Modal Logic Mechanisation in HOL4

Yiming Xu

15/6/2018

1 Getting Start

In our formalisation, we only consider the most standard modal language, called the basic modal language. The basic modal language is defined using a set Φ of propositional formulas, and only one modal operator $\Diamond$. A formula in the basic modal language is either a propositional symbol p where $p \in \Phi$, or a disjunction $\psi \vee \phi$ of prmodal formulas ψ and ϕ, or the negation $\neg\phi$ of a modal formula ϕ, or else of the form $\Diamond\phi$ obtained by putting a diamond before a modal formula. In the HOL, we create a datatype called 'form' of the formulas of this modal language.

$$
\begin{array}{l}
\alpha\ \texttt{chap1\$form}\ = \\
\quad \mathsf{VAR}\ \alpha \\
\mid \mathsf{DISJ}\ (\alpha\ \texttt{chap1\$form})\ (\alpha\ \texttt{chap1\$form}) \\
\mid \bot \\
\mid (\neg)\ (\alpha\ \texttt{chap1\$form}) \\
\mid \Diamond\ (\alpha\ \texttt{chap1\$form})
\end{array}
$$

When we say an 'α form', we mean a formula in the basic modal language defined on a set Φ with elements of type α, such a set is called an α set, and of type $\alpha \to \mathit{bool}$. Note that we have the notion of the conjunction '$\wedge$', the implication $\to$, and the truth '$\top$', but they are not primitive, and defined as:

$$\vdash \mathsf{AND}\ f_1\ f_2\ =\ \neg\mathsf{DISJ}\ (\neg f_1)\ (\neg f_2) \quad \vdash (f_1\ \to\ f_2)\ =\ \mathsf{DISJ}\ (\neg f_1)\ f_2 \quad \vdash \mathsf{TRUE}\ =\ \neg\bot$$

As an analogue of the duality of the universal quantifier is the dual of the existential quantifier, in the sense that $\exists$ is defined to be $\neg\forall\neg$, we have a modal operator that is dual to the diamond, defined by $\Box\phi := \neg\Diamond\neg\phi$.

$$\vdash \Box\ \phi\ =\ \neg\Diamond\ (\neg\phi)$$

If a modal formula does not use any diamond, then it is also a propositional formula:

$$\begin{aligned}\vdash\ & (\forall p.\ \mathsf{propform}\ (\mathsf{VAR}\ p) \iff \mathsf{T}) \wedge \\ & (\forall \phi_1\ \phi_2.\ \mathsf{propform}\ (\mathsf{DISJ}\ \phi_1\ \phi_2) \iff \mathsf{propform}\ \phi_1 \wedge \mathsf{propform}\ \phi_2) \wedge \\ & (\forall f.\ \mathsf{propform}\ (\neg f) \iff \mathsf{propform}\ f) \wedge (\mathsf{propform}\ \bot \iff \mathsf{T}) \wedge \\ & \forall f.\ \mathsf{propform}\ (\Diamond f) \iff \mathsf{F}\end{aligned}$$

Knowing how does the modal formulas we are interested in look like, now we want to consider what does they mean. If the modal formula defined using propositional symbols in an α-set Φ is a propositional formula, then to talk about its truth value, what we need is just an assignment of truth value of the propositional letters in Φ, that is, a function $\Phi \to bool$. In the HOL, to define a function from an α-set, where α is any type, we are not allowed to only assign values to the elements of the set, instead, we must define the function on for all the terms that has type α. Hence we can define the function of evaluating propositional formulas as takes a function $\sigma : \alpha \to bool$ and an α propositional formula, and let it spit out the truth value of the propositional formula under the assignment σ.

$$\begin{aligned}\vdash\ & (\forall \sigma\ p.\ \mathsf{peval}\ \sigma\ (\mathsf{VAR}\ p) \iff \sigma\ p) \wedge \\ & (\forall \sigma\ f_1\ f_2. \\ & \quad \mathsf{peval}\ \sigma\ (\mathsf{DISJ}\ f_1\ f_2) \iff \mathsf{peval}\ \sigma\ f_1 \vee \mathsf{peval}\ \sigma\ f_2) \wedge \\ & (\forall \sigma.\ \mathsf{peval}\ \sigma \bot \iff \mathsf{F}) \wedge \\ & (\forall \sigma\ f.\ \mathsf{peval}\ \sigma\ (\neg f) \iff \neg \mathsf{peval}\ \sigma\ f) \wedge \\ & \forall \sigma\ f.\ \mathsf{peval}\ \sigma\ (\Diamond f) \iff \mathsf{F}\end{aligned}$$

However, to interpret a modal formula that involves diamonds, an assignment of truth value is not enough. We need a model of our language. A model of the basic modal language consists of two pieces of informations. The first thing we need is a frame. A frame consists of two things, a set, and a relation defined on elements in this set. If the underlying set of a frame is an β-set, then the frame is called an β-frame.

$$\alpha\ \texttt{frame} = \texttt{<|}\ \mathsf{world}\ :\ \alpha\ \to\ \texttt{bool};\ \mathsf{rel}\ :\ \alpha\ \to\ \alpha\ \to\ \texttt{bool}\ \texttt{|>}$$

The second thing we need is a valuation. If we are taking about interpret an α-form on a β-frame, a valuation is a function of type $\beta \to \alpha \to bool$. The information that a valuation gives is that, for each point in the frame, an assignment of truth value of each propositional symbol. We get a model by putting a frame and a valuation together.

$$\begin{aligned}&(\alpha,\ \beta)\ \texttt{chap1\$model} = \texttt{<|} \\ &\quad \mathsf{frame}\ :\ \beta\ \texttt{frame}; \\ &\quad \mathsf{valt}\ :\ \alpha\ \to\ \beta\ \to\ \texttt{bool} \\ &\texttt{|>}\end{aligned}$$

When we say an (α, β)-model, we mean a model for α-forms with a β-set as its underlying set.

Now we can interpret modal formulas in the basic modal language on a model by defining satisfication.

$$
\begin{aligned}
\vdash\ & (\forall \mathfrak{M}\ w\ p.\\
& \quad \mathsf{satis}\ \mathfrak{M}\ w\ (\mathsf{VAR}\ p)\ \Longleftrightarrow\\
& \qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ w\ \in\ \mathfrak{M}.\mathsf{valt}\ p)\ \wedge\\
& (\forall \mathfrak{M}\ w.\ \mathsf{satis}\ \mathfrak{M}\ w\ \bot\ \Longleftrightarrow\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{F})\ \wedge\\
& (\forall \mathfrak{M}\ w\ \phi.\\
& \quad \mathsf{satis}\ \mathfrak{M}\ w\ (\neg\phi)\ \Longleftrightarrow\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \neg\mathsf{satis}\ \mathfrak{M}\ w\ \phi)\ \wedge\\
& (\forall \mathfrak{M}\ w\ \phi_1\ \phi_2.\\
& \quad \mathsf{satis}\ \mathfrak{M}\ w\ (\mathsf{DISJ}\ \phi_1\ \phi_2)\ \Longleftrightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ \phi_1\ \vee\ \mathsf{satis}\ \mathfrak{M}\ w\ \phi_2)\ \wedge\\
& \forall \mathfrak{M}\ w\ \phi.\\
& \quad \mathsf{satis}\ \mathfrak{M}\ w\ (\Diamond\ \phi)\ \Longleftrightarrow\\
& \qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\\
& \qquad \exists v.\\
& \qquad\quad \mathfrak{M}.\mathsf{frame.rel}\ w\ v\ \wedge\ v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\\
& \qquad\quad \mathsf{satis}\ \mathfrak{M}\ v\ \phi
\end{aligned}
$$

(insert an example of model and satisfication here)

Just as in first order logic and propositional logic, we can define the notion of logical equivalence of modal formulas which is an equivalence relation between formulas which use propositional symbols of the same type:

$$\vdash \mathsf{equiv0}\ tyi\ f_1\ f_2\ \Longleftrightarrow\ \forall \mathfrak{M}\ w.\ \mathsf{satis}\ \mathfrak{M}\ w\ f_1\ \Longleftrightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ f_2\ \vdash \mathsf{equiv0}\ \mu\ \mathsf{equiv\_on}\ s$$

A notable thing is that equiv0 need to take a type itself, denoted as μ, and if μ denotes the type α itself and f_1, f_2 are β formulas, equiv0 μ f_1 f_2 means for any (α, β)-model $\mathfrak{M}$ and any world w in it, we have satis $\mathfrak{M}$ w f_1 $\Longleftrightarrow$ satis $\mathfrak{M}$ w f_2. We are not allowed to omit the μ in the definition, since then we will introduce a type, namely the type of models $\mathfrak{M}$, that only appear on the right hand side but not on the left hand side of the definition, which is not allowed. Also we are not allowed to quantify over the μ, and something like 'equiv f1 f2 $\Leftrightarrow$!μ. equiv0 μ f1 f2' is also not allowed. This is actually a inconvenience of the HOL, since when we mention equivalence of formulas in usual mathematical language, we are implicitly taking about the class of all models, but the constrain here bans us from talking about all models at once.

We can immediately prove that for μ denotes any type, if equiv0 μ f g then equiv0 μ ($\Diamond$ f) ($\Diamond$ g), but the converse does not hold under our definition. In the usual mathematical notion of equivalence, if two diamond formulas $\Diamond f$ and $\Diamond g$ are equivalent, then f and g must be equivalent. It is because if f and g are not equivalent, then there eixsts a model $\mathfrak{M}$ and a world w such that w satisfies f but not g, then we can attach a world that is only linked to w, and the world we add is a witness of the fact that $\Diamond f$ and $\Diamond g$ are not equivalent. But under our definition in the HOL, if the μ denotes a finite type, then it is possible that we have equiv0 μ ($\Diamond$ f) ($\Diamond$ g) but $\neg$equiv0 μ f g. For example, consider the type that has only two inhabitants a, b, then in the model below, we have equiv0 μ ($\Diamond$ (VAR p)) ($\Diamond$ (VAR q)) but $\neg$equiv0 μ (VAR p) (VAR q).

(add the picture of example, how to draw it?)

As the situition above is only because of our special definition, such case is uninteresting. For any model, regardless its world set is of a finite type or not, we can always create a copy of the model in an infinite type. So it is harmless to just consider equivalence of formulas for models whose underlying set is of an infinite type. When μ denotes an infinite type, as we can always come up with a fresh world that allows the proof of equivalence between f and g from the equivalence of $\Diamond f$ and $\Diamond g$ as in the usual sense to go through, we do have a double implication:

$$\begin{aligned}&\vdash \mathsf{INFINITE}\ \mathcal{U}(:\beta)\ \Rightarrow\\ &\quad (\mathsf{equiv0}\ \mu\ (\Diamond\ f)\ (\Diamond\ g)\ \iff\ \mathsf{equiv0}\ \mu\ f\ g)\end{aligned}$$

Given a modal formula, we can extract the set of propositional symbols appear in it, as:

$$\begin{aligned}&\vdash (\forall p.\ \mathsf{prop\_letters}\ (\mathsf{VAR}\ p)\ =\ \{\ p\ \})\ \wedge\ \mathsf{prop\_letters}\ \bot\ =\ \emptyset\ \wedge\\ &\quad (\forall f_1\ f_2.\\ &\qquad \mathsf{prop\_letters}\ (\mathsf{DISJ}\ f_1\ f_2)\ =\\ &\qquad\quad \mathsf{prop\_letters}\ f_1\ \cup\ \mathsf{prop\_letters}\ f_2)\ \wedge\\ &\quad (\forall f.\ \mathsf{prop\_letters}\ (\neg f)\ =\ \mathsf{prop\_letters}\ f)\ \wedge\\ &\quad \forall f.\ \mathsf{prop\_letters}\ (\Diamond\ f)\ =\ \mathsf{prop\_letters}\ f\end{aligned}$$

Then we can show when evaluating a formula ϕ on a model, the only relevant information in the valuation is the assignment it makes to the propositional letters actually occurring in ϕ:

$$\begin{aligned}&\vdash \mathfrak{M}_1.\mathsf{frame}\ =\ \mathfrak{M}_2.\mathsf{frame}\ \Rightarrow\\ &\quad (\forall p.\ p\ \in\ \mathsf{prop\_letters}\ phi\ \Rightarrow\ \mathfrak{M}_1.\mathsf{valt}\ p\ =\ \mathfrak{M}_2.\mathsf{valt}\ p)\ \Rightarrow\\ &\quad \forall w.\ w\ \in\ \mathfrak{M}_1.\mathsf{frame.world}\ \Rightarrow\ (\mathsf{satis}\ \mathfrak{M}_1\ w\ phi\ \iff\ \mathsf{satis}\ \mathfrak{M}_2\ w\ phi)\end{aligned}$$

In particular, if we are interpreting a propositional formula, then we do not need the relation on the model, as we can prove from definition:

$$\begin{aligned}&\vdash \mathsf{propform}\ f\ \wedge\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ &\quad (\mathsf{satis}\ \mathfrak{M}\ w\ f\ \iff\ \mathsf{peval}\ (\lambda\, a.\ w\ \in\ \mathfrak{M}.\mathsf{valt}\ a)\ f)\end{aligned}$$

Moreover, we only need the truth values of the propositional symbols that appears in the formula:

$$\begin{aligned}&\vdash \mathsf{propform}\ f\ \wedge\ (\forall a.\ \mathsf{VAR}\ a\ \in\ \mathsf{subforms}\ f\ \Rightarrow\ a\ \in\ s)\ \wedge\\ &\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ &\quad (\mathsf{satis}\ \mathfrak{M}\ w\ f\ \iff\ \mathsf{peval}\ ((\lambda\, a.\ w\ \in\ \mathfrak{M}.\mathsf{valt}\ a)\ \cap\ s)\ f)\end{aligned}$$

One may confused why are we allowed to write $\lambda\ a.w \in M.valt\ a) \cap s$ here, for $\lambda\ a.(w \in M.valt\ a)$ is a function but s is a set. This is because both the α-set s and the function $\lambda\ a.(w \in M.valt)$ has type $\alpha \to bool$, so they can be feeded to the function $\cap$, which has type $(\alpha \to bool) \to (\alpha \to bool) \to (\alpha \to bool)$. Hence for the set s, we can either view it as a collection of elements of type α, or a function that assigns each term of type α a truth value, where the truth value assigned to $a : \alpha$ is T if and only if $a \in s$.

(change subforms to propsyms later?)

2 Invariant results and bisimulations

The key concept we are interested in this chapter is called 'modal equivalent', by 'two worlds $w \in M.frame.world, w' \in M'.frame.world$ are modal equivalent', we mean they satisfies exactly the same modal formulas.

$$\vdash \mathsf{modal\_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w' \iff \forall \phi.\ \mathsf{satis}\ \mathfrak{M}\ w\ \phi \iff \mathsf{satis}\ \mathfrak{M}'\ w'\ \phi$$

In this chapter, we investigate when can we get modal equivalence. In the first section, we talk about how can we get new models from old models without affect modal satisfication, and then we introduce morphisms between models, and find out under which condition we can map a point of a model to another model and preserve satisfication. In the second section, we investigate a kind of relation between models, called bisimulation, that gives arise to modal equivalence, in the sense of two worlds that are related by a bisimulation satisfies the same modal formulas.

2.1 Operations on models

Here we introduce two operations on models, called disjoint union and generated submodels respectively.

The operation on models that leads to invarience of formulas in the most straightforward way is the disjoint union:

$$\begin{array}{l}
\mathsf{DU}\ (f, dom) \stackrel{\mathsf{def}}{=} \\
\quad \mathsf{<|frame}\ := \\
\quad\quad \mathsf{<|world}\ := \\
\quad\quad\quad \{\ w\ \mid \mathsf{FST}\ w\ \in\ dom\ \wedge\ \mathsf{SND}\ w\ \in\ (f\ (\mathsf{FST}\ w)).\mathsf{frame.world}\ \};\\
\quad\quad\quad \mathsf{rel}\ := \\
\quad\quad\quad (\lambda\, w_1\ w_2. \\
\quad\quad\quad\quad \mathsf{FST}\ w_1\ =\ \mathsf{FST}\ w_2\ \wedge\ \mathsf{FST}\ w_1\ \in\ dom\ \wedge \\
\quad\quad\quad\quad (f\ (\mathsf{FST}\ w_1)).\mathsf{frame.rel}\ (\mathsf{SND}\ w_1)\ (\mathsf{SND}\ w_2))\ \mathsf{|>};\\
\quad \mathsf{valt}\ :=\ (\lambda\, v\ w.\ (f\ (\mathsf{FST}\ w)).\mathsf{valt}\ v\ (\mathsf{SND}\ w))\ \mathsf{|>}
\end{array}$$

Disjoint union is defined as a function that takes a function that takes a function $f : \beta \to (\alpha, \gamma)$-model and a β-set dom, which plays the role of the index set that we take the disjoint union over. The function f are defined for all terms of type β, but actually we only need its value on elements in dom. Given a family of models indexed by the set dom, the world set of the disjoint union of this family has elements of form (i, w) where $i \in dom$ and $w \in (f\ i).frame.world$. For a propositional letter p and a world (i, w), we have $DU\ (f, dom).valt\ p\ (i, w)$ iff $(f\ i).valt\ p\ w$, and $DU\ (f, dom).rel\ (i, w)\ (i', w')$ iff $i = i'$ and $(f\ i).frame.rel\ w\ w'$, means that (i, w) and (i', w') come from the same model and are related by the relation in that model.

The disjoint union does nothing except for collecting a set of models together, so it is unsurprising that we can prove by induction that:

$$\begin{array}{l}\vdash \mathsf{FST}\ w\ \in\ dom\ \Rightarrow \\ \quad (\mathsf{satis}\ (f\ (\mathsf{FST}\ w))\ (\mathsf{SND}\ w)\ phi\ \iff \\ \quad\quad \mathsf{satis}\ (\mathsf{DU}\ (f, dom))\ w\ phi)\end{array}$$

Disjoint union is the operation we use when we want to expand out scope from a smaller model to a large model. Accordingly, we have an operation that allows us to restrict our scope to a smaller model, this is called 'generated submodel' construction.

When we say 'M_1 is a submodel of M_2, we mean M_1 is a part of M_2.

$$\begin{array}{l}\mathsf{SUBMODEL}\ \mathfrak{M}_1\ \mathfrak{M}_2\ \stackrel{\mathsf{def}}{=} \\ \quad \mathfrak{M}_1.\mathsf{frame.world}\ \subseteq\ \mathfrak{M}_2.\mathsf{frame.world}\ \wedge \\ \quad \forall\, w_1. \\ \quad\quad w_1\ \in\ \mathfrak{M}_1.\mathsf{frame.world}\ \Rightarrow \\ \quad\quad (\forall\, v.\ \mathfrak{M}_1.\mathsf{valt}\ v\ w_1\ \iff\ \mathfrak{M}_2.\mathsf{valt}\ v\ w_1)\ \wedge \\ \quad\quad \forall\, w_2. \\ \quad\quad\quad w_2\ \in\ \mathfrak{M}_1.\mathsf{frame.world}\ \Rightarrow \\ \quad\quad\quad (\mathfrak{M}_1.\mathsf{frame.rel}\ w_1\ w_2\ \iff\ \mathfrak{M}_2.\mathsf{frame.rel}\ w_1\ w_2)\end{array}$$

It is not necessary that submodel construction preserves modal satisfication, since we can discard relations between worlds and hence destroy the satisfication of formulas with diamonds.

But after adding some constrains to the relations that we need to preserve, we can have a special kind of submodels, called generated submodels, that preserves modal satisfication.

$$\begin{array}{l}\vdash \mathsf{GENSUBMODEL}\ \mathfrak{M}_1\ \mathfrak{M}_2\ \iff \\ \quad \mathsf{SUBMODEL}\ \mathfrak{M}_1\ \mathfrak{M}_2\ \wedge \\ \quad \forall\, w_1. \\ \quad\quad w_1\ \in\ \mathfrak{M}_1.\mathsf{frame.world}\ \Rightarrow \\ \quad\quad \forall\, w_2. \\ \quad\quad\quad w_2\ \in\ \mathfrak{M}_2.\mathsf{frame.world}\ \wedge\ \mathfrak{M}_2.\mathsf{frame.rel}\ w_1\ w_2\ \Rightarrow \\ \quad\quad\quad w_2\ \in\ \mathfrak{M}_1.\mathsf{frame.world}\end{array}$$

Note that for a model M_1 to be a generated submodel of M_2, we only require all the worlds w' which have a link from a world w in M_1 to be also included in the world set of M_1, and we are allowed to completely ignore everthing that is linked to a world in M_1. This is because modal formulas cannot 'look back', in the sense that linking or discard links to a world does not change the set of satisfied formulas at the world.

