Expressivity of Logics of Knowledge and Action

Kuijer, Bouke

Chapter 6

Generalizing Expressivity

Chapter Summary. In this chapter we look at a number of results that were presented in [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b]. These results are not expressivity results. Yet they are in some ways very similar to expressivity results.

We therefore generalize the concept of expressivity in such a way that the aforementioned results can be seen as showing that one logic has a greater generalized expressivity than another.

6.1 Goals and Motivation

This chapter is more speculative than the other chapters in this thesis. As such it is very important, even more so than in the other chapters, to be clear about my motivations and about what I want to achieve.

I have been studying expressivity for several years now. During that time I have proven a number of expressivity results and I have seen the proofs of many more. But, while looking for expressivity results, I have also encountered several results [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] that are very similar to expressivity results but that do not quite satisfy Definition 1.1.

Much like expressivity results, these "similar" result seem to show that one logic can say everything that can be said in a different logic. Furthermore, the techniques used in the proofs of the "similar" results also strongly resemble the techniques that are typically used in the proofs of expressivity results. This suggests that the "similar" results may be *expressivity*$_g$ results, for some appropriate generalization expressivity$_g$ of expressivity. My goal in this chapter is to find a definition for expressivity$_g$.

We should note that some humility—epistemic and otherwise—is appropriate here, for two reasons. Firstly, we are defining expressivity$_g$ after the fact, based on existing examples. There is no guarantee that the authors of these examples had something like expressivity$_g$ in mind. Secondly, the definition of expressivity$_g$ requires a lot of complicated technical details. We could change some of these technical details to obtain a slightly different concept expressivity$'_g$.

With this humility in mind, the goal of this chapter is to find a reasonable generalization (as opposed to *the* generalization) expressivity$_g$ of expressivity with the

property that the results from [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] can be seen as expressivity$_g$ results.

6.2 Generalizing Expressivity

As mentioned in the previous section, our main goal is to find a generalization expressivity$_g$ of expressivity. Let us start by considering in a bit more detail why we would want to generalize expressivity. Recall that in Chapter 1 we defined expressivity as follows.

Definition 1.1. A logic $\mathcal{L}_2$ is *at least as expressive* as a logic $\mathcal{L}_1$ if for every formula φ_1 of $\mathcal{L}_1$ there is an equivalent formula φ_2 of $\mathcal{L}_2$.

Also recall that "equivalent" in this definition means "having the same truth value on every pointed model". So φ_2 must have the property that, for every pointed model $\mathcal{M}, w$ we have $\mathcal{M}, w \models \varphi_1$ if and only if $\mathcal{M}, w \models \varphi_2$. This means that we cannot compare the expressivity of two logics if they do not share the same class of models.

There are, however, a number of results in the literature that are very similar to expressivity results except that they compare two logics that have different classes of models. These results include [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b]. I consider each of these results interesting and, considering that these papers were published in journals and conference proceedings, it seems that there are other logicians that also consider them interesting. Let us therefore say that they *are* interesting.

If multiple results that follow the same pattern are all interesting it is reasonable to assume that they are interesting partially *because* they follow this pattern. Let us briefly consider a few examples.

Example 6.1. There are many published results that satisfy the following pattern.

1. Semantics for a logic are defined.
2. A proof system is defined.
3. It is shown that a formula is valid according to the semantics if and only if it is provable in the proof system.

Results satisfying this pattern are usually called "soundness and (weak) completeness results."

Example 6.2. There are also a lot of published results that satisfy this pattern:

1. Semantics for a logic $\mathcal{L}_1$ are defined.
2. Semantics are defined for a logic $\mathcal{L}_2$ that used the same class of models as $\mathcal{L}_1$.
3. A function is defined that maps every $\mathcal{L}_1$ formula φ_1 to an $\mathcal{L}_2$ formula φ_2 such that φ_1 and φ_2 are equivalent.

Such results are called "(positive) expressivity results." In another pattern, step 3 is replaced by introducing an $\mathcal{L}_1$ formula φ_1 and showing that there is no $\mathcal{L}_2$ formula equivalent to φ_1. A result following that pattern is a "(negative) expressivity result."

Let us consider one final example.

Example 6.3. A lot of results satisfy the following pattern.

1. A logic $\mathcal{L}$ is introduced, for which the computational complexity of the satisfiability (or model checking) problem is not yet known.

2. A decision problem with known computational complexity is introduced.

3. A reduction is given that transforms instances of the decision problem with known complexity into instances of the $\mathcal{L}$ satisfiability problem in polynomial time.

Results satisfying this pattern are a certain type of "computational complexity results."

Satisfying a pattern is probably not a necessary condition for a result to be interesting.[1] It is not a sufficient condition for a result to be interesting either. After all, a result that compares two uninteresting logics will generally be uninteresting, regardless of whether it satisfies an interesting pattern. But, all else being equal, results that satisfy such a pattern will usually be more interesting than results that do not. As such, the patterns can partially explain why a given result is interesting: an interesting computational complexity result is interesting *because* it is a computational complexity result. If the result had been slightly different and only shown that most (as opposed to all) instances of the decision problem with known complexity can be reduced to instances of the $\mathcal{L}$ problem then the result probably wouldn't have been interesting because it would not have been a computational complexity result.

The results that are presented in [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] are similar to each other and interesting. This suggests that they are interesting because they satisfy some pattern. The goal of this chapter is to find this pattern. The results in question are very similar to expressivity results, in a way that is explained Section 6.4. We will therefore call the pattern "generalized expressivity", or expressivity$_g$ for short. So, as mentioned at the start of this section, we are looking for a generalization of expressivity. But we do not want just any generalization; we want expressivity$_g$ to be the generalization that explains why the results from [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] are interesting. Considering that we introduce expressivity$_g$ specifically to explain why these results are interesting we call them the *prototypical* expressivity$_g$ results.

Before defining expressivity$_g$ we first need to consider the definition of expressivity in a bit more detail. Both expressivity and expressivity$_g$ are used to compare different logics. So we should start by defining what a logic is. There are many different ways to define a logic and it is outside the scope of this thesis to consider all of them. Still, let us very briefly look at three possibilities.

- A logic is the combination of a set of formulas and a subset of the formulas that are valid, so a pair (Φ, T) where $T \subseteq \Phi$.

- A logic is the combination of a set of formulas and an inference relation on the formulas, so a pair $(\Phi, \vdash)$ where $\vdash \subseteq \wp(\Phi) \times \Phi$.

[1] That is a matter of definition though. If we allow patterns of which there is only one instance then we could say that every interesting result follows a pattern.

- A logic is the combination of a set of formulas and semantics. The semantics consist of a set of (pointed) models and a satisfaction relation between (pointed) models and formulas, so a logic is a triple $(\Phi, \mathfrak{M}, \models)$ where $\models \subseteq \mathfrak{M} \times \Phi$.

Note that $(\Phi, \vdash)$ is a more detailed description of a logic than (Φ, T), in the sense that a logic $(\Phi, \vdash)$ uniquely determines a corresponding logic $(\Phi, T) = (\Phi, \{\varphi \mid \emptyset \vdash \varphi\})$ while there is no *unique* logic $(\Phi, \vdash)$ corresponding to a logic (Φ, T). Likewise, $(\Phi, \mathfrak{M}, \models)$ is a more detailed description than $(\Phi, \vdash)$, because a logic $(\Phi, \mathfrak{M}, \models)$ uniquely determines a corresponding logic $(\Phi, \vdash)$ but for given $(\Phi, \vdash)$ there may be multiple corresponding logics of the form $(\Phi, \mathfrak{M}, \models)$. These three possible ways to define a logic are therefore three different levels of detail with which we can describe logics.

None of these levels of detail is right or wrong per se; which one is best depends on the circumstances. If, for example, we wanted to study the properties of proof systems then we should probably take the first or second option and define a logic to be a pair (Φ, T) or a pair $(\Phi, \vdash)$. But here we want to study expressivity, not proof systems. That means we need to talk about equivalence, in the sense of having the same truth value on every pointed model. So the concepts we need are only available at the greatest level of detail. We should therefore take the third option: a logic is a triple $(\Phi, \mathfrak{M}, \models)$.

Definition 6.1 (Logic)**.** A *logic* $\mathcal{L}$ is a triple $\mathcal{L} = (\Phi, \mathfrak{M}, \models)$ where Φ is a set of formulas, $\mathfrak{M}$ is a class of pointed models and $\models \subseteq \mathfrak{M} \times \Phi$ is a satisfaction relation.

We say that $\mathcal{M}$ is a *model* if there is some w such that $(\mathcal{M}, w) \in \mathfrak{M}$. Furthermore, w is a *world* of $\mathcal{M}$ if $(\mathcal{M}, w) \in \mathfrak{M}$.

We sometimes abuse notation by writing $\mathcal{M} \in \mathfrak{M}$ if $\mathcal{M}$ is a model and $w \in \mathcal{M}$ if w is a world of $\mathcal{M}$. Given this definition of a logic we can give a definition of expressivity that is slightly more precise than Definition 1.1.

Definition 6.2 (Expressivity)**.** Let $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}, \models_1)$ and $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}, \models_2)$ be logics that have the same class of models. Then $\mathcal{L}_2$ is *at least as expressive* as $\mathcal{L}_1$ if for every $\varphi_1 \in \Phi_1$ there is a $\varphi_2 \in \Phi_2$ such that for every $\mathcal{M}, w \in \mathfrak{M}$ we have $\mathcal{M}, w \models_1 \varphi_1$ if and only if $\mathcal{M}, w \models_2 \varphi_2$.

