

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 propositional 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.

```

α chap1$form =
  VAR α
  | DISJ (α chap1$form) (α chap1$form)
  | ⊥
  | (¬) (α chap1$form)
  | ◇ (α chap1$form)

```

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 $\alpha \rightarrow bool$. Note that we have the notion of the conjunction ‘ \wedge ’, the implication \rightarrow , and the truth ‘ \top ’, but they are not primitive, and defined as:

$$\vdash \text{AND } f_1 f_2 = \neg \text{DISJ } (\neg f_1) (\neg f_2) \quad \vdash (f_1 \rightarrow f_2) = \text{DISJ } (\neg f_1) f_2 \quad \vdash \text{TRUE} = \neg \perp$$

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:

$$\begin{aligned}
&\vdash (\forall p. \text{propform } (\text{VAR } p) \iff \top) \wedge \\
&\quad (\forall \phi_1 \phi_2. \text{propform } (\text{DISJ } \phi_1 \phi_2) \iff \text{propform } \phi_1 \wedge \text{propform } \phi_2) \wedge \\
&\quad (\forall f. \text{propform } (\neg f) \iff \neg \text{propform } f) \wedge (\text{propform } \perp \iff \top) \wedge \\
&\quad \forall f. \text{propform } (\Diamond f) \iff \text{F}
\end{aligned}$$

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 $\Phi \rightarrow bool$. 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 $\sigma : \alpha \rightarrow bool$ and an α propositional formula, and let it spit out the truth value of the propositional formula under the assignment σ .

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

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.

$$\alpha \text{ frame} = \langle | \text{world} : \alpha \rightarrow \text{bool}; \text{rel} : \alpha \rightarrow \alpha \rightarrow \text{bool} | \rangle$$

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 $\beta \rightarrow \alpha \rightarrow \text{bool}$. 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.

$$(\alpha, \beta) \text{ chap1\$model} = \langle | \text{frame} : \beta \text{ frame}; \text{valt} : \alpha \rightarrow \beta \rightarrow \text{bool} | \rangle$$

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 satisfaction.

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

(insert an example of model and satisfaction 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:

$$\vdash \text{equiv0 } ty_i f_1 f_2 \iff \forall \mathfrak{M} w. \text{satis } \mathfrak{M} w f_1 \iff \text{satis } \mathfrak{M} w f_2 \vdash \text{equiv0 } \mu \text{ equiv_on } s$$

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 $\mathfrak{M} w f_1 \iff \text{satis } \mathfrak{M} w f_2$` . 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 exists 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 situation 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:

$$\vdash \text{INFINITE } \mathcal{U}(\cdot; \beta) \Rightarrow (\text{equiv0 } \mu (\Diamond f) (\Diamond g) \iff \text{equiv0 } \mu f g)$$

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

$$\begin{aligned} \vdash (\forall p. \text{prop_letters } (\text{VAR } p) &= \{ p \}) \wedge \text{prop_letters } \perp = \emptyset \wedge \\ (\forall f_1 f_2. & \\ \text{prop_letters } (\text{DISJ } f_1 f_2) &= \\ \text{prop_letters } f_1 \cup \text{prop_letters } f_2) \wedge & \\ (\forall f. \text{prop_letters } (\neg f) &= \text{prop_letters } f) \wedge \\ \forall f. \text{prop_letters } (\Diamond f) &= \text{prop_letters } f \end{aligned}$$

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 :

$$\begin{aligned} \vdash \mathfrak{M}_1.\text{frame} = \mathfrak{M}_2.\text{frame} \Rightarrow \\ (\forall p. p \in \text{prop_letters } \phi \Rightarrow \mathfrak{M}_1.\text{valt } p = \mathfrak{M}_2.\text{valt } p) \Rightarrow \\ \forall w. w \in \mathfrak{M}_1.\text{frame.world} \Rightarrow (\text{satis } \mathfrak{M}_1 w \phi \iff \text{satis } \mathfrak{M}_2 w \phi) \end{aligned}$$

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:

$$\vdash \text{propform } f \wedge w \in \mathfrak{M}.\text{frame.world} \Rightarrow (\text{satis } \mathfrak{M} w f \iff \text{peval } (\lambda a. w \in \mathfrak{M}.\text{valt } a) f)$$

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

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

One may confused why are we allowed to write $\lambda a. w \in M.\text{valt } a) \cap s$ here, for $\lambda a. (w \in M.\text{valt } a)$ is a function but s is a set. This is because both the α -set s and the function $\lambda a. (w \in M.\text{valt } a)$ has type $\alpha \rightarrow \text{bool}$, so they can be feeded to the function \cap , which has type $(\alpha \rightarrow \text{bool}) \rightarrow (\alpha \rightarrow \text{bool}) \rightarrow (\alpha \rightarrow \text{bool})$. 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 $w \in M.\text{frame.world}, w' \in M'.\text{frame.world}$ are modal equivalent’, we mean they satisfies exactly the same modal formulas.

$$\vdash \text{modal_eq } \mathfrak{M} \mathfrak{M}' w w' \iff \forall \phi. \text{satis } \mathfrak{M} w \phi \iff \text{satis } \mathfrak{M}' w' \phi$$

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:

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

Disjoint union is defined as a function that takes a function that takes a function $f : \beta \rightarrow (\alpha, \gamma)$ -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 $w \in (f i).\text{frame.world}$. For a propositional letter p and a world (i, w) , we have $DU(f, dom).\text{valt } p(i, w)$ iff $(f i).\text{valt } p w$, and $DU(f, dom).\text{rel}(i, w)(i', w')$ iff $i = i'$ and $(f i).\text{frame.rel } w w'$, 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:

$$\begin{aligned}
&\vdash \text{FST } w \in dom \Rightarrow \\
&\quad (\text{satis}(f(\text{FST } w))(\text{SND } w) \text{ phi} \iff \\
&\quad \quad \text{satis}(DU(f, dom)) w \text{ phi})
\end{aligned}$$

Disjoint union is the operation we use when we want to expand our 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 .

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

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

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

$$\begin{aligned}
&\vdash \text{GENSUBMODEL } \mathfrak{M}_1 \mathfrak{M}_2 \iff \\
&\quad \text{SUBMODEL } \mathfrak{M}_1 \mathfrak{M}_2 \wedge \\
&\quad \forall w_1. \\
&\quad \quad w_1 \in \mathfrak{M}_1.\text{frame.world} \Rightarrow \\
&\quad \quad \forall w_2. \\
&\quad \quad \quad w_2 \in \mathfrak{M}_2.\text{frame.world} \wedge \mathfrak{M}_2.\text{frame.rel } w_1 w_2 \Rightarrow \\
&\quad \quad \quad w_2 \in \mathfrak{M}_1.\text{frame.world}
\end{aligned}$$

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 everything 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 satisfaction under taking generated submodels:

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

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’:

$$\begin{aligned} \vdash \text{hom } f \mathfrak{M}_1 \mathfrak{M}_2 &\iff \\ \forall w. & \\ w \in \mathfrak{M}_1.\text{frame.world} \Rightarrow & \\ f w \in \mathfrak{M}_2.\text{frame.world} \wedge & \\ (\forall p. w \in \mathfrak{M}_1.\text{valt } p \Rightarrow f w \in \mathfrak{M}_2.\text{valt } p) \wedge & \\ \forall u. & \\ u \in \mathfrak{M}_1.\text{frame.world} \Rightarrow & \\ \mathfrak{M}_1.\text{frame.rel } w u \Rightarrow & \\ \mathfrak{M}_2.\text{frame.rel } (f w) (f u) & \end{aligned}$$

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 guarantee 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 equivalence, we strengthen the condition of being a homomorphism, and what we get after strengthening is called a strong homomorphism:

$$\begin{aligned} \text{strong_hom } f \mathfrak{M}_1 \mathfrak{M}_2 &\stackrel{\text{def}}{=} \\ \forall w. & \\ w \in \mathfrak{M}_1.\text{frame.world} \Rightarrow & \\ f w \in \mathfrak{M}_2.\text{frame.world} \wedge & \\ (\forall p. w \in \mathfrak{M}_1.\text{valt } p \iff f w \in \mathfrak{M}_2.\text{valt } p) \wedge & \\ \forall u. & \\ u \in \mathfrak{M}_1.\text{frame.world} \Rightarrow & \\ (\mathfrak{M}_1.\text{frame.rel } w u \iff \mathfrak{M}_2.\text{frame.rel } (f w) (f u)) & \end{aligned}$$

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

$$\begin{aligned} \vdash \text{strong_hom } f \mathfrak{M} \mathfrak{M}' \wedge f w = w' \wedge w \in \mathfrak{M}.\text{frame.world} \wedge \\ \text{SURJ } f \mathfrak{M}.\text{frame.world } \mathfrak{M}'.\text{frame.world} \Rightarrow \\ \text{modal_eq } \mathfrak{M} \mathfrak{M}' w w' \end{aligned}$$

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:

$$\begin{aligned} \text{bounded_mor } f \mathfrak{M} \mathfrak{M}' &\stackrel{\text{def}}{=} \\ \forall w. & \\ w \in \mathfrak{M}.\text{frame.world} \Rightarrow & \\ f w \in \mathfrak{M}'.\text{frame.world} \wedge & \\ (\forall a. \text{satis } \mathfrak{M} w (\text{VAR } a) \iff \text{satis } \mathfrak{M}' (f w) (\text{VAR } a)) \wedge & \\ (\forall v. & \\ v \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } w v \Rightarrow & \\ \mathfrak{M}'.\text{frame.rel } (f w) (f v)) \wedge & \\ \forall v'. & \\ v' \in \mathfrak{M}'.\text{frame.world} \wedge \mathfrak{M}'.\text{frame.rel } (f w) v' \Rightarrow & \\ \exists v. v \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } w v \wedge f v = v' & \end{aligned}$$

The invariance result that bounded morphism gives is stated as:

$$\vdash \text{bounded_mor } f \mathfrak{M} \mathfrak{M}' \wedge w \in \mathfrak{M}.\text{frame.world} \Rightarrow \\ (\text{satis } \mathfrak{M} w \phi \iff \text{satis } \mathfrak{M}' (f w) \phi)$$

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:

$$\begin{aligned} \text{tree } S \ r &\stackrel{\text{def}}{=} \\ &r \in S.\text{world} \wedge \\ &(\forall t. t \in S.\text{world} \Rightarrow (\text{RESTRICT } S.\text{rel } S.\text{world})^* r \ t) \wedge \\ &(\forall r_0. r_0 \in S.\text{world} \Rightarrow \neg S.\text{rel } r_0 \ r) \wedge \\ &\forall t. t \in S.\text{world} \wedge t \neq r \Rightarrow \exists! t_0. t_0 \in S.\text{world} \wedge S.\text{rel } t_0 \ t \end{aligned}$$

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:

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

We read ‘tree $S \ r$ ’ as ‘ S is a tree with root r ’. By definition, any tree like model is rooted. An induction on reflexive and transitive closure proves a tree has no loop:

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

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

$$\begin{aligned} \vdash \text{satis } \mathfrak{M} \ w \ \phi &\Rightarrow \\ \exists \text{MODEL } s. \text{tree } \text{MODEL}.\text{frame } s &\wedge \text{satis } \text{MODEL } s \ \phi \end{aligned}$$

Proof: Suppose $\text{satis } \mathfrak{M} \ w \ \phi$. By the invariance result of rooted model, we have $\text{satis } \mathfrak{M}' \ w \ \phi$ 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 $[w = u_0; u_1; \dots; u_n]$ such that $n > 0$ and $M.\text{frame}.\text{rel } u_i \ u_{i+1}$ for all i . Define $M''.\text{frame}.\text{rel } [w; u_1; \dots; u_n] [w; v_1; \dots; v_m]$ iff $m = n + 1$, $m_i = v_i$ for $i \leq n$ and $M.\text{frame}.\text{rel } u_n \ v_m$. The valuation is given by $[w; u_1; \dots; u_n] \in M''.\text{valt } p$ iff $u_n \in M.\text{valt } p$. In the HOL, such a model looks like:

$$\begin{aligned} \text{bounded_preimage_rooted } \mathfrak{M} \ x &\stackrel{\text{def}}{=} \\ &<| \text{frame} := \\ &<| \text{world} := \\ &\{ l \mid \\ &\quad \text{HD } l = x \wedge \text{LENGTH } l > 0 \wedge \\ &\quad \forall m. \\ &\quad \quad m < \text{LENGTH } l - 1 \Rightarrow \\ &\quad \quad \text{RESTRICT } \mathfrak{M}.\text{frame}.\text{rel } \mathfrak{M}.\text{frame}.\text{world } (\text{EL } m \ l) \\ &\quad \quad (\text{EL } (m + 1) \ l) \} ; \\ &\text{rel} := \\ &\quad (\lambda l_1 \ l_2. \\ &\quad \quad \text{LENGTH } l_1 + 1 = \text{LENGTH } l_2 \wedge \\ &\quad \quad \text{RESTRICT } \mathfrak{M}.\text{frame}.\text{rel } \mathfrak{M}.\text{frame}.\text{world } (\text{LAST } l_1) \\ &\quad \quad (\text{LAST } l_2) \wedge \\ &\quad \quad \forall m. m < \text{LENGTH } l_1 \Rightarrow \text{EL } m \ l_1 = \text{EL } m \ l_2) | > ; \\ &\text{valt} := (\lambda v \ n. \mathfrak{M}.\text{valt } v \ (\text{LAST } n)) | > \end{aligned}$$

It is straightforward to check the map that sends a world in $\text{bounded_preimage_rooted } \mathfrak{M}' \ w$ 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 approaches 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:

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

If we have $\text{bisim } Z \mathfrak{M} \mathfrak{M}'$, 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:

$$\begin{aligned}
&\vdash i \in \text{dom} \wedge w \in (f i).\text{frame.world} \Rightarrow \\
&\quad \text{bisim_world } (f i) (\text{DU } (f, \text{dom})) w (i, w) \vdash \text{GENSUBMODEL } \mathfrak{M} \mathfrak{M}' \Rightarrow \\
&\quad \forall w. w \in \mathfrak{M}.\text{frame.world} \Rightarrow \text{bisim_world } \mathfrak{M} \mathfrak{M}' w w \vdash \text{bounded_mor_image } f \mathfrak{M} \mathfrak{M}' \Rightarrow \\
&\quad \forall w. w \in \mathfrak{M}.\text{frame.world} \Rightarrow \text{bisim_world } \mathfrak{M} \mathfrak{M}' w (f w)
\end{aligned}$$

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:

$$\vdash \text{bisim_world } \mathfrak{M} \mathfrak{M}' w w' \Rightarrow \text{modal_eq } \mathfrak{M} \mathfrak{M}' w w'$$

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 $w \in \mathfrak{M}.\text{frame.world}$, there are only finitely many worlds in \mathfrak{M} that is related to w .

$$\begin{aligned}
\text{image_finite } \mathfrak{M} &\stackrel{\text{def}}{=} \\
&\forall x. \\
&\quad x \in \mathfrak{M}.\text{frame.world} \Rightarrow \\
&\quad \text{FINITE } \{ y \mid y \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } x y \}
\end{aligned}$$

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

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

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 $\lambda n_1 n_2. \forall \phi. \text{satis } \mathfrak{M} n_1 \phi \iff \text{satis } \mathfrak{M}' n_2 \phi$ gives a bisimulation. The first clause is immediate to check. For the second one, assume $\text{modal_eq } \mathfrak{M} \mathfrak{M}' n_1 n_2$ and $\mathfrak{M}.\text{frame.rel } n_1 n'_1$ for some $n'_1 \in \mathfrak{M}.\text{frame.world}$, we prove the existence of the world $n'_2 \in \mathfrak{M}'.\text{frame.world}$ such that $\mathfrak{M}'.\text{frame.rel } n_2 n'_2$ and $\text{modal_eq } \mathfrak{M} \mathfrak{M}' n'_1 n'_2$. Suppose such a n'_2 does not exist, we derive a contradiction. Consider the set $S := \{ u' \in \mathfrak{M}'.\text{frame.world} \mid \mathfrak{M}'.\text{frame.rel } n_2 u' \}$, 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 $\Box \perp$ 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 ϕ such that $\text{satis } \mathfrak{M} n'_1 \phi$ but $\neg \text{satis } \mathfrak{M}' n'_2 \phi$. As the set S is finite, the set of such ϕ s is finite. Then we can take the conjunction of such ϕ s to obtain a formula ψ . Then we will have $\text{satis } \mathfrak{M} n_1 (\Box \psi)$ but $\neg \text{satis } \mathfrak{M} n_2 (\Box \psi)$.

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:

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

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 arbitrary model, then it can be satisfied on a finite model, where finite model means the finiteness of the set of worlds. We will discuss two methods for building finite models for satisfiable modal formulas, namely via filtration and selection.

3.1 Finite model property via filtration

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:

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

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.

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

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 .