Also by induction, we can prove the invariance of modal satisfication under taking generated submodels:

$$\begin{array}{l}\vdash \mathsf{GENSUBMODEL}\ \mathfrak{M}_1\ \mathfrak{M}_2\ \wedge\ n\ \in\ \mathfrak{M}_1.\mathsf{frame.world}\ \Rightarrow \\ \quad (\mathsf{satis}\ \mathfrak{M}_1\ n\ phi\ \iff\ \mathsf{satis}\ \mathfrak{M}_2\ n\ phi)\end{array}$$

2.2 Morphisms between models

In this section, we talk about various kind of 'morphisms' between models. Similar as in mathematics, the notion of 'morphism' here is used to describe maps that preserves structures. For instance, a homomorphism is a rather notion of 'structure-preserving':

$$
\begin{array}{l}
\vdash \mathsf{hom}\ f\ \mathfrak{M}_1\ \mathfrak{M}_2 \iff \\
\quad \forall\, w. \\
\qquad w \in \mathfrak{M}_1.\mathsf{frame.world} \Rightarrow \\
\qquad f\ w \in \mathfrak{M}_2.\mathsf{frame.world} \wedge \\
\qquad (\forall\, p.\ w \in \mathfrak{M}_1.\mathsf{valt}\ p \Rightarrow f\ w \in \mathfrak{M}_2.\mathsf{valt}\ p) \wedge \\
\qquad \forall\, u. \\
\qquad\quad u \in \mathfrak{M}_1.\mathsf{frame.world} \Rightarrow \\
\qquad\quad \mathfrak{M}_1.\mathsf{frame.rel}\ w\ u \Rightarrow \\
\qquad\quad \mathfrak{M}_2.\mathsf{frame.rel}\ (f\ w)\ (f\ u)
\end{array}
$$

Although a homomorphism preserves link between worlds in the source model, we allow the existence of extra link in the target model which has no counterpart in the source. Because of this, we cannot guarentee any world in the source is mapped to a world that satisfies exactly the same set of modal formulas under a homomorphism. As an attempt to obtain an equivlence, we strongthen the condition of being a homomorphism, and what we get after strengthening is called a strong homomorphism:

$$
\begin{array}{l}
\mathsf{strong\_hom}\ f\ \mathfrak{M}_1\ \mathfrak{M}_2 \stackrel{\mathsf{def}}{=} \\
\quad \forall\, w. \\
\qquad w \in \mathfrak{M}_1.\mathsf{frame.world} \Rightarrow \\
\qquad f\ w \in \mathfrak{M}_2.\mathsf{frame.world} \wedge \\
\qquad (\forall\, p.\ w \in \mathfrak{M}_1.\mathsf{valt}\ p \iff f\ w \in \mathfrak{M}_2.\mathsf{valt}\ p) \wedge \\
\qquad \forall\, u. \\
\qquad\quad u \in \mathfrak{M}_1.\mathsf{frame.world} \Rightarrow \\
\qquad\quad (\mathfrak{M}_1.\mathsf{frame.rel}\ w\ u \iff \mathfrak{M}_2.\mathsf{frame.rel}\ (f\ w)\ (f\ u))
\end{array}
$$

A strong homomorphism that is a bijection on the world set is called an isomorphism, and an isomorphism will certainly preserves everthing. Just for the sake of modal equivalence, we do not really require an isomorphism, instead, a surjective strong morphism is enough:

$$
\begin{array}{l}
\vdash \mathsf{strong\_hom}\ f\ \mathfrak{M}\ \mathfrak{M}' \wedge f\ w = w' \wedge w \in \mathfrak{M}.\mathsf{frame.world} \wedge \\
\quad \mathsf{SURJ}\ f\ \mathfrak{M}.\mathsf{frame.world}\ \mathfrak{M}'.\mathsf{frame.world} \Rightarrow \\
\quad \mathsf{modal\_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w'
\end{array}
$$

Now we have a desired modal equivalence from strong homomorphism. The problem is that the condition of being a strong homomorphism is too strong. We want a suitably weakened condition on a morphism that gives arise of such equivalence, our answer is bounded morphism:

$$
\begin{array}{l}
\mathsf{bounded\_mor}\ f\ \mathfrak{M}\ \mathfrak{M}' \stackrel{\mathsf{def}}{=} \\
\quad \forall\, w. \\
\qquad w \in \mathfrak{M}.\mathsf{frame.world} \Rightarrow \\
\qquad f\ w \in \mathfrak{M}'.\mathsf{frame.world} \wedge \\
\qquad (\forall\, a.\ \mathsf{satis}\ \mathfrak{M}\ w\ (\mathsf{VAR}\ a) \iff \mathsf{satis}\ \mathfrak{M}'\ (f\ w)\ (\mathsf{VAR}\ a)) \wedge \\
\qquad (\forall\, v. \\
\qquad\quad v \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v \Rightarrow \\
\qquad\quad \mathfrak{M}'.\mathsf{frame.rel}\ (f\ w)\ (f\ v)) \wedge \\
\qquad \forall\, v'. \\
\qquad\quad v' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \mathfrak{M}'.\mathsf{frame.rel}\ (f\ w)\ v' \Rightarrow \\
\qquad\quad \exists\, v.\ v \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v \wedge f\ v = v'
\end{array}
$$

The invariance result that bounded morphism gives is stated as:

$$
\begin{array}{l}
\vdash \mathsf{bounded\_mor}\ f\ \mathfrak{M}\ \mathfrak{M}' \wedge w \in \mathfrak{M}.\mathsf{frame.world} \Rightarrow \\
\quad (\mathsf{satis}\ \mathfrak{M}\ w\ \phi \iff \mathsf{satis}\ \mathfrak{M}'\ (f\ w)\ \phi)
\end{array}
$$

As an application, we will use it to prove the tree-like property of modal formulas. The tree-like property of modal formula says that for any formula ϕ satisfied on any point in any model, there exists a tree-like model such that ϕ is satisfied at the root of the tree. As the name indicates, a tree-like model is a model such that its underlying frame is a tree. Formally, a tree is defined as:

$$
\begin{array}{l}
\mathsf{tree}\ S\ r \stackrel{\mathsf{def}}{=} \\
\quad r \in S.\mathsf{world} \wedge \\
\quad (\forall\, t.\ t \in S.\mathsf{world} \Rightarrow (\mathsf{RESTRICT}\ S.\mathsf{rel}\ S.\mathsf{world})^*\ r\ t) \wedge \\
\quad (\forall\, r_0.\ r_0 \in S.\mathsf{world} \Rightarrow \neg S.\mathsf{rel}\ r_0\ r) \wedge \\
\quad \forall\, t.\ t \in S.\mathsf{world} \wedge t \neq r \Rightarrow \exists!\, t_0.\ t_0 \in S.\mathsf{world} \wedge S.\mathsf{rel}\ t_0\ t
\end{array}
$$

Here RTC denotes the reflexive and transitive closure, and RESTRICT is a function that takes a relation and a set and return a relation that can only possibly hold on the set we give:

$$
\mathsf{RESTRICT}\ R\ s\ x\ y \stackrel{\mathsf{def}}{=} R\ x\ y \wedge x \in s \wedge y \in s
$$

We read '*tree S r*' as 'S is a tree with root r'. By definition, any tree like model is rooted. An induction on reflextive and transitive closure proves a tree has no loop:

$$
\vdash \mathsf{tree}\ s\ r \Rightarrow \forall\, t_0\ t.\ (\mathsf{RESTRICT}\ s.\mathsf{rel}\ s.\mathsf{world})^+\ t_0\ t \Rightarrow t_0 \neq t
$$

We now prove the tree-like property of modal formulas:

$$
\begin{array}{l}
\vdash \mathsf{satis}\ \mathfrak{M}\ w\ \phi \Rightarrow \\
\quad \exists\, \mathit{MODEL}\ s.\ \mathsf{tree}\ \mathit{MODEL}.\mathsf{frame}\ s \wedge \mathsf{satis}\ \mathit{MODEL}\ s\ \phi
\end{array}
$$

Proof: Suppose satis $\mathfrak{M}$ w *phi* By the invariance result of rooted model, we have satis $\mathfrak{M}'$ w *phi* where $\mathfrak{M}$ is the rooted model generated by w. By the invariance result for bounded morphisms, it suffices to prove $\mathfrak{M}'$ is the image of some bounded morphism from some tree-like model where the root of the tree is mapped to the root of $\mathfrak{M}'$. We construct such a model M'' as follows: Take the set of worlds to be the finite sequences $[w = u_0; u_1; \cdots; u_n]$ such that $n > 0$ and $M.frame.rel\ u_i\ u_{i+1}$ for all i. Define $M''.frame.rel\ [w; u_1; \cdots; u_n]\ [w; v_1; \cdots; v_m]$ iff $m = n+1$, $m_i = v_i$ for $i \leq n$ and $M.frame.rel\ u_n\ v_m$. The valuation is given by $[w; u_1; \cdots; u_n] \in M''.valt\ p$ iff $u_n \in M.valt\ p$. In the HOL, such a model looks like:

```
bounded_preimage_rooted 𝔐 x ⁼ᵈᵉᶠ
 <|frame :=
  <|world :=
     { l |
      HD l = x ∧ LENGTH l > 0 ∧
      ∀m.
         m < LENGTH l − 1 ⇒
         RESTRICT 𝔐.frame.rel 𝔐.frame.world (EL m l)
          (EL (m + 1) l) };
    rel :=
     (λl₁ l₂.
         LENGTH l₁ + 1 = LENGTH l₂ ∧
         RESTRICT 𝔐.frame.rel 𝔐.frame.world (LAST l₁)
          (LAST l₂) ∧
         ∀m. m < LENGTH l₁ ⇒ EL m l₁ = EL m l₂)|>;
  valt := (λv n. 𝔐.valt v (LAST n))|>
```

It is straightforward to check the map that sends a world in bounded_preimage_rooted $\mathfrak{M}'$ w to its last member is a bounded morphism, and $[w]$ in M'' is sent to w in $\mathfrak{M}'$, as desired.

2.3 Bisimulation

The three approachs to obtain modal equivalence has a common feature: all of them leads to a relation between models such that related states satisfies exactly the same set of propositional letters, and once we are able to make a transition in one model, we can make a corresponding transition in the other. This observation leads us to the concept of bisimulation:

$$
\begin{array}{l}
\mathsf{bisim}\ Z\ \mathfrak{M}\ \mathfrak{M}' \stackrel{\mathsf{def}}{=} \\
\quad \forall\, w\ w'. \\
\qquad w \in \mathfrak{M}.\mathsf{frame.world} \wedge w' \in \mathfrak{M}'.\mathsf{frame.world} \wedge Z\ w\ w' \Rightarrow \\
\qquad (\forall\, a.\ \mathsf{satis}\ \mathfrak{M}\ w\ (\mathsf{VAR}\ a) \iff \mathsf{satis}\ \mathfrak{M}'\ w'\ (\mathsf{VAR}\ a)) \wedge \\
\qquad (\forall\, v. \\
\qquad\quad v \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v \Rightarrow \\
\qquad\quad \exists\, v'.\ v' \in \mathfrak{M}'.\mathsf{frame.world} \wedge Z\ v\ v' \wedge \mathfrak{M}'.\mathsf{frame.rel}\ w'\ v') \wedge \\
\qquad \forall\, v'. \\
\qquad\quad v' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \mathfrak{M}'.\mathsf{frame.rel}\ w'\ v' \Rightarrow \\
\qquad\quad \exists\, v.\ v \in \mathfrak{M}.\mathsf{frame.world} \wedge Z\ v\ v' \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v
\end{array}
$$

If we have $\mathsf{bisim}\ Z\ \mathfrak{M}\ \mathfrak{M}'$, it means that Z is a bisimulation relation between worlds in $\mathfrak{M}$ and $\mathfrak{M}'$. We require any two worlds related by a bisimulation to have same atomic information and matching transition possibilities. We can see all of the three constructions we introduced before give arise to bisimulations:

$$
\begin{array}{l}
\vdash i \in dom \wedge w \in (f\ i).\mathsf{frame.world} \Rightarrow \\
\quad \mathsf{bisim\_world}\ (f\ i)\ (\mathsf{DU}\ (f, dom))\ w\ (i, w)\ \vdash \mathsf{GENSUBMODEL}\ \mathfrak{M}\ \mathfrak{M}' \Rightarrow \\
\quad \forall\, w.\ w \in \mathfrak{M}.\mathsf{frame.world} \Rightarrow \mathsf{bisim\_world}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w\ \vdash \mathsf{bounded\_mor\_image}\ f\ \mathfrak{M}\ \mathfrak{M}' \Rightarrow \\
\quad \forall\, w.\ w \in \mathfrak{M}.\mathsf{frame.world} \Rightarrow \mathsf{bisim\_world}\ \mathfrak{M}\ \mathfrak{M}'\ w\ (f\ w)
\end{array}
$$

Proof: For (i), the bisimulation relation is $\lambda\, a\ b.\ b = (i, a)$, which is linking a world to its corresponding copy in the disjoint union. For (ii), the relation is $\lambda\, n_1\ n_2.\ n_1 = n_2$. And for (iii), the relation is $\lambda\, n_1\ n_2.\ n_2 = f\ n_1$

The clauses on forth and back condition for a bisimulation relation provide precisely the information to push the induction on formula for proving the following invariance theorem about bisimulations through:

$$
\vdash \mathsf{bisim\_world}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w' \Rightarrow \mathsf{modal\_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w'
$$

The theorem above provides alternative proofs to the invariance theorems for disjoint union, generated submodels, and bounded morphisms. Hence we can say the three former constructions is actually 'the same thing'. But it is not the end of the story. A natural question to ask is : is bisimulation and modal equivalence the 'same thing'? More precisely, a bisimulation will always give a modal equivalence, conversely, is that the fact that a modal equivalence always give a bisimulation?

The answer is no. Nonetheless, we can prove the converse of the theorem above with an extra condition on the models. A model $\mathfrak{M}$ is called image finite if for any world $w \in \mathfrak{M}.\mathsf{frame.world}$, there are only finitely many worlds in $\mathfrak{M}$ that is related to w.

$$
\begin{array}{l}
\mathsf{image\_finite}\ \mathfrak{M} \stackrel{\mathsf{def}}{=} \\
\quad \forall\, x. \\
\qquad x \in \mathfrak{M}.\mathsf{frame.world} \Rightarrow \\
\qquad \mathsf{FINITE}\ \{\ y \mid y \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ x\ y\ \}
\end{array}
$$

Our main theorem is called Hennessy-Milner theorem, it says that for image finite models, modal equivalence and bisimulation coincides:

$$\begin{aligned}&\vdash \mathsf{image\_finite}\ \mathfrak{M}\ \wedge\ \mathsf{image\_finite}\ \mathfrak{M}'\ \wedge\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\\&\quad w'\ \in\ \mathfrak{M}'.\mathsf{frame.world}\ \Rightarrow\\&\quad (\mathsf{modal\_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w'\ \iff\ \mathsf{bisim\_world}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w')\end{aligned}$$

Proof: The implication from right to left is just the invariance theorem for bisimulation. We prove the implication from left to right. Given w and w' are worlds in $\mathfrak{M}$ and $\mathfrak{M}'$ which are modal equivalent, we prove the relation $\lambda\, n_1\ n_2.\ \forall\, \phi.\ \mathsf{satis}\ \mathfrak{M}\ n_1\ \phi\ \iff\ \mathsf{satis}\ \mathfrak{M}'\ n_2\ \phi$ gives a bisimulation. The first clause is immediate to check. For the second one, assume $\mathsf{modal\_eq}\ \mathfrak{M}\ \mathfrak{M}'\ n_1\ n_2$ and $\mathfrak{M}.\mathsf{frame.rel}\ n_1\ n_1'$ for some $n_1'\ \in\ \mathfrak{M}.\mathsf{frame.world}$, we prove the existence of the world $n_2'\ \in\ \mathfrak{M}'.\mathsf{frame.world}$ such that $\mathfrak{M}'.\mathsf{frame.rel}\ n_2\ n_2'$ and $\mathsf{modal\_eq}\ \mathfrak{M}\ \mathfrak{M}'\ n_1'\ n_2'$. Suppose such a n_2' does not exist, we derive a contradiction. Consider the set $S :=u'\ \in\ \mathfrak{M}'.\mathsf{frame.world}\ \wedge\ \mathfrak{M}'.\mathsf{frame.rel}\ n_2\ u'$, the first claim is that this set is finite and nonempty. Finiteness comes from the fact that $\mathfrak{M}'$ is image finite, and if the set is empty, then $\Box\ \bot$ will be a formula which holds for n_2 but not for n_1, contradicts the modal equivalence between n_1 and n_2. By assumption, for each world in S, there is a formula phi such that $\mathsf{satis}\ \mathfrak{M}\ n_1'\ phi$ but $\neg\mathsf{satis}\ \mathfrak{M}'\ n_2'\ phi$. As the set S is finite, the set of such phis is finite. Then we can take the conjunction of such phis to obtain a formula psi. Then we will have $\mathsf{satis}\ \mathfrak{M}\ n_1\ (\Diamond\ psi)$ but $\neg\mathsf{satis}\ \mathfrak{M}\ n_2\ (\Diamond\ psi)$.

The trick is what to do to capture the big conjunction. Certainly we can define a big conjunction inductively as a function that takes a finite set and give us the formula that conjuncts them together, but here is a much more direct approach such that allows us to directly obtain the psi. We can prove:

$$\begin{aligned}&\vdash \mathsf{FINITE}\ s\ \Rightarrow\\&\quad s\ \neq\ \emptyset\ \Rightarrow\\&\quad v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\\&\quad (\forall\, v'.\ v'\ \in\ s\ \Rightarrow\ \exists\, phi.\ \mathsf{satis}\ \mathfrak{M}\ v\ phi\ \wedge\ \neg\mathsf{satis}\ \mathfrak{M}'\ v'\ phi)\ \Rightarrow\\&\quad \exists\, psi.\ \mathsf{satis}\ \mathfrak{M}\ v\ psi\ \wedge\ \forall\, v'.\ v'\ \in\ s\ \Rightarrow\ \neg\mathsf{satis}\ \mathfrak{M}'\ v'\ psi\end{aligned}$$

Using this lemma, we obtain the psi we want by plugging in S to be the s.

3 Finite model property

In this chapter, we prove the finite model property of our modal language, which says if a modal formula is satisfied on an arbitary mode, then it can be satisfied on a finite model, where finite model means the finitness of the set of worlds. We will discuss two methods for building finite models for satisfiable modal formulas, namely via filteration and selection.

3.1 Finite model property via filteration

One way to get a finite model is by taking the quotient of the world set. The quotient model we will get is called filtration of a model. The equivalence relation that we will use for taking the quotient is defined using

the concept of 'closed under subformulas'. Firstly, subformulas is a function that takes a formula and gives a set of the subformulas of this formula, defined by:

$$\begin{aligned}
&\mathsf{subforms}\ (\mathsf{VAR}\ a) \stackrel{\mathsf{def}}{=} \{\ \mathsf{VAR}\ a\ \}\\
&\mathsf{subforms}\ \bot \stackrel{\mathsf{def}}{=} \{\ \bot\ \}\\
&\mathsf{subforms}\ (\neg f) \stackrel{\mathsf{def}}{=} \neg f\ \mathsf{INSERT}\ \mathsf{subforms}\ f\\
&\mathsf{subforms}\ (\mathsf{DISJ}\ f_1\ f_2) \stackrel{\mathsf{def}}{=}\\
&\quad \mathsf{DISJ}\ f_1\ f_2\ \mathsf{INSERT}\ \mathsf{subforms}\ f_1\ \cup\ \mathsf{subforms}\ f_2\\
&\mathsf{subforms}\ (\Diamond\ f) \stackrel{\mathsf{def}}{=} \Diamond f\ \mathsf{INSERT}\ \mathsf{subforms}\ f
\end{aligned}$$

The definition says a subformula of a formula is a part of a formula which is itself a formula. Some properties of subformulas can be proved immediately, for instance, we have any formula is a subformula of itself. Also subformulas are transitive, and the set of subformulas for any formula is finite.

$$\vdash phi\ \in\ \mathsf{subforms}\ phi\ \ \vdash f\ \in\ \mathsf{subforms}\ phi\ \wedge\ phi\ \in\ \mathsf{subforms}\ psi\ \Rightarrow\ f\ \in\ \mathsf{subforms}\ psi\ \ \vdash \mathsf{FINITE}\ (\mathsf{subforms}\ phi)$$

We say a set Σ is closed under formula if for any formula phi in the set, any subformula of phi is also in Σ.

$$\begin{aligned}
&\mathsf{CUS}\ \Sigma \stackrel{\mathsf{def}}{=}\\
&\ \forall f\ f'.\\
&\quad (\mathsf{DISJ}\ f\ f'\ \in\ \Sigma\ \Rightarrow\ f\ \in\ \Sigma\ \wedge\ f'\ \in\ \Sigma)\ \wedge\\
&\quad (\neg f\ \in\ \Sigma\ \Rightarrow\ f\ \in\ \Sigma)\ \wedge\ (\Diamond f\ \in\ \Sigma\ \Rightarrow\ f\ \in\ \Sigma)
\end{aligned}$$

We can easily check the set of subformulas of any formulas is closed under subformulas:

$$\vdash \mathsf{CUS}\ (\mathsf{subforms}\ phi)$$

And for a model $\mathfrak{M}$, the equivalence relation we will use to filtrate its world set is:

$$\begin{aligned}
&\mathsf{REL_CUS}\ \Sigma\ \mathfrak{M} \stackrel{\mathsf{def}}{=}\\
&\ (\lambda\ w\ v.\\
&\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\\
&\quad \forall\ phi.\ phi\ \in\ \Sigma\ \Rightarrow\ (\mathsf{satis}\ \mathfrak{M}\ w\ phi\ \iff\ \mathsf{satis}\ \mathfrak{M}\ v\ phi))
\end{aligned}$$

Under this equivalence relation, for any world $w\ \in\ \mathfrak{M}.\mathsf{frame.world}$, the equivalence class it belongs to looks like:

$$\mathsf{EC\_CUS}\ \Sigma\ \mathfrak{M}\ w\ \stackrel{\mathsf{def}}{=}\ \{\ v\ |\ \mathsf{REL\_CUS}\ \Sigma\ \mathfrak{M}\ w\ v\ \}$$

We can take these equivalence class to be the set of worlds for the filtrated model, but then the model we get will have different type from the original model. To avoid changing the type, we define $\mathsf{EC\_REP}\ \Sigma\ \mathfrak{M}\ w$ to be the representative from the equivalence class where w lives in, and we will use $\mathsf{EC\_REP\_SET}\ \Sigma\ \mathfrak{M}$ to be the set of worlds for the filtrated model.

$$\mathsf{EC\_REP}\ \Sigma\ \mathfrak{M}\ w \stackrel{\text{def}}{=} \mathsf{CHOICE}\ (\mathsf{EC\_CUS}\ \Sigma\ \mathfrak{M}\ w)\mathsf{EC\_REP\_SET}\ \Sigma\ \mathfrak{M} \stackrel{\text{def}}{=}$$
$$\{\ n \mid \exists\, w.\ w \in \mathfrak{M}.\mathsf{frame.world} \wedge n = \mathsf{EC\_REP}\ \Sigma\ \mathfrak{M}\ w\ \}$$

The definition of filtration of a model $\mathfrak{M}$ via a subformula-closed set is given by a relation, we read as $\mathsf{filtration}\ \mathfrak{M}\ \Sigma\ FLT$ as 'FLT is a filteration of $\mathfrak{M}$ under Σ.

