set is not empty because w has a successor v (and therefore, $\mathfrak{M}, w \models \Diamond\mathsf{true}$) and wSw' (and therefore, $\mathfrak{M}', w' \models \Diamond\mathsf{true}$ as well). Moreover, T is finite and can be enumerated:

$$T = \{u_0, u_1, u_2, \cdots\}.$$

By hypothesis, for each $u_i \in T$, there exists a formula ϕ_i such that $\mathfrak{M}, v \models \phi_i$ but $\mathfrak{M}', u\neg \models \phi_i$. Thus,

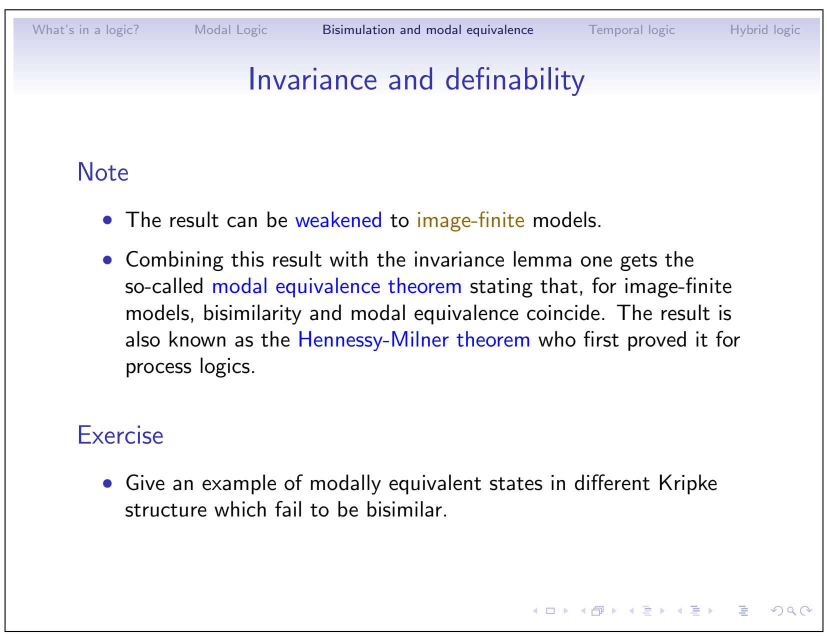
$$\mathfrak{M}, w \models \Diamond(\phi_1 \wedge \phi_2 \wedge \cdots \wedge \phi_n)$$

but

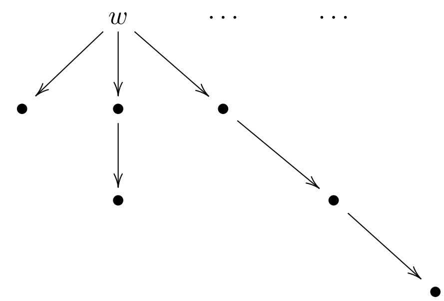
$$\mathfrak{M}', w'\neg \models \Diamond(\phi_1 \wedge \phi_2 \wedge \cdots \wedge \phi_n)$$

which contradicts the assumption that wSw'. Therefore, relation S satisfies the (zig) condition. A similar arguments shows it also satisfies the (zag) condition.

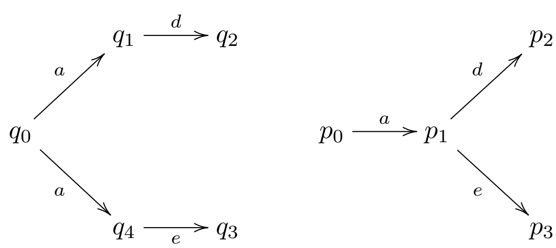
$\square$



19. Answer. Consider state w in the following tree which has ω branches with length $1, 2, 3, \cdots$. Clearly, w is modally equivalent to a state in the root of a similar tree which additionally contains an infinite branch. Such states, however, are not bisimilar.



20. EXERCISE. Prove that states q_0 and p_0 are not bisimilar by presenting a formula in the suitable process logic which holds for one of them but not for the other.



Invariance and definability

Lemma (modal logic vs first-order)

The following are equivalent for all first-order formulas $\phi(x)$ in one free variable x:

1. $\phi(x)$ is invariant for bisimulation.
2. $\phi(x)$ is equivalent to the standard translation of a basic modal formula.

Therefore:
the basic modal language corresponds to the fragment of their first-order correspondence language that is invariant for bisimulation

Invariance and definability

- the basic modal language (interpreted over the class of all models) is computationally better behaved than the corresponding first-order language (interpreted over the same models)
- ... but clearly less expressive

	model checking	satisfiability
ML	PTIME	PSPACE-complete
FOL	PSPACE-complete	undecidable

What are the trade-offs? Can this better computational behaviour be lifted to more expressive modal logics?

Richer modal logics

can be obtained in different ways, e.g.

- axiomatic extensions
- introducing more complex satisfaction relations
- support novel semantic capabilities
- ...