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

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

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

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

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

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

$$\text{EC_CUS } \Sigma \mathfrak{M} w \stackrel{\text{def}}{=} \{ v \mid \text{REL_CUS } \Sigma \mathfrak{M} w v \}$$

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 $\text{EC_REP } \Sigma \mathfrak{M} w$ to be the representative from the equivalence class where w lives in, and we will use $\text{EC_REP_SET } \Sigma \mathfrak{M}$ to be the set of worlds for the filtrated model.

$$\text{EC_REP } \Sigma \mathfrak{M} w \stackrel{\text{def}}{=} \text{CHOICE } (\text{EC_CUS } \Sigma \mathfrak{M} w) \text{EC_REP_SET } \Sigma \mathfrak{M} \stackrel{\text{def}}{=} \{ n \mid \exists w. w \in \mathfrak{M}.\text{frame.world} \wedge n = \text{EC_REP } \Sigma \mathfrak{M} w \}$$

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

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

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:

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

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:

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

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:

$$\vdash \text{FINITE } \Sigma \wedge \text{filtration } \mathfrak{M} \Sigma FLT \Rightarrow \text{CARD } FLT.\text{frame.world} \leq 2 ** \text{CARD } \Sigma$$

Proof: As Σ is finite, it suffices to give an injection from the world set of FLT to $\text{POW } \Sigma$. 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:

$$\begin{aligned}
& \vdash phi \in \Sigma \Rightarrow \\
& \quad \forall w. \\
& \quad w \in \mathfrak{M}.\text{frame.world} \wedge \text{filtration } \mathfrak{M} \Sigma FLT \Rightarrow \\
& \quad (\text{satis } \mathfrak{M} w phi \iff \text{satis } FLT (\text{EC_REP } \Sigma \mathfrak{M} w) phi)
\end{aligned}$$

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

$$\begin{aligned}
& \vdash \text{satis } \mathfrak{M} w phi \Rightarrow \\
& \quad \exists \mathfrak{M}' w'. \\
& \quad w' \in \mathfrak{M}'.\text{frame.world} \wedge \text{satis } \mathfrak{M}' w' phi \wedge \\
& \quad \text{FINITE } \mathfrak{M}'.\text{frame.world}
\end{aligned}$$

3.2 Finite model property via selection

Another method to build finite models for an arbitrary 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:

$$\begin{aligned}
& \text{DEG } (\text{VAR } p) \stackrel{\text{def}}{=} 0 \\
& \text{DEG } \perp \stackrel{\text{def}}{=} 0 \\
& \text{DEG } (\neg \phi) \stackrel{\text{def}}{=} \text{DEG } \phi \\
& \text{DEG } (\text{DISJ } \phi_1 \phi_2) \stackrel{\text{def}}{=} \text{MAX } (\text{DEG } \phi_1) (\text{DEG } \phi_2) \\
& \text{DEG } (\Diamond \phi) \stackrel{\text{def}}{=} \text{DEG } \phi + 1
\end{aligned}$$

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

$$\vdash \text{DEG } f = 0 \iff \text{propform } f$$

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

$$\begin{aligned}
& \vdash \text{FINITE } s \wedge \text{INFINITE } \mathcal{U}(\beta) \Rightarrow \\
& \quad \forall n. \\
& \quad \text{FINITE} \\
& \quad (\{ f \mid \text{DEG } f \leq n \wedge \forall a. \text{VAR } a \in \text{subforms } f \Rightarrow a \in s \} // E \\
& \quad \mu)
\end{aligned}$$

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 obtain 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 $\lambda f. \{ \sigma \mid \text{peval } \sigma f \} \cap (\text{POW } s)$, which sends a formula f to the set $\{ \sigma \mid \text{peval } \sigma f \} \cap (\text{POW } s)$ of assignments of truth value on propositional letters in s that makes f holds, is an injection to the finite set $\text{POW } (\text{POW } (\text{POW } s))$. 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 $\text{POW } (\text{POW } s)$, so a equivalence class has image as a set of elements in $\text{POW } (\text{POW } s)$, which lies in $\text{POW } (\text{POW } (\text{POW } s))$.

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

$$\begin{aligned}
& \vdash x \in \\
& \quad \{ f \mid \text{propform } f \wedge \forall a. \text{VAR } a \in \text{subforms } f \Rightarrow a \in s \} // E \mu \wedge \\
& \quad \phi \in x \Rightarrow \\
& \quad \text{IMAGE } (\lambda f. \{ \sigma \mid \text{peval } \sigma f \} \cap \text{POW } s) x = \\
& \quad \{ \{ \sigma \mid \text{peval } \sigma \phi \} \cap \text{POW } s \}
\end{aligned}$$

Indeed, if two f_1 and f_2 formulas are equivalent, then for any assignment of truth value of propositional symbols, it makes f_1 true iff it makes f_2 true:

$$\begin{aligned} &\vdash \text{propform } f_1 \wedge \text{propform } f_2 \wedge \\ &\quad (\forall \mathfrak{M} w. \text{satis } \mathfrak{M} w f_1 \iff \text{satis } \mathfrak{M} w f_2) \Rightarrow \\ &\quad \forall \sigma. \text{peval } \sigma f_1 \iff \text{peval } \sigma f_2 \end{aligned}$$

It is because if we have a $\text{peval } \sigma f_1 \wedge \neg \text{peval } \sigma f_2$, 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 $\text{satis } \mathfrak{M} w f_1 \wedge \neg \text{satis } \mathfrak{M} w f_2$, a contradiction. As a consequence, we have:

$$\begin{aligned} &\vdash \text{propform } f_1 \wedge \text{propform } f_2 \wedge \text{equiv0 } \mu f_1 f_2 \Rightarrow \\ &\quad (\lambda f s. \text{peval } s f) f_1 = (\lambda f s. \text{peval } s f) f_2 \end{aligned}$$

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 $\text{IMAGE_peval_singlton_strengthen}$.

Recall we proved the condition for a propositional formula to hold as the lemma $\text{peval_satis_strengthen}$, it gives arise to this useful lemma:

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

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

We can now prove the injection:

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

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

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 $\Diamond \text{phi}$ which also only uses propositional letters in ss where $\text{DEG } \text{phi} \leq n$ 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 $\text{IBC } f s$ reads ' f is a boolean combination of elements in the set s ':

$$\begin{aligned} &\vdash (\forall f_1 f_2 s. \text{IBC } f_1 s \wedge \text{IBC } f_2 s \Rightarrow \text{IBC } (\text{DISJ } f_1 f_2) s) \wedge \\ &\quad (\forall s. \text{IBC } \perp s) \wedge (\forall f s. \text{IBC } f s \Rightarrow \text{IBC } (\neg f) s) \wedge \\ &\quad \forall f s. f \in s \Rightarrow \text{IBC } f s \vdash (\forall f_1 f_2 s. \text{IBC}' f_1 s \wedge \text{IBC}' f_2 s \Rightarrow \text{IBC}' (\text{DISJ } f_1 f_2) s) \wedge \\ &\quad (\forall s. \text{IBC}' \perp s) \wedge (\forall f s. \text{IBC}' f s \Rightarrow \text{IBC}' (\neg f) s) \wedge \\ &\quad (\forall f s. f \in s \Rightarrow \text{IBC}' f s) \Rightarrow \\ &\quad \forall a_0 a_1. \text{IBC } a_0 a_1 \Rightarrow \text{IBC}' a_0 a_1 \vdash \text{IBC } a_0 a_1 \iff \\ &\quad (\exists f_1 f_2. a_0 = \text{DISJ } f_1 f_2 \wedge \text{IBC } f_1 a_1 \wedge \text{IBC } f_2 a_1) \vee \\ &\quad a_0 = \perp \vee (\exists f. a_0 = \neg f \wedge \text{IBC } f a_1) \vee a_0 \in a_1 \end{aligned}$$

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

$$\begin{aligned} \vdash \text{DEG } x \leq n + 1 \wedge (\forall a. \text{VAR } a \in \text{subforms } x \Rightarrow a \in s) &\iff \\ \text{IBC } x & \\ (\{ \text{VAR } v \mid v \in s \} \cup & \\ \{ \Diamond \text{psi} \mid & \\ \text{DEG } \text{psi} \leq n \wedge \forall a. \text{VAR } a \in \text{subforms } \text{psi} \Rightarrow a \in s \}) & \end{aligned}$$

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

$$\vdash \text{IBC } f \ s \Rightarrow s = \emptyset \Rightarrow \text{equiv0 } \mu \ f \ \text{TRUE} \vee \text{equiv0 } \mu \ f \ \perp$$

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:

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

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 $\text{IMAGE CHOICE } (fs // E \ \mu)$ of representatives of the equivalence classes is finite. There is a surjection $\lambda s. \{y \mid \text{IBC } y \ fs \wedge !f. f \in s \implies \text{equiv0 } \mu \ y \ f\}$ from $\{f \mid \text{IBC } f \ (\text{IMAGE CHOICE } (fs // E \ \mu))\} // E \ \mu$ to $\{f \mid \text{IBC } f \ fs\} // E \ \mu$ 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:

$$\text{DNF_OF } f \ fs \stackrel{\text{def}}{=} \text{DISJ_OF } f \ \{ c \mid \text{CONJ_OF } c \ fs \}$$

where CONJ_OF is defined by:

$$\begin{aligned} &\frac{c = c_0 \vee c = \neg c_0}{\text{CONJ_OF } c \ \{ c_0 \}} \\ &\frac{f_1 = f_0 \vee f_1 = \neg f_0 \quad f_0 \in fs \quad \text{CONJ_OF } f_2 \ (fs \ \text{DELETE } f_0)}{\text{CONJ_OF } (\text{AND } f_1 \ f_2) \ fs} \end{aligned}$$

and for a formula f , we say $\text{DISJ_OF } f \ fs$ when f is either the ' \perp ' or a formula such that $\text{DISJ_OF0 } f \ fs$

$$\begin{aligned} \vdash (\forall f \ fs. f \in fs \Rightarrow \text{DISJ_OF0 } f \ fs) \wedge & \\ \forall f_1 \ f_2 \ fs. & \\ f_1 \in fs \wedge \text{DISJ_OF0 } f_2 \ (fs \ \text{DELETE } f_1) \Rightarrow & \\ \text{DISJ_OF0 } (\text{DISJ } f_1 \ f_2) \ fs \vdash (\forall f \ fs. f \in fs \Rightarrow \text{DISJ_OF0 } f \ fs) \wedge & \\ (\forall f_1 \ f_2 \ fs. & \\ f_1 \in fs \wedge \text{DISJ_OF0 } f_2 \ (fs \ \text{DELETE } f_1) \Rightarrow & \\ \text{DISJ_OF0 } (\text{DISJ } f_1 \ f_2) \ fs) \Rightarrow & \\ \forall a_0 \ a_1. \text{DISJ_OF0 } a_0 \ a_1 \Rightarrow \text{DISJ_OF0 } a_0 \ a_1 \vdash \text{DISJ_OF0 } a_0 \ a_1 \iff & \\ a_0 \in a_1 \vee & \\ \exists f_1 \ f_2. & \\ a_0 = \text{DISJ } f_1 \ f_2 \wedge f_1 \in a_1 \wedge & \\ \text{DISJ_OF0 } f_2 \ (a_1 \ \text{DELETE } f_1) \text{DISJ_OF } f \ fs \stackrel{\text{def}}{=} f = \perp \vee \text{DISJ_OF0 } f \ fs & \end{aligned}$$

As an example, the four CONJ_OF formulas on the set $\{f_1, f_2\}$ are $\text{AND } f_1 \ f_2$, $\text{AND } (\neg f_1) \ f_2$ $\text{AND } f_1 \ (\neg f_2)$ and $\text{AND } (\neg f_1) \ (\neg f_2)$. For another example, consider the set $fs := \{f_1, f_2, f_3\}$ of three formulas, we can say $\text{CONJ_OF } (\text{AND } f_3 \ (\text{AND } f_1 \ f_2)) \ fs$, $\text{CONJ_OF } (\text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3)) \ fs$, $\text{CONJ_OF } (\text{AND } (\neg f_3) \ (\text{AND } f_2 \ f_1)) \ fs$ and so on. But it is not the fact that $\text{CONJ_OF } f_1 \ fs$, since from the definition, any element of fs or its negation must appear exactly once in a CONJ_OF formula on fs . Let $CO := \{f \mid \text{CONJ_OF } f \ fs\}$, then we can say DISJ_OF

$(\text{DISJ } (\text{AND } f_3 \ (\text{AND } f_1 \ f_2)) \ (\text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3))) \ CO$, $\text{DISJ_OF } (\text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3)) \ CO$ and so on, so both these two formulas are disjunction normal form on fs , but we cannot say DISJ_OF $(\text{DISJ } (\text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3)) \ (\text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3))) \ CO$ or DISJ_OF $(\text{DISJ } (\text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3)) \ (\neg \text{AND } f_2 \ (\text{AND } (\neg f_1) \ f_3))) \ CO$, 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:

$$\vdash \text{FINITE } s \Rightarrow \text{FINITE } \{f \mid \text{CONJ_OF } f \ s\} \quad \vdash \text{FINITE } s \Rightarrow \text{FINITE } \{f \mid \text{DISJ_OF0 } f \ s\} \quad \vdash \text{FINITE } fs \Rightarrow \text{FINITE } \{f \mid \text{DNF_OF } f \ fs\}$$

Once we prove:

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

we get `FINITE_FINITE_IBC`, which leads to a proof of the step case of `prop_2_29_strengthen` goes as follows: Suppose $\{f \mid \text{DEG } f \leq n \wedge \forall a. \text{VAR } a \in \text{subforms } f \Rightarrow a \in s\} / E \ \mu$ is finite and let $A = (\{\text{VAR } v \mid v \in s\} \cup \{\Diamond \text{psi} \mid \text{DEG } \text{psi} \leq n \wedge \forall a. (\text{VAR } a) \in \text{subforms } \text{psi} \Rightarrow a \in s\})$. By `DEG_IBC_strengthen`, we have $B := \{f \mid \text{DEG } f \leq n+1 \wedge \forall a. \text{VAR } a \in \text{subforms } f \Rightarrow a \in s\} = \{\text{phi} \mid \text{IBC } \text{phi} \ A\}$, we prove B is finite up to equivalence. If $A = \emptyset$, then by `IBC_EMPTY_lemma`, B contains only two non-equivalent formulas \perp 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` $\mu \ (\Diamond f) \ (\Diamond g)$ iff `equiv0` $\mu \ f \ g$, 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 \perp 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 f_1, f_2 are boolean combination of the set fs which are equivalent to p_1, p_2 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 \perp 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:

$$\begin{aligned} &\vdash \text{FINITE } fs \wedge fs \neq \emptyset \Rightarrow \\ &\quad \forall f. f \in fs \Rightarrow \exists p. \text{DNF_OF } p \ fs \wedge \text{equiv0 } \mu \ f \ p \end{aligned}$$

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 satisfaction of each disjuncts of p to imply the satisfaction 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 $f_1 \wedge ((f_2 \wedge f_3) \vee (\neg f_2 \wedge f_3) \vee (f_2 \wedge \neg f_3) \vee (\neg f_2 \wedge \neg f_3))$, 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:

$$\begin{aligned}
\text{negf } (f, T) &\stackrel{\text{def}}{=} f \\
\text{negf } (f, F) &\stackrel{\text{def}}{=} \neg f \text{lit_list_to_form } [] \stackrel{\text{def}}{=} \text{TRUE} \\
\text{lit_list_to_form } [fb] &\stackrel{\text{def}}{=} \text{negf } fb \\
\text{lit_list_to_form } (fb :: v_2 :: v_3) &\stackrel{\text{def}}{=} \\
&\quad \text{AND } (\text{negf } fb) (\text{lit_list_to_form } (v_2 :: v_3)) \text{lit_list_to_form2 } [] \stackrel{\text{def}}{=} \perp \\
\text{lit_list_to_form2 } [fb] &\stackrel{\text{def}}{=} fb \\
\text{lit_list_to_form2 } (fb :: v_2 :: v_3) &\stackrel{\text{def}}{=} \\
&\quad \text{DISJ } fb (\text{lit_list_to_form2 } (v_2 :: v_3))
\end{aligned}$$

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 $\text{set } (\text{MAP FST } l) = fs$ and $\text{ALL_DISTINCT } (\text{MAP FST } l)$, where MAP takes a function f and a list $[a_1; \dots, a_n]$, and return the list $[f(a_1); \dots; f(a_n)]$. 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.

$$\begin{aligned}
&\vdash l \neq [] \Rightarrow \\
&\quad \text{set } (\text{MAP FST } l) = fs \wedge \text{ALL_DISTINCT } (\text{MAP FST } l) \Rightarrow \\
&\quad \text{CONJ_OF } (\text{lit_list_to_form } l) fs \vdash \text{CONJ_OF } f fs \Rightarrow \\
&\quad \exists l. \\
&\quad \quad \text{set } (\text{MAP FST } l) = fs \wedge \text{ALL_DISTINCT } (\text{MAP FST } l) \wedge \\
&\quad \quad f = \text{lit_list_to_form } l
\end{aligned}$$

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

$$\begin{aligned}
&\vdash l \neq [] \wedge \text{set } l \subseteq fs \wedge \text{ALL_DISTINCT } l \Rightarrow \\
&\quad \text{DISJ_OF0 } (\text{lit_list_to_form2 } l) fs \vdash \text{DISJ_OF0 } f fs \Rightarrow \\
&\quad \exists l. \\
&\quad \quad l \neq [] \wedge \text{set } l \subseteq fs \wedge f = \text{lit_list_to_form2 } l \wedge \\
&\quad \quad \text{ALL_DISTINCT } l
\end{aligned}$$

Put the two things together, we can build DNF_OF formulas using lit_list_to_form and $\text{list_list_to_form2}$

$$\begin{aligned}
&\vdash ld \neq [] \wedge \text{ALL_DISTINCT } ld \wedge \\
&\quad (\forall d. \\
&\quad \quad \text{MEM } d ld \Rightarrow \\
&\quad \quad \exists lc. \\
&\quad \quad \quad lc \neq [] \wedge d = \text{lit_list_to_form } lc \wedge \\
&\quad \quad \quad \text{set } (\text{MAP FST } lc) = fs \wedge \text{ALL_DISTINCT } (\text{MAP FST } lc)) \Rightarrow \\
&\quad \text{DNF_OF } (\text{lit_list_to_form2 } ld) fs
\end{aligned}$$

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

$$\begin{aligned}
&\vdash l \neq [] \Rightarrow \\
&\quad \text{equiv0 } \mu (\text{AND } e (\text{lit_list_to_form2 } l)) \\
&\quad (\text{lit_list_to_form2 } (\text{MAP } (\lambda a. \text{AND } e a) l)) \vdash \text{set } l_1 = \text{set } l_2 \Rightarrow \\
&\quad \text{equiv0 } \mu (\text{lit_list_to_form2 } l_1) (\text{lit_list_to_form2 } l_2)
\end{aligned}$$