6.3 Related Concepts

Before continuing with the generalization of expressivity let us first briefly consider several concepts that fulfill a role similar to that of expressivity. These concepts are *defineability* (see for example [van Benthem, 1984]), *conservative translations* (see for example [Feitosa and D'Ottaviano, 2001]) and *interpretability* (see for example [Verbrugge, 1993]).

The idea behind expressivity is that a logic $\mathcal{L}_2$ is at least as expressive as a logic $\mathcal{L}_1$ if and only if everything that can be said in $\mathcal{L}_1$ can also be said in $\mathcal{L}_2$. But similar things hold for the other concepts. Everything that is definable in $\mathcal{L}_1$ is defineable in $\mathcal{L}_2$ if and only if everything that can be said in $\mathcal{L}_1$ can also be said in $\mathcal{L}_2$. There is a conservative translation from $\mathcal{L}_1$ to $\mathcal{L}_2$ if and only if everything that can be said in $\mathcal{L}_1$ can also be said in $\mathcal{L}_2$. The logic $\mathcal{L}_1$ can be interpreted in $\mathcal{L}_2$ if and only if everything that can be said in $\mathcal{L}_1$ can also be said in $\mathcal{L}_2$.

At first glance one would say that the four concepts would have to be equivalent, since they all hold if and only if everything that can be said in $\mathcal{L}_1$ can also be said in

$\mathcal{L}_2$. But they are not equivalent. The reason for this non-equivalence is that the four concepts are suitable for different definitions of logic. Conservative translations are the concept suitable for logics $(\Phi, \vdash)$. Defineability is the concept suitable for logics $(\Phi, \mathfrak{K}, \models)$, where $\mathfrak{K}$ is a class of *frames*.

The difference between expressivity and interpretability is slightly more difficult to explain, because they can both be applied to logics of the form $(\Phi, \mathfrak{M}, \models)$. Roughly speaking, a logic $\mathcal{L}_1$ is interpretable in a logic $\mathcal{L}_2$ if and only if every model of $\mathcal{L}_2$ contains a submodel that is (in some precisely defined sense) a model of $\mathcal{L}_1$. This does not make much sense if the models of $\mathcal{L}_2$ are Kripke-style possible world models. It does however make sense if the models of $\mathcal{L}_2$ are *universes* like the models of set theory or Peano arithmetic. As such, interpretability tends to be a suitable notion for mathematical logics whereas expressivity tends to be a suitable notion for modal logics with Kripke-style models.

The logics that are studied in this thesis are of the type for which expressivity is the most suitable notion, which is why we will generalize expressivity and not the other concepts. Some of the ideas we use while generalizing expressivity are, however, similar to ideas from the other concepts. We will discuss such similarities when we get to them.

6.4 Similar to Expressivity

In Section 6.1 I stated that the results from [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] are similar to expressivity results. In this section I explain in which sense these results are similar to expressivity results.

A logic $\mathcal{L}_2$ is at least as expressive as a logic $\mathcal{L}_1$ if and only if for every $\mathcal{L}_1$ formula φ_1 there is an equivalent $\mathcal{L}_2$ formula φ_2. But we can also characterize expressivity by the existence or non-existence of a certain kind of translation.

Definition 6.3 (Translation)**.** A *translation* from a logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}, \models_1)$ to a logic $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}, \models_2)$ is a function $t : \Phi_1 \to \Phi_2$.

Definition 6.4 (Truth preserving)**.** A translation $t : \Phi_1 \to \Phi_2$ from a logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}, \models_1)$ to a logic $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}, \models_2)$ is *truth preserving* if for every $\mathcal{M}, w \in \mathfrak{M}$ we have $\mathcal{M}, w \models \varphi_1$ if and only if $\mathcal{M}, w \models \varphi_2$.

We can then easily see the following.

Lemma 6.1. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be logics with the same class of models. Then $\mathcal{L}_2$ is at least as expressive as $\mathcal{L}_1$ if and only if there is a truth preserving translation t from $\mathcal{L}_1$ and $\mathcal{L}_2$.*

The lemma is not much more than a rephrasing of Definition 6.2. The proof is therefore left to the reader. The reason for introducing Lemma 6.1 is that this characterization allows us to see why the results in [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] are similar to expressivity results.

Each of these publications is about at least two logics $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ and $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ that have different classes of models. Two functions $t : \Phi_1 \to \Phi_2$ and $f : \mathfrak{M}_1 \to \mathfrak{M}_2$ are then introduced, and it is shown that for every $\mathcal{M}_1, w_1 \in \mathfrak{M}_1$ and every $\varphi_1 \in \Phi_1$ we have $\mathcal{M}_1, w_1 \models_1 \varphi_1$ if and only if $f(\mathcal{M}_1, w_1) \models_2 t(\varphi_1)$. Such pairs

(f, t) strongly resemble truth-preserving translations, so their existence shows that one logic stands to another in a relation very similar to "being at least as expressive as".

6.5 The Triviality Problem

A first attempt at generalizing expressivity could therefore be the following.

Definition 6.5 (Translation)**.** A *translation* from a logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ to a logic $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ is a pair (t, f) of functions where $f : \mathfrak{M}_1 \to \mathfrak{M}_2$ and $t : \Phi_1 \to \Phi_2$.

Definition 6.6 (Truth preserving)**.** A translation from $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ to $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ is *truth preserving* if for every $\mathcal{M}_1, w_1 \in \Phi_1$ and every $\varphi_1 \in \Phi_1$ we have $\mathcal{M}_1, w_1 \models_1 \varphi_1$ if and only if $f(\mathcal{M}_1, w_1) \models_2 t(\varphi_1)$.

Based on this generalization of truth preserving translations, we can define a generalization of expressivity. This generalization is not the expressivity$_g$ we are looking for though, so we will call it expressivity$_t$.

Definition 6.7 (Expressivity$_t$)**.** A logic $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ is at least as *expressive*$_t$ as a logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ if there is a truth-preserving translation from $\mathcal{L}_1$ to $\mathcal{L}_2$.

The prototypical translations are all truth-preserving, so they show that one logic is at least as expressive$_t$ as another. Unfortunately, expressivity$_t$ is not a very useful concept: the subscript t stands for "trivial", since almost every logic is trivially at least as expressive$_t$ as almost every other logic. The problem is that most logics have a countably infinite number of formulas as well as a countably infinite number of propositional variables. This allows us to translate every formula of one logic to a propositional variable of the other logic. Consider the following translation, for example.

Example 6.4. Let $\mathcal{L} = (\Phi, \mathfrak{M}, \models)$ be any logic such that Φ is at most countably infinite. Furthermore, let $\mathcal{L}_{\text{tm}} = (\Phi_{\text{tm}}, \mathfrak{M}_{\text{tm}}, \models_{\text{tm}})$ be a logic with a countable set of propositional variables but no connectives whatsoever that is evaluated on models with possible worlds. So a model of $\mathcal{L}_{\text{tm}}$ is a set of possible worlds together with a valuation.

The *first trivial translation from* $\mathcal{L}$ is the pair $(t_{\text{tm}}, f_{\text{tm}})$. Here t_{tm} maps a formula $\varphi \in \Phi$ to a propositional variable p_φ. Furthermore, f_{tm} maps models $\mathcal{M} \in \mathfrak{M}$ to a model $\mathcal{M}' \in \mathfrak{M}_{\text{tm}}$ by taking the set of possible worlds from $\mathcal{M}$ (stripping away all other structure of the model) together with the valuation such that $v(p_\varphi) = \{w \in \mathcal{M} \mid \mathcal{M}, w \models \varphi\}$. This $(t_{\text{tm}}, f_{\text{tm}})$ is a truth-preserving translation so $\mathcal{L}_{\text{tm}}$ is at least as expressive$_t$ as $\mathcal{L}$.

Clearly this trivial translation should not show that $\mathcal{L}_{\text{tm}}$ is at least as "expressive" as another logics. So expressivity$_t$ is not the right generalization of expressivity. This also means that expressivity$_t$ lacks the ability to explain why the prototypical results are interesting.

The prototypical results are truth preserving translations, but they are not trivial like the first trivial translation. So in addition to being truth preserving, these translations satisfy one or more other conditions that the trivial translation does not. In order to define expressivity$_g$ we should find these other conditions.

6.6 Desiderata for Expressivity$_g$

We are looking for additional constraints $X_1, \cdots, X_n$ on translations such that $\mathcal{L}_2$ is at least as expressive$_g$ are $\mathcal{L}_1$ if and only if there is a translation from $\mathcal{L}_1$ to $\mathcal{L}_2$ that is truth preserving and $X_1, \cdots, X_n$. But before we can choose such conditions we must first decide what properties expressivity$_g$ should have. Let us call these desired properties the desiderata for expressivity$_g$.

The first and most important desideratum is that the results from [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] should be expressivity$_g$ results. After all, we want to explain why these results are interesting by showing that they are expressivity$_g$ results.

We first want to introduce all the desiderata for expressivity$_g$ so we do not discuss these translations in detail here. In order to evaluate whether a proposed definition for expressivity$_g$ satisfies this desideratum, we do however need to discuss them in detail. We therefore fully define these translations in Section 6.7.