Examples

- richer temporal logics
- hybrid logic
- modal μ-calculus

4 Temporal logic

Temporal logics with $\mathcal{U}$ and $\mathcal{S}$

Until and Since

$$\mathfrak{M}, w \models \phi\,\mathcal{U}\,\psi \quad \text{iff} \quad \text{there exists } v \in W \text{ st } vRw \text{ and } \mathfrak{M}, v \models \psi, \text{ and for all } u \text{ st } uRw \text{ and } vRu, \text{ one has } \mathfrak{M}, u \models \phi$$

$$\mathfrak{M}, w \models \phi\,\mathcal{S}\,\psi \quad \text{iff} \quad \text{there exists } v \in W \text{ st } wRv \text{ and } \mathfrak{M}, v \models \psi, \text{ and for all } u \text{ st } uRv \text{ and } wRu, \text{ one has } \mathfrak{M}, u \models \phi$$

- note the $\exists\forall$ qualification pattern: these operators are neither diamonds nor boxes.
- more expressive — e.g. helpful to express guarantee properties, e.g. some event will happen, and a certain condition will hold until then

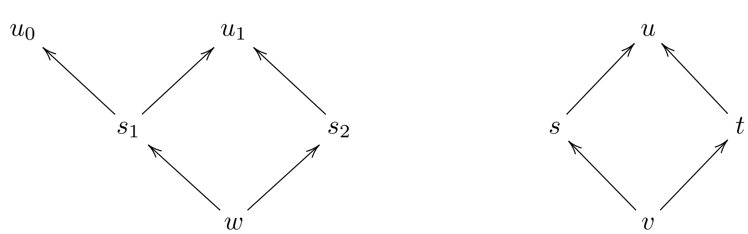
Exercise

Temporal logics

- Show that $\mathcal{U}$ is modally undefinable.
  Hint Consider the following transition structures and formula false$\mathcal{U}$ true:

- Would this be the case if we restrict ourselves to transitive, irreflexive models?

21. Answer. Yes. Consider the two models $\mathfrak{M}$ and $\mathfrak{M}'$ below, and suppose $\mathfrak{M}, s_1 \models \phi$, $\mathfrak{M}, u_0 \models \psi$, $\mathfrak{M}, u_1 \models \psi$, $\mathfrak{M}', s \models \phi$, and $\mathfrak{M}', u \models \psi$. Clearly states w and v are bisimilar. However, $\mathfrak{M}, w \models \phi\,\mathcal{U}\,\psi$, which is not the case for v in $\mathfrak{M}'$.



What's in a logic? Modal Logic Bisimulation and modal equivalence Temporal logic Hybrid logic

Linear temporal logic (LTL)

$$\phi \; := \; \mathsf{true} \mid p \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \bigcirc\phi \mid \phi_1 \,\mathcal{U}\, \phi_2$$

mutual exclusion	$\Box(\neg c_1 \vee \neg c_2)$
liveness	$\Box\Diamond c_1 \wedge \Box\Diamond c_2$
starvation freedom	$(\Box\Diamond w_1 \rightarrow \Box\Diamond c_1) \wedge (\Box\Diamond w_1 \rightarrow \Box\Diamond c_1)$
progress	$\Box(w_1 \rightarrow \Diamond c_1)$
weak fairness	$\Diamond\,\Box\, w_1 \;\rightarrow \Box\Diamond c_1$
eventually forever	$\Diamond\,\Box\, w_1$

- First temporal logic to reason about reactive systems [Pnueli, 1977]
- Formulas are interpreted over execution paths
- Express linear-time properties

22. Linear temporal logic. Linear temporal logic (LTL) was introduced in by A. Pnueli [**?**] for reasoning about reactive systems. A number of variants have been explored since then, of which L. Lamport TLA [**?**] is probably the most used in industry. LTL is a logic for formalising properties of a program execution path assuming a linear structure for time (each moment has a single successor): modality $\Diamond$ refers to a future point in an execution path, while $\Box$ captures the fact that a property holds in the current moment and forever in the future.

As an example of what can be expressed in LTL, the table in the slide concerns the mutual exclusion problem of two concurrent processes, T_1 and T_2. It is supposed that propositions c_i and w_i are valid when process i is in, or waiting to enter into its critical section, respectively. The first property is a typical *safety* requirement, while the second is a *liveness* property (each process is infinitely often in its critical section). The third one is a weaker version of the latter: every waiting process will eventually enter its critical section. The fourth example is typical in specifying communications (*if a request is sent, a message will came*). The fairness requirement states that if the first process is continuously waiting to enter it s critical section, it will be entering infinitely often. Finally, the latter is an example of a liveness property asserting a (future) invariant. Note that fairness constraints can be expressed in LTL along with any other properties of transition systems.

LTL formulas are usually interpreted over an execution path p, i.e. a sequence of states, by considering a model whose set of sates is formed by all the suffixes of p and the accessibility relations (cf, § **??**) are taken as the *suffix* order ($w'Rw$ off w' is a suffix of w) and its cover. For example, for a propositional symbol $p \in PROP$, we get

$$l \models p \;\Leftrightarrow\; \mathsf{hd}\, l \in V(l) \;,$$

and

$$\begin{aligned} & l \models \bigcirc\,\phi \;\Leftrightarrow\; \mathsf{tl}\, l \\ & l \models \Diamond\phi \;\Leftrightarrow\; \exists_{j\geq 0}.\, l(j..) \models \phi \\ & l \models \Box\,\phi \;\Leftrightarrow\; \forall_{j\geq 0}.\, l(j..) \models \phi \\ & l \models \phi\,\mathcal{U}\,\psi \;\Leftrightarrow\; \exists_{j>0}\forall_{0\leq i<j}\;.\,(l(j..) \models \psi \;\wedge\; l(i..) \models \phi) \end{aligned}$$

where, for an index $k > 0$, where $l(k)$ denotes the kth element in sequence l, and $l(k..)$ stands for the suffix of l starting at position k in the original path l. Note that $\mathsf{hd}\, l = l(0)$ and $\mathsf{tl}\, l = l(1..)$.