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

$$\begin{aligned}
&\vdash \text{FINITE } fs \Rightarrow \\
&\quad fs \neq \emptyset \Rightarrow \\
&\quad \text{equiv0 } \mu \\
&\quad \quad (\text{lit_list_to_form2 } (\text{SET_TO_LIST } \{ c \mid \text{CONJ_OF } c fs \})) \\
&\quad \quad \text{TRUE}
\end{aligned}$$

Now we launch on the proof of `IBC_DNF_EXISTS_case4`:

Given an $f \in fs$, the desired p is given by $\phi := \text{lit_list_to_form}(\text{SET_TO_LIST } \{f \wedge c \mid c \mid \text{CONJ_OF } c (fs/\{f\})\})$. 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:

$$\begin{aligned} & \text{lit_list_to_form}(\text{SET_TO_LIST } \{f \wedge c \mid c \mid \text{CONJ_OF } c (fs/\{f\})\}) \\ &= \text{lit_list_to_form}(\text{MAP } (\lambda a. \text{AND } f a)(\text{SET_TO_LIST } \{c \mid \text{CONJ_OF } c (fs/\{f\})\})) \\ &\stackrel{\text{list_demorgan}}{=} f \wedge \text{lit_list_to_form}(\text{SET_TO_LIST } \{c \mid \text{CONJ_OF } c (fs/\{f\})\}) \\ &\stackrel{\text{ALL_POSSIBLE_VALUE_TRUE}}{=} f \wedge \top \\ &\equiv f \end{aligned}$$

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.

$$\begin{aligned} & \vdash \text{DNF_OF } p1 fs \wedge \text{DNF_OF } p2 fs \Rightarrow \\ & \exists f. \text{DNF_OF } f fs \wedge \text{equiv0 } \mu f (\text{DISJ } p1 p2) \end{aligned}$$

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

$$\begin{aligned} & \vdash \text{DISJ_OF0 } p1 fs \Rightarrow \\ & \quad \forall p2. \\ & \quad \text{DISJ_OF0 } p2 fs \Rightarrow \\ & \quad \exists f. \text{DISJ_OF0 } f fs \wedge \text{equiv0 } \mu f (\text{DISJ } p1 p2) \end{aligned}$$

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` $p1 (fs \text{ DELETE } f_1)$ and `DISJ_OF0` $p2 fs$, we are asked to prove:

$$\begin{aligned} & \exists f. \\ & \text{DISJ_OF0 } f fs \wedge \text{equiv0 } \mu f (\text{DISJ } (\text{DISJ } f_1 p1) p2) \text{ from the inductive hypothesis} \\ & \quad \forall p2. \\ & \text{DISJ_OF0 } p2 (fs \text{ DELETE } f_1) \Rightarrow \\ & \exists f. \text{DISJ_OF0 } f (fs \text{ DELETE } f_1) \wedge \text{equiv0 } \mu f (\text{DISJ } p1 p2) \end{aligned}$$

If `DISJ_OF0` $p2 (fs \text{ DELETE } f_1)$, then we are done by inductive hypothesis, otherwise we need a trick: If `DISJ_OF0` $p2 fs$ but $\neg \text{DISJ_OF0 } p2 (fs \text{ DELETE } f_1)$, it must be the case that f_1 appears in $p2$, hence we can extract f_1 out of $p2$ to split $p2$ as a disjunction $t \vee f_1$, using the following lemma proved by induction on `DISJ_OF0`:

$$\begin{aligned} & \vdash \text{DISJ_OF0 } f fs \Rightarrow \\ & \quad \forall t. \\ & \quad t \in fs \wedge \neg \text{DISJ_OF0 } f (fs \text{ DELETE } t) \Rightarrow \\ & \quad \exists p. \text{DISJ_OF } p (fs \text{ DELETE } t) \wedge \text{equiv0 } \mu f (\text{DISJ } t p) \end{aligned}$$

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:

$$\begin{aligned} & \vdash \text{DNF_OF } p fs \wedge \text{FINITE } fs \wedge fs \neq \emptyset \Rightarrow \\ & \exists f. \text{DNF_OF } f fs \wedge \text{equiv0 } \mu (\neg p) f \end{aligned}$$

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 $ss \subseteq \{c \mid \text{CONJ_OF } c fs\}$. Since none of these conjunctions is satisfied, it must be the case that the satisfied conjunction is in $\{c \mid \text{CONJ_OF } c fs\}/ss$. 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 complement 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 satisfaction of a literal as:

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

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

$$\begin{aligned} \text{is_lset } c fs &\stackrel{\text{def}}{=} \\ \text{FINITE } fs \wedge \text{FINITE } c \wedge \text{CARD } c &= \text{CARD } fs \wedge \text{IMAGE FST } c = fs \end{aligned}$$

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.

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

If two `lsets` 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:

$$\begin{aligned} \vdash \text{is_lset } c_1 fs \wedge \text{is_lset } c_2 fs \wedge c_1 \neq c_2 &\Rightarrow \\ \forall \mathfrak{M} w. w \in \mathfrak{M}.\text{frame.world} &\Rightarrow \neg \text{csatis } \mathfrak{M} w (c_1 \cup c_2) \end{aligned}$$

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:

$$\begin{aligned} \vdash \text{CONJ_OF } c fs &\Rightarrow \\ \exists ls. & \\ \text{is_lset } ls fs \wedge & \\ \forall \mathfrak{M} w. & \\ w \in \mathfrak{M}.\text{frame.world} &\Rightarrow \\ (\text{csatis } \mathfrak{M} w ls \iff \text{satis } \mathfrak{M} w c) &\vdash \text{is_lset } c fs \wedge c \neq \emptyset \Rightarrow \\ \exists f. & \\ (\forall \mathfrak{M} w. & \\ w \in \mathfrak{M}.\text{frame.world} &\Rightarrow \\ (\text{satis } \mathfrak{M} w f \iff \text{csatis } \mathfrak{M} w c)) &\wedge \text{CONJ_OF } f fs \end{aligned}$$

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 satisfaction of `DISJ_OF0` formulas is captured by d -satisfaction:

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

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

$$\begin{aligned} \vdash \text{FINITE } fs &\Rightarrow \\ fs \neq \emptyset &\Rightarrow \\ \forall \mathfrak{M} w. w \in \mathfrak{M}.\text{frame.world} &\Rightarrow \text{dsatis } \mathfrak{M} w \{ c \mid \text{is_lset } c fs \} \end{aligned}$$

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.

$$\begin{aligned} \vdash \text{FINITE } fs \wedge fs \neq \emptyset \wedge (\forall c. c \in cs \Rightarrow \text{is_lset } c fs) &\Rightarrow \\ \forall \mathfrak{M} w. & \\ w \in \mathfrak{M}.\text{frame.world} &\Rightarrow \\ (\text{dsatis } \mathfrak{M} w cs \iff & \\ \neg \text{dsatis } \mathfrak{M} w (\{ c \mid \text{is_lset } c fs \} \text{ DIFF } cs)) & \end{aligned}$$

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

And we have a correspondence of set of lsets and DISJ_OF0 formulas.

$$\begin{aligned} \vdash \text{DISJ_OF0 } d fs &\Rightarrow \\ \forall fs_0. & \\ fs \subseteq \{c \mid \text{CONJ_OF } c fs_0\} &\Rightarrow \\ \exists cs. & \\ (\forall c. c \in cs \Rightarrow \text{is_lset } c fs_0) \wedge & \\ \forall \mathfrak{M} w. & \\ w \in \mathfrak{M}.\text{frame.world} \Rightarrow & \\ (\text{satis } \mathfrak{M} w d \iff \exists c. c \in cs \wedge \text{csatis } \mathfrak{M} w c) & \end{aligned}$$

Altogether, there are the correspondence results of set of lsets 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 lsets .

$$\begin{aligned} \vdash \text{DNF_OF } d fs &\Rightarrow \\ \exists cs. & \\ (\forall c. c \in cs \Rightarrow \text{is_lset } c fs) \wedge & \\ \forall \mathfrak{M} w. & \\ w \in \mathfrak{M}.\text{frame.world} \Rightarrow & \\ (\text{satis } \mathfrak{M} w d \iff \exists c. c \in cs \wedge \text{csatis } \mathfrak{M} w c) & \end{aligned}$$

The second one says each set of lsets gives a DNF_OF formula:

$$\begin{aligned} \vdash \text{FINITE } s &\Rightarrow \\ \forall fs. & \\ \text{FINITE } fs \wedge fs \neq \emptyset \Rightarrow & \\ (\forall c. c \in s \Rightarrow \text{is_lset } c fs) \Rightarrow & \\ \exists f. & \\ (\forall \mathfrak{M} w. & \\ w \in \mathfrak{M}.\text{frame.world} \Rightarrow & \\ (\text{satis } \mathfrak{M} w f \iff \text{dsatis } \mathfrak{M} w s)) \wedge \text{DNF_OF } f fs & \end{aligned}$$

Proof: The proof is by induction on $\text{FINITE } s$. For the base case that $s = \emptyset$, the required formula f is obviously \perp . Suppose for s a finite set of lsets and there exists a formula f in disjunction normal form such that $\text{satis } \mathfrak{M} w f \iff \text{dsatis } \mathfrak{M} w s$ for any model \mathfrak{M} and any $w \in \mathfrak{M}.\text{frame.world}$, we find a formula g such that $\text{satis } \mathfrak{M} w g \iff \text{dsatis } \mathfrak{M} w (e \text{ INSERT } s)$. By $\text{is_lset_CONJ_OF_EXISTS}$, there is a CONJ_OF formula f' on fs with the property that $\text{satis } \mathfrak{M} w f' \iff \text{csatis } \mathfrak{M} w e$ for arbitrary \mathfrak{M} . If f is equivalent to \perp , then f' is the g we want, and vice versa. If both of them are not equivalent to \perp , 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 $\text{DISJ_OF0 } f \{c \mid \text{CONJ_OF } c fs\}/\{f'\}$. Suppose not, then there exists a p such that $\text{DISJ_OF0 } p \{c \mid \text{CONJ_OF } c fs\}/\{f'\}$ and $\text{equiv0 } \mu f (\text{DISJ } f' p)$ 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 $w \in \mathfrak{M}.\text{frame.world}$ such that $\text{csatis } \mathfrak{M} w e$ but $\neg \text{dsatis } \mathfrak{M} w s$. The claim is any world c -satisfying e cannot d -satisfy s , otherwise it contradicts $\text{dsatis_is_lset_complement}$. Also since f' is not equivalent to \perp , a model satisfying f' exists, so we are done.

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

Proof: As $\text{DNF_OF } p fs$, by DNF_OF_cset we obtain a $\text{lset } cs$ such that $\text{satis } \mathfrak{M} w p \iff \text{dsatis } \mathfrak{M} w cs$ for any \mathfrak{M} and w . Consider its complement $\{c \mid \text{is_lset } c fs\}/cs$, by $\text{is_lset_DNF_OF_EXISTS}$ we obtain a formula f such that $\text{DNF_OF } f fs$ and $\text{satis } \mathfrak{M} w f$ iff $\text{dsatis } \mathfrak{M} w \{c \mid \text{is_lset } c fs\}/cs$ for any \mathfrak{M} and w . By $\text{dsatis_is_lset_complement}$, $\text{dsatis } \mathfrak{M} w \{c \mid \text{is_lset } c fs\}/cs$ iff $\neg \text{dsatis } \mathfrak{M} w cs$, 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 ‘satisfying exactly the same formulas’, but when we are building a finite model for a formula ϕ , we do not care about the satisfaction of any formula of degree above $\text{DEG } \phi$, since such formula cannot affect the satisfaction of ϕ . Therefore, a finite approximation of bisimulation is enough, a finite approximation of depth n is called an

n -bisimulation. Let $w \in \mathfrak{M}.\text{frame.world}$ and $w' \in \mathfrak{M}'.\text{frame.world}$, w and w' are n -bisimilar if there exists a sequence of relations $Z_n \subseteq \dots \subseteq Z_0$ such that:

- w and w' are related by Z_n
- If $v \in \mathfrak{M}.\text{frame.world}$ and $v' \in \mathfrak{M}'.\text{frame.world}$ are related by Z_0 , then v and v' satisfy the same propositional letters.
- If $v \in \mathfrak{M}.\text{frame.world}$ and $v' \in \mathfrak{M}'.\text{frame.world}$ are related by Z_{i+1} and we have $\mathfrak{M}.\text{frame.rel } v \ u$ for $u \in \mathfrak{M}.\text{frame.world}$, then there exists $u' \in \mathfrak{M}'.\text{frame.world}$ such that $\mathfrak{M}'.\text{frame.rel } v' \ u'$ with u and u' related by Z_i .
- If $v \in \mathfrak{M}.\text{frame.world}$ and $v' \in \mathfrak{M}'.\text{frame.world}$ are related by Z_{i+1} and we have $\mathfrak{M}'.\text{frame.rel } v' \ u'$ for $u' \in \mathfrak{M}'.\text{frame.world}$, then there exists $u \in \mathfrak{M}.\text{frame.world}$ such that $\mathfrak{M}.\text{frame.rel } v \ u$ 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 $f : \text{num} \rightarrow \beta \rightarrow \gamma \rightarrow \text{bool}$. Such a function assigns each natural number a relation, so the $f \ i$ is the relation Z_i in the usual mathematical definition, and $\text{nbisim } \mathfrak{M} \ \mathfrak{M}' \ f \ n \ w \ w'$ 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 .

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

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

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

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 $\lambda \ n \ n_1 \ n_2$.

$\forall \text{phi}. \text{DEG } \text{phi} \leq n \Rightarrow (\text{satis } \mathfrak{M} \ n_1 \ \text{phi} \iff \text{satis } \mathfrak{M}' \ n_2 \ \text{phi})$ gives an n -bisimulation relation linking w and w' :

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

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 $w \in \mathfrak{M}.\text{frame.world}$, 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**:

$$\begin{aligned}
& \vdash (\forall n. \text{heightLE } \mathfrak{M} x \mathfrak{M}' x n) \wedge \\
& \quad \forall n v. \\
& \quad \quad v \in \mathfrak{M}.\text{frame.world} \wedge \\
& \quad \quad (\exists w. \\
& \quad \quad \quad w \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } w v \wedge \\
& \quad \quad \quad \text{heightLE } \mathfrak{M} x \mathfrak{M}' w n) \Rightarrow \\
& \quad \quad \text{heightLE } \mathfrak{M} x \mathfrak{M}' v (n + 1) \vdash (\forall n. \text{heightLE}' x n) \wedge \\
& (\forall n v. \\
& \quad v \in \mathfrak{M}.\text{frame.world} \wedge \\
& \quad (\exists w. \\
& \quad \quad w \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } w v \wedge \\
& \quad \quad \text{heightLE}' w n) \Rightarrow \\
& \quad \quad \text{heightLE}' v (n + 1)) \Rightarrow \\
& \forall a_0 a_1. \text{heightLE } \mathfrak{M} x \mathfrak{M}' a_0 a_1 \Rightarrow \text{heightLE}' a_0 a_1 \vdash \text{heightLE } \mathfrak{M} x \mathfrak{M}' a_0 a_1 \iff \\
& \quad a_0 = x \vee \\
& \quad \exists n. \\
& \quad \quad a_1 = n + 1 \wedge a_0 \in \mathfrak{M}.\text{frame.world} \wedge \\
& \quad \quad \exists w. \\
& \quad \quad \quad w \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } w a_0 \wedge \\
& \quad \quad \quad \text{heightLE } \mathfrak{M} x \mathfrak{M}' w n
\end{aligned}$$

Recall how we defined rooted model: $\text{rooted_model } \mathfrak{M} x \mathfrak{M}'$ 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. $\text{heightLE } \mathfrak{M} x \mathfrak{M}' w n$ 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 $\text{heightLE } \mathfrak{M} x \mathfrak{M}' w n$. And a height of model is the maximum n such that there is a world of height n in the model.

$$\begin{aligned}
\text{height } \mathfrak{M} x \mathfrak{M}' w & \stackrel{\text{def}}{=} \text{MIN_SET } \{ n \mid \text{heightLE } \mathfrak{M} x \mathfrak{M}' w n \} \\
\text{model_height } \mathfrak{M} x \mathfrak{M}' & \stackrel{\text{def}}{=} \text{MAX_SET } \{ n \mid (\exists w. w \in \mathfrak{M}.\text{frame.world} \wedge \text{height } \mathfrak{M} x \mathfrak{M}' w = n) \}
\end{aligned}$$

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 $w \in \mathfrak{M}.\text{frame.world}$ has height n , then any world $w' \in \mathfrak{M}.\text{frame.world}$ such that $\mathfrak{M}.\text{frame.rel } w w'$ will have height $n + 1$, this is proved using the induction principle RTC_INDUCT_RIGHT1 :

$$\begin{aligned}
& \vdash \text{tree } \mathfrak{M}.\text{frame } x \Rightarrow \\
& \quad \forall w. \\
& \quad \quad w \in \mathfrak{M}.\text{frame.world} \wedge \text{height } \mathfrak{M} x \mathfrak{M} w = n \Rightarrow \\
& \quad \quad \forall v. \\
& \quad \quad \quad \mathfrak{M}.\text{frame.rel } w v \wedge v \in \mathfrak{M}.\text{frame.world} \Rightarrow \\
& \quad \quad \quad \text{height } \mathfrak{M} x \mathfrak{M} v = n + 1
\end{aligned}$$

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:

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

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:

$$\begin{aligned}
& \vdash \text{rooted_model } \mathfrak{M} x \mathfrak{M}' \Rightarrow \\
& \quad \forall w. \\
& \quad \quad w \in (\text{hrestriction } \mathfrak{M} x \mathfrak{M}' k).\text{frame.world} \Rightarrow \\
& \quad \quad \exists f. \\
& \quad \quad \quad \text{nbisim } (\text{hrestriction } \mathfrak{M} x \mathfrak{M}' k) \mathfrak{M} f \\
& \quad \quad \quad (k - \text{height } \mathfrak{M} x \mathfrak{M}' w) w w
\end{aligned}$$

Proof: The n -bisimilar relation is give as $\lambda n w_1 w_2. w_1 = w_2 \wedge \text{height } \mathfrak{M} x \mathfrak{M}' w_1 \leq k - n$.
Now we can start building a finite model via selection:

$$\begin{aligned} & \vdash \text{satis } \mathfrak{M}_1 w_1 \text{ phi} \Rightarrow \\ & \quad \exists FM v. \\ & \quad \text{FINITE } FM.\text{frame.world} \wedge v \in FM.\text{frame.world} \wedge \\ & \quad \text{satis } FM v \text{ phi} \end{aligned}$$

Proof: Suppose $\text{satis } \mathfrak{M}_1 w_1 \text{ phi}$ where $w_1 : \beta, \phi : \alpha \text{ form}$, 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 $\beta \text{ list}$, so all the lemmas proved before with a infinite universe assumption applies for any model with its world set a subset of $\mathfrak{M}_2.\text{frame.world}$. Define $M_3 := \text{restriction } \mathfrak{M}_2 w_2 \mathfrak{M}_2 k$ to be the restriction of \mathfrak{M}_2 to height k , then M_3 is rooted and there is a nbisim $M_3 \mathfrak{M}_2 f k w_2 w_2$ by `lemma_2_33`. Hence $\text{satis } M_3 w_2 \text{ phi}$ 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 := M3.frame.world; rel := M3.frame.rel|>`;

`valt :=`

`(\lambda p v. if VAR p \in subforms phi then M3.valt p v else F)|>`, and still have $\text{satis } M'_3 w_2 \text{ phi}$. 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 $\text{distfp} := \{f \mid \text{DEG } f \leq k \wedge \forall a. \text{VAR } a \in \text{subforms } f \implies a \in s\} // E \mu$ 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 $rs := (\text{IMAGE CHOICE}((\text{IMAGE } (\lambda s.s \cap \{d \mid ?d0.d = \Diamond d0\}) \text{ distfp}) \text{ DELETE } \{\}))$ 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, \dots, 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, \dots, 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 $\text{satis } M'_3 w_2 (\Diamond \phi)$, pick a world $u \in M'_3.\text{frame.world}$ such that $M'_3.\text{frame.rel } v u$ and $\text{satis } M'_3 u \text{ phi}$. 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:

$$\begin{aligned} & \text{PRIM_REC } \{ w_2 \} \\ & (\lambda s_0 n. \\ & \quad \{ \text{CHOICE } \text{uset} \mid \\ & \quad \quad \exists \text{phi } v. \\ & \quad \quad \quad \text{satis } M'_3 v (\Diamond \text{phi}) \wedge \\ & \quad \quad \quad \Diamond \text{phi} \in \text{IMAGE CHOICE } (\text{IMAGE } (\lambda s. s \cap \{ d \mid \exists d_0. d = \Diamond d_0 \}) \text{ distfp } \text{DELETE } \emptyset) \wedge \\ & \quad \quad \quad v \in s_0 \wedge \text{uset} = \{ u \mid M'_3.\text{frame.rel } v u \wedge u \in M'_3.\text{frame.world} \wedge \text{satis } M'_3 u \text{ phi} \} \}) \end{aligned}$$

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 := W4; rel := M3.frame.rel|>`;

`valt := M'_3.valt|>` To prove $\text{satis } M_4 w_2 \text{ phi}$, it suffices to give a k -bisimulation between M_4 and M'_3 relating w_2 to itself. Such a k -bisimulation is: $\lambda n a_1 a_2.$

$a_1 \in M_4.\text{frame.world} \wedge a_2 \in M'_3.\text{frame.world} \wedge$

$\text{height } M'_3 w_2 M'_3 a_1 = \text{height } M'_3 w_2 M'_3 a_2 \wedge$

$\text{height } M'_3 w_2 M'_3 a_1 \leq k - n \wedge$

$\forall \text{phi}.$

$\text{DEG } \text{phi} \leq n \wedge (\forall a. \text{VAR } a \in \text{subforms } \text{phi} \implies a \in s) \implies$

$(\text{satis } M'_3 a_1 \text{ phi} \iff \text{satis } M'_3 a_2 \text{ phi})$ 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 $f(t_1, \dots, t_n)$, 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:

$$\text{term} = \text{fV num} \mid \text{Fn num (term list)}$$

Hence our terms will look like $\text{fV } 6$, $\text{Fn } 1 [\text{fV } 1; \text{fV } 2]$, $\text{Fn } 2 [\text{Fn } 0 []]$, 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 :

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

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:

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

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

$$\begin{aligned} \text{LIST_UNION } [] &\stackrel{\text{def}}{=} \emptyset \\ \text{LIST_UNION (} h :: t) &\stackrel{\text{def}}{=} h \cup \text{LIST_UNION } t \end{aligned}$$

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:

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

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.

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

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

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

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 $\mathfrak{M}.\text{Dom}$. 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 `feval` $\mathfrak{M} \ \sigma \ fform$, we say ‘ $fform$ is satisfied in \mathfrak{M} under the valuation σ ’.

Definition 1 (Term Valuation). *Some text here. The* `termval` : $\alpha \ model \rightarrow (num \rightarrow \alpha) \rightarrow term \rightarrow \alpha$.

$$\begin{aligned} \text{termval } (\mathfrak{M} : \alpha \ model) \ (v : num \rightarrow \alpha) \ (\text{fV } (x : num)) &\stackrel{\text{def}}{=} v \ x \\ \text{termval } (\mathfrak{M} : \alpha \ model) \ (v : num \rightarrow \alpha) \\ &\quad (\text{Fn } (f : num) \ (l : term \ list)) &\stackrel{\text{def}}{=} \\ &\quad \mathfrak{M}.\text{Fun } f \ (\text{MAP } (\lambda (a : term). \text{termval } \mathfrak{M} \ v \ a) \ l) \end{aligned}$$

Some concluding comments here.

$$\begin{aligned} \text{fsatis } \mathfrak{M} \ \sigma \ fform &\stackrel{\text{def}}{=} \text{valuation } \mathfrak{M} \ \sigma \wedge \text{feval } \mathfrak{M} \ \sigma \ fform \\ \text{feval } \mathfrak{M} \ v \ \text{fFALSE} &\stackrel{\text{def}}{=} \text{F} \\ \text{feval } \mathfrak{M} \ v \ (\text{Pred } a \ l) &\stackrel{\text{def}}{=} \mathfrak{M}.\text{Pred } a \ (\text{MAP } (\text{termval } \mathfrak{M} \ v) \ l) \\ \text{feval } \mathfrak{M} \ v \ (f_1 \rightarrow f_2) &\stackrel{\text{def}}{=} \text{feval } \mathfrak{M} \ v \ f_1 \Rightarrow \text{feval } \mathfrak{M} \ v \ f_2 \\ \text{feval } \mathfrak{M} \ v \ (\text{FALL } x \ f) &\stackrel{\text{def}}{=} \forall a. a \in \mathfrak{M}.\text{Dom} \Rightarrow \text{feval } \mathfrak{M} \ v \ (x \mapsto a) \ f \end{aligned}$$

(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:

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

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:

$$\begin{aligned} \text{mm2folm } \mathfrak{M} &\stackrel{\text{def}}{=} \\ &\quad <| \text{Dom} := \mathfrak{M}.\text{frame.world}; \\ &\quad \text{Fun} := (\lambda n \ args. \text{CHOICE } \mathfrak{M}.\text{frame.world}); \\ &\quad \text{Pred} := \\ &\quad \quad (\lambda p \ zs. \\ &\quad \quad \quad \text{case } zs \text{ of} \\ &\quad \quad \quad \quad [] \Rightarrow \text{F} \\ &\quad \quad \quad \quad | [w] \Rightarrow w \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{valt } p \ w \\ &\quad \quad \quad \quad | [w; w_2] \Rightarrow \\ &\quad \quad \quad \quad \quad p = 0 \wedge \mathfrak{M}.\text{frame.rel } w \ w_2 \wedge w \in \mathfrak{M}.\text{frame.world} \wedge \\ &\quad \quad \quad \quad \quad w_2 \in \mathfrak{M}.\text{frame.world} \\ &\quad \quad \quad \quad | w :: w_2 :: v_{10} :: v_{11} \Rightarrow \text{F}) | > \text{folm2mm } FM \stackrel{\text{def}}{=} \\ &\quad <| \text{frame} := \\ &\quad \quad <| \text{world} := FM.\text{Dom}; \\ &\quad \quad \text{rel} := \\ &\quad \quad \quad (\lambda w_1 \ w_2. FM.\text{Pred } 0 \ [w_1; w_2] \wedge w_1 \in FM.\text{Dom} \wedge w_2 \in FM.\text{Dom}) | >; \\ &\quad \text{valt} := (\lambda v \ w. FM.\text{Pred } v \ [w] \wedge w \in FM.\text{Dom}) | > \end{aligned}$$

Note a first order model from a modal model has no interesting information about function symbols and higher-ary predicate symbols. And 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. So the above constructions are not inverses in general. Also observe 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 need the assumption $\text{form_functions } f = \emptyset$ 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:

$$\begin{aligned} & \vdash \text{form_functions } f = \emptyset \Rightarrow \\ & \quad \forall \sigma. \\ & \quad \text{IMAGE } \sigma \mathcal{U}(: \text{num}) \subseteq \mathfrak{M}.\text{Dom} \Rightarrow \\ & \quad (\forall n. \mathfrak{M}.\text{Pred } n [] \iff F) \Rightarrow \\ & \quad (\forall a \ b \ n. \mathfrak{M}.\text{Pred } n [a; b] \Rightarrow n = 0) \Rightarrow \\ & \quad (\forall a \ b \ c \ d \ n. \mathfrak{M}.\text{Pred } n (a :: b :: c :: d) \iff F) \Rightarrow \\ & \quad (\text{feval } (\text{mm2folm } (\text{folm2mm } \mathfrak{M})) \sigma f \iff \text{feval } \mathfrak{M} \sigma f) \end{aligned}$$

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:

$$\begin{aligned} ST_x(p) &= Px \\ ST_x(\perp) &= \perp \\ ST_x(\neg\phi) &= \neg ST_x(\phi) \\ ST_x(\phi \vee \psi) &= ST_x(\phi) \vee ST_x(\psi) \\ ST_x(\Diamond\phi) &= \exists y (Rxy \wedge ST_y(\phi)) \end{aligned}$$

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 $x + 1$:

$$\begin{aligned} & \vdash (\forall x \ p. \text{ST } x (\text{VAR } p) = \text{fP } p (\text{fV } x)) \wedge \\ & \quad (\forall x. \text{ST } x \perp = \text{fFALSE}) \wedge \\ & \quad (\forall x \ phi. \text{ST } x (\neg phi) = \text{fNOT } (\text{ST } x phi)) \wedge \\ & \quad (\forall x \ phi \ psi. \\ & \quad \quad \text{ST } x (\text{DISJ } phi \ psi) = \text{fDISJ } (\text{ST } x phi) (\text{ST } x psi)) \wedge \\ & \quad \forall x \ phi. \\ & \quad \quad \text{ST } x (\Diamond phi) = \\ & \quad \quad \text{fEXISTS } (x + 1) \\ & \quad \quad (\text{fAND } (\text{fR } (\text{fV } x) (\text{fV } (x + 1))) (\text{ST } (x + 1) phi)) \end{aligned}$$

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

$$\vdash \text{FV } (\text{ST } x f) \subseteq \{x\} \vdash \text{form_functions } (\text{ST } x f) = \emptyset$$

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

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

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

$$\begin{aligned} \vdash \text{IMAGE } \sigma \mathcal{U}(: \text{num}) \subseteq \mathfrak{M}.\text{frame.world} &\Rightarrow \\ (\text{satis } \mathfrak{M} (\sigma x) \text{ phi} \iff \text{fsatis } (\text{mm2folm } \mathfrak{M}) \sigma (\text{ST } x \text{ phi})) \end{aligned}$$

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:

$$\begin{aligned} \text{ST_alt } x (\text{VAR } p) &\stackrel{\text{def}}{=} \text{fP } p (\text{fV } x) \\ \text{ST_alt } x \perp &\stackrel{\text{def}}{=} \text{fFALSE} \\ \text{ST_alt } x (\neg \text{phi}) &\stackrel{\text{def}}{=} \text{fNOT } (\text{ST_alt } x \text{ phi}) \\ \text{ST_alt } x (\text{DISJ } \text{phi } \text{psi}) &\stackrel{\text{def}}{=} \text{fDISJ } (\text{ST_alt } x \text{ phi}) (\text{ST_alt } x \text{ psi}) \\ \text{ST_alt } x (\Diamond \text{phi}) &\stackrel{\text{def}}{=} \\ \text{fEXISTS } (1 - x) & \\ (\text{fAND } (\text{fR } (\text{fV } x) (\text{fV } (1 - x))) & (\text{ST_alt } (1 - x) \text{ phi})) \end{aligned}$$

(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:

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

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:

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

A model \mathfrak{M} is called m-saturated if for each $w \in \mathfrak{M}.\text{frame.world}$ 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 .

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

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:

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

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:

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

Proof. Let \mathfrak{M} and \mathfrak{M}' be (α, β) , (α, γ) models respectively. Under the assumptions, the bisimulation relation is given by $\lambda n_1 n_2. \forall \phi. \text{satis } \mathfrak{M} n_1 \phi \iff \text{satis } \mathfrak{M}' n_2 \phi$. 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 $\mathfrak{M}.\text{frame.rel } w v$ and w and w' are modal equivalent, we can find a world v' of \mathfrak{M}' such that $\mathfrak{M}'.\text{frame.rel } w' v'$ 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:

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

We have $\text{satis } \mathfrak{M} v ff$, and therefore $\text{satis } \mathfrak{M} w (\Diamond ff)$. By modal equivalence of w and w' , we then get $\text{satis } \mathfrak{M} w' (\Diamond ff)$, 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 arbitrary 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:

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

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

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.

$$\begin{aligned} \text{filter } FLT \ W &\stackrel{\text{def}}{=} \\ &W \neq \emptyset \wedge FLT \subseteq POW \ W \wedge W \in FLT \wedge \\ &(\forall X \ Y. X \in FLT \wedge Y \in FLT \Rightarrow X \cap Y \in FLT) \wedge \\ &\forall X \ Z. X \in FLT \wedge X \subseteq Z \wedge Z \subseteq W \Rightarrow Z \in FLT \end{aligned}$$

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

$$\begin{aligned} &\vdash \text{FINITE } s \Rightarrow \\ &s \neq \emptyset \Rightarrow \\ &\forall U \ W. \text{filter } U \ W \Rightarrow s \subseteq U \Rightarrow \text{BIGINTER } s \in U \end{aligned}$$

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

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

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 filter:

$$\vdash \text{proper_filter } FLT \ W \iff \text{filter } FLT \ W \wedge FLT \neq POW \ W$$

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

$$\begin{aligned} &\vdash W \neq \emptyset \wedge U \neq \emptyset \wedge (\forall A. A \in U \Rightarrow \text{proper_filter } A \ W) \wedge \\ &(\forall A \ B. A \in U \wedge B \in U \Rightarrow A \subseteq B \vee B \subseteq A) \Rightarrow \\ &\text{proper_filter } (\text{BIGUNION } U) \ W \end{aligned}$$

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 .

$$\vdash \text{generated_filter } E \ W = \text{BIGINTER } \{ G \mid E \subseteq G \wedge \text{filter } G \ W \}$$

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:

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

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.

$$\vdash \text{principle_UF } w \ W = \{ X \mid X \subseteq W \wedge w \in X \}$$

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.

$$\begin{aligned} \text{ultrafilter } U \ W &\stackrel{\text{def}}{=} \\ \text{proper_filter } U \ W \wedge \forall X. X \in POW \ W \Rightarrow (X \in U \iff W \text{ DIFF } X \notin U) \end{aligned}$$

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

$$\vdash W \neq \emptyset \wedge w \in W \Rightarrow \text{ultrafilter } (\text{principle_UF } w \ W) \ W$$

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:

$$\begin{aligned} \vdash \text{FIP } S \ W &\iff \\ S &\subseteq \text{POW } W \wedge \\ \forall S'. S' &\subseteq S \wedge \text{FINITE } S' \wedge S' \neq \emptyset \Rightarrow \text{BIGINTER } S' \neq \emptyset \end{aligned}$$

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 destroy the finite intersection property of U .

$$\vdash \text{proper_filter } U \ W \Rightarrow \text{FIP } U \ W \vdash \text{proper_filter } U \ W \wedge B \in \text{POW } W \wedge B \notin U \wedge W \text{ DIFF } B \notin U \Rightarrow \text{FIP } (\{ B \} \cup U) \ W$$

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.

$$\vdash W \neq \emptyset \wedge \text{FIP } S \ W \Rightarrow \exists V. \text{proper_filter } V \ W \wedge S \subseteq V$$

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

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.

$$\vdash \text{proper_filter } U \ W \wedge (\forall S. \text{filter } S \ W \wedge U \subset S \Rightarrow S = \text{POW } W) \Rightarrow \text{ultrafilter } U \ W$$

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.

$$\vdash \text{proper_filter } f \ w \Rightarrow \exists U. \text{ultrafilter } U \ w \wedge f \subseteq U$$

