

# LogicS: A Modal Perspective

## Spillover #2

Carlos Areces and Patrick Blackburn

`{carlos.areces,patrick.blackburn}@loria.fr`

INRIA Nancy Grand Est  
Nancy, France

NASSLLI 2010 - Bloomington - USA

## Diamond languages: Semantics

Suppose we are given a relational structure

$$\langle W, \{R_1, \dots, R_n\}, \{P_1, \dots, P_m\} \rangle$$

where we have one relation  $R$  for each diamond  $\langle R \rangle$ , and one property  $P$  for each proposition symbol  $p$ . Then we define:

$$\mathcal{M}, w \models p_i \quad \text{iff} \quad w \in P_i,$$

## Diamond languages: Semantics

Suppose we are given a relational structure

$$\langle W, \{R_1, \dots, R_n\}, \{P_1, \dots, P_m\} \rangle$$

where we have one relation  $R$  for each diamond  $\langle R \rangle$ , and one property  $P$  for each proposition symbol  $p$ . Then we define:

$$\begin{aligned} \mathcal{M}, w \models p_i & \text{ iff } w \in P_i, \\ \mathcal{M}, w \models \neg\varphi & \text{ iff } \mathcal{M}, w \not\models \varphi, \\ \mathcal{M}, w \models \varphi \wedge \psi & \text{ iff } \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi, \end{aligned}$$

## Diamond languages: Semantics

Suppose we are given a relational structure

$$\langle W, \{R_1, \dots, R_n\}, \{P_1, \dots, P_m\} \rangle$$

where we have one relation  $R$  for each diamond  $\langle R \rangle$ , and one property  $P$  for each proposition symbol  $p$ . Then we define:

$$\begin{aligned} \mathcal{M}, w \models p_i & \text{ iff } w \in P_i, \\ \mathcal{M}, w \models \neg\varphi & \text{ iff } \mathcal{M}, w \not\models \varphi, \\ \mathcal{M}, w \models \varphi \wedge \psi & \text{ iff } \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi, \\ \mathcal{M}, w \models \langle R_j \rangle \varphi & \text{ iff for some } v \in W \text{ such that } (w, v) \in R_j \\ & \text{ we have } \mathcal{M}, v \models \varphi, \end{aligned}$$

## Diamond languages: Semantics

Suppose we are given a relational structure

$$\langle W, \{R_1, \dots, R_n\}, \{P_1, \dots, P_m\} \rangle$$

where we have one relation  $R$  for each diamond  $\langle R \rangle$ , and one property  $P$  for each proposition symbol  $p$ . Then we define:

$$\begin{aligned} \mathcal{M}, w \models p_i & \text{ iff } w \in P_i, \\ \mathcal{M}, w \models \neg\varphi & \text{ iff } \mathcal{M}, w \not\models \varphi, \\ \mathcal{M}, w \models \varphi \wedge \psi & \text{ iff } \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi, \\ \mathcal{M}, w \models \langle R_j \rangle \varphi & \text{ iff for some } v \in W \text{ such that } (w, v) \in R_j \\ & \text{ we have } \mathcal{M}, v \models \varphi, \\ \mathcal{M}, w \models [R_j] \varphi & \text{ iff for all } v \in W \text{ such that } (w, v) \in R_j \\ & \text{ we have } \mathcal{M}, v \models \varphi. \end{aligned}$$

## Fundamental Semantic Concepts

- ▶ A formula  $\varphi$  is **satisfiable in a model**  $\mathcal{M}$  if there is some point in  $\mathcal{M}$  at which  $\varphi$  is satisfied, and  $\varphi$  is **satisfiable** if there is some point in some model at which it is satisfied.

# Fundamental Semantic Concepts

- ▶ A formula  $\varphi$  is **satisfiable in a model**  $\mathcal{M}$  if there is some point in  $\mathcal{M}$  at which  $\varphi$  is satisfied, and  $\varphi$  is **satisfiable** if there is some point in some model at which it is satisfied.
- ▶ A formula  $\varphi$  is **valid** if it is globally satisfied in all models, and if this is the case we write  $\models \varphi$ .

