# Finite model theory

## 7. Logics IV

# Further Logics

In this section we look into yet further logics. These are motivated both by theory and applications.

# Further Logics

## Definition 7.1

Let $\tau$ be a relational vocabulary. A **conjunctive query** (or $\mathrm{CQ}$) is a first-order $\tau$-formula $\exists x_1 \dots \exists x_n \chi$ where $\chi$ is a conjunction of relational atoms $R(t_1, \dots t_n)$ with $R \in \tau$ and $t_1, \dots, t_n$ are $\tau$-terms. Conjunctive queries do not contain equality atoms $t_1 = t_2$. If a conjunctive query does not have free variables, it is called a **Boolean conjunctive query**.

# Further Logics

Conjunctive queries are used in, e.g., database theory. For example, let $\tau$ be the relational vocabulary over the relational vocabulary

$$\{\mathrm{Book}, A_1, \ldots, A_n, B_1, \ldots, B_m, , Y_1, \ldots Y_\ell\}$$

where

- $\mathrm{Book}$ is a ternary relation symbol.
- $A_1, \ldots, A_n$ are constant symbols (intuitively denoting author names).
- $B_1, \ldots, B_m$ are constant symbols (book names).
- $Y_1, \ldots, Y_\ell$ are constant symbols (denoting the years from, say, $1700$ to $2020$.)

# Further Logics

Let $\mathfrak{D}$ be a finite $\tau$-model whose domain is

$$\{A_1^{\mathfrak{D}}, \ldots, A_n^{\mathfrak{D}}, B_1^{\mathfrak{D}}, \ldots, B_m^{\mathfrak{D}}, Y_1^{\mathfrak{D}}, \ldots, Y_\ell^{\mathfrak{D}}\}$$

such that the elements are separate, i.e., the domain size is $n + m + \ell$, and $\mathrm{Book}$ is some ternary relaition containing some seet of triples $(X, Y, Z)$ such that

- $X$ is one of the elements $A_i^{\mathfrak{D}}$,
- $Y$ is one of the elements $B_j^{\mathfrak{D}}$,
- $Z$ is one of the elements $Y_k^{\mathfrak{D}}$.

# Further Logics

The finite model $\mathfrak{D}$ can be seen as a database, possibly encoding the contents of some small library. Now, let the constant symbol $A_i$ be the name of some author, say, F. Scott Fitzgerald. Now the conjunctive query

$$\varphi(x) = \exists y \, \mathrm{Book}(A_i, x, y)$$

returns the unary relation $\varphi^{\mathfrak{D}}$ containing precisely the book titles authored by F. Scott Fitzgerald and stored in the library $\mathfrak{D}$.

Let $\mathfrak{D}'$ be the expansion of the $\tau$-model $\mathfrak{D}$ to the vocabulary $\tau \cup \{\mathrm{Album}\}$. We assume that some of the constant symbols $B_1, \ldots, B_m$ can also be album names, and we let $\mathrm{Album}^{\mathfrak{D}'}$ contain triples $(X, Y, Z)$ where $X$ is one of the elements $A_i^{\mathfrak{D}} = A_i^{\mathfrak{D}'}$ while $Y$ is some element $B_j^{\mathfrak{D}} = B_j^{\mathfrak{D}'}$ and $Z$ some element $Y_k^{\mathfrak{D}} = Y_k^{\mathfrak{D}'}$. Write a conjunctive query that returns the set of people who have published both a book and an album during the same year.

# Further Logics

The query

$$\chi(x) = \exists u \exists y \exists z \left( \mathrm{Book}(x, u, y) \wedge \mathrm{Album}(x, z, y) \right).$$

returns the set of people who have published a book and an album during some single year.

# Further Logics

Let $\mathfrak{D}$ a finite model over the vocabulary $\{\mathrm{Parent\_of}\}$ where $\mathrm{Parent\_of}$ is a binary relation. Intuitively, $\mathfrak{D}$ is a simple genealogical database of where the elements are people and $\mathrm{Parent\_of}$ contains the pairs $(u, v)$ such that $u$ is a parent of $v$. Now the conjunctive query

$$\varphi(x, y) = \exists z(\mathrm{Parent\_of}(x, z) \wedge \mathrm{Parent\_of}(z, y))$$

returns the pairs $(u, v)$ such that $u$ is a grand parent of $v$.

# Further Logics

Conjunctive queries are the most central querying framework studied in database theory. However, conjunctive queries are a rather weak fragment of first-order logic and thus come with obvious limitations. Indeed, consider the genealogical database from the previous slide. Consider the query $\psi_{ancestor}(x, y)$ that returns those pairs $(u, v)$ such that $u$ is an ancestor of $v$. We cannot write this as a conjunctive query, as we cannot define the transitive closure of a first-order formula even in first-order logic. Thus a $\mathrm{CQ}$, in particular, is definitely not expressive enough.

# Further Logics

For writing the ancestor query, we need a logic with capacities that go beyond that of first-order logic. For example the LFP-formula

$$[lfp_{(S,(x,y))}\text{Parent\_of}(x,y) \vee \exists z(\text{Parent\_of}(x,z) \wedge S(z,y))](x,y)$$

does the job. This is just the good old formula defining the transitive closure. However, for some purposes, LFP tends to be too expressive (as very expressive logics tend to be difficult and slow to work with in relation to applications, requiring generally too slow truth checking algorithms). A paradigmatic framework that srikes a balance between conjunctive queries and LFP is DATALOG which combines the idea of conjunctive queries and iterative fixed point constructions. We shall look into DATALOG.

# Further Logics

DATALOG is somewhat technically messy and thus DATALOG will not appear in the exam but may appear in the exercises.