The interpretation of LTL formulas over p can also be given in terms it individual states as follows:

$$
\begin{aligned}
& l \models p \;\Leftrightarrow\; l(0) \in V(p) \\
& l \models \bigcirc\, \phi \;\Leftrightarrow\; l(1) \\
& l \models \Diamond \phi \;\Leftrightarrow\; \exists_{j \geq 0} .\, l(j) \models \phi \\
& l \models \Box\, \phi \;\Leftrightarrow\; \forall_{j \geq 0} .\, l(j) \models \phi \\
& l \models \phi\, \mathcal{U}\, \psi \;\Leftrightarrow\; \exists_{j \geq 0} \,.\, (l(j) \models \psi \;\wedge\; (\forall_{0 \leq i < j} l(i) \models \phi))
\end{aligned}
$$

More generally, one may turn attention to all possible execution paths starting at a state s and define state satisfaction as

$$s \models \phi \;\text{ iff }\; \forall_{l \in \mathsf{Paths}(s)} \,.\, l \models \phi \tag{1}$$

But this opens the way to a different logic in which modal reasoning explores a branching time structure.

23. Exercise. Consider an elevator servicing N floors. At each floor assume the existence of a call-button and an indicator light that is on when the elevator has been called. Specify in LTL the following properties, also defining the atomic propositions you may find necessary for the specification;

- Every request, from any floor, will be served sometime.
- At each floor the doors are never open unless the elevator is there.
- Only a request is served at a time.
- Whenever an even floor issues a request it is served at once: the elevator does not stop on the way there.
- The elevator, if not serving a request, always returns to the bottom floor.

24. Exercise. Prove the following are valid formulas in LTL:

$$
\begin{gathered}
\Diamond\Diamond\phi \;\Leftrightarrow\; \Diamond\phi \\
\Diamond\,\Box\,\Diamond\phi \;\Leftrightarrow\; \Box\Diamond\phi \;\text{ and }\; \Box\,\Diamond\,\Box\,\phi \;\Leftrightarrow\; \Diamond\,\Box\,\phi \\
\Box(\phi \wedge \psi) \;\Leftrightarrow\; \Box\phi \wedge \Box\psi \\
\Diamond(\phi \vee \psi) \;\Leftrightarrow\; \Diamond\phi \vee \Diamond\psi \\
\Box\phi \;\Leftrightarrow\; \phi \wedge \bigcirc\,\Box\,\phi \;\text{ and }\; \Diamond\phi \;\Leftrightarrow\; \phi \vee \bigcirc\Diamond\phi \\
\phi\,\mathcal{U}\,(\phi\,\mathcal{U}\,\psi) \;\Leftrightarrow\; (\phi\,\mathcal{U}\,\psi)\,\mathcal{U}\,\psi \;\Leftrightarrow\; \phi\,\mathcal{U}\,\psi
\end{gathered}
$$

What's in a logic? Modal Logic Bisimulation and modal equivalence Temporal logic Hybrid logic

Computational tree logic (CTL, CTL*)

state formulas to express properties of a state:

$$\Phi \ := \ \mathsf{true} \mid \Phi \wedge \Phi \mid \neg\Phi \mid \exists\phi \mid \forall\phi$$

path formulas to express properties of a path:

$$\phi \ := \ \bigcirc\Phi \mid \Phi\,\mathcal{U}\,\Psi$$

mutual exclusion	$\forall\,\Box\,(\neg c_1 \vee \neg c_2)$
liveness	$\forall\,\Box\,\forall\Diamond c_1 \wedge \forall\,\Box\,\forall\Diamond c_2$
order	$\forall\,\Box\,(c_1 \vee \forall\,\forall\bigcirc\, c_2)$

- Branching time structure encode transitive, irreflexive but not necessarily linear flows of time
- flows are trees: past linear; branching future

25. Branching temporal structures. Equation (**??**) in § **??** paves the way for a more general way to look at temporal properties, explicitly introducing in the logic quantification over execution paths. Clarke & Emerson in a seminal paper [**?**] introduced CTL, a temporal logic that is interpreted not over a linear time structure but over a *branching* one: in other words, replacing (infinite) sequences of states by (infinite) trees of states. Each traversal of a such a tree, starting at its root, corresponds to an execution path. To explore such a structure (e.g. to assert that there exists a computation in which formula $\Diamond\phi$ holds) *path* (existential and universal) quantifiers are introduced thus inducing a double formula structure. CTL syntax includes, therefore,

- *path* formulas, whose main connective is a LTL operator, to express properties of a path,
- *state* formulas, witch include the above mentioned path quantifiers, to express properties of a state.

The way both formulas interact is not arbitrary: formally, assuming a set AP of atomic propositions, *state formulas* are generated by the following grammar:

$$\Phi \ := \ \mathsf{true} \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\phi \mid \forall\phi$$

where ϕ is a *path formula* built according to

$$\phi \ := \ \bigcirc\Phi \mid \Phi_1\,\mathcal{U}\,\Phi_2$$

where Φ, Ψ are state formulas. Note that the remaining Boolean connectives can be defined by abbreviation. One may also define $\Diamond\Phi \stackrel{\text{abv}}{=} (\mathsf{true}\,\mathcal{U}\,\Phi)$. However, what corresponds to a box connective cannot be obtained as in LTL by $\Box\Phi \stackrel{\text{abv}}{=} \neg\Diamond\neg\Phi$ since the grammar precludes propositional connectives to be applied to path formulas. What we get, however, is a richer ontology of temporal expressions:

Φ potentially holds	$\exists\Diamond\Phi$
Φ in inevitable	$\forall\Diamond\Phi$
Potentially always Φ holds	$\exists\,\Box\,\Phi$
Invariantly Φ holds	$\forall\,\Box\,\Phi$

Note the dualities: $\exists\,\Box\,\Phi = \neg\forall\Diamond\neg\Phi$ and $\forall\,\Box\,\Phi = \neg\exists\Diamond\neg\Phi$.