## Fundamental Semantic Concepts

- ▶ A formula  $\varphi$  is **satisfiable in a model**  $\mathcal{M}$  if there is some point in  $\mathcal{M}$  at which  $\varphi$  is satisfied, and  $\varphi$  is **satisfiable** if there is some point in some model at which it is satisfied.
- ▶ A formula  $\varphi$  is **valid** if it is globally satisfied in all models, and if this is the case we write  $\models \varphi$ .  
(A formula  $\varphi$  is **globally satisfied** (or **globally true**) in a model  $\mathcal{M}$  if it is satisfied at all points in  $\mathcal{M}$ , and if this is the case we write  $\mathcal{M} \models \varphi$ .)



## Two example validities

The following two formulas are validities:

- ▶  $\langle R \rangle (p \wedge q) \rightarrow \langle R \rangle p \wedge \langle R \rangle q$
- ▶  $\langle R \rangle \top \wedge \neg \langle R \rangle \neg p \rightarrow \langle R \rangle p$

Can you see why? Note: you've seen the first one already ...

## Inference and computation in the diamond language

- ▶ It should be clear that defining an inference systems (such as tableaux systems) for the diamond language is going to be trickier (and more interesting!) than for PL.
- ▶ It should also be clear that there are computational issues to be settled — there is no obvious way to prove that the diamond language is computable by “counting models” as we did with PL.
- ▶ That is, we have traded in some tractability for expressivity.
- ▶ We'll look at other inferential/computational issues in the next lecture. For now, we'll simply look briefly at model checking, which is still easy, before turning to expressivity.

# Model checking I

Use a bottom-up **labeling algorithm**. To model check a formula  $\varphi$ :

- ▶ Label every point in the model with all the subformulas of  $\varphi$  that are true at that point.
- ▶ We start with the proposition symbols: the valuation tells us where these are true, so we label all the appropriate points.
- ▶ We then label with more complex formulas. The booleans are handled in the obvious way: for example, we label  $w$  with  $\psi \wedge \theta$  if  $w$  is labeled with both  $\psi$  and  $\theta$ .
- ▶ As for the modalities, we label  $w$  with  $\diamond\varphi$  if one of its  $R$ -successors is labeled with  $\varphi$ , and we label it with  $\square\varphi$  if all of its  $R$ -successors are labeled with  $\varphi$ .

## Model Checking 2

- ▶ The beauty of this algorithm is that we never need to duplicate work: once a point is labeled as making  $\varphi$  true, that's it.
- ▶ This makes the algorithm run in time polynomial in the size of the input formula and model: the algorithm takes time of the order of

$$con(\varphi) \times nodes(\mathcal{M}) \times nodes(\mathcal{M}),$$

where  $con(\varphi)$  is the number of connectives in  $\varphi$ , and  $nodes(\mathcal{M})$  is the number of nodes in  $\mathcal{M}$ .

- ▶ To see this, note that  $con(\varphi)$  tells us how many rounds of labeling we need to perform, one of the  $nodes(\mathcal{M})$  factors is simply the upper bound on the nodes that need to be labeled, while the other is the upper bound on the number of successor nodes that need to be checked.

## Model checking is important

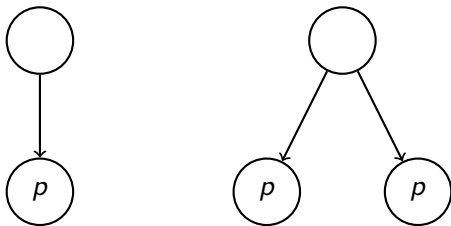
- ▶ Although this looks simple, this kind of model checking is important.
- ▶ It is possible to strengthen the diamond language in various ways to make it excellent for talking about graph structures representing hardware chips and much else besides.
- ▶ The above algorithm can (and have been) extended to such languages and applied to industrial hardware and software design problems.
- ▶ Model checking is the simplest form of inference — but it is important and useful.

## Expressivity and bisimulation

- ▶ What can we say with diamonds? And what can't we say?  
That is, how expressive is our diamond language?
- ▶ We will ask ourselves the following question: how good are diamond languages at **distinguishing between models**?
- ▶ Let's look at an example. . .