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:

$$\begin{aligned} \vdash s &\neq \emptyset \wedge \text{partial_order } r \ s \wedge \\ &(\forall t. \text{chain } t \ r \Rightarrow \text{upper_bounds } t \ r \neq \emptyset) \Rightarrow \\ &\exists x. x \in \text{maximal_elements } s \ r \end{aligned}$$

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 `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 filter 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:

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

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

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

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.

$$\vdash \text{ultrafilter } U \ \mathcal{U}(\text{: num}) \ \wedge \ (\forall n. \{n\} \notin U) \Rightarrow \text{countably_incomplete } U \ \mathcal{U}(\text{: num})$$

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:

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

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 $I = I_0 \supseteq I_1 \supseteq I_2 \supseteq \dots$ such that each I_i is in U , and $\bigcap_{n \in \mathbb{N}} I_n = \emptyset$.

$$\begin{aligned} \vdash \text{countably_incomplete } U \ I &\Rightarrow \\ \exists I_n. \\ I_n \ 0 = I \ \wedge \ (\forall n. I_n \ n \in U \ \wedge \ I_n \ (n + 1) \subseteq I_n \ n) \ \wedge \\ \text{BIGINTER } \{ I_n \ n \mid n \in \mathcal{U}(\text{: num}) \} = \emptyset \end{aligned}$$

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 $J_n := \bigcap_{m \leq n} X_m$, 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 `PRIM_REC` ($X \ 0$) ($\lambda Xn \ n. Xn \cap X \ (n + 1)$). We get the desired chain I_n by inserting I at the beginning 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 `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 explanation. For $X \subseteq \mathfrak{M}.\text{frame.world}$, 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:

$$\begin{aligned} \text{can_see } \mathfrak{M} \ X &\stackrel{\text{def}}{=} \\ \{ w \mid w \in \mathfrak{M}.\text{frame.world} \ \wedge \ \exists x. x \in X \ \wedge \ \mathfrak{M}.\text{frame.rel } w \ x \} &\text{only_see } \mathfrak{M} \ X \stackrel{\text{def}}{=} \\ \{ w \mid & \\ w \in \mathfrak{M}.\text{frame.world} \ \wedge & \\ \forall x. x \in \mathfrak{M}.\text{frame.world} \ \wedge \ \mathfrak{M}.\text{frame.rel } w \ x \Rightarrow x \in X \} &\vdash X \subseteq \mathfrak{M}.\text{frame.world} \Rightarrow \\ \text{can_see } \mathfrak{M} \ X = & \\ \mathfrak{M}.\text{frame.world DIFF only_see } \mathfrak{M} \ (\mathfrak{M}.\text{frame.world DIFF } X) &\vdash X \subseteq \mathfrak{M}.\text{frame.world} \Rightarrow \\ \text{only_see } \mathfrak{M} \ X = & \\ \mathfrak{M}.\text{frame.world DIFF can_see } \mathfrak{M} \ (\mathfrak{M}.\text{frame.world DIFF } X) & \end{aligned}$$

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 :

$$\begin{aligned} \vdash \{ w \mid w \in \mathfrak{M}.\text{frame.world} \wedge \text{satis } \mathfrak{M} w (\Diamond \phi) \} = \\ \text{can_see } \mathfrak{M} \{ v \mid v \in \mathfrak{M}.\text{frame.world} \wedge \text{satis } \mathfrak{M} v \phi \} \vdash \{ w \mid w \in \mathfrak{M}.\text{frame.world} \wedge \text{satis } \mathfrak{M} w (\Box \phi) \} = \\ \text{only_see } \mathfrak{M} \{ v \mid v \in \mathfrak{M}.\text{frame.world} \wedge \text{satis } \mathfrak{M} v \phi \} \vdash \text{only_see } \mathfrak{M} (X \cap Y) = \text{only_see } \mathfrak{M} X \cap \text{only_see } \mathfrak{M} Y \end{aligned}$$

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 :

$$\begin{aligned} \text{UE_rel } \mathfrak{M} u v &\stackrel{\text{def}}{=} \\ &\text{ultrafilter } u \mathfrak{M}.\text{frame.world} \wedge \text{ultrafilter } v \mathfrak{M}.\text{frame.world} \wedge \\ &\forall X. X \in v \Rightarrow \text{can_see } \mathfrak{M} X \in u \vdash \text{ultrafilter } u \mathfrak{M}.\text{frame.world} \wedge \text{ultrafilter } v \mathfrak{M}.\text{frame.world} \Rightarrow \\ &(\text{UE_rel } \mathfrak{M} u v \iff \\ &\{ Y \mid \text{only_see } \mathfrak{M} Y \in u \wedge Y \subseteq \mathfrak{M}.\text{frame.world} \} \subseteq v) \end{aligned}$$

The ultrafilter extension is defined as follows:

$$\begin{aligned} \text{UE } \mathfrak{M} &\stackrel{\text{def}}{=} \\ &\langle \mid \text{frame} := \\ &\quad \langle \mid \text{world} := \{ u \mid \text{ultrafilter } u \mathfrak{M}.\text{frame.world} \}; \\ &\quad \text{rel} := \text{UE_rel } \mathfrak{M} \mid \rangle; \\ \text{valt} &:= \\ &(\lambda p v. \\ &\quad \text{ultrafilter } v \mathfrak{M}.\text{frame.world} \wedge \\ &\quad \mathfrak{M}.\text{valt } p \cap \mathfrak{M}.\text{frame.world} \in v) \mid \rangle \end{aligned}$$

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

$$\begin{aligned} \vdash w \in \mathfrak{M}.\text{frame.world} \Rightarrow \\ \text{modal_eq } \mathfrak{M} (\text{UE } \mathfrak{M}) w (\text{principle_UF } w \mathfrak{M}.\text{frame.world}) \end{aligned}$$

This is actually a special case of the following proposition, where u is taken as $\text{principle_UF } w \mathfrak{M}.\text{frame.world}$. 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 .

$$\begin{aligned} \vdash \text{ultrafilter } u \mathfrak{M}.\text{frame.world} \Rightarrow \\ (\{ w \mid w \in \mathfrak{M}.\text{frame.world} \wedge \text{satis } \mathfrak{M} w \phi \} \in u \iff \\ \text{satis } (\text{UE } \mathfrak{M}) u \phi) \end{aligned}$$

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

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

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

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

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

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

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

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

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

Consider the set $\Delta := \{\{w \mid w \in M.\text{frame.world} \wedge \forall \phi. \phi \in s \implies \text{satis } M w \phi\} \mid \text{FINITE } s \wedge s \subseteq \Sigma\} \cup \{Y \mid \text{only_see } M Y \in u \wedge Y \subseteq M.\text{frame.world}\}$. 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 $a \in \{\{w \mid w \in M.\text{frame.world} \wedge \forall \phi. \phi \in s \implies \text{satis } M w \phi\} \mid \text{FINITE } s \wedge s \subseteq \Sigma\}$ and $b \in \{Y \mid \text{only_see } M Y \in u \wedge Y \subseteq M.\text{frame.world}\}$, 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 $\text{only_see } \mathfrak{M} b \in u$, we show $\{w \mid w \in M.\text{frame.world} \wedge \forall \phi. \phi \in s \implies \text{satis } M w \phi\} \cap b \neq \emptyset$. Recall Σ is finitely satisfiable in the set of successors of u , we have a world u'' such that $\text{UE_rel } \mathfrak{M} u u''$ and $\text{satis } (\text{UE } \mathfrak{M}) u'' \phi$ for all $\phi \in s$, in other words, $\{w \mid w \in M.\text{frame.world} \wedge \text{satis } M w \phi\} \in u''$ for all $\phi \in s$. Then $\{w \mid w \in M.\text{frame.world} \wedge \forall \phi. \phi \in s \implies \text{satis } M w \phi\} \cap b \neq \emptyset$ is a big intersection of sets in u'' , and hence is in u'' . By exercise_2_5_5 again, $\text{UE_rel } \mathfrak{M} u u''$ gives $\{Y \mid \text{only_see}(Y) \in u \wedge Y \subseteq M.\text{frame.world}\} \subseteq u''$, so $b \in u''$ as well. As two elements in u'' has nonempty intersection, we are done.

Hence by $\text{ultrafilter_theorem_corollary}$, 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:

$$\begin{aligned} \vdash w \in \mathfrak{M}.\text{frame.world} \wedge w' \in \mathfrak{M}'.\text{frame.world} \implies \\ (\text{modal_eq } \mathfrak{M} \mathfrak{M}' w w' \iff \\ \text{bisim_world } (\text{UE } \mathfrak{M}) (\text{UE } \mathfrak{M}')) \\ (\text{principle_UF } w \mathfrak{M}.\text{frame.world}) \\ (\text{principle_UF } w' \mathfrak{M}'.\text{frame.world})) \end{aligned}$$

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

5 Invariant under bisimulations and simulations

In this chapter, we prove two characterisation results, they are related to the concepts ‘invariant under bisimulations’ and ‘preserved by simulations’ respectively. This whole chapter relies highly on first order logic, and we will swap between modal models and first order models very frequently. Hence when we talk about first order models, we restrict our scope to the ones which contains the same amount of information as a modal model. And only talk about first order formulas which makes sense to such models. So when we say ‘a first order formula’ in this chapter, we secretly refer to a first order formula with no function symbols and only have unary predicate symbols and only one binary predicate symbol.

5.1 Standard translations vs invariant under bisimulations

In the last chapter, we see the correspondence between modal formulas and first order formulas, and get known to the fact that every modal formula is equivalent to a first order formula. But actually, standard translation is really a small fragment of modal formulas. Even in our restricted scope, there are many first order formulas which is not equivalent to the standard translation of any formulas. Here we ask: What are the first order formulas which are equivalent to some standard translation? This section is to answer this question.

The short answer to our question is that a first order formula is equivalent to a modal formula iff it is invariant under bisimulations. A first order formula $\alpha(x)$ with at most one free variable x is invariant for bisimulations if for all models \mathfrak{M} and N , with $w \in \mathfrak{M}.\text{frame.world}$ and $v \in N.\text{frame.world}$, if w and v are bisimilar, then $\alpha(x)$ holds in \mathfrak{M} with x sends to w iff it holds in N with x sends to v .

When we see a formula is invariant under bisimulation, we allow \mathfrak{M} and N to be models of any type, and the type of \mathfrak{M} and N do not need to be identical. However, just like the issue we have seen when we defined `equiv0`, we are not allowed to have new type variables in only the right hand side of a definition. Thus in the HOL, we are not allowed to talk about invariance for bisimulation for any type of models, and have to refer to the type of models which are linked by the bisimulation, the definition is stated like this:

Modal formulas is preserved under bisimulation by `thm_2_20`, and hence their standard translations by `prop_2_47_i`. To justify the above answer to the question, the rest of this section devotes to prove any formulas which is invariant under bisimulation is equivalent to a standard translation:

$$\begin{aligned} \vdash \text{FV } a \subseteq \{x\} \wedge \text{form.functions } a = \emptyset &\Rightarrow \\ \text{invar4bisim } x \ t_1 \ t_2 \ a &\Rightarrow \\ \exists \text{ phi.} & \\ \forall \mathfrak{M} \ \sigma. & \\ (\forall n. \mathfrak{M}.\text{Pred } n \ [] \iff \text{F}) \wedge & \\ (\forall a \ b \ n. \mathfrak{M}.\text{Pred } n \ [a; b] \Rightarrow n = 0) \wedge & \\ (\forall a \ b \ c \ d \ n. \mathfrak{M}.\text{Pred } n \ (a :: b :: c :: d) \iff \text{F}) \wedge & \\ (\forall n_0 \ l_0. \mathfrak{M}.\text{Fun } n_0 \ l_0 \in \mathfrak{M}.\text{Dom}) \Rightarrow & \\ \text{IMAGE } \sigma \ \mathcal{U}(\text{: num}) \subseteq \mathfrak{M}.\text{Dom} \Rightarrow & \\ (\text{feval } \mathfrak{M} \ \sigma \ (\text{ST } x \ \text{phi}) \iff \text{feval } \mathfrak{M} \ \sigma \ a) & \end{aligned}$$

The tools that this proof will use centered on saturated models, which we are introducing now:

Given a first order model \mathfrak{M} , we can add some new constants to it to equip the model with more information. By adding new constant, we mean add nullary function symbols, each corresponds to some point in its domain. A model \mathfrak{M}' is an expansion of \mathfrak{M} if it is same as \mathfrak{M} except it has more constants. In the HOL, `expansion $\mathfrak{M} \ A \ \mathfrak{M}' \ f$` means ' \mathfrak{M}' is the expansion of \mathfrak{M} obtained by adding a bunch of constant, each corresponds an element in A '.

$$\begin{aligned} \text{expansion } \mathfrak{M} \ A \ \mathfrak{M}' \ f &\stackrel{\text{def}}{=} \\ \mathfrak{M}'.\text{Dom} &= \mathfrak{M}.\text{Dom} \wedge \text{BIJ } f \ (\text{count } (\text{CARD } A)) \ A \wedge \\ \mathfrak{M}'.\text{Fun} &= \\ (\lambda n \ \text{args.} & \\ \text{if } n < \text{CARD } A \wedge \text{args} = [] &\text{ then } f \ n \text{ else CHOICE } \mathfrak{M}.\text{Dom}) \wedge \\ \mathfrak{M}'.\text{Pred} &= \mathfrak{M}.\text{Pred} \end{aligned}$$

We are interested in the case that we are adding finitely many constants to a first order model without any function. So when we use this definition, \mathfrak{M} should be a model with no interesting information about constants, and A is a finite set of worlds in \mathfrak{M} . The count n is the set of the n natural numbers from 0 to $n - 1$. For \mathfrak{M} an α -model, f is of type $\text{num} \rightarrow \alpha$, its role is to assign each natural number an element in A . Hence `expansion $\mathfrak{M} \ A \ \mathfrak{M}' \ f$` means \mathfrak{M}' is a model that obtained by adding constants `Fn n []` with $n < \text{CARD } A$ which will evaluate to $f \ n$. For fixed \mathfrak{M} and A , the model \mathfrak{M}' such that `expansion $\mathfrak{M} \ A \ \mathfrak{M}' \ f$` is not unique, since the f may varies. But it does not matter since all those models will be equivalent, so we can actually refer to 'the' expansion of \mathfrak{M}' .

A first order model \mathfrak{M} is called n -saturated if for any set $A \subseteq \mathfrak{M}.\text{frame.world}$ where $\text{CARD } A < n$, the any \mathfrak{M}' such that `expansion $\mathfrak{M} \ A \ \mathfrak{M}' \ f$` for valid f realise every set G of formulas where each formula in G only have one free variable x and is only possible to involve nullary function symbols corresponds to the elements in A . Here the definition of consistence is borrowed from first order logic. A set G is consistent to the theory of a first order model \mathfrak{M} iff any finite set of G is realisable in \mathfrak{M} . A model is countably saturated if it is n -saturated for all n .

$$\begin{aligned}
& \text{consistent } \mathfrak{M} \ G \stackrel{\text{def}}{=} \\
& \quad \forall G_0. \\
& \quad \quad \text{FINITE } G_0 \wedge G_0 \subseteq G \Rightarrow \\
& \quad \quad \exists \sigma. \\
& \quad \quad \quad \text{IMAGE } \sigma \ U(: \text{num}) \subseteq \mathfrak{M}.\text{Dom} \wedge \\
& \quad \quad \quad \forall \text{phi}. \text{phi} \in G_0 \Rightarrow \text{fsatis } \mathfrak{M} \ \sigma \ \text{phin_saturated } \mathfrak{M} \ n \stackrel{\text{def}}{=} \\
& \quad \forall A \ \mathfrak{M}' \ G \ x \ f. \\
& \quad \quad \text{IMAGE } f \ U(: \text{num}) \subseteq \mathfrak{M}.\text{Dom} \wedge \text{FINITE } A \wedge \text{CARD } A \leq n \wedge \\
& \quad \quad A \subseteq \mathfrak{M}.\text{Dom} \wedge \text{expansion } \mathfrak{M} \ A \ \mathfrak{M}' \ f \wedge \\
& \quad \quad (\forall \text{phi}. \\
& \quad \quad \quad \text{phi} \in G \Rightarrow \\
& \quad \quad \quad \forall c. \\
& \quad \quad \quad \quad c \in \text{form_functions } \text{phi} \Rightarrow \\
& \quad \quad \quad \quad \text{FST } c \in \text{count } (\text{CARD } A) \wedge \text{SND } c = 0) \wedge \\
& \quad \quad \text{ftype } x \ G \wedge \text{consistent } \mathfrak{M}' \ G \Rightarrow \\
& \quad \quad \text{frealizes } \mathfrak{M}' \ x \ G \text{countably_saturated } \mathfrak{M} \stackrel{\text{def}}{=} \forall n. \text{n.saturated } \mathfrak{M} \ n
\end{aligned}$$

The following result explains how can countably saturated models be related to bisimulations:

$$\begin{aligned}
& \vdash \text{countably_saturated } (\text{mm2folm } \mathfrak{M}) \wedge \\
& \quad \text{countably_saturated } (\text{mm2folm } \mathfrak{M}') \wedge w \in \mathfrak{M}.\text{frame.world} \wedge \\
& \quad w' \in \mathfrak{M}'.\text{frame.world} \Rightarrow \\
& \quad \text{modal_eq } \mathfrak{M} \ \mathfrak{M}' \ w \ w' \Rightarrow \\
& \quad \text{bisim_world } \mathfrak{M} \ \mathfrak{M}' \ w \ w'
\end{aligned}$$

By `prop_2_54`, to prove the above, it suffices to prove:

$$\vdash \text{countably_saturated } (\text{mm2folm } \mathfrak{M}) \Rightarrow \text{M_sat } \mathfrak{M}$$

Proof. Suppose `countably_saturated (mm2folm M)`. Let $a \in \mathfrak{M}.\text{frame.world}$ and Σ a set of modal formulas which is finitely satisfiable in the set of successors of a . We find a successor of a in \mathfrak{M} realising all the formulas in Σ .