The satisfaction relation for CTL is given, for state formulas, by

$$
\begin{aligned}
s \models p &\;\Leftrightarrow\; s \in V(p) \\
s \models \neg\Phi &\;\Leftrightarrow\; s \not\models \Phi \\
s \models \Phi \wedge \Psi &\;\Leftrightarrow\; s \models \Phi \text{ and } s \models \Psi \\
s \models \exists\phi &\;\Leftrightarrow\; p \models \phi \;\text{ for some } p \in \mathsf{Paths}(s) \\
s \models \forall\phi &\;\Leftrightarrow\; p \models \phi \;\text{ for all } p \in \mathsf{Paths}(s)
\end{aligned}
$$

and, for path formulas, by

$$
\begin{aligned}
l \models \bigcirc\, \Phi &\;\Leftrightarrow\; l(1) \models \Phi \\
l \models \Phi\, \mathcal{U}\, \Psi &\;\Leftrightarrow\; \exists_{j \geq 0}\, .\, l(j) \models \psi \;\wedge\; (\forall_{0 \leq i < j} .\, l(i) \models \Phi)
\end{aligned}
$$

26. Exercise. Discuss the meaning of the following CTL formulas:

- $\exists(\Phi\, \mathcal{U}\, \Psi)$
- $\forall \Box \forall \Diamond \Phi$
- $\forall \bigcirc \Phi$

It is not difficult to see that not all valid identities in LTL can be lifted to CTL. As an example, show that $\forall\Diamond(\Phi \vee \Psi) \not\Leftrightarrow \forall\Diamond\Phi \vee \forall\Diamond\Psi$ (whereas in LTL one has $\Diamond(\phi \vee \psi) \Leftrightarrow \Diamond\phi \vee \Diamond\psi$).

27. CTL*. The expressive power of LTL and CTL cannot be compared, both logics being able to record assertions which cannot be suitably expressed by the other. The reader is referred to [?] for an extensive, formal discussion. Just as an appetiser for such a discussion consider the following transition system and assume valuation V such that $V(r) = V(t) = \{a\}$, for a an atomic proposition, $V(s) = \emptyset$.

$$
\begin{tikzcd}
r \arrow[loop above] \arrow[r] & s \arrow[r] & t \arrow[loop above]
\end{tikzcd}
$$

Clearly $r \models \Diamond \Box a$ but considering path $p = r^\omega$ is enough to falsify $s \models \forall\Diamond\forall \Box a$.

A CTL extension, called CTL* [?], allows path quantifires to be arbitrarily nested with linear time operators. For example, $\exists \Box \Diamond\Phi$ or $\forall \bigcirc \bigcirc a$ are valid formulas in CTL* but not allowed in CTL. Moreover, path quantifier $\forall$ can be defined as $\neg\exists\neg$ which is not the case in CTL. CTL* is strictly more expressive then both LTL and CTL, and can thus be specialised to both of them. We give CTL* syntax below and let as an exercise the definition of the corresponding satisfaction relation.

$$\Phi \;:=\; \mathsf{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\phi$$

where ϕ is a *path formula* built according to

$$\phi \;:=\; \Phi \mid \phi_1 \wedge \phi_2 \mid \bigcirc\phi \mid \phi_1\, \mathcal{U}\, \phi_2$$

where, as before, capital Greek letters stand for path formulas. Note that $\Diamond\phi \overset{\text{abv}}{=} \mathsf{true}\, \mathcal{U}\, \phi$, as in both CTL and LTL, and $\Box\phi \overset{\text{abv}}{=} \neg\Diamond\neg\phi$, as in LTL.

5 Hybrid logic and applications to architectural design

Hybrid logic

Motivation
Add the possibility of naming points and reason about their identity

Compare:

$$\Diamond(r \wedge p) \ \wedge \ \Diamond(r \wedge q) \ \rightarrow \ \Diamond(p \wedge q)$$

with

$$\Diamond(i \wedge p) \ \wedge \ \Diamond(i \wedge q) \ \rightarrow \ \Diamond(p \wedge q)$$

for $i \in \mathrm{NOM}$ (a nominal)

Hybrid logic

Nominals i

- Are special propositional symbols that hold exactly on one state (the state they name)
- In a model the valuation V is extended from