Desideratum 6.1 (Prototypical Translations)**.** These translations should show that their target logic is at least as expressive$_g$ as their source logic:

- The translation from temporal logic to modal logic introduced in [Thomason, 1974].
- The translation from classical (mono-)modal logic to normal (tri-)modal logic introduced in [Gasquet and Herzig, 1996].
- The translation from Alternating-time Temporal Epistemic Logic (ATEL) to Alternating-time Temporal Logic (ATL) introduced in [Goranko and Jamroga, 2004].
- The translation from Coalition Logic (CL) to a variant of STIT introduced in [Broersen et al., 2006a].
- The translation from ATL to a different variant of STIT introduced in [Broersen et al., 2006b].

The second desideratum is that trivial translations should not show that one logic is at least as expressive$_g$ as another. Such trivial translations include the first trivial translation defined in the previous section, but let us consider two more trivial translations.

Example 6.5. Let $\mathcal{L} = (\Phi, \mathfrak{M}, \models)$ be any logic such that Φ is at most countably infinite. Furthermore, let $\mathcal{L}_{\text{tp}}$ be a logic with a countable set of propositional variables but no connectives whatsoever that is evaluated on models that consist only of a valuation.

The *second trivial translation from* $\mathcal{L}$ is the pair $(t_{\text{tp}}, f_{\text{tp}})$. Here t_{tp} maps a formula $\varphi \in \Phi$ to a propositional variable p_φ. Furthermore, f_{tp} maps pointed models $\mathcal{M}, w \in \mathfrak{M}$ to the valuation such that $v(p_\varphi) = 1$ if and only if $\mathcal{M}, w \models \varphi$. This (t, f) is a truth-preserving translation so $\mathcal{L}_{\text{tp}}$ is at least as expressive$_{\text{tr}}$ as $\mathcal{L}$.

Example 6.6. Let $\mathcal{L} = (\Phi, \mathfrak{M}, \models)$ be any logic such that Φ is at most countably infinite. Furthermore, let $\mathcal{L}_{\text{m}}$ be a standard (mono-)modal logic with a countable set of propositional variables that is evaluated on standard relational models. So a model of $\mathcal{L}_{\text{m}}$ is a triple (W, R, v).

The *third trivial translation from* $\mathcal{L}$ is the pair $(t_{\text{m}}, f_{\text{m}})$. Here t_{m} maps a formula $\varphi \in \Phi$ to $\Box p_\varphi$ where p_φ is a propositional variable. Furthermore, for $\mathcal{M} \in \mathfrak{M}$ let W

be the set of possible worlds of $\mathcal{M}$ (stripping away all other structure of the model) and let $R = \{(w, w) \mid w \in W\}$.

Then f_{m} maps models $\mathcal{M} \in \mathfrak{M}$ to (W, R, v) with the valuation v such that $v(p_\varphi) = \{w \in \mathcal{M} \mid \mathcal{M}, w \models \varphi\}$. This (t, f) is a truth-preserving translation so $\mathcal{L}_{\mathrm{m}}$ is at least as expressive$_{\mathrm{tr}}$ as $\mathcal{L}$.

Like the first trivial translation, these second and third trivial translations are clearly not sufficient to show that one logic is at least as "expressive" as another in any interesting sense.

Desideratum 6.2 (Trivial Translations)**.** The first, second and third trivial translations do not show that their target logics are at least as expressive as their source logics.

The third desideratum is that expressivity$_g$ should be a generalization of expressivity. So existing expressivity results should also be expressivity$_g$ results. We could formalize this as the requirement that if t is a truth-preserving translation from $\mathcal{L}_1 = (\Phi_2, \mathfrak{M}, \models_2)$ to $\mathcal{L}_2 = (\Phi_1, \mathfrak{M}, \models_1)$, then (t, id) should show that $\mathcal{L}_2$ is at least as expressive$_g$ as $\mathcal{L}_1$ (where id is the identity map on $\mathfrak{M}$). Such a requirement is slightly too strong, however. I am not aware of any published truth preserving translation $t : \Phi_1 \to \Phi_2$ that is trivial in the way the three trivial translations are trivial. But there is no proof that such trivial t do not exist. Only in those cases where t is truth preserving and non-trivial should we demand that (t, id) count as evidence for $\mathcal{L}_2$ being at least as expressive as $\mathcal{L}_1$.

Ideally we would characterize the non-trivial truth preserving translations t. That would be very hard and somewhat outside the scope of this chapter though. Fortunately, a good approximation is available: the non-trivial t are those that have been published as expressivity results. The desideratum is therefore as follows.

Definition 6.8 (Traditional Translation)**.** Let $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}, \models_1)$ and $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}, \models_2$). A function $t : \Phi_1 \to \Phi_2$ is a *traditional translation* if it is truth preserving and it has been published as an expressivity result.

Desideratum 6.3 (Traditional Translations)**.** If t is a traditional translation from $\mathcal{L}_1$ to $\mathcal{L}_2$ then (t, id) shows that $\mathcal{L}_2$ is at least as expressive$_g$ as $\mathcal{L}_1$.

In addition to the prototypical and traditional translations there is one more class of translations that should count as evidence for one logic being at least as expressive as another. This is the class of translations that only change the notation of the logic. Consider the following example.

Example 6.7. Suppose $\mathcal{L}_1 = (\Phi, \mathfrak{M}_1, \models_1)$ and $\mathcal{L}_2 = (\Phi, \mathfrak{M}_2, \models_2)$ are both basic modal logic but that the models of $\mathcal{L}_1$ are triple (W, R, v) whereas the models of $\mathcal{L}_2$ are triples (R, W, v). Now let id be the identity map on Φ and let $f : \mathfrak{M}_1 \to \mathfrak{M}_2$ be given by $f(W, R, v) = (R, W, v)$.

The classes $\mathfrak{M}_1$ and $\mathfrak{M}_2$ of models in the example are not identical so, strictly speaking, we cannot say that $\mathcal{L}_1$ and $\mathcal{L}_2$ are equally expressive (following the traditional definition). But it is clear that the difference between the models (W, R, v) and (R, W, v) is purely notational in nature, so even though we cannot say that $\mathcal{L}_1$ and $\mathcal{L}_2$ are equally expressive we *want* to say that $\mathcal{L}_1$ and $\mathcal{L}_2$ are equally expressive.

Unlike expressivity, expressivity$_g$ does allow us to compare logics with different classes of models. So we can say that the above $\mathcal{L}_1$ and $\mathcal{L}_2$ are equally expressive$_g$. This

is exactly what we will do: if the $\mathcal{L}_1$ and $\mathcal{L}_2$ are merely notational variants of each other, then the translation between them should show that the logics are equally expressive$_g$. Of course, much like with the trivial and traditional translations, there is no easy way to characterize exactly those translations that only make notational changes. We will deal with this by considering a number of clear examples and requiring that expressivity$_g$ works as desired for those examples. Hopefully it will then work for other notational variants as well. The first of these examples is the translation between (W, R, v) and (R, W, v) given above, which we will call the *first notation changing translation*. That particular notation changing translation is a bijection between the two classes of models. But we can also find examples that are not surjective or injective.

Example 6.8. Let $\mathcal{L}_3 = (\Phi, \mathfrak{M}_3, \models_3)$ and $\mathcal{L}_4 = (\Phi, \mathfrak{M}_4, \models_4)$ both be basic modal logic evaluated on finite models. However, let $\mathfrak{M}_3$ be the class of all finite Kripke models (W, R, v) while $\mathcal{L}_4$ is the class of all finite Kripke models (W', R, v) where $W' = \{w_1, \cdots, w_n\}$ with n being the number of worlds in the model.

The *second notation changing translation* is the pair (id, f) where $f : \mathfrak{M}_3 \to \mathfrak{M}_4$ simply renames the worlds of a model (W, R, v) to $\{w_1, \cdots, w_{|W|}\}$. The *third notation changing translation* is the pair (id, f') where $f' : \mathfrak{M}_4 \to \mathfrak{M}_3$ is the inclusion map.

Note that the second notation changing translation is not injective while the third notation changing translation is not surjective. We could also quite easily define a notation changing translation that is neither injective nor surjective.

Desideratum 6.4 (Notation Changing Translations)**.** The first, second and third notation changing translations show that their source logics are at least as expressive$_g$ as their traget logics.

We then arrive at the final desideratum, which is something of a sanity check. A logic $\mathcal{L}_2$ is supposed to be at least as expressive$_g$ as a logic $\mathcal{L}_1$ if and only if everything that can be said in $\mathcal{L}_1$ can also be said in $\mathcal{L}_2$. The "everything that can be said in X can also be said in Y" relation is transitive and reflexive, so expressivity$_g$ should also be transitive and reflexive (and therefore a preorder). Additionally, we should demand that it is a non-trivial preorder, in the sense that there are logics $\mathcal{L}_2$ and $\mathcal{L}_1$ such that $\mathcal{L}_2$ is not at least as expressive$_g$ as $\mathcal{L}_1$.[2]

Ideally, expressivity$_g$ would be transitive because $(t_2 \circ t_2, f_2 \circ f_1)$ shows that $\mathcal{L}_3$ is at least as expressive$_g$ as $\mathcal{L}_1$ whenever (t_1, f_1) shows $\mathcal{L}_2$ to be at least as expressive$_g$ as $\mathcal{L}_1$ and (t_2, f_2) shows $\mathcal{L}_3$ to be at least as expressive$_g$ as $\mathcal{L}_2$, and reflexive because (id, id) shows that $\mathcal{L}$ is as expressive$_g$ as itself. The desideratum does not require this to be the case, but it will turn out to be true nonetheless.