Define $\Sigma' = \{ \text{fR } (\text{Fn } 0 \ \square) \ (\text{fV } x) \} \cup \text{IMAGE } (\text{ST } x) \ \Sigma$ and $MA =$
`<|Dom := (mm2folm M).Dom;`
`Fun :=`
`(λ n args.`

`if n = 0 ∧ args = [] then w`
`else CHOICE (mm2folm M).Dom);`

`Pred := (mm2folm M).Pred|>`, then `expansion (mm2folm M) { w } MA (λ n. w)`. We claim `consistent MA Σ'`. Take a finite set $\Sigma_0 \subseteq \Sigma'$, we should find an element in MA realising it. For each element in Σ_0 which is a standard translation, use `CHOICE` to choose a modal formula $p \in \Sigma$ that is translated to it, note that we do need the choice function since standard translation is not injective under our construction. Collect the formulas we choose into a set ps , then ps is a finite subset of Σ . Recall we have assumed Σ is finitely satisfiable in the set of successors of a , hence there exists $b \in \mathfrak{M}.\text{frame.world}$ and $\mathfrak{M}.\text{frame.rel } a \ b$ such that `satis M b p` for any $p \in \Sigma$. It follows by `prop_2_47_i` that no matter `fR (Fn 0 □) (fV x)` is in Σ or not, we have Σ_0 is realised at b in MA .

This proves `consistent MA Σ'`. Since `mm2folm M` is countably saturated, Σ' itself is realised in some b in MA . The fact that `fR (Fn 0 □) (fV x)` is realised at b implies b is a successor of a in \mathfrak{M} , and `IMAGE (ST x) Σ` is realised at b implies that `satis M b phi` for any $\text{phi} \in \Sigma$ by `prop_2_47_i`. \square

There remains two steps from our main characterisation result, which are discussed in the following two interludes.

5.1.1 Interlude III: Countably saturated models via ultraproducts

The proceeding discussion about countably saturated models left us an important question: How do we get some countably saturated models? The fundamental construction underlying the theory of countably saturated models is that of ultraproduct. We firstly construct ultraproduct about sets, then about models.

Suppose $I \neq \emptyset$ and U is an ultrafilter on I . And for each $i \in I$, A_i is a non-empty set. The Cartesian product is the set of functions with domain I such that for all $i \in I$, $f(i) \in A_i$.

$$\text{Cart_prod } I \ A \stackrel{\text{def}}{=} \{ f \mid \forall i. i \in I \Rightarrow f \ i \in A \ i \}$$

For two functions f, g in the Cartesian product, we say f and g are U -equivalent if f and g agrees on some set in U . This defined an equivalent relation on the Cartesian product of the A_i 's:

$$\begin{aligned} \text{Uequiv } U \ I \ A \ f \ g &\stackrel{\text{def}}{=} \\ \text{ultrafilter } U \ I \ \wedge \ (\forall i. i \in I \Rightarrow A \ i \neq \emptyset) \ \wedge \ f &\in \text{Cart_prod } I \ A \ \wedge \\ g \in \text{Cart_prod } I \ A \ \wedge \ \{ i \mid i \in I \ \wedge \ f \ i = g \ i \} &\in U \vdash \text{ultrafilter } U \ I \Rightarrow \text{Uequiv } U \ I \ A \ \text{equiv_on } \text{Cart_prod } I \ A \end{aligned}$$

The ultraproduct of A_i modulo U is the set of equivalence classes partitioned by the relation $\text{Uequiv } U \mid A$:

$$\text{ultraproduct } U \ I \ A \stackrel{\text{def}}{=} \text{partition } (\text{Uequiv } U \ I \ A) (\text{Cart_prod } I \ A)$$

In the case where $A_i = A$ for all $i \in I$, the ultraproduct is called ultrapower of W modulo U .

We deal with ultraproduct of modal models first. Given a family MS of modal models indexed by I and an ultrafilter U on I , the ultraproduct model of MS modulo U is the model described as follows:

- The world set is the ultraproduct of world sets of M_i s modulo U .
- For two equivalence classes f_U, g_U of functions in the ultraproduct, they are related iff there exists $f \in f_U, g \in g_U$ such that $\{ i \in I \mid (MS \ i).\text{frame.rel } (f \ i) (g \ i) \}$ is in U .
- For a propositional letter p and an equivalence class f_U , we have p is satisfied at f_U iff there exists $g \in f_U$ such that $\{ i \mid i \in I \ \wedge \ f \ i \in (MS \ i).\text{val } p \}$ is in U .

The HOL definition looks like this:

$$\begin{aligned} \text{ultraproduct_model } U \ I \ MS &\stackrel{\text{def}}{=} \\ <\mid \text{frame} := \\ <\mid \text{world} := \text{ultraproduct } U \ I \ (\text{models2worlds } MS); \\ \text{rel} := \\ &(\lambda f \ u \ g \ u. \\ &\quad \exists f \ g. \\ &\quad f \in f_u \ \wedge \ g \in g_u \ \wedge \\ &\quad \{ i \mid i \in I \ \wedge \ (MS \ i).\text{frame.rel } (f \ i) (g \ i) \} \in U) \mid >; \\ \text{val} := \\ &(\lambda p \ f_u. \exists f. f \in f_u \ \wedge \ \{ i \mid i \in I \ \wedge \ f \ i \in (MS \ i).\text{val } p \} \in U) \mid > \end{aligned}$$

Note that since the choice of representatives does not make any difference since $\text{Uequiv } U \mid A$ is an equivalence relation, we get an equivalent definition of ultraproduct models if we replace all the existential quantifiers with universal quantifiers.

where the function `models2worlds` is used here to extract the underlying sets of models.

$$\text{models2worlds } MS \stackrel{\text{def}}{=} (\lambda i. (MS \ i).\text{frame.world})$$

Our central result about ultraproduct on modal models is a ‘modal version’ of the Los theorem. The standard version of Los theorem will be discussed in a moment.

$$\begin{aligned} \vdash \text{ultrafilter } U \ J \Rightarrow \\ \forall \phi \text{ fc.} \\ \text{fc} \in (\text{ultraproduct_model } U \ J \ MS).\text{frame.world} \Rightarrow \\ (\text{satis } (\text{ultraproduct_model } U \ J \ MS) \ \text{fc} \ \phi \text{ fc} \iff \\ \exists f. \\ f \in \text{fc} \ \wedge \\ \{ i \mid i \in J \ \wedge \ \text{satis } (MS \ i) (f \ i) \ \phi \} \in U) \end{aligned}$$

Proof. Given an ultrafilter U over J and a family Ms of models. We proceed by induction on ϕ , the base case for $\phi = \text{VAR } p$ is directly by definition, and the case for $\phi = \perp$ is by the fact that the empty set is not in the ultrafilter. The boolean cases are by basic property of ultrafilters. We only spell out the proof for diamond case. Suppose for any equivalence fc in the ultraproduct, we have $\text{satis}(\text{ultraproduct_model } U J Ms) fc \phi \iff$

$$\exists f. f \in fc \wedge \{ i \mid i \in J \wedge \text{satis}(Ms i) (f i) \phi \} \in U$$

Given a world in $\text{ultraproduct_model } U J Ms$, we prove $\text{satis}(\text{ultraproduct_model } U J Ms) fc (\Diamond \phi) \iff$
 $\exists f. f \in fc \wedge \{ i \mid i \in J \wedge \text{satis}(Ms i) (f i) (\Diamond \phi) \} \in U$

Left to right: The assumption says that there is an equivalence class that is related to fc and satisfied ϕ . Suppose the equivalence class gc is represented by a function g , and fc is represented by the function f . We check the f can be taken as our f as above. By definition of satisfaction, our task is to check the set $A =$

$$\begin{aligned} & \{ i \mid \\ & i \in J \wedge f i \in (Ms i).\text{frame.world} \wedge \\ & \exists v. \\ & (Ms i).\text{frame.rel}(f i) v \wedge v \in (Ms i).\text{frame.world} \wedge \\ & \text{satis}(Ms i) v \phi \} \text{ is in } U. \end{aligned}$$

By inductive hypothesis, the fact that ϕ is satisfied at gc implies the existence of an element in $x gc$ such that $\{ i \mid i \in J \wedge \text{satis}(Ms i) (x i) \phi \}$ is in U , but as Uequiv is an equivalence relation, this implies $\{ i \mid i \in J \wedge \text{satis}(Ms i) (g i) \phi \}$ is in U . As we can check: Since x and g lives in the same equivalence class, $\{ i \mid i \in J \wedge g i = x i \}$ is in U . As ultrafilters are closed under finite intersection, $\{ i \mid i \in J \wedge g i = x i \} \cap$

$\{ i \mid i \in J \wedge \text{satis}(Ms i) (x i) \phi \}$ is in U , this is a subset of $\{ i \mid i \in J \wedge \text{satis}(Ms i) (g i) \phi \}$, hence by upward closure, the result follows. As fc and gc is related, by a same procedure of checking independence of representatives, the set $\{ i \mid i \in J \wedge (Ms i).\text{frame.rel}(f i) (g i) \}$ is in U . Hence the intersection $\{ i \mid i \in J \wedge (Ms i).\text{frame.rel}(f i) (g i) \} \cap$
 $\{ i \mid i \in J \wedge \text{satis}(Ms i) (g i) \phi \}$ is in U . As our A is a supset of this set, A is in U as well.

Right to left: Suppose there is an $f \in fc$ such that $\{ i \mid$

$$\begin{aligned} & i \in J \wedge f i \in (Ms i).\text{frame.world} \wedge \\ & \exists v. \\ & (Ms i).\text{frame.rel}(f i) v \wedge v \in (Ms i).\text{frame.world} \wedge \\ & \text{satis}(Ms i) v \phi \} \text{ is in } U, \text{ we need to find an equivalence class which is related to } fc \text{ and satisfies } \phi, \text{ which} \\ & \text{by definition of relation in ultraproduct model, amounts to find a representative of such an equivalence class.} \\ & \text{The representative is given by : } \lambda i. \end{aligned}$$

$$\begin{aligned} & \text{if} \\ & \exists v. \\ & (Ms i).\text{frame.rel}(f i) v \wedge v \in (Ms i).\text{frame.world} \wedge \\ & \text{satis}(Ms i) v \phi \\ & \text{then} \\ & \text{CHOICE} \\ & \{ v \mid \\ & (Ms i).\text{frame.rel}(f i) v \wedge v \in (Ms i).\text{frame.world} \wedge \\ & \text{satis}(Ms i) v \phi \} \\ & \text{else CHOICE } (Ms i).\text{frame.world} \end{aligned}$$

checking this is the correct representative is tedious, but just routine, using representative independence and the property of CHOICE function. □

In the case that we are taking the ultraproduct of a constant family of models with $MS i = \mathfrak{M}$ for all $i \in I$, we get an ultrapower of \mathfrak{M} . Specializing `Los_model_thm` to the case of ultrapowers yields the following proposition.

$$\begin{aligned} & \vdash (\forall i. i \in J \Rightarrow MS i = \mathfrak{M}) \wedge \text{ultrafilter } U J \Rightarrow \\ & \forall \phi w. \\ & \text{satis}(\text{ultraproduct_model } U J Ms) \\ & \{ fw \mid \text{Uequiv } U J (\text{models2worlds } Ms) (\lambda i. w) fw \} \phi \iff \\ & \text{satis } \mathfrak{M} w \phi \end{aligned}$$

The construction of ultraproduct of first order models is similar to the construction for modal models. Restricting our scope only to first order models with ‘well-formed’ conditions in the sense of it contains the

same information as a modal model does not save much work, so we will just construct it for any first order models. This requires us to deal with function symbols and predicate symbols with any arity.

- A function with its symbol denoted by natural number n will send a list of equivalence class to the equivalence class represented by the function that sending $i \in I$ to $(FMS\ i).Fun\ n\ (MAP\ (\lambda fc.\ CHOICE\ fc\ i)\ fs)$. That is, to see where does $i \in I$ goes to, evaluate each representative in the list fs at i , collect them into a list, and use this list as the input of the function denoted by n in the model $FMS\ i$.
- A predicate with its symbol denoted by p will hold for a list zs of equivalence classes iff when we pick representatives in each member of zs and evaluate these representatives at i , the predicate p holds in $FMS\ i$ when we evaluate it with the list $MAP\ (\lambda fc.\ CHOICE\ fc\ i)\ zs$ of the outputs.

To avoid having quantifiers everywhere, we fix the representative for each equivalence class fc to be $CHOICE\ fc$.

The definition looks like this:

```

ultraproduct.folmodel  $U\ I\ FMS \stackrel{\text{def}}{=}
<|Dom := \text{ultraproduct } U\ I\ (\text{folmodels2Doms } FMS);
Fun :=
(\lambda n\ fs.
\{ y \mid
(\forall i. i \in I \Rightarrow y\ i \in (FMS\ i).Dom) \wedge
\{ i \mid
i \in I \wedge
y\ i = (FMS\ i).Fun\ n\ (MAP\ (\lambda fc.\ CHOICE\ fc\ i)\ fs) \} \in U \} \};
Pred :=
(\lambda p\ zs.
\{ i \mid
i \in I \wedge (FMS\ i).Pred\ p\ (MAP\ (\lambda fc.\ CHOICE\ fc\ i)\ zs) \} \in
U)|>$ 
```

We now prove this construction has nice semantical behaviour.

For an ultraproduct model of the family FMS of first order models, an evaluation σ assigns each variable symbol an equivalence class in the ultraproduct of the world sets of the family. A term t will be sent to the equivalence class represented by the function obtained as follows. For $i \in I$, the function that assigns a variable symbol n to the representative of $\sigma\ n$ evaluated at i will be an assignment of variable symbols to elements in $FMS\ i$, and i is sent to the element in $(FMS\ i).Dom$ we will get by evaluating t in $FMS\ i$ using this assignment of variable symbols.

```

 $\vdash \text{ultrafilter } U\ I \Rightarrow$ 
 $\forall \sigma\ FMS.$ 
 $\text{IMAGE } \sigma\ \mathcal{U}(:\text{ num}) \subseteq$ 
 $\text{ultraproduct } U\ I\ (\text{folmodels2Doms } FMS) \Rightarrow$ 
 $(\forall i\ ff\ ll. i \in I \Rightarrow (FMS\ i).Fun\ ff\ ll \in (FMS\ i).Dom) \Rightarrow$ 
 $\text{termval } (\text{ultraproduct.folmodel } U\ I\ FMS)\ \sigma\ t =$ 
 $\{ f \mid$ 
 $\text{Uequiv } U\ I\ (\text{folmodels2Doms } FMS)\ f$ 
 $(\lambda i. \text{termval } (FMS\ i)\ (\lambda n. \text{CHOICE } (\sigma\ n)\ i)\ t) \}$ 

```

Proof. By complete induction on `term_size t`. □

The nice semantical behaviour of ultraproduct is evident from the Los theorem:

```

 $\vdash \text{ultrafilter } U\ I \Rightarrow$ 
 $\forall \sigma\ FMS.$ 
 $\text{IMAGE } \sigma\ \mathcal{U}(:\text{ num}) \subseteq$ 
 $\text{ultraproduct } U\ I\ (\text{folmodels2Doms } FMS) \Rightarrow$ 
 $(\forall i\ ff\ ll. i \in I \Rightarrow (FMS\ i).Fun\ ff\ ll \in (FMS\ i).Dom) \Rightarrow$ 
 $(\text{feval } (\text{ultraproduct.folmodel } U\ I\ FMS)\ \sigma\ phi \iff$ 
 $\{ i \mid$ 
 $i \in I \wedge \text{feval } (FMS\ i)\ (\lambda x. \text{CHOICE } (\sigma\ x)\ i)\ phi \} \in$ 
 $U)$ 

```

Proof. By induction on ϕ . The base case for fFALSE comes from the fact that the empty set is not in an ultrafilter. And the atomic case reduces to check representative independence by unwinding the definitions.

We should prove $S_1 =$

```
{ i |
i ∈ I ∧
(FMS i).Pred n
(MAP
(λ x.
CHOICE
(termval (ultraproduct_folmodel U | FMS) σ x)
i) l) } is in U iff S2 =
{ i |
i ∈ I ∧
(FMS i).Pred n
(MAP (termval (FMS i) (λ x. CHOICE (σ x) i)) l) } is in U.
```

Let $I_0 =$

```
{ i |
i ∈ I ∧
MAP
(λ x.
CHOICE
(termval (ultraproduct_folmodel U | FMS) σ x) i)
l =
```

$\text{MAP } (\text{termval } (FMS \ i) \ (\lambda x. \text{CHOICE } (\sigma \ x) \ i)) \ l \}$, then $I_0 \cap S_1 = I_0 \cap S_2$. If $I_0 \in U$, suppose in addition that S_1 is in U , then S_2 is a supset of the set $I_0 \cap S_2$, which is in U , and vice versa. So it suffices to prove $I_0 \in U$. Obviously, if the two maps we are interested in agree on each member of l , then the resultant list we get will be equal. That gives $A =$

BIGINTER

```
{ { i |
i ∈ I ∧
CHOICE (termval (ultraproduct_folmodel U | FMS) σ a) i =
termval (FMS i) (λ x. CHOICE (σ x) i) a } |
```

$\text{MEM } a \ l \}$ is a subset of I_0 , reduces our task to prove A is in U . But by **thm_A_19_i**, for each member a of l , $\text{termval } (\text{ultraproduct_folmodel } U \mid FMS) \sigma \ a$ is the equivalence class represents by $\lambda i. \text{termval } (FMS \ i) \ (\lambda n. \text{CHOICE } (\sigma \ n) \ i) \ a$. Hence $\{ i \mid$

$i \in I \wedge$

```
CHOICE (termval (ultraproduct_folmodel U | FMS) σ a) i =
```

$\text{termval } (FMS \ i) \ (\lambda x. \text{CHOICE } (\sigma \ x) \ i) \ a \}$ is in U for each a . So A is in U as a finite intersection of sets in U .