# Further Logics

Let $\tau$ be a relational vocabulary. A DATALOG-program is a list

$$
\begin{aligned}
X_1(x_{1,1}, \ldots, x_{1,n_1}) \quad &:- \quad \varphi_1 \\
&\vdots \\
X_m(x_{m,1}, \ldots, x_{m,n_m}) \quad &:- \quad \varphi_m
\end{aligned}
$$

where each $X_k$ is an $n_k$-ary relation symbol not in $\tau$ and each $\varphi_k$ is a CQ over the vocabulary $\tau \cup \{X_1, \ldots, X_m\}$ and in the free variables $(x_{k,1}, \ldots, x_{k,n_k})$. The relation symbols $X_1, \ldots, X_m$ are called IDBs (intensional databases) and thus sometimes the relation symbols in $\tau$ are called EDBs (extensional databases). Note carefully, some of the symbols $X_1, \ldots, X_m$ can be the same (see the next page for an example where two of the first lines of a program begin with $X_1$).

# Further Logics

Each line

$$X_i(x_{i,1}, \ldots, x_{i,n_i}) \quad :- \quad \varphi_i$$

of a DATALOG-program is called a **rule**. The rule consists of the **head** $X_i(x_{i,1}, \ldots, x_{i,n_i})$, the symbol $:-$ and the **body** $\varphi_i$ which is a conjunctive query with exactly the free variables that occur also in the head. A rule with the **head variable** equal to the IDB $X_i$ is called an $X_i$-**rule**. There can be several $X_i$-rules in the same program. For example, the following DATALOG-program over the vocabulary $\{P, S, R\}$

$$\begin{aligned}
X_1(x,y) &:- Rxy \\
X_1(x,y) &:- \exists z(P(x) \wedge Sxzy \wedge X_1(x,y) \wedge X_2(x,x)) \\
X_2(x,y) &:- \exists z(X_2(y,x) \wedge Rzz \wedge X_3(x)) \\
X_3(x) &:- \exists z \exists u(Rxu \wedge X_1(x,z) \wedge X_2(x,u))
\end{aligned}$$

contains two $X_1$-rules. Note also that even though this is a DATALOG-program over the vocabulary $\{R, S, P\}$, the CQs are allowed to contain relation symbols that are IDBs, i.e., the IDBs $X_1, X_2, X_3$ are allowed to occur in the rule bodies in addition to the EDBs $R, S, P$.

# Further Logics

Note also that, as required, the heads of each rule of the program

$$
\begin{array}{lll}
X_1(x,y) & :- & Rxy \\
X_1(x,y) & :- & \exists z(P(x) \wedge Sxzy \wedge X_1(x,y) \wedge X_2(x,x)) \\
X_2(x,y) & :- & \exists z(X_2(y,x) \wedge Rzz \wedge X_3(x)) \\
X_3(x) & :- & \exists z \exists u(Rxu \wedge X_1(x,z) \wedge X_2(x,u))
\end{array}
$$

contain precisely the variables that occur free in the corresponding body. For example, the body of the second rule is

$$\exists z(P(x) \wedge Sxzy \wedge X_1(x,y) \wedge X_2(x,x)),$$

and the free variables of this $\mathrm{CQ}$ are $x, y$, the variables that occur also in the head of that rule.

# Further Logics

Before we define the semantics of DATALOG, let us consider the following simple example program.

$$\begin{array}{lll} X_1(x) & :- & P(x) \\ X_2(x) & :- & Q(x) \\ X_1(x) & :- & \exists y(Rxy \wedge X_1(y) \wedge X_2(y)) \end{array}$$

This program is evaluated as follows on a $\tau$-model $\mathfrak{A}$, where $\{P, Q, R\} \subseteq \tau$. Let $X_1^0$ be the empty unary relation $\emptyset$, and similarly, let $X_2^0$ also be the empty unary relation. These are both unary because $X_1(x)$ and $X_2(x)$ both have the one single free variable $x$. Now suppose we have defined $X_1^n$ and $X_2^n$ to be some unary relations over $A$. Let $\mathfrak{A}^*$ be the model obtained from $\mathfrak{A}$ by expanding the vocabulary by $X_1$ and $X_2$ and interpreting these relation symbols as the relations $X_1^n$ and $X_2^n$. We let $X_1^{n+1}$ be equal to the relation

$$\{a \in A \mid \mathfrak{A}^*, \{(x, a)\} \models P(x) \vee \exists y(Rxy \wedge X_1(y) \wedge X_2(y)) \}$$

where $\{(x, a)\}$ is of course a variable assignment.

# Further Logics

Intuitively, we therefore obtain $X_1^{n+1}$ by taking the union of the relations that the $X_1$-rule bodies define when each $\mathrm{IDB}$-symbol $X_i$ in the rule bodies is interpreted as $X_i^n$. Shortly

1. interpret each $X_i$ as $X_i^n$,
2. evaluate the rule bodies to obtain the relations defined by the bodies, and
3. let $X_i^{n+1}$ be the union of the relations whose head variable $X_i$ is.

Doing this, we observe that $X_2^{n+1}$ is interpreted as the relation defined by $Q(x)$, so it is somewhat less interesting here than $X_1^{n+1}$.

# Further Logics

To define the semantics of DATALOG formally, let $\mathfrak{A}$ be a $\tau$-model. For each of the IDBs $X_i \in \{X_1, \ldots, X_m\}$, we define a corresponding operator $F_{X_i}$. To define the operators, let $ar(X_i)$ denote the arity of $X_i$. The operator $F_{X_i}$ is the function

$$F_{X_i} : \mathcal{P}(A^{ar(X_1)}) \times \cdots \times \mathcal{P}(A^{ar(X_m)}) \quad \rightarrow \quad \mathcal{P}(A^{ar(X_i)})$$

such that for any input relations $U_1, \ldots, U_m$ of the appropriate arities (and over $A$), the function $F_{X_i}$ outputs the relation...