Desideratum 6.5 (Non-trivial Preorder)**.** If $\mathcal{L}_3$ is at least as expressive$_g$ as $\mathcal{L}_2$ and $\mathcal{L}_2$ is at least as expressive$_g$ as $\mathcal{L}_1$ then $\mathcal{L}_3$ is at least as expressive$_g$ as $\mathcal{L}_1$. Furthermore, every logic is at least as expressive$_g$ as itself. Finally, there are logics $\mathcal{L}, \mathcal{L}'$ such that $\mathcal{L}$ is not at least as expressive as $\mathcal{L}'$.

If we can define a concept of expressivity$_g$ that satisfies all of these desiderata, then I am confident that this expressivity$_g$ can explain why the prototypical results are interesting.

[2]Note that the non-triviality of expressivity$_g$ does not follow immediately from Desideratum 6.2. That desideratum demands that the trivial translations do not show one logic to be at least as expressive$_g$ as another. But in theory there could be other, nontrivial, translations between the same logics.

6.7 Existing Translations

Desideratum 6.1 given above states that five specific translations should prove that one logic is more expressive$_g$ than another, because these translations should be seen as prototypical examples of expressivity$_g$ results. Recall that the translations in question are the following.

- The translation from temporal logic to modal logic introduced in [Thomason, 1974].
- The translation from classical (mono-)modal logic to normal (tri-)modal logic introduced in [Gasquet and Herzig, 1996].
- The translation from Alternating-time Temporal Epistemic Logic (ATEL) to Alternating-time Temporal Logic (ATL) introduced in [Goranko and Jamroga, 2004].
- The translation from Coalition Logic (CL) to a variant of STIT introduced in [Broersen et al., 2006a].
- The translation from ATL to a different variant of STIT introduced in [Broersen et al., 2006b].

In order to see whether a proposed definition of expressivity$_g$ satisfies Desideratum 6.1 we need to consider these translations in some detail. In this section we therefore give a short introduction to the translations.

Most of the logics in question were introduced long before the translation between them. In some cases there are significant differences between the definition of a logic as given by the publication in which it was introduced and the definition of the logic as given by the publication in which the translation was introduced. Here we define the logics the same way (modulo some notation) as in the publication in which the translation was introduced.

Recall that we abuse notation by writing $\mathcal{M} \in \mathfrak{M}$ and $w \in \mathcal{M}$ when $\mathcal{M}, w \in \mathfrak{M}$. This allows us to be significantly more concise when introducing the model translations $f : \mathfrak{M}_1 \to \mathfrak{M}_2$.

6.7.1 Embedding Temporal Logic in Modal Logic

In [Thomason, 1974] a translation is introduced from Temporal Logic to Modal Logic. Modulo some notation[3] the logics are defined in [Thomason, 1974] as follows.

Definition 6.9 (Temporal Logic)**.** Temporal logic $\mathcal{L}_{\mathrm{t}}$ is the triple $(\Phi_{\mathrm{t}}, \mathfrak{M}_{\mathrm{t}}, \models)$. Here Φ_{t} is given recursively by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \mathrm{P}\varphi \mid \mathrm{F}\varphi$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables. We use $\wedge$, $\rightarrow$, $\leftrightarrow$, $\top$ and $\bot$ in the usual way as abbreviations.

Furthermore, $\mathfrak{M}_{\mathrm{t}}$ is the class of triples $\mathcal{M} = (W, <, v)$ where W is a set of possible worlds, $<\subseteq W \times W$ is a binary relation on W and $v : \mathcal{P} \to \wp W$ is a valuation. Finally, $\models$ is given recursively by

[3]Note that we can ignore notational differences thanks to the fact that we require expressivity$_g$ to be transitive and to allow notation changing translations.

$$\begin{array}{lcl}
\mathcal{M}, w \models p & \Leftrightarrow & w \in v(p) \text{ for } p \in \mathcal{P}, \\
\mathcal{M}, w \models \neg\varphi & \Leftrightarrow & \mathcal{M}, w \not\models \varphi, \\
\mathcal{M}, w \models \varphi_1 \vee \varphi_2 & \Leftrightarrow & \mathcal{M}, w \models \varphi_1 \text{ or } \mathcal{M}, w \models \varphi_2, \\
\mathcal{M}, w \models \mathrm{P}\varphi & \Leftrightarrow & \mathcal{M}, w' \models \varphi \text{ for some } w' \in W \text{ such that } w' < w, \\
\mathcal{M}, w \models \mathrm{F}\varphi & \Leftrightarrow & \mathcal{M}, w' \models \varphi \text{ for some } w' \in W \text{ such that } w < w'.
\end{array}$$

In the definitions of the other logics in this chapter, we omit the obvious clauses in the definition of $\models$ and only mention the relevant ones—usually the ones for the modal operators.

Note that we have not put any restrictions on $<$ of the type one would expect of a temporal logic. In particular, $<$ need not be a strict partial order.[4]

Definition 6.10 (Modal Logic)**.** Modal Logic $\mathcal{L}_{\mathrm{m}}$ is the triple $(\Phi_{\mathrm{m}}, \mathfrak{M}_{\mathrm{m}}, \models)$. Here Φ_{m} is given by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \Diamond\varphi$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables. We use $\wedge, \rightarrow, \leftrightarrow, \top, \bot$ and $\Box$ in the usual way as abbreviations.

Furthermore, $\mathfrak{M}_{\mathrm{m}}$ is the class of triples $\mathcal{M} = (W, R, v)$ where W is a set of possible worlds, $R \subseteq W \times W$ is a binary relation on W and $v : \mathcal{P} \rightarrow \wp W$ is a valuation. Finally, the relevant clause of $\models$ is given by

$$\mathcal{M}, w \models \Diamond\varphi \quad \Leftrightarrow \quad \mathcal{M}, w' \models \varphi \text{ for some } w' \in W \text{ such that } (w, w') \in R.$$

Before giving the translation from $\mathcal{L}_{\mathrm{t}}$ to $\mathcal{L}_{\mathrm{m}}$, let us first give names to a few formulas. Fix some $p \in \mathcal{P}$ and let

$$\begin{aligned}
\psi_0 &= \Box(p \wedge \neg p) \\
\psi_1 &= \Diamond\psi_0 \\
\psi_2 &= \neg\psi_0 \wedge \neg\psi_1
\end{aligned}$$

As one final thing before considering the translation, let us define the disjoint union of a set Q with itself by $Q \uplus Q := Q \times \{+\} \cup Q \times \{-\}$. We denote elements $(q, +) \in Q \uplus Q$ by q^+ and $(q, -) \in Q \uplus Q$ by q^-. Now let us look at the translation between the logics.

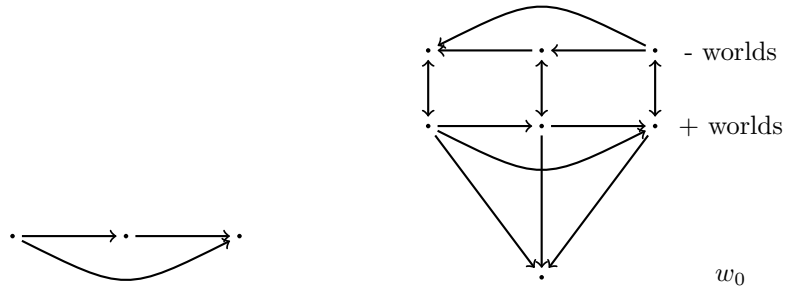
Definition 6.11 (Translation from $\mathcal{L}_{\mathrm{t}}$ to $\mathcal{L}_{\mathrm{m}}$)**.** The model translation $f : \mathfrak{M}_{\mathrm{t}} \rightarrow \mathfrak{M}_{\mathrm{m}}$ is given by $f(W, <, v) = (W', R, v')$, where

$$\begin{aligned}
W' &= W \uplus W \cup \{w_0\}, \\
R &= \{(w^+, w^-), (w^-, w^+), (w^+, w_0) \mid w \in W\} \\
&\quad \cup \{(w_1^+, w_2^+), (w_2^-, w_1^-) \mid w_1 < w_2\}, \\
v'(p) &= \{w^+, w^- \mid w \in v(p)\}.
\end{aligned}$$

The formula translation $t : \Phi_{\mathrm{t}} \rightarrow \Phi_{\mathrm{m}}$ is given by

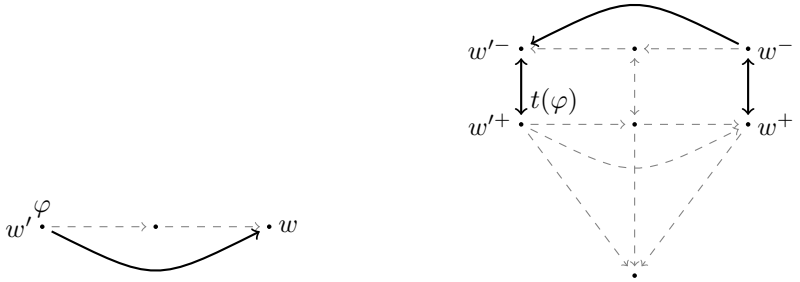
$$\begin{aligned}
t(p) &= \psi_1 \wedge p \\
t(\neg\varphi) &= \psi_1 \wedge \neg t(\varphi) \\
t(\varphi_1 \vee \varphi_2) &= t(\varphi_1) \vee t(\varphi_2) \\
t(\mathrm{F}\varphi) &= \psi_1 \wedge \Diamond t(\varphi) \\
t(\mathrm{P}\varphi) &= \psi_1 \wedge \Diamond(\psi_2 \wedge \Diamond(\psi_2 \wedge \Diamond t(\varphi))).
\end{aligned}$$

[4] As a result of the lack of restrictions on $<$, the two logics actually use the same class of models (modulo notation). The translation still doesn't show that $\mathcal{L}_{\mathrm{m}}$ is at least as expressive as $\mathcal{L}_{\mathrm{t}}$ according to Definition 1.1 though, as the model translation is not the identity map.