$$\begin{array}{l}
\mathsf{filtration}\ \mathfrak{M}\ \Sigma\ FLT \stackrel{\text{def}}{=} \\
\quad \mathsf{CUS}\ \Sigma \wedge FLT.\mathsf{frame.world} = \mathsf{EC_REP_SET}\ \Sigma\ \mathfrak{M} \wedge \\
\quad (\forall\, w\ v. \\
\qquad w \in \mathfrak{M}.\mathsf{frame.world} \wedge v \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v \Rightarrow \\
\qquad FLT.\mathsf{frame.rel}\ (\mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ w)\ (\mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ v)) \wedge \\
\quad (\forall\, w\ v. \\
\qquad w \in \mathfrak{M}.\mathsf{frame.world} \wedge v \in \mathfrak{M}.\mathsf{frame.world} \wedge \\
\qquad FLT.\mathsf{frame.rel}\ (\mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ w)\ (\mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ v) \Rightarrow \\
\qquad \forall\, phi\ psi. \\
\qquad\quad phi \in \Sigma \wedge phi = \Diamond\, psi \Rightarrow \\
\qquad\quad \mathsf{satis}\ \mathfrak{M}\ v\ psi \Rightarrow \\
\qquad\quad \mathsf{satis}\ \mathfrak{M}\ w\ phi) \wedge \\
\quad \forall\, p\ s. \\
\qquad FLT.\mathsf{valt}\ p\ s \iff \\
\qquad\quad \exists\, w.\ s = \mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ w \wedge \mathsf{satis}\ \mathfrak{M}\ w\ (\mathsf{VAR}\ p)
\end{array}$$

The definition above forces filtration to only make sense for subformula closed sets. For any model $\mathfrak{M}$ and set Σ which is closed under subformulas, we can find some model such that it is a filtration of $\mathfrak{M}$ under Σ. One construction of filtration is given by:

$$\begin{array}{l}
\mathsf{FLT}\ \mathfrak{M}\ \Sigma \stackrel{\text{def}}{=} \\
\quad \langle|\mathsf{frame} := \\
\quad\quad \langle|\mathsf{world} := \mathsf{EC_REP_SET}\ \Sigma\ \mathfrak{M}; \\
\quad\quad\quad \mathsf{rel} := \\
\quad\quad\quad\quad (\lambda\, n_1\ n_2. \\
\quad\quad\quad\quad\quad \exists\, w_1\ w_2. \\
\quad\quad\quad\quad\quad\quad w_1 \in \mathfrak{M}.\mathsf{frame.world} \wedge w_2 \in \mathfrak{M}.\mathsf{frame.world} \wedge \\
\quad\quad\quad\quad\quad\quad n_1 = \mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ w_1 \wedge n_2 = \mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ w_2 \wedge \\
\quad\quad\quad\quad\quad\quad \exists\, w'\ v'. \\
\quad\quad\quad\quad\quad\quad\quad w' \in \mathfrak{M}.\mathsf{frame.world} \wedge v' \in \mathfrak{M}.\mathsf{frame.world} \wedge \\
\quad\quad\quad\quad\quad\quad\quad w' \in \mathsf{EC_CUS}\ \Sigma\ \mathfrak{M}\ w_1 \wedge \\
\quad\quad\quad\quad\quad\quad\quad v' \in \mathsf{EC_CUS}\ \Sigma\ \mathfrak{M}\ w_2 \wedge \mathfrak{M}.\mathsf{frame.rel}\ w'\ v')|\rangle; \\
\quad\quad \mathsf{valt} := (\lambda\, p\ s. \exists\, w.\ s = \mathsf{EC_REP}\ \Sigma\ \mathfrak{M}\ w \wedge \mathsf{satis}\ \mathfrak{M}\ w\ (\mathsf{VAR}\ p))|\rangle
\end{array}$$

In fact, it is not the fact that we always have a unique filtration of a model under a subformula closed set, but we will just use the one defined above. It is routine to check the above definition does give a filtration:

$$\vdash \mathsf{CUS}\ \Sigma\ \Rightarrow\ \mathsf{filtration}\ \mathfrak{M}\ \Sigma\ (\mathsf{FLT}\ \mathfrak{M}\ \Sigma)$$

We are interested in two properties of filtration of models that directly give the proof of finite model property. Firstly, filtrating a model using a finite set gives a finite model:

$$\begin{array}{l}\vdash \mathsf{FINITE}\ \Sigma\ \wedge\ \mathsf{filtration}\ \mathfrak{M}\ \Sigma\ FLT\ \Rightarrow \\ \quad \mathsf{CARD}\ FLT.\mathsf{frame.world}\ \leq\ 2\ **\ \mathsf{CARD}\ \Sigma\end{array}$$

Proof: As Σ is finite, it suffices to give an injection from the world set of FLT to $\mathsf{POW}\ \Sigma$. Such an injection is given by sending a world w to the set of formulas in Σ that is satisfies at w.

Secondly, we need the satisfication of modal formula to be preserved under filtration, in the following sense:

$$\begin{array}{l}\vdash phi\ \in\ \Sigma\ \Rightarrow \\ \quad \forall\, w. \\ \qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{filtration}\ \mathfrak{M}\ \Sigma\ FLT\ \Rightarrow \\ \qquad (\mathsf{satis}\ \mathfrak{M}\ w\ phi\ \iff\ \mathsf{satis}\ FLT\ (\mathsf{EC\_REP}\ \Sigma\ \mathfrak{M}\ w)\ phi)\end{array}$$

Putting the last two theorems together and we get the finite model property via filtration:

$$\begin{array}{l}\vdash \mathsf{satis}\ \mathfrak{M}\ w\ phi\ \Rightarrow \\ \quad \exists \mathfrak{M}'\ w'. \\ \qquad w'\ \in\ \mathfrak{M}'.\mathsf{frame.world}\ \wedge\ \mathsf{satis}\ \mathfrak{M}'\ w'\ phi\ \wedge \\ \qquad \mathsf{FINITE}\ \mathfrak{M}'.\mathsf{frame.world}\end{array}$$

3.2 Finite model property via selection

Another method to build finite models for an arbitary model is by selection. The intuition of this method is that only finitely many diamonds can occur in a modal formula, so any formula can only capture the information of finitely depth. To make the notion of 'depth' precise, we define the degree of a modal formula, which count the diamonds appear in a model formula and hence measure how much of the model can a modal formula see from the current state:

$$\begin{array}{l}\mathsf{DEG}\ (\mathsf{VAR}\ p)\ \stackrel{\mathsf{def}}{=}\ 0 \\ \mathsf{DEG}\ \bot\ \stackrel{\mathsf{def}}{=}\ 0 \\ \mathsf{DEG}\ (\neg\phi)\ \stackrel{\mathsf{def}}{=}\ \mathsf{DEG}\ \phi \\ \mathsf{DEG}\ (\mathsf{DISJ}\ \phi_1\ \phi_2)\ \stackrel{\mathsf{def}}{=}\ \mathsf{MAX}\ (\mathsf{DEG}\ \phi_1)\ (\mathsf{DEG}\ \phi_2) \\ \mathsf{DEG}\ (\Diamond\ \phi)\ \stackrel{\mathsf{def}}{=}\ \mathsf{DEG}\ \phi\ +\ 1\end{array}$$

A modal formula of degree zero is exactly a propositional formula:

$$\vdash \mathsf{DEG}\, f \;=\; 0 \;\Longleftrightarrow\; \mathsf{propform}\, f$$

The crucial fact that we will use about the degree of formulas is the following lemma:

$$\begin{array}{l}
\vdash \mathsf{FINITE}\; s \;\wedge\; \mathsf{INFINITE}\, \mathcal{U}(:\beta) \;\Rightarrow \\
\quad \forall\, n. \\
\qquad \mathsf{FINITE} \\
\qquad\quad (\{\, f \mid \mathsf{DEG}\, f \;\leq\; n \;\wedge\; \forall\, a.\, \mathsf{VAR}\; a \;\in\; \mathsf{subforms}\, f \;\Rightarrow\; a \;\in\; s \,\}\, //E \\
\qquad\quad \mu)
\end{array}$$

The proof of this lemma is by induction on the degree n. We prove it in the following interlude:

3.3 Interlude I: Finiteness of non-equivalent modal formulas in each degree

3.3.1 Base case

For the base case, we need to prove if we only use propositional letters in s where s is a finite set, then up to equivalence, we can only onbtain finitely many propositional formulas. To prove this, it suffices to find out an injection from the set of equivalence class of propositional formulas that only uses propositional letters in s to a finite set. We will prove the function that sends an equivalence class of formulas to its image under the function $\lambda f.\{\sigma \mid peval\ \sigma\ f\} \cap (POW\ s)$, which sends a formula f to the set $\{\sigma \mid peval\ \sigma\ f\} \cap (POW\ s)$ of assignments of truth value on propositional letters in s that makes f holds, is an injection to the finite set $\mathsf{POW}\ (\mathsf{POW}\ (\mathsf{POW}\ s))$. We can see the function we define has correct codomain: A formula f is sent to a set of subsets of s, which is an element of $\mathsf{POW}\ (\mathsf{POW}\ s)$, so a equivalence class has image as a set of elements in $\mathsf{POW}\ (\mathsf{POW}\ s)$, which lies in $\mathsf{POW}\ (\mathsf{POW}\ (\mathsf{POW}\ s))$.

Let us firstly investigate what the function $\lambda f.\{\sigma \mid peval\ \sigma\ f\} \cap (POW\ s)$ does. We claim the image of each equivalence class of this function is a singleton:

$$\begin{array}{l}
\vdash x \;\in \\
\quad \{\, f \mid \mathsf{propform}\, f \;\wedge\; \forall\, a.\, \mathsf{VAR}\; a \;\in\; \mathsf{subforms}\, f \;\Rightarrow\; a \;\in\; s \,\}\, //E\; \mu \;\wedge \\
\quad \phi \;\in\; x \;\Rightarrow \\
\quad \mathsf{IMAGE}\; (\lambda f.\; \{\; \sigma \mid \mathsf{peval}\; \sigma\; f \;\}\; \cap\; \mathsf{POW}\; s)\; x \;= \\
\qquad \{\; \{\; \sigma \mid \mathsf{peval}\; \sigma\; \phi \;\}\; \cap\; \mathsf{POW}\; s \;\}
\end{array}$$

Indeed, if two $f1$ and $f2$ formulas are equivalent, then for any assignment of truth value of propositional symbols, it makes $f1$ true iff it makes $f2$ true:

$$\begin{array}{l}
\vdash \mathsf{propform}\; f_1 \;\wedge\; \mathsf{propform}\; f_2 \;\wedge \\
\quad (\forall\, \mathfrak{M}\; w.\; \mathsf{satis}\; \mathfrak{M}\; w\; f_1 \;\Longleftrightarrow\; \mathsf{satis}\; \mathfrak{M}\; w\; f_2) \;\Rightarrow \\
\quad \forall\, \sigma.\; \mathsf{peval}\; \sigma\; f_1 \;\Longleftrightarrow\; \mathsf{peval}\; \sigma\; f_2
\end{array}$$

It is because if we have a $\mathsf{peval}\ \sigma\ f_1 \wedge \neg\mathsf{peval}\ \sigma\ f_2$, then we can construct a model with only one world w, no relation, and define the valuation at w for propositional symbols to be σ. Then by `peval_satis`, we get $\mathsf{satis}\ \mathfrak{M}\ w\ f_1 \wedge \neg\mathsf{satis}\ \mathfrak{M}\ w\ f_2$, a contradiction. As a consequence, we have:

$$\vdash \mathsf{propform}\ f_1\ \wedge\ \mathsf{propform}\ f_2\ \wedge\ \mathsf{equiv0}\ \mu\ f_1\ f_2\ \Rightarrow$$
$$(\lambda f\ s.\ \mathsf{peval}\ s\ f)\ f_1\ =\ (\lambda f\ s.\ \mathsf{peval}\ s\ f)\ f_2$$

which says if two formulas are equivalent, then the sets of valuations that makes them hold are identical. As a consequence, these two sets will still be equal after taking inter section with $POW\ s$. This proves `IMAGE_peval_singlton_strengthen`.

Recall we proved the condition for a propositional formula to hold as the lemma `peval_satis_strengthen`, it gives arise to this useful lemma:

$$\begin{aligned}\vdash\ & \mathsf{propform}\ f_1\ \wedge\ \mathsf{propform}\ f_2\ \wedge\\ & (\forall a.\ \mathsf{VAR}\ a\ \in\ \mathsf{subforms}\ f_1\ \Rightarrow\ a\ \in\ s)\ \wedge\\ & (\forall a.\ \mathsf{VAR}\ a\ \in\ \mathsf{subforms}\ f_2\ \Rightarrow\ a\ \in\ s)\ \Rightarrow\\ & (\forall \sigma.\ \sigma\ \in\ \mathsf{POW}\ s\ \Rightarrow\ (\mathsf{peval}\ \sigma\ f_1\ \Longleftrightarrow\ \mathsf{peval}\ \sigma\ f_2))\ \Rightarrow\\ & \forall \mathfrak{M}\ w.\ \mathsf{satis}\ \mathfrak{M}\ w\ f_1\ \Longleftrightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ f_2\end{aligned}$$

Proof: Suppose $\mathsf{satis}\ \mathfrak{M}\ w\ f_1$ for some model $\mathfrak{M}$, then by `peval_satis_strengthen`, we have $\mathsf{peval}\ ((\lambda a.\ w\ \in\ \mathfrak{M}.\mathsf{valt}\ a)\ \cap\ s)\ f_1$. As $(\lambda a.\ w\ \in\ \mathfrak{M}.\mathsf{valt}\ a)\ \cap\ s$ gives a subset of s, we conclude $\mathsf{peval}\ ((\lambda a.\ w\ \in\ \mathfrak{M}.\mathsf{valt}\ a)\ \cap\ s)\ f_2$ by assumption, and then $\mathsf{satis}\ \mathfrak{M}\ w\ f_2$ by `peval_satis_strengthen` again. The other direction is the same.

We can now prove the injection:

$$\begin{aligned}\vdash\ & \mathsf{INJ}\ (\lambda\ eqc.\ \mathsf{IMAGE}\ (\lambda f.\ \{\ s\ |\ \mathsf{peval}\ s\ f\ \}\ \cap\ \mathsf{POW}\ s)\ eqc)\\ & (\{\ f\ |\ \mathsf{propform}\ f\ \wedge\ \forall a.\ \mathsf{VAR}\ a\ \in\ \mathsf{subforms}\ f\ \Rightarrow\ a\ \in\ s\ \}\ //E\ \mu)\\ & (\mathsf{POW}\ (\mathsf{POW}\ (\mathsf{POW}\ s)))\end{aligned}$$

Proof: If two equivalence classes represented by x and x' respectively has identical image under the function $\lambda f.\{s \mid peval\ s f\} \cap POW\ s$, then by `IMAGE_peval_singlton_strengthen`, it gives $\{\{\sigma \mid peval\ \sigma x\} \cap POW\ s\}$. This means $\forall \sigma.\ \sigma\ \in\ \mathsf{POW}\ s\ \Rightarrow\ \mathsf{peval}\ \sigma\ x\ \Longleftrightarrow\ \mathsf{peval}\ \sigma\ x'$, which means x and x' are equivalent by `equiv0_peval_strengthen`.

3.3.2 Step case

The key observation for proving the step case is that any formulas which only uses propositional letters in a fixed finite set ss and of degree no more than $n+1$ is obtained by boolean combination on the set s formed by formulas of form $\Diamond\ phi$ which also only uses propositional letters in ss where $\mathsf{DEG}\ phi\ \leq\ n$ and propositional letters in ss. Saying a formula f is a boolean combination of a set s of formulas is to say f is either itself an element in s, or can be obtained by combining elements in s using the connectives $\neg$ and $\vee$. We define boolean combination as an inductive relation, where $\mathsf{IBC}\ f\ s$ reads 'f is a boolean combination of elements in the set s':

$$\begin{array}{l}
\vdash (\forall f_1\ f_2\ s.\ \mathsf{IBC}\ f_1\ s\ \wedge\ \mathsf{IBC}\ f_2\ s\ \Rightarrow\ \mathsf{IBC}\ (\mathsf{DISJ}\ f_1\ f_2)\ s)\ \wedge \\
\quad (\forall s.\ \mathsf{IBC}\ \bot\ s)\ \wedge\ (\forall f\ s.\ \mathsf{IBC}\ f\ s\ \Rightarrow\ \mathsf{IBC}\ (\neg f)\ s)\ \wedge \\
\quad \forall f\ s.\ f\ \in\ s\ \Rightarrow\ \mathsf{IBC}\ f\ s\ \vdash (\forall f_1\ f_2\ s.\ IBC'\ f_1\ s\ \wedge\ IBC'\ f_2\ s\ \Rightarrow\ IBC'\ (\mathsf{DISJ}\ f_1\ f_2)\ s)\ \wedge \\
\quad (\forall s.\ IBC'\ \bot\ s)\ \wedge\ (\forall f\ s.\ IBC'\ f\ s\ \Rightarrow\ IBC'\ (\neg f)\ s)\ \wedge \\
\quad (\forall f\ s.\ f\ \in\ s\ \Rightarrow\ IBC'\ f\ s)\ \Rightarrow \\
\quad \forall a_0\ a_1.\ \mathsf{IBC}\ a_0\ a_1\ \Rightarrow\ IBC'\ a_0\ a_1\ \vdash \mathsf{IBC}\ a_0\ a_1\ \Longleftrightarrow \\
\quad\ (\exists f_1\ f_2.\ a_0\ =\ \mathsf{DISJ}\ f_1\ f_2\ \wedge\ \mathsf{IBC}\ f_1\ a_1\ \wedge\ \mathsf{IBC}\ f_2\ a_1)\ \vee \\
\quad\quad a_0\ =\ \bot\ \vee\ (\exists f.\ a_0\ =\ \neg f\ \wedge\ \mathsf{IBC}\ f\ a_1)\ \vee\ a_0\ \in\ a_1
\end{array}$$

We can prove our claim in the last paragraph with induction:

$$\begin{array}{l}
\vdash \mathsf{DEG}\ x\ \leq\ n\ +\ 1\ \wedge\ (\forall a.\ \mathsf{VAR}\ a\ \in\ \mathsf{subforms}\ x\ \Rightarrow\ a\ \in\ s)\ \Longleftrightarrow \\
\quad \mathsf{IBC}\ x \\
\quad\quad (\{\ \mathsf{VAR}\ v\ |\ v\ \in\ s\ \}\ \cup \\
\quad\quad \{\ \Diamond\ psi\ | \\
\quad\quad\ \mathsf{DEG}\ psi\ \leq\ n\ \wedge\ \forall a.\ \mathsf{VAR}\ a\ \in\ \mathsf{subforms}\ psi\ \Rightarrow\ a\ \in\ s\ \}\ )
\end{array}$$

Observe a formula which is a boolean combinations on an empty set is either equivalent to $\bot$ or $\top$:

$$\vdash \mathsf{IBC}\ f\ s\ \Rightarrow\ s\ =\ \emptyset\ \Rightarrow\ \mathsf{equiv0}\ \mu\ f\ \mathsf{TRUE}\ \vee\ \mathsf{equiv0}\ \mu\ f\ \bot$$

so we are interested in boolean combination of non-empty sets along our proof, and leave the empty case to be treat separately at the very end. Our aim is to prove the set of boolean combinations on a set which contains only finitely many non-equivalent formulas only contain finitely many non-equivalent formulas:

$$\vdash fs\ \neq\ \emptyset\ \Rightarrow\ \mathsf{FINITE}\ (fs//E\ \mu)\ \Rightarrow\ \mathsf{FINITE}\ (\{\ f\ |\ \mathsf{IBC}\ f\ fs\ \}\ //E\ \mu)$$

Note that it suffices to prove the set of formulas obtained by boolean combination on a finite set is finite up to equivalence. Since then as fs have only finitely many non-equivalent formulas, the set $\mathsf{IMAGE}\ \mathsf{CHOICE}\ (fs//E\ \mu)$ of representatives of the equivalence classes is finite. There is a surjection $\lambda s.\{y \mid IBC\ y\ fs \wedge !f.f \in s\ \Longrightarrow\ equiv0\ \mu\ y\ f\}$ from $\{f \mid IBC\ f\ (IMAGE\ CHOICE\ (fs//E\ \mu))\}//E\ \mu$ to $\{f \mid IBC\ f\ fs\}//E\ \mu$ is a surjection, showing the codomain is finite.