# Further Logics

...the relation

$$\{\,(a_1,\ldots,a_{ar(X_i)})\mid \mathfrak{A}[X_1\mapsto U_1,\ldots,X_m\mapsto U_m],\{(x_{i,1},a_1),\ldots,(x_{i,ar(X_i)},a_{ar(X_i)})\}\models\bigvee_{i\in I}\varphi_i\,\}$$

where $\bigvee_{i\in I}\varphi_i$ is the disjunction of those rule bodies that have $X_i$ as head. Note indeed that $\{(x_{i,1},a_1),\ldots,(x_{i,ar(X_i)},a_{ar(X_i)})\}$ is simply a variable assignment and $\mathfrak{A}[X_1\mapsto U_1,\ldots,X_m\mapsto U_m]$ the expansion of $\mathfrak{A}$ with the symbols $X_1,\ldots,X_m$ interpreted as the relations $U_1,\ldots,U_m$. (If some symbol $X$ appears more than once in the list $X_1,\ldots,X_m$ but two corresponding relations $U_p$ and $U_q$ are different in the list $U_1,\ldots,U_m$, then the output of the operator $F_{X_i}$ can be defined arbitrarily on that input tuple $(U_1,\ldots,U_m)$. Such an output tuple is irrelevant for us here.)

# Further Logics

We define $X_i^0 = \emptyset$, i.e., the empty relation of the same arity as $X_i$. We define $X_i^{n+1}$ to be $F_{X_i}(X_1^n, \ldots, X_m^n)$.

# Further Logics

Note that, as defined so far, a DATALOG-program simply corresponds to a system that makes the relations corresponding to the rule head variables $X_1, \ldots, X_m$ evolve in a stepwise fashion. This is ok, but we would also like to make sense of statements of the form $\mathfrak{A} \models \Pi$, where $\Pi$ is a DATALOG-program. To do this, the first step is to allow **nullary** IDB**s**.

Let $\mathfrak{A}$ be a model. A nullary relation $Y$ over $A$ is a set $Y \subseteq A^0 = \{\emptyset\}$. Thus either $Y = \emptyset$ or $Y = \{\emptyset\}$. If $Y = \emptyset$, we associate it with the truth value **false**, and if $Y = \{\emptyset\}$, we associate it with the truth value **true**. We typically write $Y = \top$ and $Y = \bot$ instead of $Y = \{\emptyset\}$ and $Y = \emptyset$. A model $\mathfrak{A}$ whose vocabulary contains a nullary relation symbol $Y$, satisfies the formula $Y$ (i.e., $\mathfrak{A} \models Y$) if and only if $Y$ is interpreted as $\top$.

# Further Logics

The syntax of DATALOG with nullary IDB is the same as the syntax specified above, with the addition that nullary IDBs can be head variables and the bodies of rules with nullary head variables must be Boolean conjunctive queries, i.e., conjunctive queries without free variables. For example

$$\begin{array}{lll} X_1(x) & :- & P(x) \\ X_2 & :- & \exists x Q(x) \\ X_3 & :- & \exists x \exists y (Rxy \wedge X_1(y) \wedge X_2) \end{array}$$

is a DATALOG-program where the IDBs $X_2$ and $X_3$ are nullary and thus the corresponding CQs have no free variables.

# Further Logics

Also the semantics remains the same. The nullary IDBs are first equal to $\bot$, and then we start iterating (i.e., running) the program. A nullary IDB $X_i$ becomes true if one of the bodies of the $X_i$-rules becomes true.

# Further Logics

Formally, let $\mathfrak{A}$ be a $\tau$-model. Let $\Pi$ be program with $m$ rules and with the IDBs $X_1, \ldots, X_m$ some of which can be nullary (and some of these IDBs can be the same). For each of the IDBs $X_i \in \{X_1, \ldots, X_m\}$, we define a corresponding operator $F_{X_i}$. Let $ar(X_i)$ denote the arity of $X_i$. The operator $F_{X_i}$ is the function

$$F_{X_i} : \mathcal{P}(A^{ar(X_1)}) \times \cdots \times \mathcal{P}(A^{ar(X_m)}) \quad \rightarrow \quad \mathcal{P}(A^{ar(X_i)})$$

such that for any input relations $U_1, \ldots, U_m$ of the appropriate arities (and over $A$), the function $F_{X_i}$ outputs the relation

$$\{ (a_1, \ldots, a_{ar(X_i)}) \mid \mathfrak{A}[X_1 \mapsto U_1, \ldots, X_m \mapsto U_m], \{(x_{i,1}, a_1), \ldots, (x_{i,ar(X_i)}, a_{ar(X_i)})\} \models \bigvee_{i \in I} \varphi_i \}$$

where $\bigvee_{i \in I} \varphi_i$ is the disjunction of the rule bodies whose head IDB is $X_i$. Note that if $X_j$ is nullary, then $\mathfrak{A}[X_1 \mapsto U_1, \ldots, X_m \mapsto U_m] \models X_j$ if and only if $U_j = \top$. Note also that a nullary IDB $X_j$ evaluates to $\top$ if at least one of the $X_j$-rules has a body that evaluates to true. Continues...

# Further Logics

Note also that if $X_i$ is nullary, then $F_{X_i}$ outputs $\top$ if

$$\mathfrak{A}[X_1 \mapsto U_1, \ldots, X_m \mapsto U_m] \models \bigvee_{i \in I} \varphi_i$$

and otherwise $F_{X_i}$ outputs $\bot$.

# Further Logics

If $X_i$ is nullary, we define $X_i^0 = \bot$, and when $X_i$ is not nullary, we define $X_i^0 = \emptyset$, i.e., the empty relation of the same arity as $X_i$. We define $X_i^{n+1}$ to be $F_{X_i}(X_1^n, \ldots, X_m^n)$ in both cases (nullary and not nullary).

# Further Logics

In addition to the usual $\mathrm{IDB}$s, we let $X_{goal}$ be a special, distinguished nullary $\mathrm{IDB}$. If $\Pi$ is a $\textsc{Datalog}$-program, we define that $\mathfrak{A} \models \Pi$ if there exists some $n$ such that $X_{goal}^{n} = \top$.