(a) $\mathcal{M}$. (b) $\mathcal{M}'$

Figure 6.1: A simple $\mathcal{L}_{\mathrm{t}}$ model $\mathcal{M}$ and its $\mathcal{L}_{\mathrm{m}}$ translation $\mathcal{M}' = f(\mathcal{M})$.



(a) $\mathcal{M}$. (b) $\mathcal{M}'$

Figure 6.2: Evaluation of $\mathrm{P}\varphi$ on $\mathcal{M}, w$ and $t(\mathrm{P}\varphi)$ on $\mathcal{M}', w^+$. Arrows that are not immediately relevant are drawn dashed an in gray.

This translation is truth-preserving.

Lemma 6.2. *For every $\mathcal{M} \in \mathfrak{M}_{\mathrm{t}}$, every $w \in \mathcal{M}$ and every $\varphi \in \Phi_{\mathrm{t}}$ we have $\mathcal{M}, w \models \varphi$ if and only if $f(\mathcal{M}), w^+ \models t(\varphi)$.*

Sketch of proof. Instead of a full proof let us consider an example that shows quite clearly why the translation works. Let $\mathcal{M} = (W, <, v)$ be a very simple model with three linearly ordered worlds, see Figure 6.1a.

The translated model $f(\mathcal{M}) = \mathcal{M}'$ shown in Figure 6.1b consists of three parts. First, there are the + worlds, which are a copy of $(W, <, v)$. Secondly, there are the − worlds, which are a copy of $(W, <^{-1}, v)$. Finally there is a single world w_0 that allows us to tell + and − worlds apart.

The world w_0 is the only one that has no outgoing arrows, so it is the only world satisfying ψ_0. This means that the + worlds, being the ones that can access w_0 are the ψ_1 worlds, leaving the − worlds as the ψ_2 worlds.

For propositional variables as well as the connectives $\neg, \vee$ and F, the translation t only adds a few ψ_1 clauses. This means that in the evaluation of a formula containing no P clauses we remain in the + part of the model. The + part is a copy of $(W, <, v)$, so the truth of such formulas is preserved.

Let us then consider a formula of the form $\mathrm{P}\varphi$. Suppose we evaluate this formula in the rightmost world w and that the leftmost world w' satisfies φ, see Figure 6.2.

The P operator acts like a $\Diamond$ but on the relation $<^{-1}$, so from the fact that $w' < w$ it follows that $\mathcal{M}, w \models \mathrm{P}\varphi$.

Now let us look at what happens if we evaluate $t(\mathrm{P}\varphi) = \psi_1 \wedge \Diamond(\psi_2 \wedge \Diamond(\psi_2 \wedge \Diamond t(\varphi)))$ on $\mathcal{M}', w^+$. We have $\mathcal{M}', w^+ \models \psi_1$ so we do not have to worry about the first conjunct ψ_1. So we get to the second conjunct, which starts with a $\Diamond$. We therefore have to take one of the arrows out of w^+. This arrow should end in a world satisfying (among others) ψ_2, so it should end in a $-$ world. The only $-$ world accessible from w^+ is w^-, so we go to w^-.

The question is then, do we have $\mathcal{M}, w^- \models \psi_2 \wedge \Diamond(\psi_2 \wedge \Diamond t(\varphi))$? We do have $\mathcal{M}, w^- \models \psi_2$ so we can focus on the second conjunct. So we should once again go to a different world, and once again this world should satisfy ψ_2. This means we should stay in the $-$ part. The relation in the minus part is a copy of $<^{-1}$, so we can go to w'^- because $w' < w$.

Let us then evaluate $\mathcal{M}, w'^- \models \psi_2 \wedge \Diamond t(\varphi)$. Every translated formula $t(\varphi)$ starts with a ψ_1 conjunct, so we should let the $\Diamond$ take us to a $+$ world. The only $+$ world accessible from w'^- is w'^+, so let's go there. And we have $\mathcal{M}, w'^+ \models t(\varphi)$.

It should be clear that the example can be generalized to all models, and that for all $\mathcal{M}, w$ and φ we have $\mathcal{M}, w \models \mathrm{P}\varphi$ if and only if $f(\mathcal{M}), w^+ \models t(\mathrm{P}\varphi)$. So the translation is truth preserving. □

The translation (t, f) from $\mathcal{L}_{\mathrm{t}}$ to $\mathcal{L}_{\mathrm{m}}$ should count as showing that modal logic is at least as expressive$_g$ as temporal logic. In fact nothing in the construction depends on the fact that the relation for P is the inverse of the relation for F. So we can easily construct very similar translations from any bi-modal logic to a mono-modal logic, and even from any n-modal logic to mono-modal logic. So if we accept that the translation given above shows that $\mathcal{L}_{\mathrm{m}}$ is at least as expressive$_g$ as $\mathcal{L}_{\mathrm{t}}$, we should also accept that $\mathcal{L}_{\mathrm{m}}$ is at least as expressive$_g$ as any basic multi-modal logic with a finite number of modalities.

6.7.2 Embedding Classical Modal Logic in Normal Modal Logic

Another translation that should count as showing that a logic is at least as expressive$_g$ as another is the one by [Gasquet and Herzig, 1996], which shows that classical (mono-)modal logic can be embedded in normal (tri-)modal logic.

Again modulo some notation these two logics are defined as follows.

Definition 6.12 (Classical modal logic)**.** Classical (mono-)modal logic $\mathcal{L}_{\mathrm{cm}}$ is the triple $(\Phi_{\mathrm{cm}}, \mathfrak{M}_{\mathrm{cm}}, \models)$. Here Φ_{cm} is given recursively by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \Box\varphi$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables. We use $\wedge, \rightarrow, \leftrightarrow, \top, \bot$ and $\Diamond$ in the usual way as abbreviations.

Furthermore, $\mathfrak{M}_{\mathrm{cm}}$ is the class of triples $\mathcal{M} = (W, N, v)$ where W is a set of possible worlds, $N : W \rightarrow \wp\wp W$ is a neighborhood function that assigns to each world a set of sets of worlds called a neighborhood and $v : \mathcal{P} \rightarrow \wp W$ is a valuation. Finally, the important clause of $\models$ is given by

$$\begin{array}{lcl} \mathcal{M}, w \models \Box\varphi & \Leftrightarrow & \exists S \in N(w) \text{ such that } \mathcal{M}, w' \models \varphi \text{ for all } w' \in S \\ & & \text{and } \mathcal{M}, w'' \not\models \varphi \text{ for all } w'' \in W \setminus S. \end{array}$$

Definition 6.13 (Normal tri-modal logic)**.** Normal (tri-)modal logic $\mathcal{L}_{\mathrm{n}}$ is the triple $(\Phi_{\mathrm{n}}, \mathfrak{M}_{\mathrm{n}}, \models)$. Here Φ_{n} is given recursively by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \Box_1\varphi \mid \Box_2\varphi \mid \Box_3\varphi$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables. We use $\wedge, \rightarrow, \leftrightarrow, \top, \bot$ and $\Diamond_i$ in the usual way as abbreviations.

Furthermore, $\mathfrak{M}_{\mathrm{n}}$ is the class of tuples $\mathcal{M} = (W, R_1, R_2, R_3, v)$, where W is a set of possible worlds, $R_i \subseteq W \times W$ is an accessibility relation for $1 \leq i \leq 3$ and $v : \mathcal{P} \rightarrow \wp W$ is a valuation. Finally, the important clause of $\models$ is given by

$$\mathcal{M}, w \models \Box_i\varphi \quad \Leftrightarrow \quad \mathcal{M}, w' \models \varphi \text{ for all } w' \text{ such that } (w, w') \in R_i$$

for $1 \leq i \leq 3$.

The translation is by letting the worlds of $\mathcal{M}' = f(\mathcal{M})$ be the worlds of $\mathcal{M}$ together with the sets of worlds of $\mathcal{M}$. The three modal operators $\Box_1, \Box_2, \Box_3$ can then encode a set of worlds being in the neighborhood of a world, a world being in a set of worlds and a world not being in a set of worlds respectively.

Definition 6.14 (Translation from $\mathcal{L}_{\mathrm{cm}}$ to $\mathcal{L}_{\mathrm{n}}$)**.** The model translation $f : \mathfrak{M}_{\mathrm{cm}} \rightarrow \mathfrak{M}_{\mathrm{n}}$ is given by

$$f(W, N, v) = (W', R_1, R_2, R_3, v')$$

where

$$\begin{aligned} W' &= W \cup \wp W, \\ R_1 &= \{(w, S) \mid w \in W, S \in \wp W, S \in N(w)\} \\ R_2 &= \{(S, w) \mid S \in \wp W, w \in W, w \in S\} \\ R_3 &= \{(S, w) \mid S \in \wp W, w \in W, w \notin S\} \\ v'(p) &= v(p). \end{aligned}$$

The formula translation $t : \Phi_{\mathrm{cm}} \rightarrow \Phi_{\mathrm{n}}$ is given by

$$\begin{aligned} t(p) &= p \\ t(\neg\varphi) &= \neg t(\varphi) \\ t(\varphi_1 \vee \varphi_2) &= t(\varphi_1) \vee t(\varphi_2) \\ t(\Box\varphi) &= \Diamond_1(\Box_2 t(\varphi) \wedge \Box_3 \neg t(\varphi)) \end{aligned}$$

This translation is also truth preserving.