The implication case is trivial from inductive hypothesis. Finally, we prove the case for universal quantifier. From left to right, suppose ϕ is satisfied at any equivalence class in the ultraproduct model for FMS .

We need $\{ i \mid$

$i \in I \wedge \text{feval } (FMS \ i) \ (\lambda x. \text{CHOICE } (\sigma \ x) \ i) \ (\text{FALL } n \ \phi) \} \in U$. Suppose not, then as U is an ultrafilter,

$B =$

```
{ i |
i ∈ I ∧
```

```
feval (FMS i) (λ x. CHOICE (σ x) i) (fEXISTS n (fNOT phi)) } ∈
```

U . Use choice, define the function f to send $i \in I$ to a chosen point in $(FMS \ i).\text{Dom}$ where ϕ is not satisfied. Such a function in HOL looks like: $f =$

```
(λ i.
```

```
if
```

```
∃ a.
```

```
a ∈ (FMS i).Dom ∧
```

```
¬feval (FMS i) (λ x. CHOICE (σ x) i) (n ↦ a) phi
```

```
then
```

```
CHOICE
```

```
{ a |
```

```
a ∈ (FMS i).Dom ∧
```

```
¬feval (FMS i) (λ x. CHOICE (σ x) i) (n ↦ a) phi }
```

```
else CHOICE (FMS i).Dom). Then { i |
```

$i \in I \wedge \neg \text{feval}(FMS\ i) (\lambda x. \text{CHOICE}(\sigma\ x)\ i) (n \mapsto f\ i)\ \text{phi} \} =$
 B , and hence is in U . Then by the inductive hypothesis, we can show the equivalence class represented by f does not satisfy phi , contradiction.

From right to left. It is straightforward to check $\{ i \mid$
 $i \in I \wedge \text{feval}(FMS\ i) (\lambda x. \text{CHOICE}(\sigma\ x)\ i) (\text{FALL}\ n\ \text{phi}) \}$ is a subset of $\{ i \mid$
 $i \in I \wedge \text{feval}(FMS\ i) (\lambda x. \text{CHOICE}(\sigma(n \mapsto a)\ x)\ i) \text{phi} \}$, for any equivalence class a . So the former one is in U implies the later one is in U . \square

The way we stated the Los theorem will look not so satisfactory, since it seems to restrict us for the choice of representatives, but it is not. We can be very free with our choice of representatives, as indicated in the following result:

$$\begin{aligned} & \vdash \text{ultrafilter } U\ I \Rightarrow \\ & \text{IMAGE } \sigma\ \mathcal{U}(\text{: num}) \subseteq \text{ultraproduct } U\ I\ (\text{folmodels2Doms } FMS) \Rightarrow \\ & \forall \text{phi } rv. \\ & (\forall v. v \in \text{FV } \text{phi} \Rightarrow rv\ v \in \sigma\ v) \Rightarrow \\ & (\{ i \mid i \in I \wedge \text{feval}(FMS\ i) (\lambda x. \text{CHOICE}(\sigma\ x)\ i) \text{phi} \} \in \\ & U \iff \\ & \{ i \mid i \in I \wedge \text{feval}(FMS\ i) (\lambda v. rv\ v\ i) \text{phi} \} \in U) \end{aligned}$$

If the world of models in FMS are of type β , then rv here is of type $\text{num} \rightarrow \beta$. The σ here is an assignment of variable symbols to worlds of the ultraproduct model, which are equivalence classes. And rv here assigns each variable symbol v an element $rv\ v$ in the equivalence class assigned to v by σ . For its proof, it actually amounts to check representative independency, which is of the same flavor of the second base case in the proof of Los theorem.

The independence of representative provide us convenience of actually putting our Los theorem into usage. Since then if we want to find an assignments of elements in a ultraproduct model satisfying a first order formula ϕ , instead of assigning equivalence classes directly, it suffices to use the rv below to assign each v a suitable representative functions in the Cartesian product :

$$\begin{aligned} & \vdash \text{ultrafilter } U\ I \Rightarrow \\ & (\forall i\ ff\ ll. i \in I \Rightarrow (FMS\ i).\text{Fun } ff\ ll \in (FMS\ i).\text{Dom}) \Rightarrow \\ & \forall rv. \\ & (\forall v\ i. i \in I \Rightarrow rv\ v\ i \in (FMS\ i).\text{Dom}) \Rightarrow \\ & \forall \text{phi}. \\ & \{ i \mid i \in I \wedge \text{feval}(FMS\ i) (\lambda v. rv\ v\ i) \text{phi} \} \in U \Rightarrow \\ & \text{feval}(\text{ultraproduct_folmodel } U\ I\ FMS) \\ & (\lambda v. \\ & \{ g \mid \\ & \text{Uequiv } U\ I\ (\text{folmodels2Doms } FMS)\ g\ (rv\ v) \}) \\ & \text{phi} \end{aligned}$$

With the help of `ultraproduct_rep_independence_lemma`, we can prove a classical corollary of Los theorem, which says any first order model \mathfrak{M} is embedded in any of its ultraproduct model by sending a world to the equivalence class represented by the constant function on that world:

$$\begin{aligned} & \vdash \text{ultrafilter } U\ I \Rightarrow \\ & (\forall i. i \in I \Rightarrow FMS\ i = FM) \Rightarrow \\ & (\forall i\ ff\ ll. i \in I \Rightarrow FM.\text{Fun } ff\ ll \in FM.\text{Dom}) \Rightarrow \\ & \forall \sigma. \\ & \text{IMAGE } \sigma\ \mathcal{U}(\text{: num}) \subseteq FM.\text{Dom} \Rightarrow \\ & \forall \text{phi}. \\ & \text{feval } FM\ \sigma\ \text{phi} \iff \\ & \text{feval}(\text{ultraproduct_folmodel } U\ I\ FMS) \\ & (\lambda x. \\ & \{ g \mid \\ & \text{Uequiv } U\ I\ (\text{folmodels2Doms } FMS)\ g \\ & (\lambda i. \sigma\ x) \}) \text{phi} \end{aligned}$$

All the construction we done above serves to paving a way to getting a countably saturated model:

$$\begin{aligned} &\vdash (\forall i. i \in I \Rightarrow (MS\ i).frame.world \neq \emptyset) \Rightarrow \\ &\quad \text{countably_incomplete } U\ I \Rightarrow \\ &\quad \text{countably_saturated } (\text{mm2folm } (\text{ultraproduct_model } U\ I\ MS)) \end{aligned}$$

With all the setups about ultraproduct models, we may feel confident that this will be a consequence of Los theorem. But if we take a closer look of it, we will find out the Los theorem cannot be directly applied here. The definition of countably saturated asks us to prove the realisation of a set of first order formulas in an expanded first order model, and the first order model itself is from turning a modal model into a first order model. The obstacles here will become clear when we compare what we want to prove to the statement of Los theorem: The Los theorem tells us the result for an ultraproduct of first order models, and says nothing about expansion. But we are proving a statement for an expanded model obtained from viewing an ultraproduct of modal models as a first order model. However, as we shall see now, it cannot stop us from applying the Los theorem.

The first issue is to remove the expansion on the outmost layer of expansion. The key observation is that we have an alternative approach to capture the idea of ‘constants’. Constants are nothing more than forcing some symbols to be sent to some point in a model under any valuation, hence rather than use nullary function symbols, we fixed a set of variable letters, each corresponds to a function symbol, and only consider the valuations that sends these variable letters to fixed certain points. With this idea, we can remove all the constants in a formula, and hence swap from an expanded model back to the unexpanded model. For our aim for proving `lemma_2_73`, we only care about the case that we only have finitely many function symbols. To get rid of n -function symbols, we need to come up with n fresh variable symbols to be sent to certain places. The easiest way to ensure that the variable symbols we use for constants do not clash with the variable symbols we use for usual aim is to force those n -variable symbols to be represented by $0, \dots, n$, and then add n to all the variable symbols which originally appear in a formula. The construction of discarding constants in a formula can be done by a function:

$$\begin{aligned} \text{shift_term } n\ (fV\ m) &\stackrel{\text{def}}{=} fV\ (m + n) \\ \text{shift_term } n\ (Fn\ m\ l) &\stackrel{\text{def}}{=} \\ &\quad \text{if } l = [] \text{ then } fV\ m \text{ else } Fn\ m\ (\text{MAP } (\lambda a. \text{shift_term } n\ a)\ l) \\ \text{shift_form } n\ (Pred\ m\ l) &\stackrel{\text{def}}{=} Pred\ m\ (\text{MAP } (\text{shift_term } n)\ l) \\ \text{shift_form } n\ (f_1 \rightarrow f_2) &\stackrel{\text{def}}{=} \text{shift_form } n\ f_1 \rightarrow \text{shift_form } n\ f_2 \\ \text{shift_form } n\ (FALL\ x\ f) &\stackrel{\text{def}}{=} FALL\ (x + n)\ (\text{shift_form } n\ f) \end{aligned}$$

Here we will need to prove the termination for the function `shift_term`. The termination relation is given by `measure (term.size \circ SND)`.

We get the syntactical result we want: All the function symbols are removed by applying the shifting construction, and the set of free variables is also under control.

$$\begin{aligned} &\vdash (\forall c. \\ &\quad c \in \text{form_functions } phi \Rightarrow \\ &\quad \text{FST } c \in \text{count } (CARD\ A) \wedge \text{SND } c = 0) \Rightarrow \\ &\quad \text{form_functions } (\text{shift_form } (CARD\ A)\ phi) = \emptyset \vdash FV\ phi \subseteq s \wedge \\ &\quad (\forall c. \\ &\quad \quad c \in \text{form_functions } phi \Rightarrow \\ &\quad \quad \text{FST } c \in \text{count } (CARD\ A) \wedge \text{SND } c = 0) \Rightarrow \\ &\quad \quad FV\ (\text{shift_form } (CARD\ A)\ phi) \text{ DIFF count } (CARD\ A) \subseteq \\ &\quad \quad \{x + CARD\ A \mid x \in s\} \end{aligned}$$

Now if we want to use an arbitrary valuation to evaluate a shifted formula, something will go wrong, because $0, \dots, n-1$ are now designed to be sent to fixed place $f\ 0, \dots, f\ n$, it does not make sense to assign it to anywhere else. Hence to talk about valuations, the first thing is to make sure that they sends the variables which actually denotes constants to the right place, therefore, we need to shift them accordingly:

$$\begin{aligned} \text{shift_valuation } n\ \sigma\ f &\stackrel{\text{def}}{=} \\ &\quad (\lambda m. \text{if } m < n \text{ then } f\ m \text{ else } \sigma\ (m - n)) \end{aligned}$$

Recall our aim is to get rid of constants and hence avoid talking about expansion of models, the following proposition proves our construction does a good job:

$$\text{shift_valuation } n \sigma f \stackrel{\text{def}}{=} (\lambda m. \text{if } m < n \text{ then } f \text{ } m \text{ else } \sigma (m - n))$$

The shifting construction get us out of the expansion, leaving us in a model obtained by coverting a ultraproduct modal model to a first model. How can we apply Los theorem to it? We do have a modal version of Los theorem, but we are not restricted to talking about standard translations, so `Los_modal_thm` is unhelpful. The correct way to apply first order version Los theorem on modal ultraproduct model is actually prove the ultraproduct construction on modal and first order models are capatible, in the sense that they are equivalent when we talk about formulas which makes sense to both of them. The capitability results are also witnesses of our success on construction of ultraproduct models.

Firstly, if we start with a family of modal models and take their ultraproduct in modal sense, then turn the resultant model into a first order model, then this model satisfies the same well-formed formula as the model we get by firstly turning the family of modal models into a family of first order models, then apply the ultraproduct construction in first order sense.

$$\begin{aligned} &\vdash \text{ultrafilter } U \ I \Rightarrow \\ &\quad \text{form.functions } phi = \emptyset \Rightarrow \\ &\quad \forall \sigma. \\ &\quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \text{ultraproduct } U \ I \ (\text{models2worlds } MS) \Rightarrow \\ &\quad (\text{feval } (\text{mm2folm } (\text{ultraproduct.model } U \ I \ MS)) \ \sigma \ phi \iff \\ &\quad \quad \text{feval} \\ &\quad \quad (\text{ultraproduct.folmodel } U \ I \ (\lambda i. \text{mm2folm } (MS \ i))) \\ &\quad \quad \sigma \ phi) \end{aligned}$$

Moreover, for a family of well-formed first order models, if we take their ultraproducts of them in first order sense, it satisfies the same well-formed formulas as the model we get by firstly regard this family as a family of modal models, take their ultraproduct in modal sense, and then turn the resultant ultraproduct modal model into a first order model.

$$\begin{aligned} &\vdash \text{ultrafilter } U \ I \Rightarrow \\ &\quad \text{form.functions } phi = \emptyset \Rightarrow \\ &\quad (\forall i \ n. i \in I \Rightarrow ((MS \ i).\text{Pred } n \ [] \iff F)) \Rightarrow \\ &\quad (\forall i \ a \ b \ n. i \in I \Rightarrow (MS \ i).\text{Pred } n \ [a; b] \Rightarrow n = 0) \Rightarrow \\ &\quad (\forall i \ a \ b \ c \ d \ n. i \in I \Rightarrow ((MS \ i).\text{Pred } n \ (a :: b :: c :: d) \iff F)) \Rightarrow \\ &\quad (\forall n_0 \ l_0 \ i. i \in I \Rightarrow (MS \ i).\text{Fun } n_0 \ l_0 \in (MS \ i).\text{Dom}) \Rightarrow \\ &\quad \forall \sigma. \\ &\quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\ &\quad \text{ultraproduct } U \ I \ (\text{folmodels2Doms } MS) \Rightarrow \\ &\quad (\text{feval } (\text{ultraproduct.folmodel } U \ I \ MS) \ \sigma \ phi \iff \\ &\quad \quad \text{feval} \\ &\quad \quad (\text{mm2folm} \\ &\quad \quad (\text{ultraproduct.model } U \ I \ (\lambda i. \text{folm2mm } (MS \ i)))) \\ &\quad \quad \sigma \ phi) \end{aligned}$$

Actually, the `ultraproduct_comm_feval` and `expansion_shift_feval` reduces our task into prove:

$$\begin{aligned}
& \vdash \text{countably_incomplete } U \ I \Rightarrow \\
& \quad \forall f. \\
& \quad \quad \text{IMAGE } f \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad \text{ultraproduct } U \ I \ (\text{folmodels2Doms } FMS) \Rightarrow \\
& \quad \quad (\forall i \text{ ff } ll. i \in I \Rightarrow (FMS \ i). \text{Fun } \text{ff } ll \in (FMS \ i). \text{Dom}) \Rightarrow \\
& \quad \quad \forall s. \\
& \quad \quad \quad (\forall phi. \\
& \quad \quad \quad \quad phi \in s \Rightarrow \\
& \quad \quad \quad \quad \text{form_functions } phi = \emptyset \wedge \text{FV } phi \text{ DIFF } N \subseteq \{x\} \Rightarrow \\
& \quad \quad \quad (\forall ss. \\
& \quad \quad \quad \quad \text{FINITE } ss \wedge ss \subseteq s \Rightarrow \\
& \quad \quad \quad \quad \exists \sigma. \\
& \quad \quad \quad \quad \quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad \quad \quad \quad (\text{ultraproduct_folmodel } U \ I \ FMS). \text{Dom} \wedge \\
& \quad \quad \quad \quad \quad (\forall n. n \in N \Rightarrow \sigma \ n = f \ n) \wedge \\
& \quad \quad \quad \quad \quad \forall phi. \\
& \quad \quad \quad \quad \quad \quad phi \in ss \Rightarrow \\
& \quad \quad \quad \quad \quad \quad \text{feval } (\text{ultraproduct_folmodel } U \ I \ FMS) \\
& \quad \quad \quad \quad \quad \quad \sigma \ phi \Rightarrow \\
& \quad \quad \quad \exists \sigma. \\
& \quad \quad \quad \quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad \quad \quad (\text{ultraproduct_folmodel } U \ I \ FMS). \text{Dom} \wedge \\
& \quad \quad \quad \quad (\forall n. n \in N \Rightarrow \sigma \ n = f \ n) \wedge \\
& \quad \quad \quad \quad \forall phi. \\
& \quad \quad \quad \quad \quad phi \in s \Rightarrow \\
& \quad \quad \quad \quad \quad \text{feval } (\text{ultraproduct_folmodel } U \ I \ FMS) \sigma \\
& \quad \quad \quad \quad \quad phi
\end{aligned}$$

Proof. Suppose for a set s of formulas, each element of it has only one free variable x except the ones that are representing constants, and for each subset $ss \subseteq s$, we have a valuation that realises this subset in the ultraproduct model, with the ‘constants’ delivered to the correct places, we will prove the whole set s is realised in the ultraproduct model by a valuation which sends the ‘constants’ to the right places. As the value of the constants are fixed, it amounts to find somewhere to send the only ‘real’ free variable x to. If s is finite then there is nothing to prove, so we assume s is infinite in the following discussion.

Recall we have restricted ourself to countably first order language, then as proved in ??? the infinite set s of first order formulas are countable. Hence we obtain a bijection $enum$ from the universe of natural numbers to our set s . Define $conj$ to be the primitive recursive function $\text{PRIM_REC True } (\lambda conjn \ n. \text{fAND } conjn \ (enum \ n))$, then $conj \ n$ will be the formula $enum(0) \wedge \dots \wedge enum(n)$. Then it suffices to prove the existence of a point that realise any of these conjunctions. By `ultraproduct_suffices_rep` and the Los theorem, it amounts to find out an a function to represent this equivalence class, we need some setups to find out the function.