# Further Logics

Let $P$ and $Q$ be unary relations and $R$ a binary relation. Define a DATALOG-program $\Pi$ such that $\mathfrak{A} \models \Pi$ if and only if there is some node $u$ in $P$ and some node $v$ in $Q$ such that $u = v$ or $(u, v) \in R$ or

$$(u, a_1) \in R \wedge (a_1, a_2) \in R \wedge \cdots \wedge (a_i, v) \in R$$

for some $i$.

# Further Logics

$$
\begin{array}{lll}
X_1(x) & :- & Q(x) \\
X_1(x) & :- & \exists y(Rxy \wedge X_1(x)) \\
X_{goal} & :- & \exists x(Px \wedge X_1(x))
\end{array}
$$

# Further Logics

Write a DATALOG-program $\Pi$ such that $\mathfrak{A} \models \Pi$ if the binary relation $S$ has a tuple $(u, v) \in S$ such that the inverse tuple $(v, u)$ belongs to the transitive closure of the binary relation $R$.

# Further Logics

$$
\begin{array}{lll}
X(x,y) & :- & R(x,y) \\
X(x,y) & :- & \exists z(R(x,z) \wedge X(z,y)) \\
X_{goal} & :- & \exists x \exists y(Syx \wedge X(x,y))
\end{array}
$$

# Further Logics

Write a DATALOG-program $\Pi$ such that $\mathbb{G} \models \Pi$ if the graph $\mathbb{G}$ is not two-colorable.

# Further Logics

$$
\begin{array}{lll}
Odd(x,y) & :- & E(x,y) \\
Odd(x,y) & :- & \exists z \exists u (E(x,z) \wedge E(z,u) \wedge Odd(u,y)) \\
X_{goal} & :- & \exists x Odd(x,x)
\end{array}
$$

# Further Logics

To simplify notation when writing DATALOG programs, we typically do not write the existential quantifier $\exists x$ explicitly, as it is clear that every variable that is not in the head must be existentially quantified. Also, we can use commas instead of conjunctions. Thus

$$\begin{array}{lll} Odd(x,y) & :- & E(x,y) \\ Odd(x,y) & :- & \exists z \exists u (E(x,z) \wedge E(z,u) \wedge Odd(u,y)) \\ X_{goal} & :- & \exists x Odd(x,x) \end{array}$$

becomes

$$\begin{array}{lll} Odd(x,y) & :- & E(x,y) \\ Odd(x,y) & :- & E(x,z), E(z,u), Odd(u,y) \\ X_{goal} & :- & Odd(x,x) \end{array}$$

# Further Logics

Despite their limitations, CQs are nevertheless a highly flexible framework capable of expressing the most typical queries. Also, being simple (and not too expressive), they are efficient to implement. One of the most common extensions of CQs (and still a fragment of FO unlike DATALOG) is the collection of queries known as UCQs, with UCQ standing for an **union of conjunctive queries**. A UCQ is simply a disjunction of conjunctive queries. A query is a **Boolean** UCQ if it has no free variables.

# Further Logics

Consider the vocabulary $\{P, Q\}$ where both $P$ and $Q$ are unary. Now, $Px \vee Qx$ is a simple $\mathrm{UCQ}$ that returns all elements in $P$ and $Q$. It is easy to see that there exists no $\mathrm{CQ}$ that can do that job.

# Further Logics: Modal Logic

We next discuss modal logic, which is probably the most popular framework of logics from the point of view of current applications. The field has its roots in philosophy, and the related research is still very active. However, the main applications nowadays lie in computer science—in particulal artificial intelligence, verification and knowledge representation. There are also various systems of modal logic that deal with the foundations of mathematics, e.g., *provability logic* that deals with the notions of provability and non-provability.

# Further Logics: Modal Logic

Modal logic is actually not a single logic, but a framework for different kinds of systems with similar features. However, there is a core modal logic, often also (confusingly) called **modal logic**. We shall call this core system $\mathrm{ML}$. Let $\mathrm{PROP}$ be a countably infinite set of unary relation symbols, called **proposition symbols** in modal logic. The set of formulae of $\mathrm{ML}$ is the smallest set $F$ such that the following conditions are satisfied.

1. If $P \in \mathrm{PROP}$, then $P \in F$.
2. If $\varphi \in F$, then $\neg\varphi \in F$.
3. If $\varphi, \psi \in F$, then $\varphi \wedge \psi \in F$.
4. If $\varphi \in F$, then $\Diamond\varphi \in F$.

# Further Logics: Modal Logic

Recall that a **Kripke model** or **Kripke structure** is a model

$$\mathfrak{M} = (W, R, (P_i)_{i \in I})$$

where $R \subseteq W \times W$ is a binary relation and each $P_i \subseteq W$ is a unary relation. The modal logic $\mathrm{ML}$ is interpreted with respect to **pointed models** $(\mathfrak{M}, w)$, where $\mathfrak{M}$ is a Kripke model and $w \in W$ is an element in the domain $W$ of the Kripke model.

# Further Logics: Modal Logic

The semantics of the modal logic $\mathrm{ML}$ is given as follows.

$$
\begin{array}{lcl}
(\mathfrak{M}, w) \models P_i & \Leftrightarrow & w \in P_i^{\mathfrak{M}} \\
(\mathfrak{M}, w) \models \varphi \wedge \psi & \Leftrightarrow & (\mathfrak{M}, w) \models \varphi \text{ and } (\mathfrak{M}, w) \models \psi \\
(\mathfrak{M}, w) \models \neg\varphi & \Leftrightarrow & (\mathfrak{M}, w) \not\models \varphi \\
(\mathfrak{M}, w) \models \Diamond\varphi & \Leftrightarrow & \text{there exists some } v \text{ such that} \\
 & & (w, v) \in R^{\mathfrak{M}} \text{ and } (\mathfrak{M}, v) \models \varphi
\end{array}
$$

# Further Logics: Modal Logic

We use the familiar abbreviations

- $\varphi \vee \psi = \neg(\neg\varphi \wedge \neg\psi)$,
- $\varphi \to \psi = \neg\varphi \quad \vee \quad \psi$,
- $\varphi \leftrightarrow \psi = (\varphi \to \psi) \wedge (\psi \to \varphi)$.