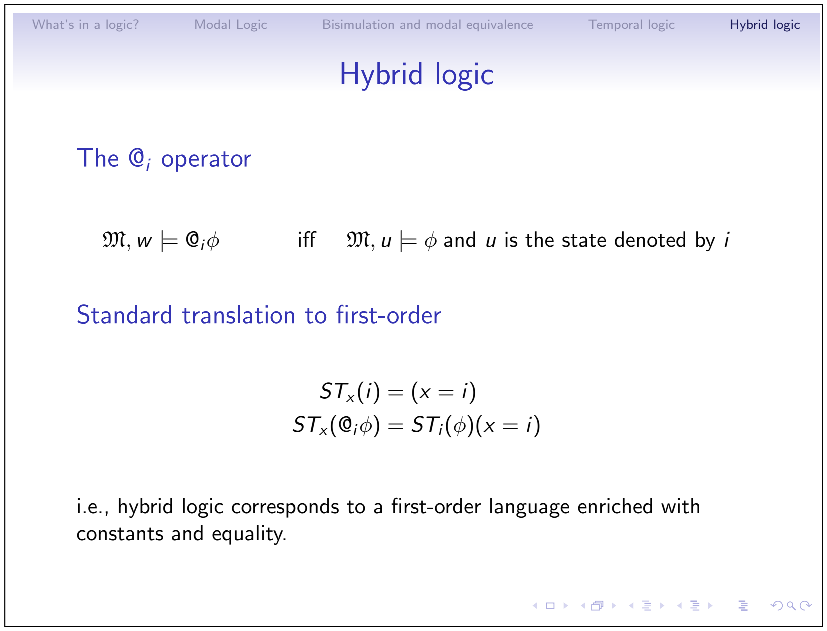
  $$V : \mathrm{PROP} \longrightarrow \mathcal{P}(W)$$

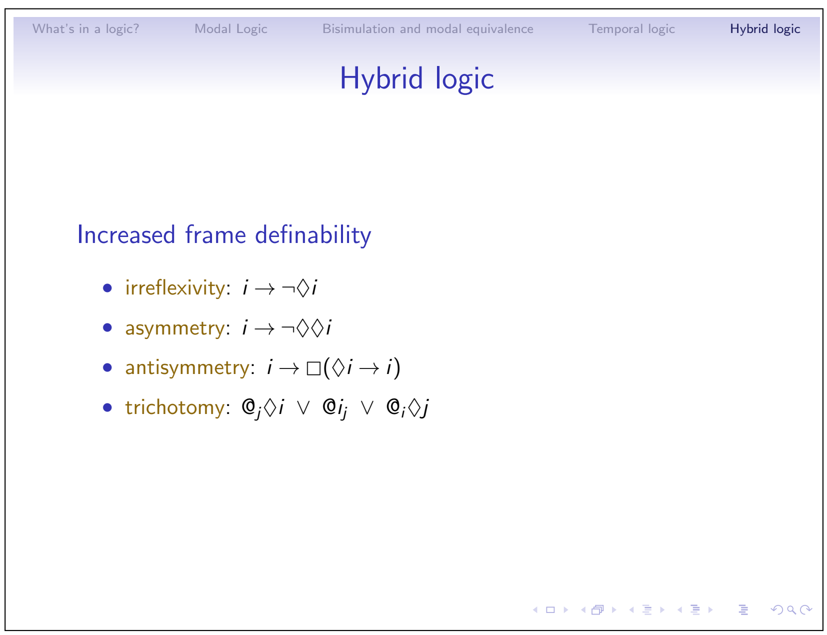
  to

  $$V : \mathrm{PROP} \longrightarrow \mathcal{P}(W) \quad \text{and} \quad V : \mathrm{NOM} \longrightarrow W$$

  where NOM is the set of nominals in the model
- Satisfaction:

  $$\mathfrak{M}, w \models i \qquad\qquad \text{iff} \ \ w = V(i)$$





28. Hybrid logic. Standard modal logic is unable to explicitly mention specific states in a model, i.e. to 'name' them. Therefore, there is no way to assert the equality between two particular states or the existence of a transition between them. Clearly, this can be done in a first order language, resorting to constants to identify whatever one wants to name, and equality. In modal logic, however, there is a number of properties, for example irreflexivity of the underlying accessibility relation, that can not be axiomatised by the same reason. Hybrid logic [**?**, **?**, **?**] overcomes this limitation by introducing a new kind of symbols NOM, called *nominals*, to make explicitly reference to states in models. Sentences are then enriched in two directions. On the one hand, each nominal is used as a simple sentence holding exclusively in the state it names; on the other hand, sentences $@_i\, \rho$, for $i \in NOM$, state the validity of ρ at the state named by i.

What's in a logic? Modal Logic Bisimulation and modal equivalence Temporal logic **Hybrid logic**

Bisimulation with nominals

Definition

Given two models $\mathfrak{M} = \langle\langle W, R\rangle, V\rangle$ and $\mathfrak{M}' = \langle\langle W', R'\rangle, V'\rangle$, a bisimulation is a non-empty binary relation $S \subseteq W \times W'$ st whenever wSw' one has that

- points w and w' satisfy the same propositional symbols and nominals
- if vRw, then there is a point v' in $\mathfrak{M}'$ st $v'Rw'$ and vSv' (zig)
- if $v'R'w'$, then there is a point v in $\mathfrak{M}$ st vRw and vSv' (zag)
- $V(i)\ R\ V'(i)$ for all nominal i (name consistency)

An invariance theorem and its dual (for image finite models) can also be proved

29. Exercise. A result relating bisimulation and modal equivalence for hybrid logic (along the same lines of §§**??** and **??**) also holds here. Prove it. Actually, hybrid logic, captures exactly the first-order fragment with constants and equality.