Consider the family of sets: $J_n =$

$$\begin{aligned}
& (\lambda n. \\
& \quad \{ i \mid \\
& \quad i \in I \wedge \\
& \quad \forall \sigma.
\end{aligned}$$

$$(\forall k. k \in N \Rightarrow \sigma \ k = \text{CHOICE } (f \ k) \ i) \Rightarrow$$

$\text{feval } (FMS \ i) \ \sigma \ (\text{fEXISTS } x \ (conj \ n)) \}$ such J_n ’s forms a desending chain. As by assumption, each finite subset of s is realisable, each such J is realisable, so all of them are in U by Los theorem. By `countably_incomplete_chain`, we get another descending chain I_n ’s of sets U with $\bigcap_{n \in \mathbb{N}} I_n = \emptyset$. Define $X_n := I_n \cap J_n$, then X_n is a descending chain in U where $\bigcap_{n \in \mathbb{N}} X_n = \emptyset$. Hence for any $i \in I$, it exists a number $N(i)$ such that $i \in X_{N(i)}$ and for any $a > N(i)$, $i \notin X_a$. The equivalence class we will send x to is represented by the function $\lambda i.$

CHOICE

$$\{ a \mid$$

$$a \in (FMS \ i). \text{Dom} \wedge$$

$$\text{feval } (FMS \ i) \ (\lambda n. \text{if } n \in N \text{ then CHOICE } (f \ n) \ i \text{ else } a)$$

$$(conj \ (Ni \ i)) \}, \text{ call it } ff. \text{ By chosing such an } ff, \text{ we have } \forall k.$$

$$\{ i \mid i \in In \ 0 \wedge \text{feval } (FMS \ i) \ (\lambda v. rv \ v \ i) \ (conj \ k) \} \in U$$

□

We can step on the two stairs we built above to get `lemma_2_73`, using the capatibility lemma, we migrant the above lemma to ultraproduct of modal models:

$$\begin{aligned}
& \vdash \text{countably_incomplete } U \ I \Rightarrow \\
& (\forall i. i \in I \Rightarrow (MS \ i).\text{frame.world} \neq \emptyset) \Rightarrow \\
& \text{IMAGE } f \ \mathcal{U}(: \text{num}) \subseteq \text{ultraproduct } U \ I \ (\text{models2worlds } MS) \Rightarrow \\
& \forall s. \\
& (\forall phi. \\
& \quad phi \in s \Rightarrow \\
& \quad \text{form_functions } phi = \emptyset \wedge \text{FV } phi \text{ DIFF } N \subseteq \{x\}) \Rightarrow \\
& (\forall ss. \\
& \quad \text{FINITE } ss \wedge ss \subseteq s \Rightarrow \\
& \quad \exists \sigma. \\
& \quad \quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)).\text{Dom} \wedge \\
& \quad \quad (\forall n. n \in N \Rightarrow \sigma \ n = f \ n) \wedge \\
& \quad \quad \forall phi. \\
& \quad \quad \quad phi \in ss \Rightarrow \\
& \quad \quad \quad \text{feval} \\
& \quad \quad \quad (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)) \\
& \quad \quad \quad \sigma \ phi) \Rightarrow \\
& \quad \exists \sigma. \\
& \quad \quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)).\text{Dom} \wedge \\
& \quad \quad (\forall n. n \in N \Rightarrow \sigma \ n = f \ n) \wedge \\
& \quad \quad \forall phi. \\
& \quad \quad \quad phi \in s \Rightarrow \\
& \quad \quad \quad \text{feval } (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)) \ \sigma \\
& \quad \quad \quad phi
\end{aligned}$$

And use the shifting, we then migrant to expanded models:

$$\begin{aligned}
& \vdash \text{countably_incomplete } U \ I \Rightarrow \\
& (\forall i. i \in I \Rightarrow (MS \ i).\text{frame.world} \neq \emptyset) \Rightarrow \\
& \forall A \ \mathfrak{M}' \ f. \\
& \quad \text{expansion } (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)) \ A \ \mathfrak{M}' \ f \Rightarrow \\
& \quad \text{IMAGE } f \ \mathcal{U}(: \text{num}) \subseteq \text{ultraproduct } U \ I \ (\text{models2worlds } MS) \Rightarrow \\
& \quad \forall s. \\
& \quad (\forall phi. \\
& \quad \quad phi \in s \Rightarrow \\
& \quad \quad (\forall c. \\
& \quad \quad \quad c \in \text{form_functions } phi \Rightarrow \\
& \quad \quad \quad \text{FST } c \in \text{count } (\text{CARD } A) \wedge \text{SND } c = 0) \wedge \\
& \quad \quad \quad \text{FV } phi \subseteq \{x\}) \Rightarrow \\
& \quad (\forall ss. \\
& \quad \quad \text{FINITE } ss \wedge ss \subseteq s \Rightarrow \\
& \quad \quad \exists \sigma. \\
& \quad \quad \quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad \quad (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)).\text{Dom} \wedge \\
& \quad \quad \quad \forall phi. phi \in ss \Rightarrow \text{feval } \mathfrak{M}' \ \sigma \ phi) \Rightarrow \\
& \quad \exists \sigma. \\
& \quad \quad \text{IMAGE } \sigma \ \mathcal{U}(: \text{num}) \subseteq \\
& \quad \quad (\text{mm2folm } (\text{ultraproduct_model } U \ I \ MS)).\text{Dom} \wedge \\
& \quad \quad \forall phi. phi \in s \Rightarrow \text{feval } \mathfrak{M}' \ \sigma \ phi
\end{aligned}$$

we can deduce `lemma_2_73` trivially from here.

The last step towards the main theorem is used to pass from infinite set to finite set.

5.1.2 Interlude IV: Compactness theorem

With all these set-ups, we give a fully justified answer to the question in the begining of the section.

$$\begin{aligned}
& \vdash \text{FV } a \subseteq \{ x \} \wedge \text{form_functions } a = \emptyset \Rightarrow \\
& \text{invar4bisim } x \ t_1 \ t_2 \ a \Rightarrow \\
& \exists \text{phi.} \\
& \quad \forall \mathfrak{M} \sigma. \\
& \quad (\forall n. \mathfrak{M}.\text{Pred } n \ [] \iff \text{F}) \wedge \\
& \quad (\forall a \ b \ n. \mathfrak{M}.\text{Pred } n \ [a; b] \Rightarrow n = 0) \wedge \\
& \quad (\forall a \ b \ c \ d \ n. \mathfrak{M}.\text{Pred } n \ (a :: b :: c :: d) \iff \text{F}) \wedge \\
& \quad (\forall n_0 \ l_0. \mathfrak{M}.\text{Fun } n_0 \ l_0 \in \mathfrak{M}.\text{Dom}) \Rightarrow \\
& \quad \text{IMAGE } \sigma \ \mathcal{U}(\text{: num}) \subseteq \mathfrak{M}.\text{Dom} \Rightarrow \\
& \quad (\text{feval } \mathfrak{M} \ \sigma \ (\text{ST } x \ \text{phi}) \iff \text{feval } \mathfrak{M} \ \sigma \ a)
\end{aligned}$$

Proof.

□

5.2 Positive existential formula vs preserved under simulations

In this section, we use a similar proof strategy as we did in `thm_2_68_half1` to present a result about the concept ‘half of a bisimulation’, which is called ‘simulation’.

$$\begin{aligned}
& \text{sim } Z \ \mathfrak{M} \ \mathfrak{M}' \stackrel{\text{def}}{=} \\
& \quad \forall w \ w'. \\
& \quad w \in \mathfrak{M}.\text{frame.world} \wedge w' \in \mathfrak{M}'.\text{frame.world} \wedge Z \ w \ w' \Rightarrow \\
& \quad (\forall p. w \in \mathfrak{M}.\text{valt } p \Rightarrow w' \in \mathfrak{M}'.\text{valt } p) \wedge \\
& \quad \forall v. \\
& \quad v \in \mathfrak{M}.\text{frame.world} \wedge \mathfrak{M}.\text{frame.rel } w \ v \Rightarrow \\
& \quad \exists v'. v' \in \mathfrak{M}'.\text{frame.world} \wedge Z \ v \ v' \wedge \mathfrak{M}'.\text{frame.rel } w' \ v'
\end{aligned}$$

The concept corresponds to ‘invariant for bisimulation’ is called ‘preserved by simulation’, it takes the types of models as parameters by exactly the same reason why we need those parameters for the definition `invar4bisim`:

$$\begin{aligned}
& \text{preserved_under_sim } \mu \ \nu \ \text{phi} \stackrel{\text{def}}{=} \\
& \quad \forall \mathfrak{M} \ \mathfrak{M}' \ Z \ w \ w'. \\
& \quad w \in \mathfrak{M}.\text{frame.world} \wedge w' \in \mathfrak{M}'.\text{frame.world} \wedge \text{sim } Z \ \mathfrak{M} \ \mathfrak{M}' \wedge \\
& \quad Z \ w \ w' \Rightarrow \\
& \quad \text{satis } \mathfrak{M} \ w \ \text{phi} \Rightarrow \\
& \quad \text{satis } \mathfrak{M}' \ w' \ \text{phi}
\end{aligned}$$

The aim of the rest of the section is to characterise formulas preserved under bisimulation as positively existential formulas. A positive existential formula is a modal formula which it either equal to \top or \perp , or is built up using only contains positive connectives, namely ‘ \wedge ’, ‘ \vee ’ or ‘ \Diamond ’. Its definition is formalised as an inductive relation.

$$\begin{aligned}
& \vdash \text{PE } \perp \wedge \text{PE } \text{TRUE} \wedge (\forall p. \text{PE } (\text{VAR } p)) \wedge \\
& \quad (\forall f_1 \ f_2. \text{PE } f_1 \wedge \text{PE } f_2 \Rightarrow \text{PE } (\text{AND } f_1 \ f_2)) \wedge \\
& \quad (\forall f_1 \ f_2. \text{PE } f_1 \wedge \text{PE } f_2 \Rightarrow \text{PE } (\text{DISJ } f_1 \ f_2)) \wedge \\
& \quad \forall f. \text{PE } f \Rightarrow \text{PE } (\Diamond f)
\end{aligned}$$

A useful syntactical feature of such formulas is that any finite conjunction or disjunction of positive existential formulas is again a positive existential formula. Again, instead of explicitly define big conjunction and big disjunction, we use the following propositions to captures this idea:

$$\begin{aligned}
& \vdash \text{FINITE } ss \Rightarrow \\
& (\forall f. f \in ss \Rightarrow \text{PE } f) \Rightarrow \\
& \exists ff. \\
& \quad \text{PE } ff \wedge \\
& \quad \forall \mathfrak{M} w. \\
& \quad \quad w \in \mathfrak{M}.\text{frame.world} \Rightarrow \\
& \quad \quad (\text{satis } \mathfrak{M} w ff \iff \forall f. f \in ss \Rightarrow \text{satis } \mathfrak{M} w f) \vdash \text{FINITE } ss \Rightarrow \\
& (\forall f. f \in ss \Rightarrow \text{PE } f) \Rightarrow \\
& \exists ff. \\
& \quad \text{PE } ff \wedge \\
& \quad \forall \mathfrak{M} w. \\
& \quad \quad w \in \mathfrak{M}.\text{frame.world} \Rightarrow \\
& \quad \quad (\text{satis } \mathfrak{M} w ff \iff \exists f. f \in ss \wedge \text{satis } \mathfrak{M} w f)
\end{aligned}$$

By induction using `PE_ind`, it is easy to proof half of our main theorem:

$$\vdash \text{PE } phi \Rightarrow \forall \mu \nu. \text{preserved_under_sim } \mu \nu \text{ } phi$$

The reverse direction of `thm_2_78_half1_lemma` will get m-saturated to be involved, since m-saturated models are not only good for bisimulations, but also give nice result for simulations.

$$\begin{aligned}
& \vdash \text{M_sat } \mathfrak{M} \wedge \text{M_sat } \mathfrak{M}' \wedge w \in \mathfrak{M}.\text{frame.world} \wedge w' \in \mathfrak{M}'.\text{frame.world} \Rightarrow \\
& (\forall phi. \text{PE } phi \Rightarrow \text{satis } \mathfrak{M} w phi \Rightarrow \text{satis } \mathfrak{M}' w' phi) \Rightarrow \\
& \exists Z. \text{sim } Z \mathfrak{M} \mathfrak{M}' \wedge Z w w'
\end{aligned}$$

Proof. Under the assumptions, $\lambda w_1 w_2.$

$\forall phi. \text{PE } phi \Rightarrow \text{satis } \mathfrak{M} w_1 phi \Rightarrow \text{satis } \mathfrak{M}' w_2 phi$ gives a simulation relation. Checking it is indeed a simulation is completely analogue to the proof of `prop_2_54`. \square

We will take a detour through m-saturated models again, but since this time we are working within a modal language instead of first order logic, we only need to use ultrafilter extensions rather than ultrapowers.

$$\begin{aligned}
& \vdash \text{preserved_under_sim } \nu \nu \text{ } phi \Rightarrow \\
& \exists phi_0. \text{equiv0 } \mu \text{ } phi \text{ } phi_0 \wedge \text{PE } phi_0
\end{aligned}$$

At first glance, the statement may looks awkward because the assumption is about preservsion under simulation for models of type $(\beta \rightarrow \text{bool}) \rightarrow \text{bool}$, but the conclusion is about equivalent to a formula on models of type β . This is because our detour through ultrafilter extensions, which embeds a model with β -worlds into a model with $(\beta \rightarrow \text{bool}) \rightarrow \text{bool}$ -worlds. It means that for the usual language statement, although the hypothesis is talking about preserved under simulation for any types, what we actually require in the proof is only the fact that the formula is preserved under simulation for a certain type.

Proof. Suppose ϕ is preserved under simulation. Consider the set of positive existential consequence of ϕ , defined as $PEC =$

$$\begin{aligned}
& \{ psi \mid \\
& \text{PE } psi \wedge
\end{aligned}$$

$\forall \mathfrak{M} w. w \in \mathfrak{M}.\text{frame.world} \Rightarrow \text{satis } \mathfrak{M} w phi \Rightarrow \text{satis } \mathfrak{M} w psi \}$ By `modal_compactness_corollary` proved by the last interlude, if we can prove for any model \mathfrak{M} and $w \in \mathfrak{M}.\text{frame.world}$, $\text{satis } \mathfrak{M} w psi$ for all $psi \in PEC$ implies $\text{satis } \mathfrak{M} w phi$, then there exists a finite subset S of PEC that entails ϕ . This will prove ϕ is equivalent to the conjunction of all the formulas in S , which is again a positive existential formula.

So we prove the entailment from PEC to ϕ . Suppose $\text{satis } \mathfrak{M} w psi$ for all $psi \in PEC$, we prove $\text{satis } \mathfrak{M} w phi$. Define $\Gamma = \{ \neg psi \mid \text{PE } psi \wedge \text{satis } \mathfrak{M} w (\neg psi) \}$. We claim that there exists a model that realises the set $\Gamma \cup \{ phi \}$. By `modal_compactness_thm`, it suffices to prove each finite subset of $\Gamma \cup \{ phi \}$ is realised by some model. Suppose there exists a finite subset of $\Gamma \cup \{ phi \}$ which cannot be realised by any model, then there exists $\neg\psi_0, \dots, \neg\psi_n \in \Gamma$ such that for any model N and any world v of it, $\text{satis } N v psi_i$ for some i . By `PE_BIGDISJ`, as all these ψ 's are positive existential, there exists a positive existential formula ψ which is satisfied precisely when some ψ_i is satisfied. Hence $psi \in PEC$. As \mathfrak{M} entails PEC , we have $\text{satis } \mathfrak{M} w psi$. But then it means $\text{satis } \mathfrak{M} w psi_i$ for some i , but from the definition of the ψ 's, $\text{satis } \mathfrak{M} w (\neg psi_i)$ for any ψ_i , this is a contradiction.

Hence we obtain a model N realising $\Gamma \cup \{ phi \}$ at point v . For any positive existential formula ψ such that $\neg \text{satis } \mathfrak{M} w psi$, we have $\neg psi \in \Gamma$, so $\text{satis } N v (\neg psi)$. Hence for any positive existential ψ , if $\text{satis } N v psi$, then $\text{satis } \mathfrak{M} w psi$. We migrant this implication onto ultrafilter extensions: Now

take the ultrafilter extension of \mathfrak{M} and N respectively. By `prop_2_59_ii`, for any positive existential ψ , $\text{satis } (\text{UE } N) (\text{principle_UF } v \ N.\text{frame.world}) \ \psi$ implies $\text{satis } (\text{UE } \mathfrak{M}) (\text{principle_UF } w \ \mathfrak{M}.\text{frame.world}) \ \psi$. As $\text{UE } \mathfrak{M}$ and $\text{UE } N$ are m-saturated by `prop_2_61`, by `exercise_2_7_1`, we have a simulation linking $\text{principle_UF } v \ N.\text{frame.world}$ to $\text{principle_UF } w \ \mathfrak{M}.\text{frame.world}$.

As $\text{satis } N \ v \ \phi$, `prop_2_59_ii` gives $\text{satis } (\text{UE } N) (\text{principle_UF } v \ N.\text{frame.world}) \ \phi$, as ϕ is preserved under simulation, we have $\text{satis } (\text{UE } \mathfrak{M}) (\text{principle_UF } w \ \mathfrak{M}.\text{frame.world}) \ \phi$. Again by `prop_2_59_ii`, it implies $\text{satis } \mathfrak{M} \ w \ \phi$. This completes the proof. \square