The strategy we used to prove this is to prove that any formula obtained by a boolean combination on a set is equivalent to a formula in disjunction normal form on the same set. The disjunction normal form is defined by:

$$\mathsf{DNF\_OF}\ f\ fs\ \stackrel{\mathsf{def}}{=}\ \mathsf{DISJ\_OF}\ f\ \{\ c\ |\ \mathsf{CONJ\_OF}\ c\ fs\ \}$$

where $\mathsf{CONJ\_OF}$ is defined by:

$$
\begin{array}{l}
\vdash (\forall\, c_0\ c.\ c\ =\ c_0\ \vee\ c\ =\ \neg c_0\ \Rightarrow\ \mathsf{CONJ\_OF}\ c\ \{\ c_0\ \})\ \wedge \\
\quad \forall\, f_0\ f_1\ f_2\ fs. \\
\qquad (f_1\ =\ f_0\ \vee\ f_1\ =\ \neg f_0)\ \wedge\ f_0\ \in\ fs\ \wedge \\
\qquad \mathsf{CONJ\_OF}\ f_2\ (fs\ \mathsf{DELETE}\ f_0)\ \Rightarrow \\
\qquad \mathsf{CONJ\_OF}\ (\mathsf{AND}\ f_1\ f_2)\ fs\ \vdash (\forall\, c_0\ c.\ c\ =\ c_0\ \vee\ c\ =\ \neg c_0\ \Rightarrow\ CONJ\_OF'\ c\ \{\ c_0\ \})\ \wedge \\
\quad (\forall\, f_0\ f_1\ f_2\ fs. \\
\qquad (f_1\ =\ f_0\ \vee\ f_1\ =\ \neg f_0)\ \wedge\ f_0\ \in\ fs\ \wedge \\
\qquad CONJ\_OF'\ f_2\ (fs\ \mathsf{DELETE}\ f_0)\ \Rightarrow \\
\qquad CONJ\_OF'\ (\mathsf{AND}\ f_1\ f_2)\ fs)\ \Rightarrow \\
\forall\, a_0\ a_1.\ \mathsf{CONJ\_OF}\ a_0\ a_1\ \Rightarrow\ CONJ\_OF'\ a_0\ a_1\ \vdash \mathsf{CONJ\_OF}\ a_0\ a_1\ \iff \\
\quad (\exists\, c_0.\ a_1\ =\ \{\ c_0\ \}\ \wedge\ (a_0\ =\ c_0\ \vee\ a_0\ =\ \neg c_0))\ \vee \\
\quad \exists\, f_0\ f_1\ f_2. \\
\qquad a_0\ =\ \mathsf{AND}\ f_1\ f_2\ \wedge\ (f_1\ =\ f_0\ \vee\ f_1\ =\ \neg f_0)\ \wedge\ f_0\ \in\ a_1\ \wedge \\
\qquad \mathsf{CONJ\_OF}\ f_2\ (a_1\ \mathsf{DELETE}\ f_0)
\end{array}
$$

and for a formula f, we say $\mathsf{DISJ\_OF}\ f\ fs$ when f is either the '$\bot$' or a formula such that $\mathsf{DISJ\_OF0}\ f\ fs$

$$
\begin{array}{l}
\vdash (\forall f\ fs.\ f\ \in\ fs\ \Rightarrow\ \mathsf{DISJ\_OF0}\ f\ fs)\ \wedge \\
\quad \forall\, f_1\ f_2\ fs. \\
\qquad f_1\ \in\ fs\ \wedge\ \mathsf{DISJ\_OF0}\ f_2\ (fs\ \mathsf{DELETE}\ f_1)\ \Rightarrow \\
\qquad \mathsf{DISJ\_OF0}\ (\mathsf{DISJ}\ f_1\ f_2)\ fs\ \vdash (\forall f\ fs.\ f\ \in\ fs\ \Rightarrow\ DISJ\_OF_0'\ f\ fs)\ \wedge \\
\quad (\forall\, f_1\ f_2\ fs. \\
\qquad f_1\ \in\ fs\ \wedge\ DISJ\_OF_0'\ f_2\ (fs\ \mathsf{DELETE}\ f_1)\ \Rightarrow \\
\qquad DISJ\_OF_0'\ (\mathsf{DISJ}\ f_1\ f_2)\ fs)\ \Rightarrow \\
\quad \forall\, a_0\ a_1.\ \mathsf{DISJ\_OF0}\ a_0\ a_1\ \Rightarrow\ DISJ\_OF_0'\ a_0\ a_1\ \vdash \mathsf{DISJ\_OF0}\ a_0\ a_1\ \iff \\
\qquad a_0\ \in\ a_1\ \vee \\
\qquad \exists\, f_1\ f_2. \\
\qquad\quad a_0\ =\ \mathsf{DISJ}\ f_1\ f_2\ \wedge\ f_1\ \in\ a_1\ \wedge \\
\qquad\quad \mathsf{DISJ\_OF0}\ f_2\ (a_1\ \mathsf{DELETE}\ f_1)\mathsf{DISJ\_OF}\ f\ fs\ \stackrel{\mathsf{def}}{=}\ f\ =\ \bot\ \vee\ \mathsf{DISJ\_OF0}\ f\ fs
\end{array}
$$

As an example, the four $\mathsf{CONJ\_OF}$ formulas on the set $\{f1, f2\}$ are $\mathsf{AND}\ f_1\ f_2$, $\mathsf{AND}\ (\neg f_1)\ f_2$ $\mathsf{AND}\ f_1\ (\neg f_2)$ and $\mathsf{AND}\ (\neg f_1)\ (\neg f_2)$. For another example, consider the set $fs := \{f1, f2, f3\}$ of three formulas, we can say $\mathsf{CONJ\_OF}\ (\mathsf{AND}\ f_3\ (\mathsf{AND}\ f_1\ f_2))\ fs$, $\mathsf{CONJ\_OF}\ (\mathsf{AND}\ f_2\ (\mathsf{AND}\ (\neg f_1)\ f_3))\ fs$, $\mathsf{CONJ\_OF}\ (\mathsf{AND}\ (\neg f_3)\ (\mathsf{AND}\ f_2\ f_1))\ fs$ and so on. But it is not the fact that $\mathsf{CONJ\_OF}\ f_1\ fs$, since from the definition, any element of fs or its negation must appear exactly once in a $\mathsf{CONJ\_OF}$ formula on fs. Let $CO := \{f\ |\mathsf{CONJ\_OF}\ f\ fs\}$, then we can say $\mathsf{DISJ\_OF}$
$(\mathsf{DISJ}\ (\mathsf{AND}\ f_3\ (\mathsf{AND}\ f_1\ f_2))\ (\mathsf{AND}\ f_2\ (\mathsf{AND}\ (\neg f_1)\ f_3)))\ CO$, $\mathsf{DISJ\_OF}\ (\mathsf{AND}\ f_2\ (\mathsf{AND}\ (\neg f_1)\ f_3))\ CO$ and so on, so both these two formulas are disjunction normal form on fs, but we cannot say $\mathsf{DISJ\_OF}$
$(\mathsf{DISJ}\ (\mathsf{AND}\ f_2\ (\mathsf{AND}\ (\neg f_1)\ f_3))\ (\mathsf{AND}\ f_2\ (\mathsf{AND}\ (\neg f_1)\ f_3)))\ CO$ or $\mathsf{DISJ\_OF}$

(DISJ (AND f_2 (AND ($\neg f_1$) f_3)) ($\neg$AND f_2 (AND ($\neg f_1$) f_3))) CO, since any formula in CO or its negation is only allowed to appear in a DISJ_OF formula on CO for at most once, so these two are not in disjunction normal form.

If we start with a finite set, by induction on finiteness using `FINITE_COMPLETE_INDUCTION` , we conclude the set of CONJ_OF formulas and DISJ_OF0, hence the DNF_OF formulas on this set is finite:

$\vdash$ FINITE s $\Rightarrow$ FINITE { f | CONJ_OF f s } $\vdash$ FINITE s $\Rightarrow$ FINITE { f | DISJ_OF0 f s } $\vdash$ FINITE fs $\Rightarrow$ FINITE { f | DNF

Once we prove:

$$\begin{aligned}&\vdash \mathsf{IBC}\ f\ fs\ \Rightarrow\\ &\quad \mathsf{FINITE}\ fs\ \wedge\ fs\ \neq\ \emptyset\ \Rightarrow\\ &\quad \exists p.\ \mathsf{DNF\_OF}\ p\ fs\ \wedge\ \mathsf{equiv0}\ \mu\ f\ p\end{aligned}$$

we get `FINITE_FINITE_IBC`, which leads to a proof of the step case of `prop_2_29_strengthen` goes as follows: Suppose $\{f|DEG\ f\ \leq\ n \wedge \forall a.VAR\ a\ \in\ subforms\ f\ \implies\ a\ \in\ s\}//E\ \mu$ is finite and let $A = (\{VAR\ v|v \in s\} \cup \{\Diamond psi|DEG\ psi \leq n \wedge \forall a.(VAR\ a) \in subforms\ psi\ \implies\ a \in s\})$. By `DEG_IBC_strengthen`, we have $B := \{f|DEG\ f \leq n+1 \wedge \forall a.VAR\ a \in subforms\ f\ \implies\ a \in s\} = \{phi|IBC\ phi\ A\}$, we prove B is finite up to equivalence. If $A = \emptyset$, then by `IBC_EMPTY_lemma`, B contains only two non-equivaelent formulas $\bot$ and $\top$. Otherwise, by `FINITE_FINITE_IBC`, it suffices to prove A is finite up to equivalence. Under the assumption INFINITE $\mathcal{U}(:\beta)$, equiv0 μ ($\Diamond$ f) ($\Diamond$ g) iff equiv0 μ f g, so the inductive hypothesis together with the assumption FINITE s gives the finiteness of A.

Therefore, it remains to give a proof of `IBC_DNF_EXISTS`. The proof is by rule induction on IBC, the things to prove are:

Base case:

- The falsity $\bot$ is equivalent to a disjunction normal form on fs.
- For any element $f \in fs$, it is equivalent to a disjunction normal form on fs.

Step case:

- Under the assumption that fs is finite and non-empty, if $f1, f2$ are boolean combination of the set fs which are equivalent to $p1, p2$ of disjunction normal form respectively, then DISJ f_1 f_2 is equivalent to a formula in disjunction normal form.
- With the same assumption on fs, if f is a boolean combination of fs and equivalent to p in disjunction normal form, then $\neg f$ is equivalent to a formula in disjunction normal form.

The first item of base case is trivial since $\bot$ itself is in disjunction normal form, we begin by proving the second item of the base case.

3.3.3 Case for $f \in fs$

We aim to prove:

$$\vdash \mathsf{FINITE}\ fs\ \wedge\ fs\ \neq\ \emptyset\ \Rightarrow$$
$$\forall f.\, f\ \in\ fs\ \Rightarrow\ \exists p.\, \mathsf{DNF\_OF}\ p\ fs\ \wedge\ \mathsf{equiv0}\ \mu\ f\ p$$

Let us consider what does the DNF_OF formula p on fs that is equivalent to and element of fs look like: We require p to be satisfied if and only if f is satisfied. As p is of disjunction normal form, p is satisfied once some of its disjuncts is satisfied. Hence we require the satisfication of each disjuncts of p to imply the satisfication of f, that is, f is a conjunct of each disjunct of p. So up to rearrangement, p is a disjunction of conjunctions $f \wedge f_1 \cdots f_n$ where each formula in $fs/\{f\}$ or its negation appears exactly once in these f_i's. Such a formula is equivalent to $f \wedge g$, where g is the disjunction of the formulas obtained by taking f out of each conjunct of p. And $f \wedge g$ is equivalent to f if and only if g is equivalent to $\top$. Hence, g must be the disjunction of all the CONJ_OF formulas on $fs/\{f\}$. By conclusion, a disjunction normal form we need can be taken as the disjunction of conjunctions starting with f, with its tail ranging over all possible combination of negated and unnegated formulas in $fs/\{f\}$. Use the set $\{f_1, f_2, f_3\}$ as example again, an example of disjunction normal form equivalent to f_1 is the formula $(f_1 \wedge f_2 \wedge f_3) \vee (f_1 \wedge \neg f_2 \wedge f_3) \vee (f_1 \wedge f_2 \wedge \neg f_3) \vee (f_1 \wedge \neg f_2 \wedge \neg f_3)$, note that such a formula is equivalent to $f_1 \wedge ((f_2 \wedge f_3) \vee (\neg f_2 \wedge f_3) \vee (f_2 \wedge \neg f_3) \vee (\neg n_2 \wedge \neg f_3))$, where $(f_2 \wedge f_3) \vee (\neg f_2 \wedge f_3) \vee (f_2 \wedge \neg f_3) \vee (\neg f_2 \wedge \neg f_3)$ is equivalent to TRUE.

To formalise the idea above, we want to be able to building disjunction normal form piece by piece. We use the following definitions as our tool:

$$\mathsf{negf}\ (f, \mathsf{T})\ \stackrel{\mathsf{def}}{=}\ f$$
$$\mathsf{negf}\ (f, \mathsf{F})\ \stackrel{\mathsf{def}}{=}\ \neg f \mathsf{lit\_list\_to\_form}\ []\ \stackrel{\mathsf{def}}{=}\ \mathsf{TRUE}$$
$$\mathsf{lit\_list\_to\_form}\ [fb]\ \stackrel{\mathsf{def}}{=}\ \mathsf{negf}\ fb$$
$$\mathsf{lit\_list\_to\_form}\ (fb :: v_2 :: v_3)\ \stackrel{\mathsf{def}}{=}$$
$$\mathsf{AND}\ (\mathsf{negf}\ fb)\ (\mathsf{lit\_list\_to\_form}\ (v_2 :: v_3))\mathsf{lit\_list\_to\_form2}\ []\ \stackrel{\mathsf{def}}{=}\ \bot$$
$$\mathsf{lit\_list\_to\_form2}\ [fb]\ \stackrel{\mathsf{def}}{=}\ fb$$
$$\mathsf{lit\_list\_to\_form2}\ (fb :: v_2 :: v_3)\ \stackrel{\mathsf{def}}{=}$$
$$\mathsf{DISJ}\ fb\ (\mathsf{lit\_list\_to\_form2}\ (v_2 :: v_3))$$

A pair (f, tv) where $f \in fs$ and tv is a truth value is called a literal, it encodes a formula in fs or its negation. Such a pair is turned to a formula f or $\neg f$ by negf, depends on the truth value given in its second coordinate. For a finite fs, we are interested in non-empty lists of literals with the condition that set (MAP FST l) $=$ fs and ALL_DISTINCT (MAP FST l), where MAP takes a function f and a list $[a_1; \cdots, a_n]$, and return the list $[f(a_1); \cdots; f(a_n)]$. The function set takes a list and gives the set of its members, and ALL_DISTINCT takes a list and return the truth value of whether all the member of the list. These conditions precisely say that each member of fs occurs exactly once in the list. As a consequence, we can build CONJ_OF formulas from such list using `lit_list_to_form`. Conversely, each CONJ_OF formula can be obtained from such a list.

$$
\begin{array}{l}
\vdash l \neq [] \Rightarrow \\
\quad \mathsf{set}\ (\mathsf{MAP\ FST}\ l) = fs \wedge \mathsf{ALL\_DISTINCT}\ (\mathsf{MAP\ FST}\ l) \Rightarrow \\
\quad \mathsf{CONJ\_OF}\ (\mathsf{lit\_list\_to\_form}\ l)\ fs \vdash \mathsf{CONJ\_OF}\ f\ fs \Rightarrow \\
\quad \exists l. \\
\qquad \mathsf{set}\ (\mathsf{MAP\ FST}\ l) = fs \wedge \mathsf{ALL\_DISTINCT}\ (\mathsf{MAP\ FST}\ l) \wedge \\
\qquad f = \mathsf{lit\_list\_to\_form}\ l
\end{array}
$$

Under the same condition on the list, the corresponding result holds for `lit_list_to_form` and DISJ_OF0:

$$
\begin{array}{l}
\vdash l \neq [] \wedge \mathsf{set}\ l \subseteq fs \wedge \mathsf{ALL\_DISTINCT}\ l \Rightarrow \\
\quad \mathsf{DISJ\_OF0}\ (\mathsf{lit\_list\_to\_form2}\ l)\ fs \vdash \mathsf{DISJ\_OF0}\ f\ fs \Rightarrow \\
\quad \exists l. \\
\qquad l \neq [] \wedge \mathsf{set}\ l \subseteq fs \wedge f = \mathsf{lit\_list\_to\_form2}\ l \wedge \\
\qquad \mathsf{ALL\_DISTINCT}\ l
\end{array}
$$

Put the two things together, we can build DNF_OF formulas using `lit_list_to_form` and `list_list_to_form2`

$$
\begin{array}{l}
\vdash ld \neq [] \wedge \mathsf{ALL\_DISTINCT}\ ld \wedge \\
\quad (\forall d. \\
\qquad \mathsf{MEM}\ d\ ld \Rightarrow \\
\qquad \exists lc. \\
\qquad\quad lc \neq [] \wedge d = \mathsf{lit\_list\_to\_form}\ lc \wedge \\
\qquad\quad \mathsf{set}\ (\mathsf{MAP\ FST}\ lc) = fs \wedge \mathsf{ALL\_DISTINCT}\ (\mathsf{MAP\ FST}\ lc)) \Rightarrow \\
\quad \mathsf{DNF\_OF}\ (\mathsf{lit\_list\_to\_form2}\ ld)\ fs
\end{array}
$$

By induction on list, we can prove the distributivity and symmetry of `lit_list_to_form2`:

$$
\begin{array}{l}
\vdash l \neq [] \Rightarrow \\
\quad \mathsf{equiv0}\ \mu\ (\mathsf{AND}\ e\ (\mathsf{lit\_list\_to\_form2}\ l)) \\
\quad\ (\mathsf{lit\_list\_to\_form2}\ (\mathsf{MAP}\ (\lambda a.\ \mathsf{AND}\ e\ a)\ l)) \vdash \mathsf{set}\ l_1 = \mathsf{set}\ l_2 \Rightarrow \\
\quad \mathsf{equiv0}\ \mu\ (\mathsf{lit\_list\_to\_form2}\ l_1)\ (\mathsf{lit\_list\_to\_form2}\ l_2)
\end{array}
$$

Use the lemmas above, by induction on finiteness, we prove the disjunction of all possible conjunctions is equivalent to the truth:

$$
\begin{array}{l}
\vdash \mathsf{FINITE}\ fs \Rightarrow \\
\quad fs \neq \emptyset \Rightarrow \\
\quad \mathsf{equiv0}\ \mu \\
\quad\ (\mathsf{lit\_list\_to\_form2}\ (\mathsf{SET\_TO\_LIST}\ \{\ c \mid \mathsf{CONJ\_OF}\ c\ fs\ \}\ )) \\
\quad\ \mathsf{TRUE}
\end{array}
$$

Now we launch on the proof of `IBC_DNF_EXISTS_case4`:

Given an $f \in fs$, the desired p is given by $\phi :=$lit_list_to_form(SET_TO_LIST $\{f \wedge c \mid c \mid CONJ\_OF\ c\ (fs/\{f\})\}$. Here SET_TO_LIST is a function takes a set and form a list with all the elements of the set as members. There are two things to check: Using $CONJ\_OF\_AND\_lemma$ and `list_to_DNF_lemma`, it is immediate to prove ϕ is in disjunction normal form. To prove ϕ is equivalent to f, we have:

lit_list_to_form(SET_TO_LIST $\{f \wedge c \mid c \mid CONJ\_OF\ c\ (fs/\{f\})\}$

$=$lit_list_to_form(MAP $(\lambda\, a.$ AND $f\ a)$(SET_TO_LIST $\{c \mid CONJ\_OF\ c\ (fs/\{f\})\}$))

$\overset{\text{list\_demorgan}}{\equiv} f \wedge$ lit_list_to_form(SET_TO_LIST$\{c \mid CONJ_OF\ c\ (fs/\{f\})\}$)

$\overset{\text{ALL\_POSSIBLE\_VALUE\_TRUE}}{\equiv} f \wedge \top$

$\equiv f$

Hence we are done with the base case of `IBC_DNF_EXISTS`.

3.3.4 Case for disjunction

The following lemma directly implies our goal by the fact that if $f1$ is equivalent $p1$ and $f2$ is equivalent to $p2$, then $f1 \vee f2$ is equivalent to $p1 \vee p2$ and the transitivity of being equivalent.

$$\begin{aligned} &\vdash \mathsf{DNF\_OF}\ p_1\ fs\ \wedge\ \mathsf{DNF\_OF}\ p_2\ fs\ \Rightarrow \\ &\quad \exists f.\ \mathsf{DNF\_OF}\ f\ fs\ \wedge\ \mathsf{equiv0}\ \mu\ f\ (\mathsf{DISJ}\ p_1\ p_2) \end{aligned}$$

Expanding the definition of DNF_OF and DISJ_OF0 gives four cases, the only interesting one among them is stated as this lemma:

$$\begin{aligned} &\vdash \mathsf{DISJ\_OF0}\ p_1\ fs\ \Rightarrow \\ &\quad \forall p_2. \\ &\qquad \mathsf{DISJ\_OF0}\ p_2\ fs\ \Rightarrow \\ &\qquad \exists f.\ \mathsf{DISJ\_OF0}\ f\ fs\ \wedge\ \mathsf{equiv0}\ \mu\ f\ (\mathsf{DISJ}\ p_1\ p_2) \end{aligned}$$

Proof: By induction on DISJ_OF0, the base case is by a straightfoward two-layer induction, for the step case, suppose $f_1 \in fs$ and DISJ_OF0 p_1 (fs DELETE f_1) and DISJ_OF0 $p_2\ fs$, we are asked to prove:

$\exists f.$

DISJ_OF0 $f\ fs\ \wedge$ equiv0 $\mu\ f$ (DISJ (DISJ $f_1\ p_1$) p_2) from the inductive hypothesis

$\forall p_2.$

DISJ_OF0 p_2 (fs DELETE f_1) $\Rightarrow$

$\exists f.$ DISJ_OF0 f (fs DELETE f_1) $\wedge$ equiv0 $\mu\ f$ (DISJ $p_1\ p_2$)

If DISJ_OF0 p_2 (fs DELETE f_1), then we are done by inductive hypothesis, otherwise we need a trick: If DISJ_OF0 $p_2\ fs$ but $\neg$DISJ_OF0 p_2 (fs DELETE f_1), it must be the case that f_1 appears in p_2, hence we can extract f_1 out of p_2 to split p_2 as a disjunction $t \vee f_1$, using the following lemma proved by induction on DISJ_OF0:

$$\vdash \mathsf{DISJ\_OF0}\ f\ fs\ \Rightarrow$$
$$\forall\, t.$$
$$t\ \in\ fs\ \wedge\ \neg\mathsf{DISJ\_OF0}\ f\ (fs\ \mathsf{DELETE}\ t)\ \Rightarrow$$
$$\exists\, p.\ \mathsf{DISJ\_OF}\ p\ (fs\ \mathsf{DELETE}\ t)\ \wedge\ \mathsf{equiv0}\ \mu\ f\ (\mathsf{DISJ}\ t\ p)$$

Applying the lemma above allows us to finish the proof by using the inductive hypothesis.