What's in a logic? Modal Logic Bisimulation and modal equivalence Temporal logic **Hybrid logic**

Hybrid logic

Summing up

- basic hybrid logic is a simple notation for capturing the bisimulation-invariant fragment of first-order logic with constants and equality, i.e., a mechanism for equality reasoning in propositional modal logic.
- comes cheap: up to a polynomial, the complexity of the resulting decision problem is no worse than for the basic modal language

30. Further reading and applications. For the interested reader, C.Areces & D. ten Cate chapter in the Handbook of Modal Logic [**?**] or T. Brauner book [**?**] provide a comprehensive survey of hybrid logic, its semantics, variants, applications and history. In the context of this course, however, it is worth to mention a number of recent applications of hybrid logic to modelling architectural problems. This is done in the sequel taking two examples of recent PhD theses at HASLab INESC TEC.

Hybrid logic

Applications to architectural design

- layout of coordination circuits (e.g. in Reo)
- reconfigurable architectures (parametric on a specification logic)
- hierarchical architectures (e.g. UML statecharts)
- ...

[recent research at HASLab: projects MONDRIAN and NASONI]

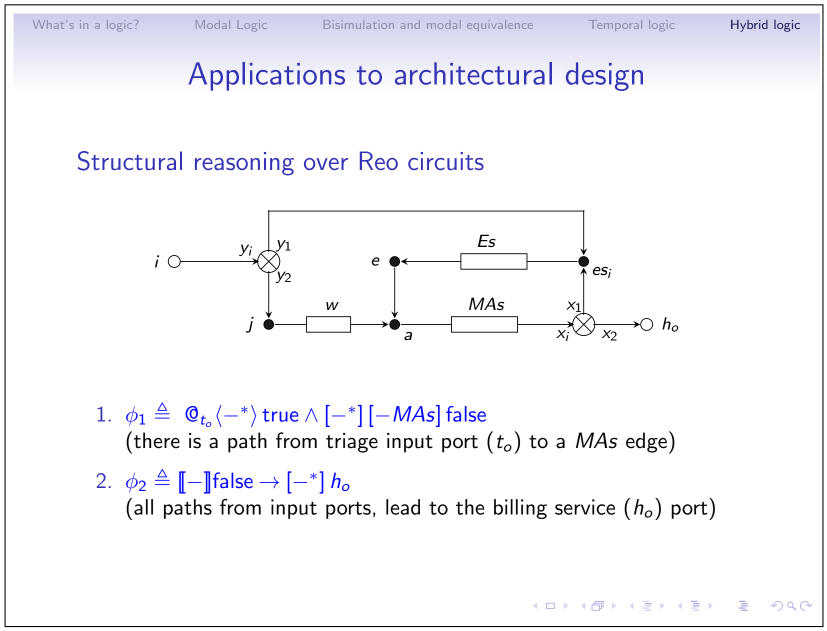
Applications to architectural design

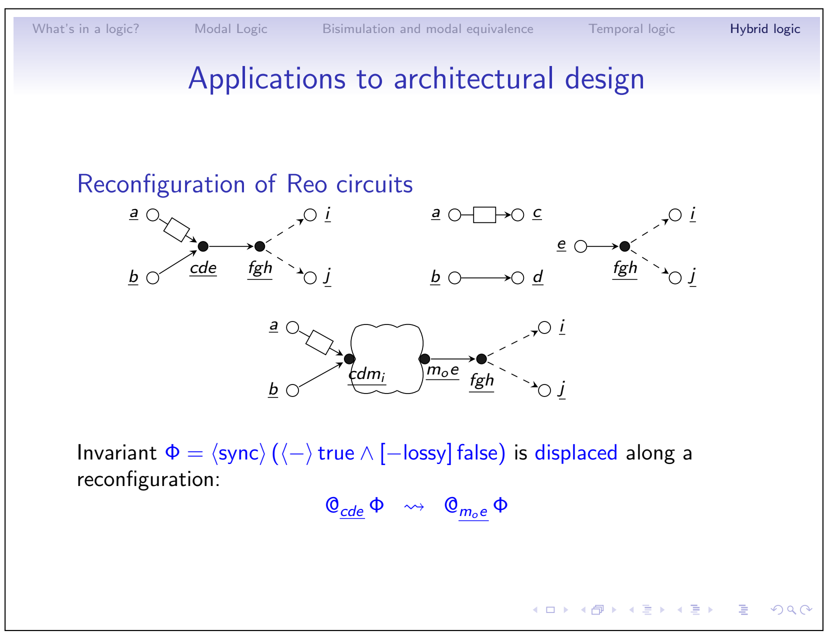
Structural reasoning over Reo circuits

$$\phi \ :== \ p \ | \ i \ | \ \neg\phi \ | \ \phi_1 \wedge \phi_2 \ | \ [K]\,\phi \ | \ [\![K]\!]\phi \ | \ @_i\phi$$

- modalities are indexed by regular expressions over channel types;
- $\langle K \rangle$ and $[K]$ (reps., $\langle\!\langle K \rangle\!\rangle$ and $[\![K]\!]$) express properties of outgoing (resp., incoming) connections from the node in which they are evaluated.