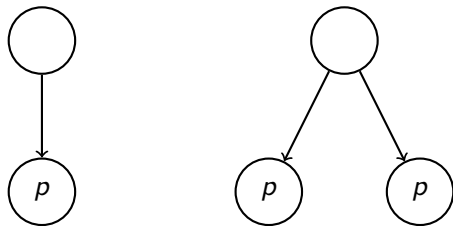
## Are these model diamond distinguishable?

Are these two models diamond distinguishable? To make the question more precise: is there an diamond formula that is true at the root node of one model, and not at the other?



## Are these model diamond distinguishable?

Are these two models diamond distinguishable? To make the question more precise: is there an diamond formula that is true at the root node of one model, and not at the other?

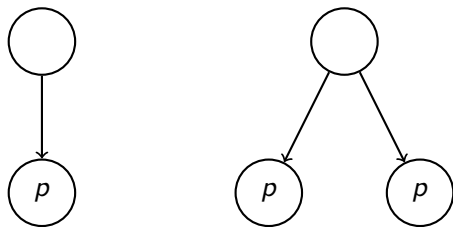


No. There is no diamond formula that distinguishes them. The diamond language, it seems, cannot count!



## Are these model diamond distinguishable?

Are these two models diamond distinguishable? To make the question more precise: is there an diamond formula that is true at the root node of one model, and not at the other?



No. There is no diamond formula that distinguishes them. The diamond language, it seems, cannot count!

Now the key question: why exactly can't the diamond language distinguish the two models?

## Bisimulations (informal)

Two models are bisimilar if their points can be related in so that:

- ▶ Related points make the same propositional symbols true.
- ▶ If you make a transition in one model, you can make “matching” transition in the other. Here “matching” means that the points you reach by making the transitions are related.

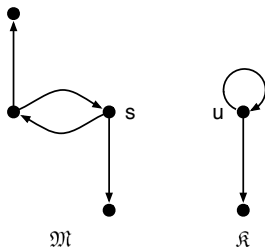
## Bisimulations (formal definition)

Here is the formal definition of bisimulation (for models with one relation  $R$ ). A bisimulation between models  $\mathcal{M}$  and  $\mathcal{M}'$  is a non-empty binary relation  $E$  between their domains (that is,  $E \subseteq D \times D'$ ) such that whenever  $(w, w') \in E$  we have that:

- ▶ **Atomic harmony:**  $w$  and  $w'$  satisfy the same proposition symbols,
- ▶ **Zig:** if  $(w, v) \in R$ , then there exists a point  $v'$  (in  $\mathcal{M}'$ ) such that  $(w', v') \in R'$  and  $(v, v') \in E$ , and
- ▶ **Zag:** if  $(w', v') \in R'$ , then there exists a point  $v$  (in  $\mathcal{M}$ ) such that  $(w, v) \in R$  and  $(v, v') \in E$ .

## Bisimilar or not?

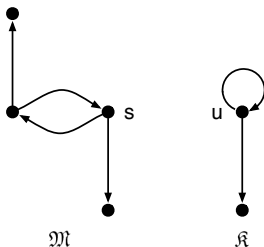
Are points  $s$  and  $u$  in these two models bisimilar or not (assume all propositional symbols are false in all points)?



If they are bisimilar, what is the bisimulation? If they are not bisimilar, what is a formula that distinguishes them?

## Bisimilar or not?

Are points  $s$  and  $u$  in these two models bisimilar or not (assume all propositional symbols are false in all points)?

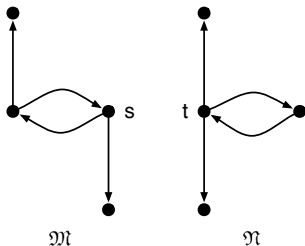


If they are bisimilar, what is the bisimulation? If they are not bisimilar, what is a formula that distinguishes them?

Yes, they are bisimilar; to see this, bend the upward-pointing arrow on the left of the left-hand model downwards.