We also use the abbreviation $\Box\varphi = \neg\Diamond\neg\varphi$.

# Further Logics: Modal Logic

Modal logic was originally motivated by the philosophy of the notions of **possibility** and **necessity**. In a Kripke model $\mathfrak{M} = (W, R, (P_i)_{i=I})$, every domain element $w \in W$ is considered to be a **world**, or a **state of affairs**. Indeed, we say that $w$ satisfies the proposition $P_i$ if and only if $w \in P^{\mathfrak{M}}$ (i.e., we have $(\mathfrak{M}, w) \models P_i$). Thus $w$ is associated with precisely those propositions that $w$ satisfies.

# Further Logics: Modal Logic

Let $\mathbb{P}_w$ denote the set of propositions satisfied by $w$. The world $w$, or the *state of affairs* $w$, is thus specified by—or somehow identified with—the set $\mathbb{P}_w$ that $w$ satisfies. For example, suppose

$$(P_i)_{i \in I} \;=\; (P_1,\, P_2) \;=\; (\mathrm{It\_is\_raining},\, \mathrm{it\_is\_shining}),$$

so we have two propositions, one stating that it is raining and the other one that it is shining. Suppose $W = \{w_1, w_2\}$ such that $w_1$ satisfies $P_1$ and $\neg P_2$ while $w_2$ satisfies $\neg P_1$ and $P_2$. Thus we have two worlds, one where it is raining and one where it is shining.

# Further Logics: Modal Logic

Now, suppose $(\mathfrak{M}, w) \models \Diamond P_i$. This means that there exists some $v$ such that $(w, v) \in R^{\mathfrak{M}}$ and we have $(\mathfrak{M}, v) \models P_i$. The inclusion $(w, v) \in R^{\mathfrak{M}}$ is interpreted to mean that $v$ is a possible alternative (to $w$) state of affairs. Thereby $(\mathfrak{M}, w) \models \Diamond P_i$ states that there exists an alternative (to $w$) state of affairs $v$ that satisfies $P_i$. In other words, $(\mathfrak{M}, w) \models \Diamond P_i$ states that in $w$, **it is possible that** $P_i$. The operator $\Diamond$ is thus called the **possibility operator**. We can read $\Diamond P_i$ as '*it is possible that* $P_i$'. Similarly for any formula $\varphi$ of $\mathrm{ML}$, we can read $\Diamond\varphi$ as '*it is possible that* $\varphi$'.

# Further Logics: Modal Logic

Recall that $\Box$ is an abbreviation for $\neg\Diamond\neg$. Thereby $\Box P_i$ states that $\neg P_i$ is impossible, i.e., $P_i$ is *necessarily* true. Thus $\Box$ is called the *necessity* operator, and we can give $\Box P_i$ the reading that '*it is necessary that* $P_i$'.

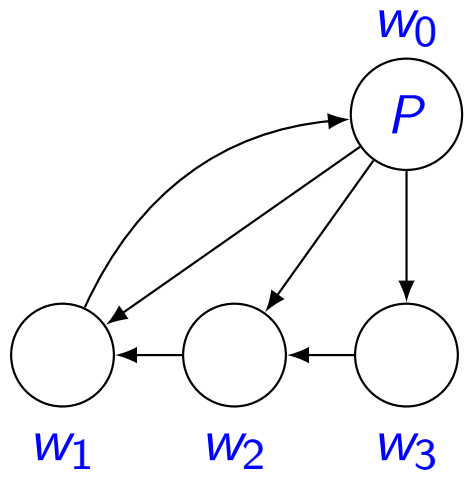
The possibility operator $\Diamond$ is also called the **diamond** and the necessity operator $\Box$ the **box**.

In addition to using proposition symbols $P_i$ with indices $i$, we often use symbols $P$ and $Q$ to denote propositions.

# Further Logics: Modal Logic

Consider the pictured Kripke model with domain $W = \{w_0, w_1, w_2, w_3\}$ and such that the proposition $P$ is satisfied by $w_0$ only, i.e., $P^{\mathfrak{M}} = \{w_0\}$.

$\mathfrak{M}$:



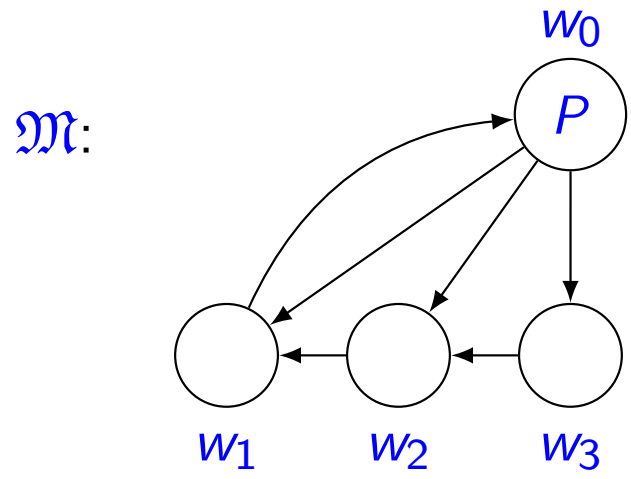

Now $(\mathfrak{M}, w_0) \models \Diamond\neg P$ because there exists a world accessible (in one step from $w_0$ via $R$) that satisfies $\neg P$. In other words, the condition

$$\exists v : (w, v) \in R \quad \text{and} \quad (\mathfrak{M}, v) \models \neg P$$

holds. This is because, e.g., $(\mathfrak{M}, w_1) \models \neg P$.