[Nuno Oliveira PhD thesis (MAP-i, 2015)]





31. Applications: Structural reasoning over Reo architectural reconfigurations. Reo [?] is a coordination model used later in this course for representing architectural configurations and interaction. In his PhD thesis [?], Nuno Oliveira introduced a hybrid language, called $Hp\mathcal{E}$, interpreted over the graph-like structure of Reo circuits, to express *structural*, or 'syntactic' properties such as

i) every $\mathsf{fifo_e}$ *channel from a node* n *is connected to at least a* lossy *channel* or
ii) node i *is a connector's output node.*

$$\phi \;:==\; p \mid i \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid [K]\,\phi \mid [\![K]\!]\phi \mid @_i\phi$$

Modalities are indexed by regular expressions over channel *types*. Operator $[K]$ quantifies universally over the edges of $\mathcal{G}(\rho)$ labelled by channel types in K; its dual $\langle K \rangle \triangleq \neg[K]\neg$ provides an existential quantification. Modalities $\langle K \rangle$ and $[K]$ express properties of *outgoing* connections from the node in which they are evaluated in a Reo circuit. Dually, modalities $\langle\!\langle K \rangle\!\rangle$ and $[\![K]\!]$ express properties of *incoming* connections. Finally, the satisfaction operator @ *redirects* the formula evaluation to the context of a specific node. Nominals make possible to express proprieties *local* to a specific node. The two properties above are expressed as $@_n[\mathsf{fifo_e}]\,\langle \mathsf{lossy} \rangle\,\mathsf{true}$, and $@_i[-]\,\mathsf{false}$, respectively. Other typical examples, include:

- Absence of a loop formed by a sync followed by a lossy channel at i:

$$i \rightarrow \neg\langle \mathsf{sync} \rangle\,\langle \mathsf{lossy} \rangle\, i.$$

- All output nodes are accessible through a sync channel but never through a $\mathsf{fifo_e}$ channel:

$$[-]\,\mathsf{false}\ \rightarrow\ (\langle\!\langle \mathsf{sync} \rangle\!\rangle\,\mathsf{true} \wedge [\![\mathsf{fifo_e}]\!]\mathsf{false})$$

- A channel of type t is accessible from a node referred to by i

$$@_i\langle -^*.t\rangle\,\mathsf{true}$$

- All input ports lead to an output port via, at least, one $\mathsf{fifo_e}$ channel

$$[\![-]\!]\mathsf{false} \rightarrow \langle -^*.\mathsf{fifo_e}.-^* \rangle\,[-]\,\mathsf{false}$$

The first example in the slides is part of a case study in architectural design in the e-Health domain. It is concerned with the *structural* counterpart of two, essential behavioural requirements to keep the system consistent with the main workflow: 1. a patient always meets a doctor in a medical appointment after triage; 2. the patient is always routed to a billing service at the end of the procedure. From a structural point of view the question becomes to know if such a data flow is possible, *i.e.*if there exist in the graph the necessary connections to make the intended flow *possible*. The requirements are then rephrased as: 1. there is a path from triage input port (t_o) to a MAs edge; 2. all paths from input ports, lead to the billing service (h_o) output port, which can be expressed in $Hp\mathcal{E}$ as follows:

- $\phi_1 \triangleq\ @_{t_o}\langle -^* \rangle\,\mathsf{true} \wedge [-^*]\,[-MAs]\,\mathsf{false}$
- $\phi_2 \triangleq [\![-]\!]\mathsf{false} \rightarrow [-^*]\,h_o$

As illustrated in the slides, a main use of this logic is to analyse structural properties of Reo circuits under reconfiguration processes, expressing, for example, the displacement of an invariant along a reconfiguration or the preservation of some structural patterns. A typical objective is to check whether both reconfigurations are structurally equivalent for a given set of hybrid properties. Further details in [**?**, **?**].

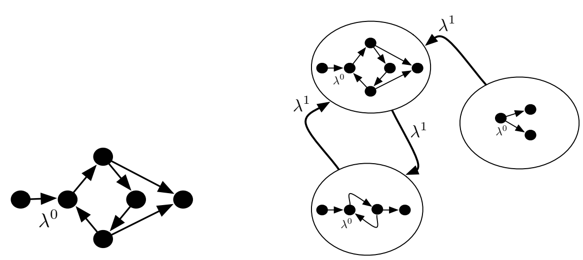
Applications to architectural design

Specifying reconfigurable architectures

- Reconfigurable architectures are represented as structured transition systems whose
- states are endowed with local specifications and
- the global transition structure models system's evolution through possible configurations.
- The hybrid language is developed on top whichever logic is taken for the local configurations (*e.g.*, equational, first-order, fuzzy, etc.) — by hybridisation.

[Alexandre Madeira PhD thesis (MAP-i, 2013)]

Applications to architectural design



- $\mathcal{H}$: pure hybrid formulas
- $\mathcal{H}^2$: hierarchical structures, e.g.

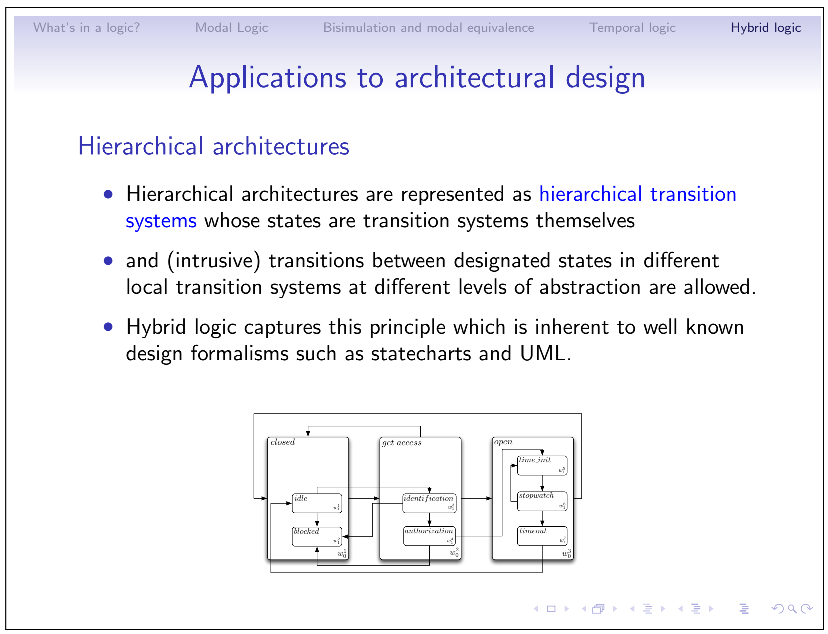
$$@_{j^1} k^0 \wedge^1 [\lambda^1](\rho_1, \dots, \rho_n)$$