Lemma 6.3. *For every $\mathcal{M} \in \mathfrak{M}_{\mathrm{cm}}$, every $w \in \mathcal{M}$ and every $\varphi \in \Phi_{\mathrm{cm}}$ we have $\mathcal{M}, w \models \varphi$ if and only if $f(\mathcal{M}), w \models t(\varphi)$.*

Sketch of proof. Fix any $\mathcal{M} \in \mathfrak{M}_{\mathrm{cm}}$, $w \in \mathcal{M}$ and $\varphi \in \Phi_{\mathrm{cm}}$. It should be immediately clear that (t, f) is truth-preserving for operators other than $\Box$. So let us focus on the translation of formulas of the form $\Box\varphi$.

Recall that $\mathcal{M}, w \models \Box\varphi$ if and only if there is an $S \in N(w)$ such that $\mathcal{M}, w' \models \varphi$ for all $w' \in S$ and $\mathcal{M}, w'' \not\models \varphi$ for all $w'' \in W \setminus S$.

The relation R_1 connects w to all sets S that are in $N(w)$. So $\mathcal{M}, w \models \Box\varphi$ if and only if there is an $S \in W'$ such that $(w, S) \in R_1$, $\mathcal{M}, w' \models \varphi$ for all $w' \in S$ and $\mathcal{M}, w'' \not\models \varphi$ for all $w'' \in W \setminus S$.

The relations R_2 and R_3 connect S to those worlds that are in S and those worlds that are not in S, respectively. So $\mathcal{M}, w \models \Box\varphi$ if and only if there is an $S \in W'$ such that $(w, S) \in R_1$ and for every $w', w'' \in W'$ if $(S, w') \in R_2$ then $\mathcal{M}, w' \models \varphi$ and if $(S, w'') \in R_3$ then $\mathcal{M}, w'' \not\models \varphi$.

But that condition is equivalent to $f(\mathcal{M}), w \models \Diamond_1(\Box_2 t(\varphi) \wedge \Box_3 \neg t(\varphi))$, so to $f(\mathcal{M}), w \models t(\Box\varphi)$. □

This translation should count as evidence for $\mathcal{L}_{\mathrm{n}}$ being at least as expressive$_g$ as $\mathcal{L}_{\mathrm{cm}}$. And of course by the result in the previous subsection, we should then also say that $\mathcal{L}_{\mathrm{m}}$ is at least as expressive$_g$ as $\mathcal{L}_{\mathrm{cm}}$.

6.7.3 Embedding ATEL in ATL

The third existing translation that should count as showing that one logic is at least as expressive$_g$ as another is the translation from Alternating-time Temporal Epistemic Logic to Alternating-time Temporal Logic given in [Goranko and Jamroga, 2004]. Let us once again start with the definitions of the logics in question. It is convenient to start with the simpler logic ATL and introduce ATEL after that, so that is what we will do. The definitions given here require the set of worlds of every model to be finite. This is not strictly necessary, but doing so simplifies the translation a lot.

Definition 6.15 (ATL)**.** Alternating-time Temporal Logic (ATL) is the logic $\mathcal{L}_{\mathrm{ATL}} = (\Phi_{\mathrm{ATL}}, \mathfrak{M}_{\mathrm{ATL}}, \models)$. Here Φ_{ATL} is given by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \langle\langle A\rangle\rangle X\varphi \mid \langle\langle A\rangle\rangle G\varphi \mid \langle\langle A\rangle\rangle(\varphi U \varphi)$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables, a ranging over a finite set $\mathcal{A} = \{a_1, \cdots, a_k\}$ of agents and A ranging over $\wp\mathcal{A}$. We use $\wedge, \rightarrow, \leftrightarrow, \top, \bot, \bigwedge$ and $\bigvee$ in the usual way as abbreviations.

Furthermore, $\mathfrak{M}_{\mathrm{ATL}}$ is the class of tuples $\mathcal{M} = (W, \Sigma, o, v)$, where W is a finite set of possible worlds, $\Sigma : W \times \mathcal{A} \rightarrow \mathbb{N}$ is a function that returns the number of choices available to an agent at a world, $o : W \times \mathbb{N}^k \rightarrow W$ is an outcome function and $v : \mathcal{P} \rightarrow \wp W$ is a valuation.

A *choice* for a set A of agents is a function $c_A : W \times A \rightarrow \mathbb{N}$ with the property that for all $w \in W$ and $a \in A$ we have $c_A(w, a) \leq \Sigma(w, a)$. A completion of a choice c_A is a choice $c_{\mathcal{A}}$ for the entire set of agents that is equal to c_A when restricted to $W \times A$. The set of all choices for A is denoted C_A.

A *strategy* for a set A of agents is a function $s_A : W^+ \rightarrow C_A$ that assigns to each finite sequence of worlds a choice for A. A completion of a strategy s_A is a strategy $s_{\mathcal{A}}$ such that for each finite sequence $w^+ \in W^+$ the choice $s_{\mathcal{A}}(w^+)$ is a completion of the choice $s_A(w^+)$.

The *outcome* $O(w, c_{\mathcal{A}})$ of a choice $c_{\mathcal{A}}$ in a world w is the world $o(w, c_{\mathcal{A}}(w, a_1), \cdots, c_{\mathcal{A}}(w, a_k))$. The *computation* $\Omega(w_0, s_{\mathcal{A}})$ associated with a world w_0 and a strategy $s_{\mathcal{A}}$ is the unique sequence $(w_0, w_1, w_2, \cdots)$ such that for each $i \in \mathbb{N}$ we have $w_{i+1} = O(w_i, s_{\mathcal{A}}(w_0, \cdots, w_i))$.

Finally the important clauses of $\models$ are given by

$$
\begin{array}{lcl}
\mathcal{M}, w \models \langle\langle A\rangle\rangle X\varphi & \Leftrightarrow & \text{there is a choice } c_A \text{ such that for every} \\
 & & \text{completion } c_{\mathcal{A}} \text{ of } c_A \text{ we have } \mathcal{M}, O(w, c_{\mathcal{A}}) \models \varphi, \\
\mathcal{M}, w \models \langle\langle A\rangle\rangle G\varphi & \Leftrightarrow & \text{there is a strategy } s_A \text{ such that for every} \\
 & & \text{completion } s_{\mathcal{A}} \text{ of } s_A \text{ we have } \mathcal{M}, w' \models \varphi \\
 & & \text{for every } w' \text{ in } \Omega(w, s_{\mathcal{A}}), \\
\mathcal{M}, w \models \langle\langle A\rangle\rangle \varphi_1 U \varphi_2 & \Leftrightarrow & \text{there is a strategy } s_A \text{ such that for every} \\
 & & \text{completion } s_{\mathcal{A}} \text{ of } s_A \text{ there is some} \\
 & & w_i \in \Omega(w, s_{\mathcal{A}}) = (w_0, w_1, \cdots) \text{ with the} \\
 & & \text{property that } \mathcal{M}, w_j \models \varphi_1 \text{ for all } w_j \\
 & & \text{with } j < i \text{ and } \mathcal{M}, w_i \models \varphi_2.
\end{array}
$$

These semantics are unfortunately a bit complicated, but they can be made clearer by mentioning the intuition behind them. We have $\mathcal{M}, w \models \langle\langle A\rangle\rangle X\varphi$ iff there is some choice for the group A that guarantees that no matter the choice of $\mathcal{A} \setminus A$, the next world will satisfy φ.

We have $\mathcal{M}, w \models \langle\langle A\rangle\rangle G\varphi$ iff there is a strategy for the group A that guarantees that no matter the strategy of $\mathcal{A} \setminus A$, all worlds that will ever be reached in the future will satisfy φ.[5]

We have $\mathcal{M}, w \models \langle\langle A\rangle\rangle \varphi_1 U \varphi_2$ iff there is a strategy for the group A that guarantees that no matter the strategy of $\mathcal{A} \setminus A$ there will be some world in the future where φ_2 holds, and φ_1 will hold in every world up to that φ_2 world.[6]

With ATL introduced let us consider ATEL.

Definition 6.16 (ATEL)**.** Alternating-time Temporal Logic (ATEL) is the logic $\mathcal{L}_{\text{ATEL}} = (\Phi_{\text{ATEL}}, \mathfrak{M}_{\text{ATEL}}, \models)$. Here Φ_{ATEL} is given by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \langle\langle A\rangle\rangle X\varphi \mid \langle\langle A\rangle\rangle G\varphi \mid \langle\langle A\rangle\rangle(\varphi U \varphi) \mid K_a\varphi \mid C_A\varphi \mid D_A\varphi$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables, a ranging over a finite set $\mathcal{A} = \{a_1, \cdots, a_k\}$ of agents and A ranges over $\wp\mathcal{A}$. We use $\wedge, \rightarrow, \leftrightarrow, \top, \bot, \bigwedge$ and $\bigvee$ in the usual way as abbreviations. Furthermore, we use $E_A\varphi$ as an abbreviation for $\bigwedge_{a\in A} K_a\varphi$.

Furthermore, $\mathfrak{M}_{\text{ATEL}}$ is the class of tuples $\mathcal{M} = (W, \Sigma, o, R, v)$, where $(W, \Sigma, o, v) \in \mathfrak{M}_{\text{ATL}}$ and $R : \mathcal{A} \rightarrow \wp(W \times W)$ assigns to each agent an equivalence relation on W.