# Further Logics: Modal Logic

Let us further investigate the model $\mathfrak{M}$, pictured again below.




We have $(\mathfrak{M}, w_0) \models \Box\neg P$ because for all worlds $v$ accessible (in one step from $w_0$ via $R$), we have $(\mathfrak{M}, v) \models \neg P$.

Exercise: For each world $v \in \{w_0, w_1, w_2.w_3\}$, find a formula $\varphi_v$ satisfied only by $v$ but not the other worlds.

# Further Logics: Modal Logic

A possible solution:

- $(\mathfrak{M}, w_0) \models P$ while none of the other worlds satisfies $P$.
- $(\mathfrak{M}, w_1) \models \Diamond P$ while no other world satisfies this formula.
- $(\mathfrak{M}, w_2) \models \neg P \ \wedge \ \Diamond\Diamond P$ while no other world satisfies this formula.
- $(\mathfrak{M}, w_3) \models \neg P \ \wedge \ \Box\Box\Box P$ while no other world satisfies this formula.

# Further Logics: Modal Logic

The modal logic $\mathrm{ML}$ is a variable-free system. However, it can be conceived simply as a simple fragment of first-order logic. The **standard translation** is a function that maps formulae of $\mathrm{ML}$ to formulae of first-order logic as follows.

- $\mathrm{St}_x(P) = P(x)$
- $\mathrm{St}_y(P) = P(y)$
- $\mathrm{St}_x(\neg\varphi) = \neg\mathrm{St}_x(\varphi)$
- $\mathrm{St}_y(\neg\varphi) = \neg\mathrm{St}_y(\varphi)$
- $\mathrm{St}_x(\varphi \wedge \psi) = \mathrm{St}_x(\varphi) \wedge \mathrm{St}_x(\psi)$
- $\mathrm{St}_y(\varphi \wedge \psi) = \mathrm{St}_y(\varphi) \wedge \mathrm{St}_y(\psi)$
- $\mathrm{St}_x(\Diamond\varphi) = \exists y\Big(R(x,y) \wedge \mathrm{St}_y(\varphi)\Big)$
- $\mathrm{St}_y(\Diamond\varphi) = \exists x\Big(R(y,x) \wedge \mathrm{St}_x(\varphi)\Big)$

Note that $\mathrm{St}_x(\varphi)$ has the free variable $x$ and $\mathrm{St}_y(\varphi)$ the free variable $y$.

# Further Logics: Modal Logic

This translation leads to the following theorem.

## Theorem 7.2

*Let $\varphi$ be a formula of $\mathrm{ML}$. Consider a Kripke model $\mathfrak{M}$ and an arbitrary variable assignment $f$. Suppose $\mathfrak{M}$ contains a point $w$. We have*

$$(\mathfrak{M}, w) \models \varphi \quad \Leftrightarrow \quad \mathfrak{M}, f[x \mapsto w] \models \mathrm{St}_x(\varphi).$$

Before justifying the theorem, notice that intuitively, the theorem states that in some sense $\mathrm{ML}$ can be seen as a fragment of first-order logic $\mathrm{FO}$. At least the modal logic $\mathrm{ML}$ translates into formulae $\psi(x) = \mathrm{St}_x(\varphi)$ of modal logic that can—in some sense—be seen equivalent to the original formulae $\varphi$.

# Further Logics: Modal Logic

**Proof.** For proving Theorem 7.2, simply recall the semantics of $\mathrm{ML}$:

$$\begin{array}{lcl}
(\mathfrak{M}, w) \models P_i & \Leftrightarrow & w \in P_i^{\mathfrak{M}} \\
(\mathfrak{M}, w) \models \varphi \wedge \psi & \Leftrightarrow & (\mathfrak{M}, w) \models \varphi \text{ and } (\mathfrak{M}, w) \models \psi \\
(\mathfrak{M}, w) \models \neg\varphi & \Leftrightarrow & (\mathfrak{M}, w) \not\models \varphi \\
(\mathfrak{M}, w) \models \Diamond\varphi & \Leftrightarrow & \text{there exists some } v \text{ such that} \\
 & & (w, v) \in R^{\mathfrak{M}} \text{ and } (\mathfrak{M}, v) \models \varphi
\end{array}$$

It is easy to observe that the translation $\mathrm{St}$ simply translates the semantics of $\mathrm{ML}$ into first-order logic. The nontrivial case is the clause

$$\mathrm{St}_x(\Diamond\varphi) = \exists y\Big(R(x, y) \wedge \mathrm{St}_y(\varphi)\Big)$$

for the diamond $\Diamond$. The other cases are indeed clear.$\square$

# Further Logics: Modal Logic

The most common way of proving undefinability results for $\mathrm{ML}$ is by using **bisimulations**. A bisimulation is a binary relation $Z \subseteq A \times B$ between the domains of two Kripke models $\mathfrak{A}$ and $\mathfrak{B}$ (over the same vocabulary $\tau$) that satisfies the following conditions.

1. If $(a, b) \in Z$, then the elements $a$ and $b$ satisfy the same propositions, i.e., we have

$$(\mathfrak{A}, a) \models P_i \;\Leftrightarrow\; (\mathfrak{B}, b) \models P_i$$

   for all propositions $P_i$ in $\tau$.
2. If $(a, b) \in Z$ and $(a, a') \in R^{\mathfrak{A}}$, then there exists a point $b' \in B$ such that $(b, b') \in R^{\mathfrak{B}}$ and $(a', b') \in Z$.
3. If $(a, b) \in Z$ and $(b, b') \in R^{\mathfrak{B}}$, then there exists a point $a' \in A$ such that $(a, a') \in R^{\mathfrak{A}}$ and $(a', b') \in Z$.