32. Hybrid logic as a lingua franca for reconfigurability. An architecture is qualified as *reconfigurable* if the emerging system behaves differently in different modes of operation (*configurations*) and commutes between them along its lifetime. Alexandre Madeira's PhD thesis [?] introduced an approach to the specification of reconfigurable architectures as *structured* transition systems whose states are endowed with *local* specifications and the *global* transition structure models system's evolution through possible configurations.

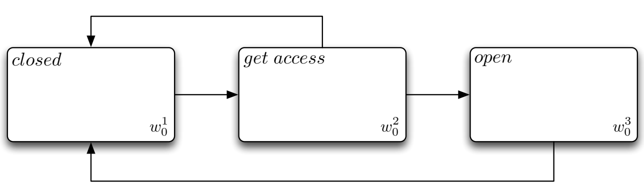
At present, such systems the norm rather than the exception. A typical, everyday example is provided by cloud based applications that elastically react to client demand levels, for example by allocating new server units to meet higher rates of service requests. Modern cars offer a second example: inside hundreds of electronic control units must operate in different modes, depending on the current situation — such as driving on a highway or in town, where different speed regulations apply. Switching between these modes is a typical example of a dynamic reconfiguration.

Specifications of this sort of systems are supposed to make assertions both about the transition dynamics and, locally, about each particular configuration. This leads to the adoption of hybrid logic, which adds to the modal description of transition structures the ability to refer to specific states, as the specification *lingua franca* for reconfigurable systems.

However, because specific problems may require specific logics to describe their configurations (*e.g.*, equational, first-order, fuzzy, etc.), instead of choosing a particular version of hybrid logic, a hybrid language is developed on top of whatever logic is chosen for specifying the system configurations. This process is called *hybridisation*, and constitutes a main contribution of [**?**], where it is framed in the very general setting of the theory of institutions of J. Goguen and R. Burstall [**?**, **?**]. The interested reader is referred to [**?**, **?**].



33. Hierarchical architectures. Hierarchical transition systems, inherent to well known design formalisms such as David Harel's statecharts [**?**] and the UML hierarchical state-machines, can be described in a *multi-layer hybrid logic*: one for capturing each hierarchical level. The slide shows a description of a strongbox controller resulting from the decomposition of the following more abstract description:

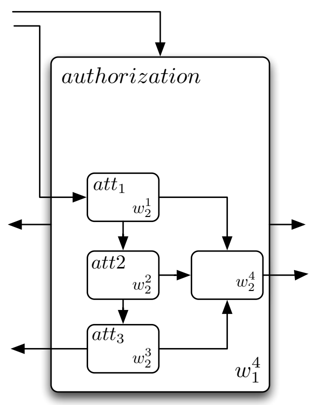


At this level one may express the dynamics depicted in the diagram above, *e.g.*,

- that the state *get access* is accessible from the state *closed*, with $@_{closed}\Diamond get\ access$, or
- that the state *open* is not directly accessible from *closed*, with $\Diamond open \to \neg closed$.

In the refined version shown in the slide each 'high-level' state gives rise to a new, local transition system, and each 'high-level'-transition is decomposed into a number of 'intrusive' transitions from sub-states of the 'down level'-transition system corresponding to the refinement of the original source state, to sub-states of the corresponding refinements of original target states. For instance, the (upper) *close* state can be refined into a (inner) transition system with two (sub) states, one, *idle*, representing the system waiting for the order to proceed for the *get access* state and, another one, *blocked*, capturing a system which is unable to proceed with the opening process (e.g. when authorised access for a given user was definitively denied). In this scenario, the upper level transition from *closed* to *get access* can be realised by, at least, one intrusive transition between the *closed* sub-state *idle* and the *get access* sub-state *identification* where e user identification to proceed is supposed to be checked.

Still the architect may go even further. For example, he may like to refine the *get access* sub-state *authorisation* into the following more fine-grained transition structure:



This third-level view includes a sub-state corresponding to each one of the possible three attempts of password validation, as well as an auxiliary state to represent the authentication success.

This application of hybrid logic to architectural design is also a 'side product' of A. Madeira PhD thesis in which a 'controlled', well-behaved version of the logic was proposed coming out of the successive hybridisation of hybrid logic [**?**]. A less strict, but rather more expressive version [**?**] captures truly intrusive transitions. For example, one may express inner-outer relations between named states (e.g. $@_{idle_1} closed_0$ or $@_{att_{1_2}} open_0$) as well as a variety of transitions. Those include, for example, the layered transition $@_{get\_access_0} \diamond_0 open_0$, the 0-internal transition $@_{identification_1} \diamond_1$ $authorisation_1$ or the 0-intrusive transitions $@_{idle_1} \diamond_1 authorisation_1$ and $get\_access_0 \to \diamond_1 open_0$. Both logics come equipped with suitable notions of bisimulation, and corresponding invariance results, as well as layered and hierarchical refinement results.