The semantics for the operators that also exist in $\mathcal{L}_{\text{ATL}}$ are as in $\mathcal{L}_{\text{ATL}}$. The clauses of $\models$ for the other operators are given by

$$
\begin{array}{lcl}
\mathcal{M}, w \models K_a\varphi & \Leftrightarrow & \mathcal{M}, w' \models \varphi \text{ for all } w' \text{ such that } (w, w') \in R(a), \\
\mathcal{M}, w \models C_A\varphi & \Leftrightarrow & \mathcal{M}, w' \models \varphi \text{ for all } w' \text{ such that } (w, w') \text{ is} \\
 & & \text{in the transitive closure of } \bigcup_{a\in A} R(a). \\
\mathcal{M}, w \models D_A\varphi & \Leftrightarrow & \mathcal{M}, w' \models \varphi \text{ for all } w' \text{ such that } (w, w') \in \bigcup_{a\in A} R(a).
\end{array}
$$

Now that we have definitions of the two logics, we can define the translation between them that was introduced in [Goranko and Jamroga, 2004]. One thing to note about this translation is that it uses different sets of agents for $\mathcal{L}_{\text{ATEL}}$ and $\mathcal{L}_{\text{ATL}}$. In fact, the set of agents we use for $\mathcal{L}_{\text{ATL}}$ is slightly larger than the powerset of the agents of $\mathcal{L}_{\text{ATEL}}$. It is therefore important that we take the set of agents to be finite and not, say, countably infinite.

[5] Actually, to be slightly more precise, φ should hold on the current world as well as all worlds that will ever be reached in the future.

[6] Again, the future might include the present, the φ_2 world might be w itself.

Definition 6.17 (Translation from ATEL to ATL)**.** Let $\mathcal{A} = \{a_1, \cdots, a_k\}$ be the set of agents in $\mathcal{L}_{\mathrm{ATEL}}$ and let $\mathcal{A}_e = \wp(\mathcal{A}) \setminus \{\emptyset\} = \{e_1, \cdots, e_m\}$. The set $\mathcal{A}'$ of $\mathcal{L}_{\mathrm{ATL}}$ agents is then given by $\mathcal{A}' = \mathcal{A} \cup \mathcal{A}_e$.

The sets of propositional variables for the two logics can be taken the same, but it is notationally convenient to take the set of propositional variables of $\mathcal{L}_{\mathrm{ATEL}}$ to be $\mathcal{P}$ and the set of propositional variables of $\mathcal{L}_{\mathrm{ATL}}$ to be $\mathcal{P}' = \mathcal{P} \cup \{p_e \mid e \in \mathcal{A}_e\}$.

For every $e \in \mathcal{A}_e$ and every w fix some ordering for those worlds that are accessible from w for all $a \in e$, so $\{w' \mid (w, w') \in \bigcap_{a \in e} R(a)\} = \{w_1, \cdots, w_n\}$ for some $n \in \mathbb{N}$. The model translation $f : \mathfrak{M}_{\mathrm{ATEL}} \to \mathfrak{M}_{\mathrm{ATL}}$ is then given by $f(W, \Sigma, o, R, v) = (W', \Sigma', o', v')$ where

$$
\begin{aligned}
W' &= W \cup \{(w, e) \mid w \in W, e \in \mathcal{A}_e\} \\
\Sigma'(w, a) &= \Sigma(w, a) \text{ for } a \in \mathcal{A} \\
\Sigma'((w, e), a) &= \Sigma(w, a) \text{ for } a \in \mathcal{A} \text{ and } e \in \mathcal{A}_e \\
\Sigma'(w, e) &= 1 + |\{w' \mid (w, w') \in \bigcap_{a \in e} R(a)\}| \text{ for } e \in \mathcal{A}_e \\
\Sigma'((w, e'), e) &= \Sigma'(w, e') \text{ for } e, e' \in \mathcal{A}_e \\
o'(w, x_1, \cdots, x_{k+m}) &= o(w, x_1, \cdots, x_k) \text{ if } x_j = 0 \text{ for all } j > k \\
o'(w, x_1, \cdots, x_{k+m}) &= (w_{x_j}, e_i) \text{ where } w_{x_j} \in \{w' \mid (w, w') \in \bigcap_{a \in e} R(a)\} \\
&\qquad \text{if } j > k, x_j \neq 0 \text{ and } x_l = 0 \text{ for all } l > j, \\
o'((w, e'), x_1, \cdots, x_{k+m}) &= o'(w, x_1, \cdots, x_{k+m}) \text{ for } e' \in \mathcal{A}_e \\
v'(p) &= v(p) \cup \{(w, e) \mid w \in v(p), e \in \mathcal{A}_e\} \text{ for } p \in \mathcal{P} \\
v'(p_e) &= \{(w, e) \mid w \in W\} \text{ for } e \in \mathcal{A}_e
\end{aligned}
$$

Let $act = \bigwedge_{e \in \mathcal{A}_e} \neg p_e$. The formula translation $t : \Phi_{\mathrm{ATEL}} \to \Phi_{\mathrm{ATL}}$ is then given by

$$
\begin{aligned}
t(p) &= p \text{ for } p \in \mathcal{P} \\
t(\neg\varphi) &= \neg t(\varphi) \\
t(\varphi_1 \vee \varphi_2) &= t(\varphi_1) \vee t(\varphi_2) \\
t(\langle\langle A \rangle\rangle X\varphi) &= \langle\langle A \cup \mathcal{A}_e \rangle\rangle X(act \wedge t(\varphi)) \\
t(\langle\langle A \rangle\rangle G\varphi) &= t(\varphi) \wedge \langle\langle A \cup \mathcal{A}_e \rangle\rangle X \langle\langle A \cup \mathcal{A}_e \rangle\rangle G(act \wedge t(\varphi)) \\
t(\langle\langle A \rangle\rangle \varphi_1 U \varphi_2) &= t(\varphi_2) \vee (t(\varphi_1) \wedge \langle\langle A \cup \mathcal{A}_e \rangle\rangle X \\
&\qquad \langle\langle A \cup \mathcal{A}_e \rangle\rangle (act \wedge t(\varphi_1)) U (act \wedge t(\varphi_2))) \\
t(K_a\varphi) &= \neg\langle\langle \mathcal{A}_e \rangle\rangle X(p_{\{a\}} \wedge \neg t(\varphi)) \\
t(C_A\varphi) &= \neg\langle\langle \mathcal{A}_e \rangle\rangle X \langle\langle \mathcal{A}_e \rangle\rangle (\bigvee_{a \in A} p_{\{a\}}) U (\neg t(\varphi) \wedge \bigvee_{a \in A} \{a\}) \\
t(D_A\varphi) &= \neg\langle\langle \mathcal{A}_e \rangle\rangle X(p_A \wedge \neg t(\varphi))
\end{aligned}
$$

Note that the $p_{\{a\}}$ in the translations of $K_a\varphi$ and $C_A\varphi$ and the p_A in the translation of $D_A\varphi$ are propositional variables, as we took $\mathcal{P}' = \mathcal{P} \cup \{p_e \mid e \in \mathcal{A}_e\}$.

This translation (t, f) is a little more complicated than the other translations discussed so far. But it is truth-preserving.

Lemma 6.4. *For every $\mathcal{M} \in \mathfrak{M}_{\mathrm{ATEL}}$, every $w \in \mathcal{M}$ and every $\varphi \in \Phi_{\mathrm{ATEL}}$ we have $\mathcal{M}, w \models \varphi$ if and only if $f(\mathcal{M}), w \models t(\varphi)$.*

Sketch of Proof. It is not too hard to see why (t, f) is truth-preserving, if we first look at exactly what the translation actually does.

The intuition behind the translation is as follows. We took $\mathcal{A}'$ to be $\mathcal{A} \cup \mathcal{A}_e$. The $\mathcal{A} \subset \mathcal{A}'$ remain the agents that do actual actions. The epistemic agents $\mathcal{A}_e = \wp(\mathcal{A}) \setminus \emptyset$ on the other hand get to do epistemic actions; an agent $e \in \mathcal{A}_e$ is (sometimes) allowed to go from a world w to any world w' that is accessible from w for every $a \in e$ (in $\mathcal{M}$).

This is one of the places where the assumption of finite models comes in handy. It implies that the set $\{w' \mid (w, w') \in \bigcap a \in eR(a)\}$ is finite, so we can assign these worlds the numbers $1, \cdots, n$ for some n. This means we can take the set of choices for this epistemic agent to be $\{0, 1, \cdots, n\}$ where 0 is a "pass" choice and each other choice i is a vote for going to world w_i.

All agents get to choose their action independently, so we have to assign some kind of priority system to which agent e can perform an epistemic action or whether the original agents $\mathcal{A} \subset \mathcal{A}'$ can perform an actual action. We assign this priority back to front. We first look at the final agent e_m. If e_m makes a choice $i \neq 0$ then the next state is w_i, but if e_m chooses 0 we look at the choice for e_{m-1}. If e_{m-1} then chooses $i \neq 0$ we go to the i-th world that is accessible for e_{m-1} but if it chooses 0 we look at e_{m-2} and so on. If all epistemic agents pass we ignore them and determine the outcome by looking at the original outcome function o.

The system as described so far already contains most of the epistemic structure from $\mathcal{M}$. If we want to translate a formula like $D_A\varphi$, we have to look up the i such that $e_i = A$. The formula $D_A\varphi$ then holds if and only if whenever e_i gets to choose the next world (so all e_j with $j > i$ pass) it is impossible for e_i to choose a world where the translation of φ does not hold.