3.3.5 Case for negation

Let us consider our last case, which amounts to prove:

$$\vdash \mathsf{DNF\_OF}\ p\ fs\ \wedge\ \mathsf{FINITE}\ fs\ \wedge\ fs\ \neq\ \emptyset\ \Rightarrow$$
$$\exists f.\ \mathsf{DNF\_OF}\ f\ fs\ \wedge\ \mathsf{equiv0}\ \mu\ (\neg p)\ f$$

The idea of this proof is to use the concept 'complement'. Consider a disjunction normal form p on a finite set fs, we find a formula of disjunction normal form f which is its negation, that is, we require f to be satisfied if and only if $\neg p$ is satisfied. As p is a disjunction, $\neg p$ is satisfied once none of p's conjuncts is satisfied. As a disjunction normal form, all of p's disjuncts are taken from the set of all of the CONJ_OF formulas on fs, and these disjuncts form a subset of $ss \subseteq \{c \mid CONJ\_OF\ c\ fs\}$. Since none of these conjunctions is satisfied, it must be the case that the satisfied conjunction is in $\{c \mid CONJ\_OF\ c\ fs\}/ss$. Again, let $fs = \{f_1, f_2\}$ and $p := (f_1 \wedge f_2) \vee (\neg f_1 \wedge f_2)$ is in disjunction normal form. For $\neg p$ is satisfied, neither $f_1 \wedge f_2$ nor $f_1 \wedge \neg f_2$ is satisfied, so it must be the case that either $\neg f_1 \wedge \neg f_2$ or $\neg f_1 \wedge f_2$. We take the conjunction of the elements in the set $\{\neg f_1 \wedge \neg f_2, \neg f_1 \wedge f_2\}$ formed by taking out the disjuncts appear in p from the total set $\{f_1 \wedge f_2, (\neg f_1 \wedge f_2, \neg f_1 \wedge \neg f_2, \neg f_1 \wedge f_2\}$, it gives the formula $(\neg f_1 \wedge \neg f_2) \vee (\neg f_1 \wedge f_2)$, which is equivalent to the negation of p.

Although it may be possible to stick to using our former tools `lit_list_to_form` and `list_list_to_form2`, it is much more natural to directly use the completement of set to deal with our issue here. Again, for a finite set fs, we will use literals to encode formulas in fs or its negation. But instead of using list of literals, we will use sets of literals this time.

We define satisfication of a literal as:

$$\mathsf{lsatis}\ \mathfrak{M}\ w\ (f, b)\ \stackrel{\text{def}}{=}\ (\mathsf{satis}\ \mathfrak{M}\ w\ f\ \iff\ b)$$

That is, for a model $\mathfrak{M}$ and a world w, a literal (f, T) is satisfied at w if satis $\mathfrak{M}$ w f, and (f, F) is satisfied at w if satis $\mathfrak{M}$ w $(\neg f)$. We want to construct CONJ_OF formulas from well-formed sets of literals, such a well-formed set is called an 'lset':

$$\mathsf{is\_lset}\ c\ fs\ \stackrel{\text{def}}{=}$$
$$\mathsf{FINITE}\ fs\ \wedge\ \mathsf{FINITE}\ c\ \wedge\ \mathsf{CARD}\ c\ =\ \mathsf{CARD}\ fs\ \wedge\ \mathsf{IMAGE\ FST}\ c\ =\ fs$$

Note the condition of being an `lset` corresponds the condition on the list as in the story about the base case. An `lset` corresponds a conjunction, for instance, $ls := \{(f_1, T), (f_2, T)\}$ corresponds to the formula $f_1 \wedge f_2$. We care about when all the literals in the set are satisfied, and call a set c-satisfied for this situation.

$$\mathsf{csatis}\ \mathfrak{M}\ w\ c \stackrel{\mathsf{def}}{=} \forall l.\ l\ \in\ c\ \Rightarrow\ \mathsf{lsatis}\ \mathfrak{M}\ w\ l$$

If two `lset`s are different, the union of them must contains a literal of opposite sign, hence the union of two distinct `lsets` can never be c-satisfied:

$$\begin{array}{l}\vdash \mathsf{is\_lset}\ c_1\ fs\ \wedge\ \mathsf{is\_lset}\ c_2\ fs\ \wedge\ c_1\ \neq\ c_2\ \Rightarrow\\ \quad \forall \mathfrak{M}\ w.\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\ \neg\mathsf{csatis}\ \mathfrak{M}\ w\ (c_1\ \cup\ c_2)\end{array}$$

By induction on CONJ_OF and the finiteness of `lset` respectively, we prove for any `lset`, it is c-satisfied at a world in a model iff its corresponding CONJ_OF formula is satisfied at the same point. Conversely, for any CONJ_OF formula, there is an `lset` that corresponds to it:

$$\begin{array}{l}\vdash \mathsf{CONJ\_OF}\ c\ fs\ \Rightarrow\\ \quad \exists ls.\\ \quad\quad \mathsf{is\_lset}\ ls\ fs\ \wedge\\ \quad\quad \forall \mathfrak{M}\ w.\\ \quad\quad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ \quad\quad\quad (\mathsf{csatis}\ \mathfrak{M}\ w\ ls\ \Longleftrightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ c)\ \vdash \mathsf{is\_lset}\ c\ fs\ \wedge\ c\ \neq\ \emptyset\ \Rightarrow\\ \quad \exists f.\\ \quad\quad (\forall \mathfrak{M}\ w.\\ \quad\quad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ \quad\quad\quad (\mathsf{satis}\ \mathfrak{M}\ w\ f\ \Longleftrightarrow\ \mathsf{csatis}\ \mathfrak{M}\ w\ c))\ \wedge\ \mathsf{CONJ\_OF}\ f\ fs\end{array}$$

A disjunction of CONJ_OF formula is captured by sets of literal sets. For instance, the set $\{\{(f_1,T),(f_2,T)\},\{(f_1,F),(f_2,T)$ encodes the formula $(f_1 \wedge f_2) \vee (\neg f_1 \wedge f_2)$ of disjunction normal form on the set $\{f_1, f_2\}$. And the satisfication of DISJ_OF0 formulas is captured by d-satisfication:

$$\mathsf{dsatis}\ \mathfrak{M}\ w\ cs \stackrel{\mathsf{def}}{=} \exists c.\ c\ \in\ cs\ \wedge\ \mathsf{csatis}\ \mathfrak{M}\ w\ c$$

There is a set-version of the the previous proposition `ALL_POSSIBLE_VALUE_TRUE`, looks like:

$$\begin{array}{l}\vdash \mathsf{FINITE}\ fs\ \Rightarrow\\ \quad fs\ \neq\ \emptyset\ \Rightarrow\\ \quad \forall \mathfrak{M}\ w.\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\ \mathsf{dsatis}\ \mathfrak{M}\ w\ \{\ c\ |\ \mathsf{is\_lset}\ c\ fs\ \}\end{array}$$

Using this as a lemma, we can prove the a critical statement about our idea of 'complement', saying that a set of `lsets` are d-satisfied iff its complement in the 'total set' of all the `lsets` is not d-satisfied.

$$\begin{array}{l}\vdash \mathsf{FINITE}\ fs\ \wedge\ fs\ \neq\ \emptyset\ \wedge\ (\forall c.\ c\ \in\ cs\ \Rightarrow\ \mathsf{is\_lset}\ c\ fs)\ \Rightarrow\\ \quad \forall \mathfrak{M}\ w.\\ \quad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ \quad\quad (\mathsf{dsatis}\ \mathfrak{M}\ w\ cs\ \Longleftrightarrow\\ \quad\quad\quad \neg\mathsf{dsatis}\ \mathfrak{M}\ w\ (\{\ c\ |\ \mathsf{is\_lset}\ c\ fs\ \}\ \mathsf{DIFF}\ cs))\end{array}$$

Proof: For the implication from left to right, suppose both dsatis $\mathfrak{M}\ w\ cs$ and dsatis $\mathfrak{M}\ w\{c \mid is\_lset\ c\ fs\}/cs$, then by definition of dsatis, it means two distinct `lset`s are c-satisfied at the same point, which contradicts `NEQ_lsets_FALSE`. For the implication from right to left, suppose $\neg$dsatis $\mathfrak{M}\ w\ cs$, by `dsatis_ALL_POSSIBLE_VALUE_TRUE`, the `lset` which holds on w must lies in $\{c \mid is\_lset\ c\ fs\}/cs$. This completes the proof.

And we have a correspondence of set of `lset`s and DISJ_OF0 formulas.

$$
\begin{array}{l}
\vdash \mathsf{DISJ\_OF0}\ d\ fs\ \Rightarrow \\
\quad \forall fs_0. \\
\qquad fs\ \subseteq\ \{\ c \mid \mathsf{CONJ\_OF}\ c\ fs_0\ \}\ \Rightarrow \\
\qquad \exists cs. \\
\qquad\quad (\forall c.\ c\ \in\ cs\ \Rightarrow\ \mathsf{is\_lset}\ c\ fs_0)\ \wedge \\
\qquad\quad \forall \mathfrak{M}\ w. \\
\qquad\qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\qquad\qquad (\mathsf{satis}\ \mathfrak{M}\ w\ d\ \Longleftrightarrow\ \exists c.\ c\ \in\ cs\ \wedge\ \mathsf{csatis}\ \mathfrak{M}\ w\ c)
\end{array}
$$

Altoghther, there are the correspondence results of set of `lset`s and DNF_OF formulas. The first one, which is a immediate consequence of `DISJ_OF0_cset`, says each DNF_OF formula corresponds a set of `lset`s.

$$
\begin{array}{l}
\vdash \mathsf{DNF\_OF}\ d\ fs\ \Rightarrow \\
\quad \exists cs. \\
\qquad (\forall c.\ c\ \in\ cs\ \Rightarrow\ \mathsf{is\_lset}\ c\ fs)\ \wedge \\
\qquad \forall \mathfrak{M}\ w. \\
\qquad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\qquad\quad (\mathsf{satis}\ \mathfrak{M}\ w\ d\ \Longleftrightarrow\ \exists c.\ c\ \in\ cs\ \wedge\ \mathsf{csatis}\ \mathfrak{M}\ w\ c)
\end{array}
$$

The second one says each set of `lset`s gives a DNF_OF formula:

$$
\begin{array}{l}
\vdash \mathsf{FINITE}\ s\ \Rightarrow \\
\quad \forall fs. \\
\qquad \mathsf{FINITE}\ fs\ \wedge\ fs\ \neq\ \emptyset\ \Rightarrow \\
\qquad (\forall c.\ c\ \in\ s\ \Rightarrow\ \mathsf{is\_lset}\ c\ fs)\ \Rightarrow \\
\qquad \exists f. \\
\qquad\quad (\forall \mathfrak{M}\ w. \\
\qquad\qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\qquad\qquad (\mathsf{satis}\ \mathfrak{M}\ w\ f\ \Longleftrightarrow\ \mathsf{dsatis}\ \mathfrak{M}\ w\ s))\ \wedge\ \mathsf{DNF\_OF}\ f\ fs
\end{array}
$$

Proof: The proof is by induction on FINITE s. For the base case that $s = \emptyset$, the required formula f is obviously $\bot$. Suppose for s a finite set of `lset`s and there exists a formula f in disjunction normal form such that satis $\mathfrak{M}\ w\ f\ \Longleftrightarrow$ dsatis $\mathfrak{M}\ w\ s$ for any model $\mathfrak{M}$ and any $w\ \in$ $\mathfrak{M}$.frame.world, we find a formula g such that satis $\mathfrak{M}\ w\ g\ \Longleftrightarrow$ dsatis $\mathfrak{M}\ w\ (e$ INSERT $s)$. By `is_lset_CONJ_OF_EXISTS`, there is a CONJ_OF formula f' on fs with the property that satis $\mathfrak{M}\ w\ f'\ \Longleftrightarrow$ csatis $\mathfrak{M}\ w\ e$ for arbitary $\mathfrak{M}$. If

f is equivalent to $\bot$, then f' is the g we want, and vice versa. If both of them are not equivalent to $\bot$, we can take $f' \vee f$ as the g we want. The first desired condition is easy to check, it remains to prove $f' \vee f$ is in disjunction normal form, which is suffices by proving DISJ_OF0 f $\{c \mid CONJ\_OF\ c\ fs\}/\{f'\}$. Suppose not, then there exists a p such that DISJ_OF0 p $\{c \mid CONJ\_OF\ c\ fs\}/\{f'\}$ and equiv0 $\mu\ f$ (DISJ f' p) by `DISJ_OF0_split`. To obtain a contradiction, we find a model with a world satisfies f' but not f. With the equivalence conditions on f and f', it amounts to find a modal $\mathfrak{M}$ with a world $w \in \mathfrak{M}$.frame.world such that csatis $\mathfrak{M}\ w\ e$ but $\neg$dsatis $\mathfrak{M}\ w\ s$. The claim is any world c-satisfying e cannot d-satisfy s, otherwise it contradicts `dsatis_is_lset_completment`. Also since f' is not equivalent to $\bot$, a model satisfying f' exists, so we are done.

Now we have all the ingredients to prove our last case.

Proof: As DNF_OF p fs, by `DNF_OF_cset` we obtain a `lset` cs such that satis $\mathfrak{M}\ w\ p \iff$ dsatis $\mathfrak{M}\ w\ cs$ for any $\mathfrak{M}$ and w. Consider its complement $\{c \mid is\_lset\ c\ fs\}/cs$, by `is_lset_DNF_OF_EXISTS` we obtain a formula f such that DNF_OF f fs and satis $\mathfrak{M}\ w\ f$ iff dsatis $\mathfrak{M}\ w\ \{c \mid is\_lset\ c\ fs\}/cs$ for any $\mathfrak{M}$ and w. By `dsatis_is_lset_complement`, dsatis $\mathfrak{M}\ w\ \{c \mid is\_lset\ c\ fs\}/cs$ iff $\neg$dsatis $\mathfrak{M}\ w\ cs$, so f is equivalent to $\neg p$.

Here we are done with the proof of `prop_2_29_strengthen`. This is the end of the interlude.

This is time to return to discussion about the proof of finite model property. Recall in the last chapter, we have seen that a bisimulation gives arise to modal equivalence, modal equivalence is 'satisfiying exactly the same formulas', but when we are building a finite model for a formula ϕ, we do not care about the satisfication of any formula of degree above DEG phi, since such formula cannot affect the satisfication of ϕ. Therefore, a finite approximation of bisimulation is enough, a finite approximation of depth n is called an n-bisimulation. Let $w \in \mathfrak{M}$.frame.world and $w' \in \mathfrak{M}'$.frame.world, w and w' are n-bisimilar if there exists a sequence of relations $Z_n \subseteq \cdots \subseteq Z_0$ such that:

- w and w' are related by Z_n
- If $v \in \mathfrak{M}$.frame.world and $v' \in M'frame$.world are related by Z_0, then v and v' satisfy the same propositional letters.
- If $v \in \mathfrak{M}$.frame.world and $v' \in M'frame$.world are related by Z_{i+1} and we have $\mathfrak{M}$.frame.rel $v\ u$ for $u \in \mathfrak{M}$.frame.world, then there exists $u' \in \mathfrak{M}'$.frame.world such that $\mathfrak{M}'$.frame.rel $v'\ u'$ with u and u' related by Z_i.
- If $v \in \mathfrak{M}$.frame.world and $v' \in M'frame$.world are related by Z_{i+1} and we have $\mathfrak{M}'$.frame.rel $v'\ u'$ for $u' \in \mathfrak{M}$.frame.world, then there exists $u \in \mathfrak{M}$.frame.world such that $\mathfrak{M}$.frame.rel $v\ u$ with u and u' related by Z_i.

Such a sequence of Z_i is a family of relations indexed by natural numbers. When the world set of $\mathfrak{M}$ has type β and the world set of $\mathfrak{M}'$ has type γ, we encode such a family using functions $f : num \to \beta \to \gamma \to bool$. Such a function assigns each natural number a relation, so the $f\ i$ is the relation Z_i in the usual mathematical definition, and nbisim $\mathfrak{M}\ \mathfrak{M}'\ f\ n\ w\ w'$ means w and w' are worlds in $\mathfrak{M}$ and $\mathfrak{M}'$ respectively which are n bisimilar via the family of relations given by f.

$$
\begin{aligned}
&\mathsf{nbisim}\ \mathfrak{M}\ \mathfrak{M}'\ f\ n\ w\ w' \stackrel{\mathsf{def}}{=} \\
&\quad w \in \mathfrak{M}.\mathsf{frame.world} \wedge w' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \\
&\quad (\forall m\ a\ b. \\
&\qquad a \in \mathfrak{M}.\mathsf{frame.world} \wedge b \in \mathfrak{M}'.\mathsf{frame.world} \Rightarrow \\
&\qquad m + 1 \leq n \Rightarrow \\
&\qquad f\ (m + 1)\ a\ b \Rightarrow \\
&\qquad f\ m\ a\ b) \wedge f\ n\ w\ w' \wedge \\
&\quad (\forall v\ v'. \\
&\qquad v \in \mathfrak{M}.\mathsf{frame.world} \wedge v' \in \mathfrak{M}'.\mathsf{frame.world} \Rightarrow \\
&\qquad f\ 0\ v\ v' \Rightarrow \\
&\qquad \forall p.\ \mathsf{satis}\ \mathfrak{M}\ v\ (\mathsf{VAR}\ p) \iff \mathsf{satis}\ \mathfrak{M}'\ v'\ (\mathsf{VAR}\ p)) \wedge \\
&\quad (\forall v\ v'\ u\ i. \\
&\qquad i + 1 \leq n \wedge v \in \mathfrak{M}.\mathsf{frame.world} \wedge v' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \\
&\qquad u \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ v\ u \wedge f\ (i + 1)\ v\ v' \Rightarrow \\
&\qquad \exists u'.\ u' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \mathfrak{M}'.\mathsf{frame.rel}\ v'\ u' \wedge f\ i\ u\ u') \wedge \\
&\quad \forall v\ v'\ u'\ i. \\
&\qquad i + 1 \leq n \wedge v \in \mathfrak{M}.\mathsf{frame.world} \wedge v' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \\
&\qquad u' \in \mathfrak{M}'.\mathsf{frame.world} \wedge \mathfrak{M}'.\mathsf{frame.rel}\ v'\ u' \wedge f\ (i + 1)\ v\ v' \Rightarrow \\
&\qquad \exists u.\ u \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ v\ u \wedge f\ i\ u\ u'
\end{aligned}
$$

By induction on n, we see if two worlds $w \in \mathfrak{M}.\mathsf{frame.world}$ and $w' \in \mathfrak{M}'.\mathsf{frame.world}$ are related by an n-bisimulation, then they agree on all modal formulas up to degree n:

$$
\begin{aligned}
&\vdash (\exists f.\ \mathsf{nbisim}\ \mathfrak{M}\ \mathfrak{M}'\ f\ n\ w\ w') \Rightarrow \\
&\qquad \forall phi.\ \mathsf{DEG}\ phi \leq n \Rightarrow (\mathsf{satis}\ \mathfrak{M}\ w\ phi \iff \mathsf{satis}\ \mathfrak{M}'\ w'\ phi)
\end{aligned}
$$

The converse of the above holds when our modal language is defined on a finite set of propositional letters and the type of world sets of our models are infinite. It can be proved by an argument analogue to the proof of Hennessy-Milner theorem, with $\lambda\, n\ n_1\ n_2$.
$\forall phi.\ \mathsf{DEG}\ phi \leq n \Rightarrow (\mathsf{satis}\ \mathfrak{M}\ n_1\ phi \iff \mathsf{satis}\ \mathfrak{M}'\ n_2\ phi)$ gives an n-bisimulation relation linking w and w':

$$
\begin{aligned}
&\vdash \mathsf{INFINITE}\ \mathcal{U}(:\beta) \wedge \mathsf{INFINITE}\ \mathcal{U}(:\gamma) \wedge \mathsf{FINITE}\ \mathcal{U}(:\alpha) \wedge \\
&\quad w \in \mathfrak{M}.\mathsf{frame.world} \wedge w' \in \mathfrak{M}'.\mathsf{frame.world} \Rightarrow \\
&\quad (\forall phi.\ \mathsf{DEG}\ phi \leq n \Rightarrow (\mathsf{satis}\ \mathfrak{M}\ w\ phi \iff \mathsf{satis}\ \mathfrak{M}'\ w'\ phi)) \Rightarrow \\
&\quad \exists f.\ \mathsf{nbisim}\ \mathfrak{M}\ \mathfrak{M}'\ f\ n\ w\ w'
\end{aligned}
$$

DEG is a concept that measure the depth of a formula, we also want a concept that measure the depth of a model, as 'depth' is a relative concept measuring the distance of two points. To talk about the depth of a world $w \in \mathfrak{M}.\mathsf{frame.world}$, we need $\mathfrak{M}$ to be naturally equipped with a base point, so the 'height' of a world only makes sense to rooted model. To tell the HOL about this definition, we start by defining heightLE:

$$
\begin{array}{l}
\vdash (\forall\, n.\ \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ x\ n)\ \wedge \\
\quad \forall\, n\ v. \\
\qquad v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge \\
\qquad (\exists\, w. \\
\qquad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathfrak{M}.\mathsf{frame.rel}\ w\ v\ \wedge \\
\qquad\quad \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ n)\ \Rightarrow \\
\qquad \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ v\ (n\ +\ 1)\ \vdash (\forall\, n.\ \mathit{heightLE}'\ x\ n)\ \wedge \\
\quad (\forall\, n\ v. \\
\qquad v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge \\
\qquad (\exists\, w. \\
\qquad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathfrak{M}.\mathsf{frame.rel}\ w\ v\ \wedge \\
\qquad\quad \mathit{heightLE}'\ w\ n)\ \Rightarrow \\
\qquad \mathit{heightLE}'\ v\ (n\ +\ 1))\ \Rightarrow \\
\quad \forall\, a_0\ a_1.\ \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ a_0\ a_1\ \Rightarrow\ \mathit{heightLE}'\ a_0\ a_1\ \vdash \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ a_0\ a_1\ \Longleftrightarrow \\
\quad a_0\ =\ x\ \vee \\
\qquad \exists\, n. \\
\qquad\quad a_1\ =\ n\ +\ 1\ \wedge\ a_0\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge \\
\qquad\quad \exists\, w. \\
\qquad\qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathfrak{M}.\mathsf{frame.rel}\ w\ a_0\ \wedge \\
\qquad\qquad \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ n
\end{array}
$$