# Further Logics: Modal Logic

Theorem 7.3

*Let $Z \subseteq A \times B$ be a bisimulation between $\mathfrak{A}$ and $\mathfrak{B}$. If $(a, b) \in Z$, then we have*

$$(\mathfrak{A}, a) \models \varphi \;\Leftrightarrow\; (\mathfrak{B}, b) \models \varphi$$

*for all formulae $\varphi$ of the modal logic $\mathrm{ML}$.*

**Proof.** We prove the claim by induction on the structure of formulae. Suppose $\varphi$ is a proposition symbol $P$. Then, directly by the definition of bisimulation, $(\mathfrak{A}, a)$ and $(\mathfrak{B}, b)$ satisfy the same propositions. Continues...

# Further Logics: Modal Logic

Now suppose $\varphi$ is $\psi \wedge \chi$. Now

$$
\begin{aligned}
& (\mathfrak{A}, a) \models \psi \wedge \chi \\
\Leftrightarrow \quad & (\mathfrak{A}, a) \models \psi \text{ and } (\mathfrak{A}, a) \models \chi \\
\overset{\text{ind.hypot.}}{\Leftrightarrow} \quad & (\mathfrak{B}, b) \models \psi \text{ and } (\mathfrak{B}, b) \models \chi \\
\Leftrightarrow \quad & (\mathfrak{B}, b) \models \psi \wedge \chi.
\end{aligned}
$$

# Further Logics: Modal Logic

Now suppose $\varphi$ is $\neg\psi$. We have

$$
\begin{aligned}
& (\mathfrak{A}, a) \models \neg\psi \\
\Leftrightarrow \quad & (\mathfrak{A}, a) \not\models \psi \\
\overset{\textit{ind.hypot.}}{\Leftrightarrow} \quad & (\mathfrak{B}, b) \not\models \psi \\
\Leftrightarrow \quad & (\mathfrak{B}, b) \models \neg\psi.
\end{aligned}
$$

# Further Logics: Modal Logic

Suppose finally that $\varphi$ is $\Diamond\psi$. Assume that $(\mathfrak{A}, a) \models \Diamond\psi$. Thus there exists $a'$ such that $(a, a') \in R^{\mathfrak{A}}$ and $(\mathfrak{A}, a') \models \psi$. By the definition of the bisimulation relation $Z$, since $(a, a') \in R^{\mathfrak{A}}$ and $(a, b) \in Z$, there exists $b'$ such that $(b, b') \in R^{\mathfrak{B}}$ and $(a', b') \in Z$. Since $(a', b') \in Z$ and $(\mathfrak{A}, a') \models \psi$, we have $(\mathfrak{B}, b') \models \psi$ by the induction hypothesis. Therefore, since $(b, b') \in R^{\mathfrak{B}}$, we have $(\mathfrak{B}, b) \models \Diamond\psi$. We have therefore proved that $(\mathfrak{A}, a) \models \Diamond\psi \Rightarrow (\mathfrak{B}, b) \models \Diamond\psi$. The converse implication is proved similarly.$\square$

# Further Logics: Modal Logic

## Example 7.4

Let $A = \{a_1, a_2, a_3\}$ and $B = \{b_1, b_2\}$ and define $R^{\mathfrak{A}} = \{(a_1, a_2), (a_1, a_3)\}$ and $R^{\mathfrak{B}} = \{(b_1, b_2)\}$. Let $P^{\mathfrak{A}} = \{a_2, a_3\}$ and $P^{\mathfrak{B}} = \{b_2\}$. Consider the Kripke models $\mathfrak{A} = (A, R^{\mathfrak{A}}, P^{\mathfrak{A}})$ and $\mathfrak{B} = (B, R^{\mathfrak{B}}, P^{\mathfrak{B}})$ (sketch the models). Consider the relation $Z = \{(a_1, b_1), (a_2, b_2), (a_3, b_2)\}$ (sketch the relation). This relation is clearly a bisimulation. It connects both $a_2$ and $a_3$ to the point $b_2$, so $(\mathfrak{A}, a_2)$ and $(\mathfrak{A}, a_3)$ satisfy the same modal formulae as $(\mathfrak{B}, b_2)$. The bisimulation $Z$ also connects $a_1$ to $b_1$, so $(\mathfrak{A}, a_1)$ and $(\mathfrak{B}, b_1)$ satisfy the same modal formulae.

# Further Logics: Modal Logic

## Definition 7.5

The **modal depth** $md(\varphi)$ of a formula $\varphi$ of $\mathrm{ML}$ is defined as follows.

- $md(P_i) = 0$
- $md(\varphi \wedge \psi) = max(md(\varphi), md(\psi))$
- $md(\neg\varphi) = md(\varphi)$
- $md(\Diamond\varphi) = md(\varphi) + 1$

Therefore, the modal depth of a formula is simply the maximum nesting depth of diamonds in the formula.

# Further Logics: Modal Logic

Earlier on we proved that there exist only finitely many non-equivalent formulae of first-order logic over a finite vocabulary $\tau$ and with quantifier-rank at most $m$. The modal logic $\mathrm{ML}$ can be seen as a fragment of first-order logic (recall the translation $\mathrm{St}$), and therefore it is obvious that an analogous result holds also for $\mathrm{ML}$. We shall now formulate and prove that result (without resorting to the earlier result on $\mathrm{FO}$).

# Further Logics: Modal Logic

## Definition 7.6

Let $\varphi$ and $\psi$ be formulae of the modal logic $\mathrm{ML}$. We say that $\varphi$ and $\psi$ are **equivalent** if for all pointed Kripke models $(\mathfrak{A}, w)$ and $(\mathfrak{B}, u)$ that interpret the proposition symbols in $\varphi$ and $\psi$, we have

$$(\mathfrak{A}, w) \models \varphi \quad \Leftrightarrow \quad (\mathfrak{B}, u) \models \psi.$$

We denote this by writing $\varphi \equiv \psi$.