## Bisimilar or not?

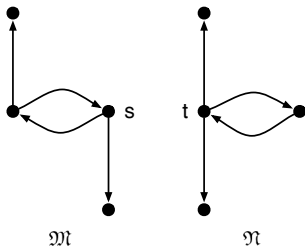
Are points  $s$  and  $t$  in these two models bisimilar or not (assume all propositional symbols are false in all points)?



If they are bisimilar, what is the bisimulation? If they are not bisimilar, what is a formula that distinguishes them?

## Bisimilar or not?

Are points  $s$  and  $t$  in these two models bisimilar or not (assume all propositional symbols are false in all points)?



If they are bisimilar, what is the bisimulation? If they are not bisimilar, what is a formula that distinguishes them?

$\Box(\Box \perp \vee \Diamond \Box \perp)$  is a formula that distinguishes these models: it is true in  $\mathcal{M}$  at  $s$ , but false in  $\mathcal{N}$  at  $t$ .

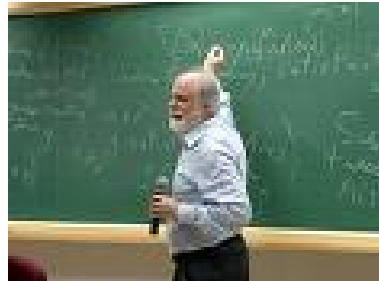
## What we Covered Today

- ▶ We motivated the diamond language, defined its syntax and semantics, and gave example of the concepts of satisfaction at a point in a model, and of validity.
- ▶ We showed that model checking with diamonds is a simple task.
- ▶ We discussed expressivity, introducing and illustrating the concept of bisimulation.
- ▶ However we also briefly remarked that it is not immediately obvious if the diamond language is computable (there is no obvious way of counting models as there was for PL), and it is not yet clear how to handle satisfiability/validity checking.



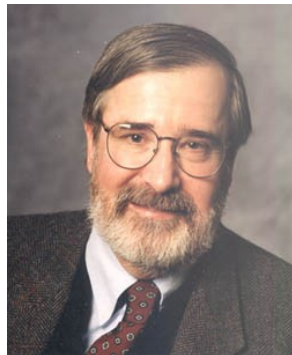
## Relevant Bibliography I

Saul Kripke is the person largely responsible for the relational semantics for the diamond language. In fact, when working with diamond languages, relational structures are often called Kripke models. Kripke, a child prodigy, was 15 when he developed his first ideas on the topic. Kripke has many other substantial contributions to logic and philosophy.



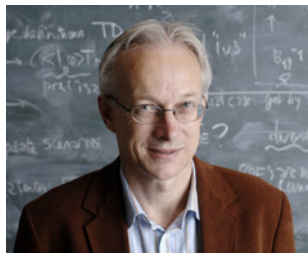
## Relevant Bibliography II

Nowadays, diamond languages (and sophisticated extensions and variants) are used far more in computer science than philosophy. Moreover, in that community they are prized for their model checking properties. Model checking is one of the big success stories of computer science: theory and applications combine beautifully. Ed Clarke, one of the pioneers, won the Turing Prize in 2007 for his contributions to the field.



## Relevant Bibliography III

He wears a crown of diamonds — Johan van Benthem, the king of bisimulation. The researcher who, more than any other, made people see how general and useful diamond languages and their relatives actually are, and who is largely responsible for creating the LLI community.



## Relevant Bibliography IV

- ▶ A New Introduction to Modal Logic, by George Hughes and Max Cresswell, Routledge, 1996.
- ▶ Modal Logic: an Introduction, by Brian Chellas, Cambridge University Press. 1996.
- ▶ First steps in Modal Logic, by Sally Popkorn, Cambridge University Press, 1994.
- ▶ Modal logic: a Semantic Perspective, by Patrick Blackburn and Johan van Benthem, pages 1–84 in *Handbook of Modal Logic*, edited by Blackburn, van Benthem, and Wolter. Elsevier, 2007. Download at <http://www.loria.fr/~blackbur/papers/blackvanben.pdf>

---

# The Next Lecture

## Lecture #5 Trees, and how to cut them