Recall how we defined rooted model: $\mathsf{rooted\_model}\ \mathfrak{M}\ x\ \mathfrak{M}'$ means '$\mathfrak{M}$ is a rooted model generated by the world x in the ambient model $\mathfrak{M}'$'. As heightLE is designed to be only make sense for rooted models, we encode the information about the rootedness of the model we are talking about into this definition. $\mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ n$ reads 'for the rooted model $\mathfrak{M}$ generated by the root x in $\mathfrak{M}'$, the distance from the world w to the root x is less or equal to n. The height of such a world w is the smallest natural number n such that $\mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ n$. And a height of model is the maximum n such that there is a world of height n in the model.

$$
\begin{array}{l}
\mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ \stackrel{\mathsf{def}}{=}\ \mathsf{MIN\_SET}\ \{\ n\ |\ \mathsf{heightLE}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ n\ \}\ \mathsf{model\_height}\ \mathfrak{M}\ x\ \mathfrak{M}'\ \stackrel{\mathsf{def}}{=} \\
\quad \mathsf{MAX\_SET}\ \{\ n\ |\ (\exists\, w.\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ =\ n)\ \}
\end{array}
$$

Obviously, the root of any rooted model have height 0. We are particularly interested in talking about heights in tree-like model. When $\mathfrak{M}$ is tree-like, if $w\ \in\ \mathfrak{M}.\mathsf{frame.world}$ has height n, then any world $w'\ \in\ \mathfrak{M}.\mathsf{frame.world}$ such that $\mathfrak{M}.\mathsf{frame.rel}\ w\ w'$ will have height $n+1$, this is proved using the induction principle `RTC_INDUCT_RIGHT1`:

$$
\begin{array}{l}
\vdash \mathsf{tree}\ \mathfrak{M}.\mathsf{frame}\ x\ \Rightarrow \\
\quad \forall\, w. \\
\qquad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}\ w\ =\ n\ \Rightarrow \\
\qquad \forall\, v. \\
\qquad\quad \mathfrak{M}.\mathsf{frame.rel}\ w\ v\ \wedge\ v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\qquad\quad \mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}\ v\ =\ n\ +\ 1
\end{array}
$$

The restriction of a rooted model $\mathfrak{M}$ to the height k is the submodel consisting of all the worlds in $\mathfrak{M}$ of height up to k. The restriction of a tree-like model is again a tree-like model:

$$
\begin{array}{l}
\mathsf{hrestriction}\ \mathfrak{M}\ x\ \mathfrak{M}'\ n\ \stackrel{\mathsf{def}}{=} \\
\quad \mathsf{<|frame}\ :=\ \\
\quad\ \mathsf{<|world}\ :=\ \{\ w\ \mid\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w\ \leq\ n\ \}\,; \\
\quad\quad \mathsf{rel}\ :=\ (\lambda\, n_1\ n_2.\ \mathfrak{M}.\mathsf{frame.rel}\ n_1\ n_2)\,\mathsf{|>}; \\
\quad \mathsf{valt}\ :=\ (\lambda\, phi\ n.\ \mathfrak{M}.\mathsf{valt}\ phi\ n)\,\mathsf{|>}\ \vdash \mathsf{tree}\ \mathfrak{M}.\mathsf{frame}\ x\ \Rightarrow\ \forall\, n.\ \mathsf{tree}\ (\mathsf{hrestriction}\ \mathfrak{M}\ x\ \mathfrak{M}\ n).\mathsf{frame}\ x
\end{array}
$$

Restriction of rooted model gives arise of n-bisimulation: If we restrict a rooted model $\mathfrak{M}$ to height k, then a world w of height m in the restricted model is $k-m$-bisimilar to itself in the original model:

$$
\begin{array}{l}
\vdash \mathsf{rooted\_model}\ \mathfrak{M}\ x\ \mathfrak{M}'\ \Rightarrow \\
\quad \forall\, w. \\
\qquad w\ \in\ (\mathsf{hrestriction}\ \mathfrak{M}\ x\ \mathfrak{M}'\ k).\mathsf{frame.world}\ \Rightarrow \\
\qquad \exists\, f. \\
\qquad\quad \mathsf{nbisim}\ (\mathsf{hrestriction}\ \mathfrak{M}\ x\ \mathfrak{M}'\ k)\ \mathfrak{M}\ f \\
\qquad\quad\ (k\ -\ \mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w)\ w\ w
\end{array}
$$

Proof: The n-bisimilar relation is give as $\lambda\, n\ w_1\ w_2.\ w_1\ =\ w_2\ \wedge\ \mathsf{height}\ \mathfrak{M}\ x\ \mathfrak{M}'\ w_1\ \leq\ k\ -\ n$.

Now we can start building a finite model via selection:

$$
\begin{array}{l}
\vdash \mathsf{satis}\ \mathfrak{M}_1\ w_1\ phi\ \Rightarrow \\
\quad \exists\, FM\ v. \\
\qquad \mathsf{FINITE}\ FM.\mathsf{frame.world}\ \wedge\ v\ \in\ FM.\mathsf{frame.world}\ \wedge \\
\qquad \mathsf{satis}\ FM\ v\ phi
\end{array}
$$

Proof: Suppose $\mathsf{satis}\ \mathfrak{M}_1\ w_1\ phi$ where $w_1 : \beta, \phi : \alpha\ form$, then by `prop_2_15_corollary`, there exists a tree-like model $\mathfrak{M}_2$ with phi satisfied at its root w_2. Such an $\mathfrak{M}_2$ obtained from `prop_2_15_corollary` has its world set of type $\beta\ list$, so all the lemmas proved before with a infinite universe assumption applies for any model with its world set a subset of $\mathfrak{M}_2$.frame.world . Define M_3 :=$\mathsf{hrestriction}\ \mathfrak{M}_2\ w_2\ \mathfrak{M}_2\ k$ to be the restriction of $\mathfrak{M}_2$ to height k, then M_3 is rooted and there is a $\mathsf{nbisim}\ M_3\ \mathfrak{M}_2\ f\ k\ w_2\ w_2$ by `lemma_2_33`. Hence $\mathsf{satis}\ M_3\ w_2\ phi$ by `prop_2_31_half1`. By `exercise_1_3_1` proved in the first chapter, if a propositional letter does not appear in ϕ, then it has no effect to the satisfication of ϕ, so we can discard all the propositional letters in M_3 which does not occur in ϕ to obtain the model $M_3'\ =$

<|frame := <|world := M_3.frame.world; rel := M_3.frame.rel|>;
valt :=
($\lambda\, p\ v.$ `if` VAR p $\in$ subforms phi `then` M_3.valt p v `else` F)|>, and still have satis M_3' w_2 phi. We select a finite model inductively from M_3'.

Let s denote the set of propositional letters used by ϕ, so s is finite. By `prop_2_29_strengthen`, there are only finitely many non-equivalent formulas of degree less or equal to k which only use propositional letters in s, that is, the set $distfp := \{f \mid DEG\ f \leq k \wedge \forall a. VAR\ a \in subforms\ f \implies a \in s\}//E\ \mu$ is a finite. We care about the equivalence classes in $distfp$ which are equivalence classes of formulas starting with a $\Diamond$. For such a equivalence class, taking the intersection with the set $\{d \mid ?d0.d = \Diamond d0\}$ does not give the empty set. Take the image of $distfp$ under the function $\lambda s.s \cap \{d \mid ?d0.d = \Diamond d0\}$ and delete the empty set from the image, we obtain a set fs of sets of formulas, where for each $x \in fs$, x consists of equivalent formulas of degree less or equal to k, only use propositional letters in s, and starts with diamond. Hence $rs := (IMAGE\ CHOICE((IMAGE\ (\lambda s.s \cap \{d \mid ?d0.d = \Diamond d0\})\ distfp)\ DELETE\ \{\})$ can be taken as the set of representatives of formulas with desired properties, it is finite as an image of a finite set.

We will construct sets $S_0, \cdots S_k$ of worlds in M_3', where the points in S_n have height n. Start with $S_0 := \{w_2\}$, and inductively, assume $S_0, \cdots S_n$ has been defined, construct S_{n+1} as follows: Consider an element in $v \in S_n$, for each $\Diamond\phi \in rs$ such that satis M_3' w_2 ($\Diamond$ phi), pick a world $u \in M_3'$.frame.world such that M_3'.frame.rel v u and satis M_3' u phi. Do the same thing to all the $v \in S_n$, then S_{n+1} is the set of all the such u's which are seleted in this way.

The we we taken to formalise the definition of such S_i's is to define a primitive recursive function: $ss := PRIM_REC\ \{w2\}(\lambda s0 : \beta\ list\ set\ n.\{CHOICE\ uset \mid ?phi\ v.satis\ M3'\ v\ (\Diamond phi) \wedge ((\Diamond phi) \in (IMAGE\ CHOICE\ ((IMAGE\ (\lambda s.s \cap \{d \mid ?d0.d = \Diamond d0\})distfp)DELETE\emptyset)) \wedge v \in s0 \wedge uset = \{u \mid M3'.frame.rel\ v\ u \wedge u \in M3'.frame.world \wedge satis\ M3'\ u\ phi\})\})$

For each $i \leq k$, $ss\ i$ will be our S_i. By induction on i, we can prove each $ss\ i$ is finite, so the set $W4 := \bigcup_i \{ss\ i \mid i \leq k\}$ is finite. The resultant finite model we select is: M_4 =
<|frame := <|world := W_4; rel := M_3.frame.rel|>;
valt := M_3'.valt|> To prove satis M_4 w_2 phi, it suffices to give a k-bisimulation between M_4 and M_3' relating $w2$ to itself. Such a k-bisimulation is: $\lambda\, n\ a_1\ a_2$.
$a_1 \in M_4$.frame.world $\wedge$ $a_2 \in M_3'$.frame.world $\wedge$
height M_3' w_2 M_3' a_1 = height M_3' w_2 M_3' a_2 $\wedge$
height M_3' w_2 M_3' $a_1 \leq k - n \wedge$
$\forall\, phi.$
DEG $phi \leq n \wedge (\forall\, a.$ VAR $a \in$ subforms $phi \Rightarrow a \in s) \Rightarrow$
(satis M_3' a_1 phi $\iff$ satis M_3' a_2 phi) The rest of the proof amounts to check the above indeed gives a k-bisimulation, the proof is not hard using a similar argument as we proved the Hennessy-Milner theorem.

As we took a detour through `prop_2_15_corollary`, this construction of finite model changes the type of model.

4 Reaching out to the world of first order logic

Modal logic is not an isolated formal system, in this chapter, we start linking modal logic with the wider logical world by discussing about the relation to first order logic. In the first half of the chapter, we define standard translation as our link between modal logic and first order logic, and in the second half of the chapter, with the help of standard translation, we introduce another construction on models which will give modal equivalence, and lead to an elegant result about bisimulation: modal equivalence implies bisimilarity-somewhere-else.

4.1 Standard Translation

To discuss the relationship between modal logic and first order logic, firstly we need to build some basics of first order logic in the HOL. First order logic is formalised in HOL light in 1998 as in (reference), we take our construction of first-order model theory as in the paper.

For a first order language, a term is either a variable letter x or a function symbol f applied on a list of terms, which looks like $f(t_1, \cdots, t_n)$, where the t's can either be variable letters or itself a function applied on some terms. To avoid specifying the type every where, we restrict our scope to countable language, by using only natural numbers to denote variable and function symbols:

$$\texttt{term} \;=\; \mathsf{fV}\ \texttt{num} \mid \mathsf{Fn}\ \texttt{num}\ (\texttt{term list})$$

Hence our terms will look like fV 6, Fn 1 [fV 1; fV 2], Fn 2 [Fn 0 []], etc. A constant is just a nullary function symbol.

Our formulas are defined inductively as well, using minimal amount of logical connectives, by choosing the falsity, atoms, implication and universal quantification as primitive, predicate symbols are also represented by natural numbers. In particular, an n-ary relation symbol is a predicate symbol that takes lists of length n:

$$\phi \;=\; \mathsf{fFALSE} \mid \mathsf{Pred}\ \texttt{num}\ (\texttt{term list}) \mid \mathsf{IMP}\ \phi\ \phi \mid \mathsf{FALL}\ \texttt{num}\ \phi$$

A quantified variable is called a bounded variable, otherwise it is called free. We define a function that returns the set of free variables of the formula, starting by collecting variables occur in the terms of formulas, and then delete the bounded ones:

$$\begin{aligned}
&\mathsf{FVT}\ (\mathsf{fV}\ v) \stackrel{\text{def}}{=} \{\ v\ \}\\
&\mathsf{FVT}\ (\mathsf{Fn}\ s\ ts) \stackrel{\text{def}}{=} \mathsf{LIST_UNION}\ (\mathsf{MAP}\ (\lambda\, a.\ \mathsf{FVT}\ a)\ ts)\mathsf{FV}\ \mathsf{fFALSE} \stackrel{\text{def}}{=} \emptyset\\
&\mathsf{FV}\ (\mathsf{Pred}\ v_0\ ts) \stackrel{\text{def}}{=} \mathsf{LIST_UNION}\ (\mathsf{MAP}\ \mathsf{FVT}\ ts)\\
&\mathsf{FV}\ (f_1\ \rightarrow\ f_2) \stackrel{\text{def}}{=} \mathsf{FV}\ f_1\ \cup\ \mathsf{FV}\ f_2\\
&\mathsf{FV}\ (\mathsf{FALL}\ x\ f) \stackrel{\text{def}}{=} \mathsf{FV}\ f\ \mathsf{DELETE}\ x
\end{aligned}$$

Here LIST_UNION is a function that takes a list of sets, and give us the union of all the sets in the list:

$$\mathsf{LIST\_UNION}\ [] \stackrel{\text{def}}{=} \emptyset$$
$$\mathsf{LIST\_UNION}\ (h :: t) \stackrel{\text{def}}{=} h \cup \mathsf{LIST\_UNION}\ t$$

Similarly, we have functions called `form_functions` and `form_predicates`, that take a formula and give the set of functions and predicates appear in the formula respectively.

The non-primitive connectives are defined in the canonical way:

$$\mathsf{fNOT}\ f \stackrel{\text{def}}{=} f \to \mathsf{fFALSE} \quad \mathsf{True} \stackrel{\text{def}}{=} \mathsf{fNOT}\ \mathsf{fFALSE} \quad \mathsf{fDISJ}\ p\ q \stackrel{\text{def}}{=} (p \to q) \to q \quad \mathsf{fAND}\ p\ q \stackrel{\text{def}}{=} \mathsf{fNOT}\ (\mathsf{fDISJ}\ (\mathsf{fNOT}\ p)\ (\mathsf{fNOT}\ q)) \quad \mathsf{fEXIS}$$

To interpret these formulas, we need models for first order logic. A first order model of type α is a triple consists of an α-set which is its domain, a interpretation of function symbols, and a interpretation of predicate symbols.

$$\begin{aligned}&\alpha\ \texttt{model} = \texttt{<|}\\ &\quad \mathsf{Dom} : \alpha \to \texttt{bool};\\ &\quad \mathsf{Fun} : \texttt{num} \to \alpha\ \texttt{list} \to \alpha;\\ &\quad \mathsf{Pred} : \texttt{num} \to \alpha\ \texttt{list} \to \texttt{bool}\\ &\texttt{|>}\end{aligned}$$

Given a first order model $\mathfrak{M}$, we can interpret formulas or terms by assigning each variable symbol an element in $\mathfrak{M}$.Dom. Such an assignment is given by a valuation, which is a function from the universe of natural numbers to the domain of $\mathfrak{M}$:

$$\mathsf{valuation}\ \mathfrak{M}\ v \stackrel{\text{def}}{=} \forall n.\ v\ n \in \mathfrak{M}.\mathsf{Dom}$$

Interpretation of terms and formulas are given as termval and `feval`, where termval takes a model, an assignment of variable letters and a term, gives us an element of $\mathfrak{M}$.Dom. And `feval` takes a model, an assignment of variable letters and a first order formula, gives us the truth value whether the formula we give holds on the model under the current assignment of variable letters, we only care about when the an assignment of variable letters is indeed a valuation as defined above, when σ is a valuation and $\mathsf{feval}\ \mathfrak{M}\ \sigma\ \mathit{fform}$, we say '$\mathit{fform}$ is satisfied in $\mathfrak{M}$ under the valuation σ'.

$$\mathsf{termval}\ \mathfrak{M}\ v\ (\mathsf{fV}\ x) \stackrel{\text{def}}{=} v\ x$$
$$\mathsf{termval}\ \mathfrak{M}\ v\ (\mathsf{Fn}\ f\ l) \stackrel{\text{def}}{=} \mathfrak{M}.\mathsf{Fun}\ f\ (\mathsf{MAP}\ (\lambda a.\ \mathsf{termval}\ \mathfrak{M}\ v\ a)\ l) \quad \mathsf{fsatis}\ \mathfrak{M}\ \sigma\ \mathit{fform} \stackrel{\text{def}}{=} \mathsf{valuation}\ \mathfrak{M}\ \sigma \wedge \mathsf{feval}\ \mathfrak{M}\ \sigma\ \mathit{fform}$$

(Why the holds def not work?!)

By induction on first order formula, we show that the truth value of a first order formula only depends on where the valuation sends the free variables to:

$$\begin{aligned}\vdash (\forall x.\ x \in \mathsf{FV}\ p \Rightarrow v_1\ x = v_2\ x) \Rightarrow\\ (\mathsf{feval}\ \mathfrak{M}\ v_1\ p \iff \mathsf{feval}\ \mathfrak{M}\ v_2\ p)\end{aligned}$$

Models for modal language can be viewed as a first order model and hence be used to interpret some first order formula, and a first order model can be also viewed as a modal model. These conversions can be done using the following two functions:

$$
\begin{array}{l}
\mathsf{mm2folm}\ \mathfrak{M} \stackrel{\mathsf{def}}{=} \\
\quad \langle|\mathsf{Dom} := \mathfrak{M}.\mathsf{frame.world}; \\
\quad \mathsf{Fun} := (\lambda\, n\ args.\ \mathsf{CHOICE}\ \mathfrak{M}.\mathsf{frame.world}); \\
\quad \mathsf{Pred} := \\
\quad\ (\lambda\, p\ zs. \\
\qquad \mathtt{case}\ zs\ \mathtt{of} \\
\qquad\ [] \Rightarrow \mathsf{F} \\
\qquad | \ [w] \Rightarrow w \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{valt}\ p\ w \\
\qquad | \ [w;\ w_2] \Rightarrow \\
\qquad\ p = 0 \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ w_2 \wedge w \in \mathfrak{M}.\mathsf{frame.world} \wedge \\
\qquad\ w_2 \in \mathfrak{M}.\mathsf{frame.world} \\
\qquad | \ w :: w_2 :: v_{10} :: v_{11} \Rightarrow \mathsf{F})|\rangle\mathsf{folm2mm}\ FM \stackrel{\mathsf{def}}{=} \\
\quad \langle|\mathsf{frame} := \\
\quad\ \langle|\mathsf{world} := FM.\mathsf{Dom}; \\
\quad\ \mathsf{rel} := \\
\qquad (\lambda\, w_1\ w_2.\ FM.\mathsf{Pred}\ 0\ [w_1;\ w_2] \wedge w_1 \in FM.\mathsf{Dom} \wedge w_2 \in FM.\mathsf{Dom})|\rangle; \\
\quad \mathsf{valt} := (\lambda\, v\ w.\ FM.\mathsf{Pred}\ v\ [w] \wedge w \in FM.\mathsf{Dom})|\rangle
\end{array}
$$

Note that the above constructions are not inverses in general, since by viewing a first order model as a modal model, we lose all the function symbols and predicate symbols except for the unary ones, and the binary one denoted by 0. Also the range of formulas which makes sense to first order model obtained by converting a modal model is quite limited, we can only use these model to interpret formulas with only one binary predicate symbol corresponds to the relation in the model, denoted by 0, and no function symbol. A first order model which do not have 'superfluous' symbols contains the same amount of information as a modal model. For such a first order model, converting it to a modal model and then to a first order model again get the original model back, in the sense of the resultant model satisfied exactly the same first order formulas without function symbols. The fact that we that we need the assumption $\mathsf{form\_functions}\ f = \emptyset$ is not a real constrain, but merely an assumption for well-formedness, since any formulas with function symbol does not make sense to both the original model and the resultant model we get by conversion:

$$
\begin{array}{l}
\vdash \mathsf{form\_functions}\ f\ =\ \emptyset\ \Rightarrow \\
\quad \forall\, \sigma. \\
\qquad \mathsf{IMAGE}\ \sigma\ \mathcal{U}(:\mathtt{num})\ \subseteq\ \mathfrak{M}.\mathsf{Dom}\ \Rightarrow \\
\qquad (\forall\, n.\ \mathfrak{M}.\mathsf{Pred}\ n\ []\ \Longleftrightarrow\ \mathsf{F})\ \Rightarrow \\
\qquad (\forall\, a\ b\ n.\ \mathfrak{M}.\mathsf{Pred}\ n\ [a;\ b]\ \Rightarrow\ n\ =\ 0)\ \Rightarrow \\
\qquad (\forall\, a\ b\ c\ d\ n.\ \mathfrak{M}.\mathsf{Pred}\ n\ (a :: b :: c :: d)\ \Longleftrightarrow\ \mathsf{F})\ \Rightarrow \\
\qquad (\mathsf{feval}\ (\mathsf{mm2folm}\ (\mathsf{folm2mm}\ \mathfrak{M}))\ \sigma\ f\ \Longleftrightarrow\ \mathsf{feval}\ \mathfrak{M}\ \sigma\ f)
\end{array}
$$

Let us build intuition about how does modal formulas corresponds to first order formulas. Observe that unlike modal formulas which atomic formulas are propositional letters standing alone, even the atomic first order formula have a variable symbols involved. Hence to translate a modal formula into a first order formula, we must get variables involved as well. This variables is used to mark the state we are looking at. And moreover, once we see a $\Diamond$ in a modal formula, we start talking about some other state which is related to the current state, so we will need another variable symbol to mark the new state of interest. Hence in mathematical language, the natural correspondence of modal and first order formulas is given by:

$ST_x(p) = Px$

$ST_x(\bot) = \bot$

$ST_x(\neg\phi) = \neg ST_x(\phi)$

$ST_x(\phi \vee \psi) = ST_x(\phi) \vee ST_x(\psi)$

$ST_x(\Diamond\phi) = \exists y(Rxy \wedge ST_y(\phi))$

We can read $ST_x\phi$ as 'the standard translation of ϕ at x'. In the following definition, the x, which is a variable symbol in our construction, is a natural number. Each time we see a $\Diamond$, we come with a fresh variable denotes by $x + 1$:

$$
\begin{array}{l}
\vdash (\forall\, x\ p.\ \mathsf{ST}\ x\ (\mathsf{VAR}\ p)\ =\ \mathsf{fP}\ p\ (\mathsf{fV}\ x))\ \wedge \\
\quad (\forall\, x.\ \mathsf{ST}\ x\ \bot\ =\ \mathsf{fFALSE})\ \wedge \\
\quad (\forall\, x\ phi.\ \mathsf{ST}\ x\ (\neg phi)\ =\ \mathsf{fNOT}\ (\mathsf{ST}\ x\ phi))\ \wedge \\
\quad (\forall\, x\ phi\ psi. \\
\qquad \mathsf{ST}\ x\ (\mathsf{DISJ}\ phi\ psi)\ =\ \mathsf{fDISJ}\ (\mathsf{ST}\ x\ phi)\ (\mathsf{ST}\ x\ psi))\ \wedge \\
\quad \forall\, x\ phi. \\
\qquad \mathsf{ST}\ x\ (\Diamond\ phi)\ = \\
\qquad\ \mathsf{fEXISTS}\ (x\ +\ 1) \\
\qquad\quad (\mathsf{fAND}\ (\mathsf{fR}\ (\mathsf{fV}\ x)\ (\mathsf{fV}\ (x\ +\ 1)))\ (\mathsf{ST}\ (x\ +\ 1)\ phi))
\end{array}
$$

Here $\lambda\, p\ \ t.\ \mathsf{fP}\ \ p\ \ t$ is the abbrevation of $\lambda\, p\ \ t.\ \mathsf{fP}\ \ p\ \ t$, and $\lambda\, w_1\ \ w_2.\ \mathsf{fR}\ \ w_1\ \ w_2$ is the abbrevation of $\lambda\, w_1\ \ w_2.\ \mathsf{fR}\ \ w_1\ \ w_2$. Any formula obtained by standard translation only has one free variable x, and a standard translation can never have function symbols:

$$
\vdash \mathsf{FV}\ (\mathsf{ST}\ x\ f)\ \subseteq\ \{\ x\ \}\quad \vdash \mathsf{form\_functions}\ (\mathsf{ST}\ x\ f)\ =\ \emptyset
$$

We can conjunct standard tranalations to get a standard translation. And the negation of standard translation is again a standard translation:

$$
\begin{array}{l}
\vdash \mathsf{FINITE}\ s\ \Rightarrow \\
\quad \forall x. \\
\qquad (\forall f.\, f\ \in\ s\ \Rightarrow\ \exists\, phi.\, f\ =\ \mathsf{ST}\ x\ phi)\ \Rightarrow \\
\qquad \exists\, cf. \\
\qquad\quad (\forall \mathfrak{M}\ \sigma. \\
\qquad\qquad \mathsf{IMAGE}\ \sigma\ \mathcal{U}(:\mathtt{num})\ \subseteq\ \mathfrak{M}.\mathsf{Dom}\ \Rightarrow \\
\qquad\qquad (\mathsf{feval}\ \mathfrak{M}\ \sigma\ cf\ \iff\ \forall f.\, f\ \in\ s\ \Rightarrow\ \mathsf{feval}\ \mathfrak{M}\ \sigma\ f))\ \wedge \\
\qquad\quad \exists\, psi.\ cf\ =\ \mathsf{ST}\ x\ psi\ \vdash \mathsf{ST}\ x\ (\neg f)\ =\ \mathsf{fNOT}\ (\mathsf{ST}\ x\ f)
\end{array}
$$

Standard translation defined like this can be regarded as a first-order reformulation of modal satisfication, since we have the precise correspondence of modal satisfication and first-order satisfication for standard translations. A modal formula is satisfied a point w in a modal model iff its standrad translation at x is satisfied at the same modal viewed as a first order model when x is valuated to w:

$$
\begin{array}{l}
\vdash \mathsf{IMAGE}\ \sigma\ \mathcal{U}(:\mathtt{num})\ \subseteq\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\quad (\mathsf{satis}\ \mathfrak{M}\ (\sigma\ x)\ phi\ \iff\ \mathsf{fsatis}\ (\mathsf{mm2folm}\ \mathfrak{M})\ \sigma\ (\mathsf{ST}\ x\ phi))
\end{array}
$$

Hence we say the current definition of standard translation makes good semantical sense, but it has syntactical deficiency. The formula $\Diamond\ (\Diamond\ f)$ with a_1 marking its state is translated to $\exists a_2(Ra_1a_2 \wedge \exists a_3(Ra_2a_3 \wedge ST_{a_3}(f)))$, it uses three variables a_1, a_2 and a_3, which is unnecessary: the formula above is equivalent to $\exists a_2(Ra_1a_2 \wedge \exists a_1(Ra_2a_1 \wedge ST_{a_1}(f)))$. With this observation, we conclude that we do not need to always come up with a variable symbol for each $\Diamond$, instead, we can use only two variable symbols alternating in each layer. As a consequence, we can redefine standard translation as follows:

$$
\begin{array}{l}
\mathsf{ST\_alt}\ x\ (\mathsf{VAR}\ p)\ \stackrel{\mathsf{def}}{=}\ \mathsf{fP}\ p\ (\mathsf{fV}\ x) \\
\mathsf{ST\_alt}\ x\ \bot\ \stackrel{\mathsf{def}}{=}\ \mathsf{fFALSE} \\
\mathsf{ST\_alt}\ x\ (\neg phi)\ \stackrel{\mathsf{def}}{=}\ \mathsf{fNOT}\ (\mathsf{ST\_alt}\ x\ phi) \\
\mathsf{ST\_alt}\ x\ (\mathsf{DISJ}\ phi\ psi)\ \stackrel{\mathsf{def}}{=}\ \mathsf{fDISJ}\ (\mathsf{ST\_alt}\ x\ phi)\ (\mathsf{ST\_alt}\ x\ psi) \\
\mathsf{ST\_alt}\ x\ (\Diamond\ phi)\ \stackrel{\mathsf{def}}{=} \\
\quad \mathsf{fEXISTS}\ (1\ -\ x) \\
\qquad (\mathsf{fAND}\ (\mathsf{fR}\ (\mathsf{fV}\ x)\ (\mathsf{fV}\ (1\ -\ x)))\ (\mathsf{ST\_alt}\ (1\ -\ x)\ phi))
\end{array}
$$

(prove only two variables?)

Note that the above definition is designed to only use 0 or 1 as the x. It is evident from the follow proposition that our new definition of translation is equally nice from the semantical aspect as the former one:

$$\begin{aligned}
&\vdash \mathsf{IMAGE}\ \sigma\ \mathcal{U}(:\mathtt{num}) \subseteq \mathfrak{M}.\mathsf{frame.world} \Rightarrow \\
&\quad (\mathsf{satis}\ \mathfrak{M}\ (\sigma\ 1)\ phi \iff \\
&\qquad \mathsf{fsatis}\ (\mathsf{mm2folm}\ \mathfrak{M})\ \sigma\ (\mathsf{ST_alt}\ 1\ phi)) \wedge \\
&\quad (\mathsf{satis}\ \mathfrak{M}\ (\sigma\ 0)\ phi \iff \\
&\qquad \mathsf{fsatis}\ (\mathsf{mm2folm}\ \mathfrak{M})\ \sigma\ (\mathsf{ST_alt}\ 0\ phi))
\end{aligned}$$

By conclusion, every modal formula is equivalent to a first order formula containing only two variables.

4.2 Modal Saturation via ultrafilter extensions

In the second chapter, we have seen bisimilarity implies modal equivalence, but only proved the converse for image finite models. In this section, we are interested in another particular class of models which modal equivalence implies bisimilarity, which is the class of m-saturated models.

A set of formulas Σ is called satisfiable in a set of worlds X of a model $\mathfrak{M}$ if there exists a world in X such that all the formulas in Σ are satisfied, and is called finitely satisfiable if any finite subset of Σ is satisfiable:

$$\begin{aligned}
&\mathsf{satisfiable_in}\ \Sigma\ X\ \mathfrak{M} \stackrel{\mathsf{def}}{=} \\
&X \subseteq \mathfrak{M}.\mathsf{frame.world} \wedge \exists x.\ x \in X \wedge \forall \phi.\ \phi \in \Sigma \Rightarrow \mathsf{satis}\ \mathfrak{M}\ x\ \phi \mathsf{fin_satisfiable_in}\ \Sigma\ X\ \mathfrak{M} \stackrel{\mathsf{def}}{=} \\
&\forall S.\ S \subseteq \Sigma \wedge \mathsf{FINITE}\ S \Rightarrow \mathsf{satisfiable_in}\ S\ X\ \mathfrak{M}
\end{aligned}$$

A model $\mathfrak{M}$ is called m-saturated if for each $w \in \mathfrak{M}.\mathsf{frame.world}$ and any set Σ of formulas, if Σ is finitely satisfiable in the set of successors of w, then it is satisfiable in the set of successors of w.

$$\begin{aligned}
&\mathsf{M_sat}\ \mathfrak{M} \stackrel{\mathsf{def}}{=} \\
&\forall w\ \Sigma. \\
&\quad w \in \mathfrak{M}.\mathsf{frame.world} \Rightarrow \\
&\quad \mathsf{fin_satisfiable_in}\ \Sigma \\
&\qquad \{\ v \mid v \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v\ \}\ \mathfrak{M} \Rightarrow \\
&\quad \mathsf{satisfiable_in}\ \Sigma \\
&\qquad \{\ v \mid v \in \mathfrak{M}.\mathsf{frame.world} \wedge \mathfrak{M}.\mathsf{frame.rel}\ w\ v\ \}\ \mathfrak{M}
\end{aligned}$$

In fact, a class of model where modal equivalence implies bisimilarity has a name, it is called a Hennessy-Milner class. If we want to formalise the definition of Hennessy-Milner class in the HOL, we could write:

$$\begin{aligned}
&\vdash \mathsf{HM_class}\ K \iff \\
&\quad \forall \mathfrak{M}\ \mathfrak{M}'\ w\ w'. \\
&\qquad \mathfrak{M} \in K \wedge \mathfrak{M}' \in K \wedge w \in \mathfrak{M}.\mathsf{frame.world} \wedge \\
&\qquad w' \in \mathfrak{M}'.\mathsf{frame.world} \Rightarrow \\
&\qquad \mathsf{modal_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w' \Rightarrow \\
&\qquad \mathsf{bisim_world}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w'
\end{aligned}$$

But in fact, such a definition is useless, since we are not allowed to have a 'class' in the HOL, we can only have set, and any set is only allowed to have elements of the same type. Hence if we use this definition, we will only be allowed to talk about bisimulations between models with the same type. As a consequence, we do not make usage of this definition in the following formalisation of the proposition which says the class of m-saturation has the Hennessy-Milner property, instead, we state it as:

$$\begin{aligned}
&\vdash \mathsf{M_sat}\ \mathfrak{M}\ \wedge\ \mathsf{M_sat}\ \mathfrak{M}'\ \wedge\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ w'\ \in\ \mathfrak{M}'.\mathsf{frame.world}\ \Rightarrow \\
&\quad \mathsf{modal_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w'\ \Rightarrow \\
&\quad \mathsf{bisim_world}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w'
\end{aligned}$$

Proof: Let $\mathfrak{M}$ and $\mathfrak{M}'$ be $(\alpha, \beta), (\alpha, \gamma)$ models respectively. Under the assumptions, the bisimulation relation is given by $\lambda\, n_1\ n_2.\ \forall\, \phi.\ \mathsf{satis}\ \mathfrak{M}\ n_1\ \phi\ \iff\ \mathsf{satis}\ \mathfrak{M}'\ n_2\ \phi$. To only non-trivial clause of being a bisimulation to check is that for worlds w, v of $\mathfrak{M}$ and world w' of $\mathfrak{M}'$ such that $\mathfrak{M}.\mathsf{frame.rel}\ w\ v$ and w and w' are modal equivalent, we can find a world v' of $\mathfrak{M}'$ such that $\mathfrak{M}'.\mathsf{frame.rel}\ w'\ v'$ and v and v' are modal equivalent.

Let Σ denote the set of formulas satisfied by v, it suffices to find a successor of w' that realises Σ. As $\mathfrak{M}'$ is m-saturated, it suffices to prove each finite subset $\Delta \subseteq \Sigma$ is satisfied in some successor of w'. Take such a Δ, then it is satisfied at v. As Δ is finite, we can prove that there exists a formula ff such that for all (α, β)-models, ff is satisfied at a world if and only if all elements in Δ are satisfied:

$$\begin{aligned}
&\vdash \mathsf{FINITE}\ s\ \Rightarrow \\
&\quad \exists\, ff. \\
&\qquad \forall\, w\ \mathfrak{M}. \\
&\qquad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
&\qquad\quad (\mathsf{satis}\ \mathfrak{M}\ w\ ff\ \iff\ \forall f.\ f\ \in\ s\ \Rightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ f)
\end{aligned}$$

We have $\mathsf{satis}\ \mathfrak{M}\ v\ ff$, and therefore $\mathsf{satis}\ \mathfrak{M}\ w\ (\Diamond\ ff)$. By modal equivalence of w and w', we then get $\mathsf{satis}\ \mathfrak{M}\ w'\ (\Diamond\ ff)$, so there exists a successor of w' that satisfies ff.

But it is not what we want! Instead, we need a successor of w' which satisfies all elements in Σ, but we cannot get it if we just use the lemma above, since the equivalence of the set of formulas and its big conjunction we proved only works for (α, β) models, but we are in $\mathfrak{M}'$ which is an (α, γ)-model. By embedding any arbitary model into a common infinite type, it may possible to prove that under some condition on universe, if a big conjunction works for a model of one type, then it works for any other types. But it is way too complicated and will involve an ugly assumption. The key observation is that what we need is no more than a big conjunction formula that simutinously works for two distinct types. Therefore, it suffices to prove:

$$
\begin{array}{l}
\vdash \mathsf{FINITE}\ s\ \Rightarrow \\
\quad \exists ff. \\
\quad\quad (\forall\, w\ \mathfrak{M}. \\
\quad\quad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\quad\quad\quad (\mathsf{satis}\ \mathfrak{M}\ w\ ff\ \iff\ \forall f.\ f\ \in\ s\ \Rightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ f))\ \wedge \\
\quad\quad \forall\, w\ \mathfrak{M}. \\
\quad\quad\quad w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\quad\quad\quad (\mathsf{satis}\ \mathfrak{M}\ w\ ff\ \iff\ \forall f.\ f\ \in\ s\ \Rightarrow\ \mathsf{satis}\ \mathfrak{M}\ w\ f)
\end{array}
$$

This lemma works perfectly, so we will not get stuck at the former point and we are done with the proof.

Since m-saturated models are nice, here is a natural question: How can we get such models. It turns out that every models can give arise to a m-saturated model, by taking its ultrafilter extension. In order to talk about ultrafilter extesnions, we need to build a theory of ultrafilers in HOL.

4.3 Interlude II: Ultrafilters

As its name suggests, an ultrafilter is a special kind of filter. Given a non-empty set W, a set F which is a subset of the power set of W, written as $POW(W)$ in the HOL, is called a filter if it contains W itself, closed under binary intersection, and closed upward.

$$
\begin{array}{l}
\mathsf{filter}\ FLT\ W\ \stackrel{\mathsf{def}}{=} \\
\quad W\ \neq\ \emptyset\ \wedge\ FLT\ \subseteq\ \mathsf{POW}\ W\ \wedge\ W\ \in\ FLT\ \wedge \\
\quad (\forall X\ Y.\ X\ \in\ FLT\ \wedge\ Y\ \in\ FLT\ \Rightarrow\ X\ \cap\ Y\ \in\ FLT)\ \wedge \\
\quad \forall X\ Z.\ X\ \in\ FLT\ \wedge\ X\ \subseteq\ Z\ \wedge\ Z\ \subseteq\ W\ \Rightarrow\ Z\ \in\ FLT
\end{array}
$$

By induction, closure under binary intersection implies that a filter is closed under finite intersection.

$$
\begin{array}{l}
\vdash \mathsf{FINITE}\ s\ \Rightarrow \\
\quad s\ \neq\ \emptyset\ \Rightarrow \\
\quad \forall\, U\ W.\ \mathsf{filter}\ U\ W\ \Rightarrow\ s\ \subseteq\ U\ \Rightarrow\ \mathsf{BIGINTER}\ s\ \in\ U
\end{array}
$$

Obviously for any set W, $POW(W)$ is a filter on W. By upward closure, any filter which contains the empty set must be $POW(W)$.

$$
\vdash\ W\ \neq\ \emptyset\ \Rightarrow\ \mathsf{filter}\ (\mathsf{POW}\ W)\ W\ \vdash\ \mathsf{filter}\ U\ W\ \wedge\ \emptyset\ \in\ U\ \Rightarrow\ U\ =\ \mathsf{POW}\ W
$$

But $POW(W)$ is very boring as a filter, we are mainly interested in filters which are not the whole power set, these filters are called proper filer:

$$
\vdash\ \mathsf{proper\_filter}\ FLT\ W\ \iff\ \mathsf{filter}\ FLT\ W\ \wedge\ FLT\ \neq\ \mathsf{POW}\ W
$$

If we have a set proper filters such that for any two members A, B of it, either $A \subset B$ or $B \subseteq A$, the union of this set is a proper filter.

$$\vdash W \neq \emptyset \wedge U \neq \emptyset \wedge (\forall A.\ A \in U \Rightarrow \mathsf{proper\_filter}\ A\ W) \wedge (\forall A\ B.\ A \in U \wedge B \in U \Rightarrow A \subseteq B \vee B \subseteq A) \Rightarrow \mathsf{proper\_filter}\ (\mathsf{BIGUNION}\ U)\ W$$

Another important kind of filter is the generated filter. For $W \neq \emptyset$ and $F_0 \subseteq POW(W)$, the generated filter is the minimal filter on W containing F_0.

$$\vdash \mathsf{generated\_filter}\ E\ W = \mathsf{BIGINTER}\ \{\ G \mid E \subseteq G \wedge \mathsf{filter}\ G\ W\ \}$$

The advantage of the above definition is that we get the fact that a generated filter is indeed a filter for free.

But sometimes we want to explicitly talk about elements in generated filter, then we can just spell out the its construction: We put any subset W that contains a finite intersection of elements in F in the generated filter of F, such subsets are precisely the ones that are required by the definition. We can prove generated filter is indeed a filter under this definition:

$$\vdash W \neq \emptyset \wedge F \subseteq \mathsf{POW}\ W \Rightarrow \mathsf{filter}\ \{\ X \mid X \subseteq W \wedge (X = W \vee \exists S.\ S \subseteq F \wedge \mathsf{FINITE}\ S \wedge S \neq \emptyset \wedge \mathsf{BIGINTER}\ S \subseteq X)\ \}\ W$$

For any element $w \in W$, the set of all the subset containing w forms a filter, it is called the principle filter generated by w. In fact, principle filters are filter generated by singletons.

$$\vdash \mathsf{principle\_UF}\ w\ W = \{\ X \mid X \subseteq W \wedge w \in X\ \}$$

An ultrafilter on a set W is a proper filter F such that for any $S \subseteq W$, either S or W/S is in F, but not both.

$$\mathsf{ultrafilter}\ U\ W \stackrel{\mathsf{def}}{=} \mathsf{proper\_filter}\ U\ W \wedge \forall X.\ X \in \mathsf{POW}\ W \Rightarrow (X \in U \iff W\ \mathsf{DIFF}\ X \notin U)$$

Principle filters are examples of the type of filters of our main interest, that is, the ultrafilters.

$$\vdash W \neq \emptyset \wedge w \in W \Rightarrow \mathsf{ultrafilter}\ (\mathsf{principle\_UF}\ w\ W)\ W$$

Since ultrafilter are important, we may ask when can we get an ultrafilter on W from a subset of $POW(W)$. We will prove later that the answer is for any subset of $POW(W)$ with finite intersection property, we can extend it into an ultrafilter. A subset of $POW(W)$ has finite intersection property if it is closed under finite intersection:

$$\vdash \mathsf{FIP}\ S\ W \iff$$
$$S \subseteq \mathsf{POW}\ W \wedge$$
$$\forall S'.\ S' \subseteq S \wedge \mathsf{FINITE}\ S' \wedge S' \neq \emptyset \Rightarrow \mathsf{BIGINTER}\ S' \neq \emptyset$$

Any proper filter U has the finite intersection property. Moreover, for $B \subseteq W$ such that neither B nor W/B is in U, then inserting B to U does not destory the finite intersection property of U.

$$\vdash \mathsf{proper\_filter}\ U\ W \Rightarrow \mathsf{FIP}\ U\ W \vdash \mathsf{proper\_filter}\ U\ W \wedge B \in \mathsf{POW}\ W \wedge B \notin U \wedge W\ \mathsf{DIFF}\ B \notin U \Rightarrow \mathsf{FIP}\ (\{\ B\ \} \cup U)\ W$$

Proof: `proper_filter_FIP` is immediate from closure under finite intersection. We prove `proper_filter_INSERT_FIP`. Take a finite subset $S \subseteq \{B\} \cup U$, if $B \notin S$, then we are done by `proper_filter_FIP`. Otherwise, if we have $\bigcap S = B \cap (\bigcap U')$ for some finite $U' \subseteq U$, which implies $\bigcap U' \subseteq W/B$, hence $W/B \in U$ by upward closure, contradicting the assumption that $W/B \notin U$.

Towards the goal of extending a set with finite intersection property, we firstly prove that such sets can be extended into a proper filter.

$$\vdash W \neq \emptyset \wedge \mathsf{FIP}\ S\ W \Rightarrow \exists V.\ \mathsf{proper\_filter}\ V\ W \wedge S \subseteq V$$

Proof: The desired proper filter is given by the generated filter of S, checking its property is straightfoward.