# Further Logics: Modal Logic

## Theorem 7.7

*Let $\pi \subseteq \mathrm{PROP}$ be a finite, nonempty set of proposition symbols. Let $n \in \mathbb{N}$. There exist only finitely many non-equivalent formulae up to modal depth $n$ and with the proposition symbols in $\pi$. More rigorously, there exists a finite set $F$ of formulae of $\mathrm{ML}$ using the proposition symbols in $\pi$ and with modal depth up to $n$ such that the following holds:*

*If $\varphi$ is an arbitrary formula with the proposition symbols in $\pi$ and with modal depth up to $n$, then there exists a formula $\psi \in F$ such that $\varphi \equiv \psi$.*

Before we work through the proof, it is indeed worth noting that the claim of the theorem follows also from Theorem 3.8.

# Further Logics: Modal Logic

**Proof (Sketch).** Let $\pi = \{P_1, \ldots, P_k\}$. We prove the claim by induction on modal depth. We begin with some auxiliary definitions.

Let $F$ be a finite set $\{\varphi_1, \ldots, \varphi_k\}$ of formulae of $\mathrm{ML}$. Now, a **full description** is a formula $\chi_1 \wedge \cdots \wedge \chi_k$ where each $\chi_i$ is either $\varphi_i$ or $\neg\varphi_i$. Clearly, as in our related argument for first-order logic earlier on, we see that every Boolean combination of formulae in $F$ is equivalent to a disjunction of full descriptions.[1]

[1] Note that the disjunction $\bigvee \emptyset$ is defined equivalent to a contradiction (as disjunction requires that at least one of the given formulae holds). We can always encode $\bigvee \emptyset$ by some formula, for example $P_1 \wedge \neg P_1$.

# Further Logics: Modal Logic

Now define that a 0-**type** is a full description over $\pi = \{P_1, \ldots, P_k\}$. Recursively, suppose we have defined the set of $m$-types, and denote this set by $F_m$. An $(m+1)$-type is any formula

$$\psi_m \wedge \bigwedge_{\psi \in G} \Diamond\psi \wedge \bigwedge_{\psi \in F_m \setminus G} \neg\Diamond\psi$$

where $G \subseteq F_m$ is a set of $m$-types and $\psi_m \in F_m$ some $m$-type. Thus an $(m+1)$-type specifies the following:

1. The $m$-type $\psi_m$ of the current point (of a pointed model).
2. The complete set $G \subseteq F_m$ of $m$-types that are accessible via $R$ from the current point (i.e., positive information about $m$-types of directly accessible points).
3. The complete set $F_m \setminus G$ of $m$-types that are not accessible from the current point via $R$ (i.e., negative information about $m$-types of directly accessible points).

# Further Logics: Modal Logic

Thus an $(m+1)$-type can be seen as a complete description of a domain point in terms of formulae of modal depth up to $(m+1)$. It is not difficult to see that therefore every formula of modal depth up to $\ell$ and with propositions in $\pi = \{P_1, \ldots, P_k\}$ is equivalent to some disjunction of $\ell$-types. By the construction of $\ell$-types, it is easy to see that there are finitely many $\ell$-types.$\square$

# Further Logics: Modal Logic

We now *informally sketch* a proof that $\mathrm{ML}$ has the **finite model property**, i.e., the property that for all formulae $\varphi$ of $\mathrm{ML}$, if there exists a (possibly infinite) pointed model $(\mathfrak{M}, w)$ such that $(\mathfrak{M}, w) \models \varphi$, then there exists a *finite* model $\mathfrak{N}$ and a point $u$ of $\mathfrak{N}$ such that $(\mathfrak{N}, u) \models \varphi$. The full details of the argument are skipped. The argument requires an understanding of equivalence relations and equivalence classes, so a reader not familiar with these notions can safely skip over the argument.

So, let $\varphi$ be a formula of $\mathrm{ML}$ and suppose $(\mathfrak{M}, w) \models \varphi$. Let $\pi$ be the set of proposition symbols in $\varphi$ and $n$ the modal depth $md(\varphi)$ of $\varphi$. Define an equivalence relation $\sim \subseteq M \times M$ such that $(u, v) \in \sim$ if and only if $(\mathfrak{M}, u)$ and $(\mathfrak{M}, v)$ satisfy the same $n$-type. (It can easily be shown that every element $u$ of any model $\mathfrak{A}$ satisfies precisely one $n$-type.) For each element $u$ of $\mathfrak{M}$, let $[u]$ denote the set

$$\{v \in M \mid v \text{ satisfies the } n\text{-type of } u\}.$$

That is, $[u]$ is the equivalence class of $u$ with respect to $\sim$.

# Further Logics: Modal Logic

Define the following Kripke model $\mathfrak{N}$.

1. The domain $N$ is the set $\{\,[u] \mid u \in M\,\}$ of equivalence classes $[u]$ of elements $u$ of the model $\mathfrak{M}$.
2. $([u],[v]) \in R^{\mathfrak{N}}$ if and only if the $(n-1)$-type $\psi_v^{n-1}$ of $(\mathfrak{M}, v)$ is such that the $n$-type of $(\mathfrak{M}, u)$ has the conjunct $\Diamond\psi_v^{n-1}$.
3. We have $[u] \in P^{\mathfrak{N}}$ if and only if the 0-type of $u$ has the conjunct $P$.

It can be shown with a relatively straightforward argument (skipped here) that

$$(\mathfrak{M}, w) \models \varphi \;\Leftrightarrow\; (\mathfrak{N}, [w]) \models \varphi$$

holds for all formulae $\varphi$ with proposition symbols in $\pi$ and up to modal depth $n$. Therefore $\mathrm{ML}$ indeed has the finite model property: if $\varphi$ is true in a pointed model $(\mathfrak{M}, w)$, construct the model $(\mathfrak{N}, [w])$ which is clearly finite (as there are finitely many $n$-types) and observe that $(\mathfrak{N}, [w]) \models \varphi$.$\square$