A proper filter which is not properly contained by any proper filter is called a maximal filter. We can readily check ultrafilters are maximal. With the last two lemmas about finite intersection property, we can prove maximal filters are ultrafilters, so maximal filters and ultrafilters turns out to be the same thing.

$$\vdash \mathsf{proper\_filter}\ U\ W \wedge (\forall S.\ \mathsf{filter}\ S\ W \wedge U \subset S \Rightarrow S = \mathsf{POW}\ W) \Rightarrow \mathsf{ultrafilter}\ U\ W$$

Proof: If a maximal filter U on W is not an ultrafilter, then for some A, either $A \in U \wedge W/A \in U$ or $A \notin U \wedge W/A \notin U$. In the first case we have $\emptyset \in U$, contradicts the properness of maximal filter. In the second case, by `proper_filter_INSERT_FIP`, the set $U \cup \{A\}$ has the finite intersection property, hence extends to a proper filter U' by `FIP_PSUBSET_proper_filter`. But then U is properly contained in U', contradicts the maximality of U. This completes the proof.

Together with the Zorn's lemma which is already in the HOL library, now we have all ingredients of the proof of ultrafilter.

$$\vdash \mathsf{proper\_filter}\ f\ w \Rightarrow \exists U.\ \mathsf{ultrafilter}\ U\ w \wedge f \subseteq U$$

Proof:Given a property filter F, we will find out a ultrafilter containing it. Consider the set S of all the proper filters containing F, ordered by inclusion, we will apply Zorn's lemma on S. The Zorn's lemma in the HOL looks like:

$$\begin{aligned}\vdash\ & s\ \neq\ \emptyset\ \wedge\ \mathsf{partial\_order}\ r\ s\ \wedge \\ & (\forall\, t.\ \mathsf{chain}\ t\ r\ \Rightarrow\ \mathsf{upper\_bounds}\ t\ r\ \neq\ \emptyset)\ \Rightarrow \\ & \exists\, x.\ x\ \in\ \mathsf{maximal\_elements}\ s\ r\end{aligned}$$

Here s is a set, r is a relation that takes a pair of elements (a, b) with $a, b \in s$ and return the truth value whether a and b are related. For a set t with the same type as s, we have $\mathsf{chain}\ t\ r$ if for any $a, b \in t$, we have either $(a, b) \in r$ or $(b, a) \in r$. Here our s is the set S defined above. r is defined by inclusion. For $A, B \in S$, $(A, B) \in r$ iff $A \subseteq B$, so a chain T on S is a subset of S such that $A \subseteq B$ or $B \subseteq A$ for any $A, B \in T$.

The thing to check is that any chain T on S has an upper bound. If the chain is empty, then the upper bound is clearly F. Otherwise, we claim the union of the chain is the upper bound. To prove the claim, it amounts to check:

- $\bigcup T$ is a proper filer containing F.
- Any element in T is a subset of $\bigcup T$.

The second item is trivial, the first one is by `UNION_proper_proper`.

Applying the Zorn's lemma gives a proper filter $X \in S$ which is an maximal element of S, it suffices to prove X is an ultrafilter. The fact that X is the maximal element of S proves X is a maximal filter, hence we are done by `maximal_ultrafilter`.

As a corollary, we can extend any set with finite intersection into an ultrafilter:

$$\vdash\ \mathsf{FIP}\ s\ W\ \wedge\ W\ \neq\ \emptyset\ \Rightarrow\ \exists\, u.\ \mathsf{ultrafilter}\ u\ W\ \wedge\ s\ \subseteq\ u$$

Proof: For $S \subseteq POW(W)$ with finite intersection property, by `FIP_PSUBSET_proper_filter`, we have a proper filter P containing S, and P extends to an ultrafiler $S \subseteq P \subseteq U$ by `ultrafilter_theorem`.

As an application of `ultrafilter_theorem_corollary`, we prove the existence of countably incomplete ultrafilters. A countably incomplete ultrafilter is an ultrafilter which is not closed under countably infinite intersection.

$$\begin{aligned}&\mathsf{countably\_incomplete}\ U\ W\ \stackrel{\mathsf{def}}{=} \\ &\ \mathsf{ultrafilter}\ U\ W\ \wedge \\ &\ \exists\, IFS\ f.\ IFS\ \subseteq\ U\ \wedge\ \mathsf{BIJ}\ f\ \mathcal{U}(:\mathtt{num})\ IFS\ \wedge\ \mathsf{BIGINTER}\ IFS\ =\ \emptyset\end{aligned}$$

It is not hard to see any ultrafilter U on the natural numbers $\mathbb{N}$ which does not contain any singleton is countably incomplete, since the countable intersection of the sets $\mathbb{N}/\{n\}$ which are members of U yields the empty set.

$$\begin{aligned}\vdash\ & \mathsf{ultrafilter}\ U\ \mathcal{U}(:\mathtt{num})\ \wedge\ (\forall\, n.\ \{\ n\ \}\ \notin\ U)\ \Rightarrow \\ & \mathsf{countably\_incomplete}\ U\ \mathcal{U}(:\mathtt{num})\end{aligned}$$

We find out an ultrafilter on $\mathbb{N}$ which does not contain any singleton, we will done if we can prove the existence of an ultrafilter on $\mathbb{N}$ which does not contain any finite set. By `ultrafilter_theorem_corollary`, it suffices to prove:

$$\vdash \mathsf{FIP}\ \{\ \mathcal{U}(:\mathtt{num})\ \mathsf{DIFF}\ X\ \mid\ \mathsf{FINITE}\ X\ \}\ \mathcal{U}(:\mathtt{num})$$

Proof: By unfolding the definition of finite intersection property and induct on finiteness of the set we are intersecting.

A countably incomplete ultrafilter has useful features, one of them is that such an ultrafilter U has a chain $I = I_0 \supseteq I_1 \supseteq I_2 \supseteq \cdots$ such that each I_i is in U, and $\bigcap_{n\in\mathbb{N}} I_n = \emptyset$.

$$\begin{array}{l}\vdash \mathsf{countably\_incomplete}\ U\ I\ \Rightarrow\\ \quad \exists\, In.\\ \qquad In\ 0\ =\ I\ \wedge\ (\forall\, n.\ In\ n\ \in\ U\ \wedge\ In\ (n\ +\ 1)\ \subseteq\ In\ n)\ \wedge\\ \qquad \mathsf{BIGINTER}\ \{\ In\ n\ \mid\ n\ \in\ \mathcal{U}(:\mathtt{num})\ \}\ =\ \emptyset\end{array}$$

Proof: By definition of countably incompleteness, there exists a family X_n in U indexed by natural numbers such that $\bigcap_{n\in\mathbb{N}} X_n = \emptyset$. Define $J_n := \bigcap_{m\leq n} X_n$, so $J_{n+1} \supseteq J_n$ for all $n \in \mathbb{N}$, and moreover $\bigcap_{n\in\mathbb{N}} I_n = \emptyset$. In the HOL, the family J_n is defined using a primitive recursive function, defined by $\mathsf{PRIM\_REC}\ (X\ 0)\ (\lambda\, Xn\ n.\ Xn\ \cap\ X\ (n\ +\ 1))$. We get the desired chain I_n by inserting I at the begining of J_n.

Our theory of ultrafilters built above is to enable us to talk about ultrafilter extensions. Ultrafilter extension is defined as a function that takes a model $\mathfrak{M}$, and give the extended model. The world set of $\mathsf{UE}\ \mathfrak{M}$ is the set of all the ultrafilters on the world set of $\mathfrak{M}$. And the relation of these worlds requires some explaination. For $X\ \subseteq\ \mathfrak{M}.\mathsf{frame.world}$, there comes two interesting sets of worlds, one is the set of worlds that is linked to some world in X, and the other one is the set of worlds that are only linked to worlds in X. These two sets are dual to each other:

$$\begin{array}{l}\mathsf{can\_see}\ \mathfrak{M}\ X\ \stackrel{\mathsf{def}}{=}\\ \{\ w\ \mid\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \exists\, x.\ x\ \in\ X\ \wedge\ \mathfrak{M}.\mathsf{frame.rel}\ w\ x\ \}\ \mathsf{only\_see}\ \mathfrak{M}\ X\ \stackrel{\mathsf{def}}{=}\\ \{\ w\ \mid\\ \ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\\ \ \forall\, x.\ x\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathfrak{M}.\mathsf{frame.rel}\ w\ x\ \Rightarrow\ x\ \in\ X\ \}\ \vdash X\ \subseteq\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ \quad \mathsf{can\_see}\ \mathfrak{M}\ X\ =\\ \qquad \mathfrak{M}.\mathsf{frame.world}\ \mathsf{DIFF}\ \mathsf{only\_see}\ \mathfrak{M}\ (\mathfrak{M}.\mathsf{frame.world}\ \mathsf{DIFF}\ X)\ \vdash X\ \subseteq\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow\\ \quad \mathsf{only\_see}\ \mathfrak{M}\ X\ =\\ \qquad \mathfrak{M}.\mathsf{frame.world}\ \mathsf{DIFF}\ \mathsf{can\_see}\ \mathfrak{M}\ (\mathfrak{M}.\mathsf{frame.world}\ \mathsf{DIFF}\ X)\end{array}$$

Directly from their definitions, the worlds satisfying $\Diamond\ phi$ are the worlds that can see a world satisfying ϕ. And the worlds satisfying $\Box\ phi$ are the worlds that can only see worlds that satisfies ϕ, and the set of worlds which only see $X \cap Y$ is the intersection of worlds only see X and worlds only see Y:

$$\begin{array}{l}\vdash \{\ w\ \mid\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{satis}\ \mathfrak{M}\ w\ (\Diamond\ \phi)\ \}\ =\\ \quad \mathsf{can\_see}\ \mathfrak{M}\ \{\ v\ \mid\ v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{satis}\ \mathfrak{M}\ v\ \phi\ \}\ \vdash \{\ w\ \mid\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{satis}\ \mathfrak{M}\ w\ (\Box\ \phi)\ \}\ =\\ \quad \mathsf{only\_see}\ \mathfrak{M}\ \{\ v\ \mid\ v\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{satis}\ \mathfrak{M}\ v\ \phi\ \}\ \vdash \mathsf{only\_see}\ \mathfrak{M}\ (X\ \cap\ Y)\ =\ \mathsf{only\_see}\ \mathfrak{M}\ X\ \cap\ \mathsf{only\_see}\ \mathfrak{M}\ Y\end{array}$$

We define two ultrafilter u, v to be related if any world set of $\mathfrak{M}$ in v can only see world sets in u. Such a definition has a reformulation: for any world set of $\mathfrak{M}$, if the set of worlds that it can only see is in u, then this set is in v:

$$
\begin{array}{l}
\mathsf{UE\_rel}\ \mathfrak{M}\ u\ v\ \stackrel{\mathsf{def}}{=} \\
\quad \mathsf{ultrafilter}\ u\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{ultrafilter}\ v\ \mathfrak{M}.\mathsf{frame.world}\ \wedge \\
\quad \forall X.\ X\ \in\ v\ \Rightarrow\ \mathsf{can\_see}\ \mathfrak{M}\ X\ \in\ u\ \vdash \mathsf{ultrafilter}\ u\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{ultrafilter}\ v\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\quad\quad (\mathsf{UE\_rel}\ \mathfrak{M}\ u\ v\ \iff \\
\quad\quad\quad \{\ Y\ \mid\ \mathsf{only\_see}\ \mathfrak{M}\ Y\ \in\ u\ \wedge\ Y\ \subseteq\ \mathfrak{M}.\mathsf{frame.world}\ \}\ \subseteq\ v)
\end{array}
$$

The ultrafilter extension is defined as follows:

$$
\begin{array}{l}
\mathsf{UE}\ \mathfrak{M}\ \stackrel{\mathsf{def}}{=} \\
\quad \langle|\mathsf{frame}\ := \\
\quad\quad \langle|\mathsf{world}\ :=\ \{\ u\ \mid\ \mathsf{ultrafilter}\ u\ \mathfrak{M}.\mathsf{frame.world}\ \};\\
\quad\quad\quad \mathsf{rel}\ :=\ \mathsf{UE\_rel}\ \mathfrak{M}|\rangle; \\
\quad\quad \mathsf{valt}\ := \\
\quad\quad\quad (\lambda\, p\ v. \\
\quad\quad\quad\quad\quad \mathsf{ultrafilter}\ v\ \mathfrak{M}.\mathsf{frame.world}\ \wedge \\
\quad\quad\quad\quad\quad \mathfrak{M}.\mathsf{valt}\ p\ \cap\ \mathfrak{M}.\mathsf{frame.world}\ \in\ v)|\rangle
\end{array}
$$

The ultrafilter extension also changes the type of model, namely, it change the type of world set from β to $(\beta \to \mathit{bool}) \to \mathit{bool}$, it is indeed an extension, in the sense that $\mathfrak{M}$ is embedded in $\mathsf{UE}\ \mathfrak{M}$ by the function sending $w\ \in\ \mathfrak{M}.\mathsf{frame.world}$ to the principle ultrafilter $\mathsf{principle\_UF}\ w\ \mathfrak{M}.\mathsf{frame.world}$ generated by w. In general, this embedding does not necessarily give a generated submodel, nevertheless, we have an invariance result for this embedding:

$$
\begin{array}{l}
\vdash w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\quad \mathsf{modal\_eq}\ \mathfrak{M}\ (\mathsf{UE}\ \mathfrak{M})\ w\ (\mathsf{principle\_UF}\ w\ \mathfrak{M}.\mathsf{frame.world})
\end{array}
$$

This is actually a special case of the following proposition, where u is taken as $\mathsf{principle\_UF}\ w\ \mathfrak{M}.\mathsf{frame.world}$. This proposition captures the idea that ultrafilters are used to describe the sense of 'most of', where the sets in the ultrafilters can be regarded as 'sets which are large enough'. From this aspect, the proposition says that a formula ϕ is satisfied in an ultrafilter u iff ϕ is satisfied at most of worlds in $\mathfrak{M}$, where the sense of 'most of' is measured by u.

$$
\begin{array}{l}
\vdash \mathsf{ultrafilter}\ u\ \mathfrak{M}.\mathsf{frame.world}\ \Rightarrow \\
\quad (\{\ w\ \mid\ w\ \in\ \mathfrak{M}.\mathsf{frame.world}\ \wedge\ \mathsf{satis}\ \mathfrak{M}\ w\ \mathit{phi}\ \}\ \in\ u\ \iff \\
\quad\quad \mathsf{satis}\ (\mathsf{UE}\ \mathfrak{M})\ u\ \mathit{phi})
\end{array}
$$

Proof: By induction on phi. Everything except for the diamond case is by unwinding the definitions of $\mathsf{UE}\ \mathfrak{M}$ and ultrafilter. We only give the proof of the diamond case.

From right to left: Suppose satis (UE $\mathfrak{M}$) u ($\Diamond$ phi), then satis (UE $\mathfrak{M}$) u' psi for some $UE\_REL\ \mathfrak{M}\ u\ u'$. By inductive hypothesis, satis (UE $\mathfrak{M}$) u' phi implies $\{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\} \in u'$. And then by definition of UE_rel $\mathfrak{M}$, this implies can_see $\mathfrak{M}$ $\{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\} \in u$. But this set is exactly the set $\{w | w \in M.frame.world \wedge satis\ M\ w\ (\Diamond\phi)\}$ by the observation in `valt_can_see`.

From left to right: Suppose $\{w | w \in M.frame.world \wedge satis\ M\ w\ (\Diamond\phi)\} \in u$, we need to find out an ultrafilter u' such that UE_rel $\mathfrak{M}$ u u' and satis (UE $\mathfrak{M}$) u' phi. Using `exercise_2_5_5`, UE_rel $\mathfrak{M}$ u u' is equivalent to $s := \{Y \mid only\_see(Y) \in u \wedge Y \subseteq M.frame.world\} \subseteq u'$. We prove $s \cup \{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\}$ has finite intersection property. It suffices to check (1) s is closed under intersection, and (2) The interesection of any element in s with $\{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\}$ is non-empty.

For (1), if $a, b \in s$, then both only_see $\mathfrak{M}$ a and only_see $\mathfrak{M}$ b are in u, and hence there intersection. By `only_see_INTER`, only_see $\mathfrak{M}$ a $\cap$ only_see $\mathfrak{M}$ b $=$ only_see $\mathfrak{M}$ $(a \cap b)$, hence $a \cap b \in u$.

For (2), let $Y \in s$, then only_see $\mathfrak{M}$ $Y \in u$, as also $\{w | w \in M.frame.world \wedge satis\ M\ w\ (\Diamond\phi)\} \in u$, by closure under intersection for an ultrafilter, $only\_see(Y) \cap \{w | w \in M.frame.world \wedge satis\ M\ w\ (\Diamond\phi)\} \neq \emptyset$, we obtaine an element of $Y \cap \{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\}$ from any element x of it. As $x \in \{w | w \in M.frame.world \wedge satis\ M\ w\ (\Diamond\phi)\}$, there exists $y \in \mathfrak{M}$.frame.world such that satis $\mathfrak{M}$ y phi, as $x \in only\_see(Y)$, we have $y \in Y$.

The rest of the proof of finite intersection property is done by induction. Hence by `ultrafilter_theorem_corollary`, there exists an ultrafilter u' such that $s \cup \{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\} \subseteq u'$. This is the u' we want: We have UE_rel $\mathfrak{M}$ u u' since $\{Y \mid only\_see(Y) \in u\} = s \subseteq u'$, and $\{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\} \in u'$, we get satis (UE $\mathfrak{M}$) u' phi by inductive hypothesis.

The above proposition leads to a proof of m-saturatedness of ultrafilter extensions.

$$\vdash \text{M\_sat (UE } \mathfrak{M})$$

Proof: Suppose Σ is a set of formulas which is finitely satisfiable in the set of successors of a world $u \in$ (UE $\mathfrak{M}$).frame.world, we need to find a world $u' \in$ (UE $\mathfrak{M}$).frame.world such that UE_rel $\mathfrak{M}$ u u' and satis (UE $\mathfrak{M}$) u' phi for all $phi \in \Sigma$. By `exercise_2_5_5` and `prop_2_59_i`, it amounts to find an ultrafilter u' on $\mathfrak{M}$.frame.world such that $\{Y \mid only\_see(Y) \in u\} \subseteq u'$ and $\{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\} \in u'$ for all $phi \in \Sigma$.

Consider the set $\Delta := \{\{w \mid w \in M.frame.world \wedge \forall\phi.\phi \in s \implies satis\ M\ w\ \phi\} \mid FINITE\ s \wedge s \subseteq \Sigma\} \cup \{Y \mid only\_see\ M\ Y \in w \wedge Y \subseteq M.frame.world\}$. Similar as in the proof of `prop_2_59_i`, we check Δ has the finite intersection property. The only nontrivial thing to check is that for $a \in \{\{w \mid w \in M.frame.world \wedge \forall\phi.\phi \in s \implies satis\ M\ w\ \phi\} \mid FINITE\ s \wedge s \subseteq \Sigma\}$ and $b \in \{Y \mid only\_see\ M\ Y \in w \wedge Y \subseteq M.frame.world\}$, we have $a \cap b \neq \emptyset$.

Suppose $s \subseteq \Sigma$ is finite, and b is a set of worlds in $\mathfrak{M}$ such that only_see $\mathfrak{M}$ $b \in u$, we show $\{w \mid w \in M.frame.world \wedge \forall\phi.\phi \in s \implies satis\ M\ w\phi\} \cap b \neq \emptyset$. Recall Σ is finitely satisfiable in the set of successors of u, we have a world u'' such that UE_rel $\mathfrak{M}$ u u'' and satis (UE $\mathfrak{M}$) u'' phi for all $\phi \in s$, in other worlds, $\{w \mid w \in M.frame.world \wedge satis\ M\ w\ phi\} \in u''$ for all $\phi \in s$. Then $\{w \mid w \in M.frame.world \wedge \forall\phi.\phi \in$

$s \implies satis\ M\ w\phi\} \cap b \neq \emptyset$ is a big intersection of sets in u'', and hence is in u''. By `exercise_2_5_5` again, UE_rel $\mathfrak{M}\ u\ u''$ gives $\{Y \mid only\_see(Y) \in u \wedge Y \subseteq M.frame.world\} \subseteq u''$, so $b \in u''$ as well. As two elements in u'' has nonempty intersection, we are done.

Hence by `ultrafilter_theorem_corollary`, there exists an ultrafilter u' contains Δ, it is trivial to check u' is what we want.

Finally, as claimed at the begining of this chapter, we arrive at the characterisation of modal equivalence as bisimilarity somewhere else:

$$\begin{aligned}
&\vdash w \in \mathfrak{M}.\mathsf{frame.world} \wedge w' \in \mathfrak{M}'.\mathsf{frame.world} \Rightarrow \\
&\quad (\mathsf{modal_eq}\ \mathfrak{M}\ \mathfrak{M}'\ w\ w' \iff \\
&\qquad \mathsf{bisim_world}\ (\mathsf{UE}\ \mathfrak{M})\ (\mathsf{UE}\ \mathfrak{M}') \\
&\qquad\quad (\mathsf{principle_UF}\ w\ \mathfrak{M}.\mathsf{frame.world}) \\
&\qquad\quad (\mathsf{principle_UF}\ w'\ \mathfrak{M}'.\mathsf{frame.world}))
\end{aligned}$$

Proof:Bisimulation implies modal equivalence by `thm_2_20`. For the reverse direction, if $w \in \mathfrak{M}.\mathsf{frame.world}$ and $w' \in \mathfrak{M}'.\mathsf{frame.world}$ are modal equivalence, then $\mathsf{principle\_UF}\ w\ \mathfrak{M}.\mathsf{frame.world} \in (\mathsf{UE}\ \mathfrak{M}).\mathsf{frame.world}$ is modal equivalent to $\mathsf{principle\_UF}\ w'\ \mathfrak{M}'.\mathsf{frame.world} \in (\mathsf{UE}\ \mathfrak{M}').\mathsf{frame.world}$. As $\mathsf{UE}\ \mathfrak{M}$ and $\mathsf{UE}\ \mathfrak{M}'$ are m-saturated by `prop_2_61`, the result follows by `prop_2_54_DIST_TYPE`.

5 Invariant under bisimulations and simulations