Unfortunately there is a complication. In ATL we cannot express the condition that e_i is the agent that gets to choose a world. This is why we need $W' = W \cup \{(w, e) \mid w \in W, e \in \mathcal{A}_e\}$. The worlds w and (w, e) are almost entirely indistinguishable; the number of choices, the outcomes of the choices and the values of all but one of the propositional variables are the same. The only difference is that for each copy $W \times \{e\}$, there is a unique propositional variable p_e that holds on those worlds and only on those worlds.

These extra worlds allow us to express the condition that e_i chooses. The trick is that whenever an epistemic action is taken that should take us to some world w' we do not go to w' but to (w', e_i) where e_i is the epistemic agent that chose the world. If on the other hand all epistemic agents pass, we go to a world $w'' \in W \subseteq W'$.

The condition that e_i is the choosing agent then simply means that p_{e_i} holds in the next world. In this light let us consider the translations of the epistemic connective K_a.

We have $t(K_a\varphi) = \neg\langle\langle \mathcal{A}_e \rangle\rangle X(p_{\{a\}} \wedge \neg t(\varphi))$. Unraveling the meaning of this translation we get that $t(K_a\varphi)$ holds if and only if there is no collective choice for $\mathcal{A}_e$ that guarantees that the next state will be a $p_{\{a\}} \wedge \neg t(\varphi)$ one. Here the $p_{\{a\}}$ condition means that it is the singleton set $\{a\}$ that actually chose the world. So $t(K_a\varphi)$ holds if and only if there is no world that was accessible for a in $\mathcal{M}$ where $t(\varphi)$ holds. So as long as our translation is truth-preserving for φ it is also truth-preserving for $K_a\varphi$.

The translations for the other epistemic connectives work similarly. So let us take a look at the translations of the non-epistemic connectives. Consider $t(\langle\langle A \rangle\rangle X\varphi) = \langle\langle A \cup \mathcal{A}_e \rangle\rangle X(act \wedge t(\varphi))$. Unraveling the meaning of this translation we get that $t(\langle\langle A \rangle\rangle X\varphi)$ holds if and only if there is a collective choice for $A \cup \mathcal{A}_e$ that guarantees that $act \wedge t(\varphi)$ holds in the next world. But act is an abbreviation for $\bigwedge_{e \in \mathcal{A}_e} \neg p_e$ so

if *act* holds in the next world, all epistemic agents have to pass. This means that the next outcome of the choice by $A \cup \mathcal{A}_e$ in $\mathcal{M}'$ will be the same as the outcome of the choice by A in $\mathcal{M}$. If the translation is truth-preserving for φ, this implies that the translation is also truth-preserving for $\langle\langle A\rangle\rangle X\varphi$.

The translations for the other non-epistemic modal connectives work similarly.[7] The entire translation is therefore truth-preserving. □

This translation is probably not the most elegant one you have ever seen, but it should count as showing that $\mathcal{L}_{\mathrm{ATL}}$ is at least as expressive$_g$ as $\mathcal{L}_{\mathrm{ATEL}}$.

6.7.4 Embedding CL in STIT

The final two translations that should count as evidence for one logic being at least as expressive$_g$ as another are the ones from Coalition Logic (CL) to a variant of STIT and from ATL to a different variant of STIT introduced in [Broersen et al., 2006a] and [Broersen et al., 2006b] respectively.

Introducing both translations would require quite a lot of extra definitions. Considering that the two translations are very similar to each other, we introduce only one of them here, namely the translation from CL to STIT from [Broersen et al., 2006a].

As usual, let us start with definitions. We already have a definition of ATL so let us consider the other logics.

Definition 6.18 (CL)**.** Coalition Logic (CL) is the logic $\mathcal{L}_{\mathrm{CL}} = (\Phi_{\mathrm{CL}}, \mathfrak{M}_{\mathrm{CL}}, \models)$. Here Φ_{CL} is given by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid [A]\varphi$$

with p ranging over a countable set $\mathcal{P}$ of propositional variables and A ranging over the powerset of a finite set $\mathcal{A}$ of agents. We use $\wedge, \rightarrow, \leftrightarrow, \top$ and $\bot$ in the usual way as abbreviations.

Furthermore, $\mathfrak{M}_{\mathrm{CL}} = \mathfrak{M}_{\mathrm{ATL}}$. Choices, strategies and outcomes are defined as in ATL. Finally, the relevant clause of $\models$ is given by

$$\mathcal{M}, w \models [A]\varphi \quad \Leftrightarrow \quad \mathcal{M}, w \models \langle\langle A\rangle\rangle X\varphi.$$

The variant of STIT that is used in [Broersen et al., 2006a] is slightly unusual in that it uses discrete time and a next operator, unlike most STIT logics. For influential examples of more usual variants of STIT, see for example [Belnap et al., 2001, Horty, 2001].

Definition 6.19 (STIT)**.** Discrete Seeing to it That Logic (STIT) is the logic $\mathcal{L}_{\mathrm{STIT}} = (\Phi_{\mathrm{STIT}}, \mathfrak{M}_{\mathrm{STIT}}, \models)$. Here Φ_{STIT} is given by

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \Box\varphi \mid X\varphi \mid [A \text{ cstit}]\varphi.$$

We use $\wedge, \rightarrow, \leftrightarrow, \top, \bot$ and $\Diamond$ in the usual way as abbreviations. Furthermore, $\mathfrak{M}_{\mathrm{STIT}}$ is the class of tuples $\mathcal{M} = (W, \leq, H, C, v)$ where

- W is a set of possible worlds,

[7] There is a small complication in the translations of $\langle\langle A\rangle\rangle G\varphi$ and $\langle\langle A\rangle\rangle\varphi_1 U\varphi_2$. The problem is that these connectives are reflexive. We cannot require the initial world to satisfy *act* as we may have reached it by, say, a K_a move. All other worlds in the computation should satisfy *act* though, as the actions in the computation should be non-epistemic. This problem is solved by treating the first world separately.

- $\leq$ is a partial order on W such that
  - for every $w_1, w_2 \in W$ there is a w_3 such that $w_3 \leq w_1$ and $w_3 \leq w_2$,
  - for every $w_1, w_2, w_3 \in W$ if $w_1 \leq w_3$ and $w_2 \leq w_3$ then either $w_1 \leq w_2$ or $w_2 \leq w_1$,
  - for every $w_1 \in W$ there is at least one $w_2 \in W$ such that $w_1 \leq w_2$, $w_2 \not\leq w_1$ and for every $w_3 \in W$ if $w_1 \leq w_3$ and $w_3 \leq w_2$ then $w_1 = w_3$ or $w_2 = w_3$.
- $H \subseteq \wp W$ is the set of maximal sets of linearly ordered worlds,
- $C : \mathcal{A} \times W \to \wp\wp H$ is a function that maps each (a, w) to a partition of $\{h \in H \mid w \in h\}$ with the property that for every $\{a_1, \cdots, a_n\} \subseteq \mathcal{A}$, every $w \in W$ and every $S_1, \cdots, S_n$ such that $S_i \in C(a_i, w)$ for $1 \leq i \leq n$ the set $\bigcap_{1 \leq i \leq n} S_i$ is nonempty,
- $v : \mathcal{P} \to \wp(W \times H)$ is a valuation function.

Unlike the other logics where points of pointed models are elements of W, points of pointed STIT models are pairs w/h where $w \in W, h \in H$ and $w \in h$. The relevant clauses of $\models$ are given by

$$
\begin{array}{lcl}
\mathcal{M}, w/h \models p & \Leftrightarrow & (w,h) \in v(p) \text{ for } p \in \mathcal{P}, \\
\mathcal{M}, w/h \models \Box\varphi & \Leftrightarrow & \mathcal{M}, w/h' \models \varphi \text{ for all } h' \in H \text{ such that } w \in h', \\
\mathcal{M}, w/h \models X\varphi & \Leftrightarrow & \mathcal{M}, w'/h \models \varphi \text{ for the } w' \text{ that is the successor} \\
 & & \text{of } w \text{ in the linearly ordered set } h, \\
\mathcal{M}, w/h \models [A \text{ cstit}]\varphi & \Leftrightarrow & \mathcal{M}, w/h' \models \varphi \text{ for all } h' \in \bigcap_{a \in A} S_a \text{ where} \\
 & & S_a \in C(a, w) \text{ is the element of } C(a, w) \\
 & & \text{that contains } h.
\end{array}
$$

There is one further restriction placed on $\mathfrak{M}_{\text{STIT}}$ in [Broersen et al., 2006a], namely that if $S_a \in C(a, w)$ for each $a \in \mathcal{A}$ then $\bigcap_{a \in \mathcal{A}} S_a$ is not only non-empty but also a singleton. This means that the transition from one state to the next is deterministic; given a choice for each agent, there is a unique next state the system is guaranteed to be in.

This assumption is highly unusual for STIT logics and is not necessary to make the translation truth-preserving, but it is required to make the translation preserve validity.

Before defining the translation we need a few more concepts.

Definition 6.20 (CL notation)**.** Let $\mathcal{M} = (W, \Sigma, o, v)$ be a CL model.

The *outcome graph* $G(\mathcal{M})$ of $\mathcal{M}$ is the graph (W, E) where $(w_1, w_2) \in E$ if and only if there is a choice $c_{\mathcal{A}}$ such that $O(w_1, c_{\mathcal{A}}) = w_2$.

An *i-choice function* $c_{i,w,a}$ for $w \in W$ and $a \in \mathcal{A}$ is a function $c_{\{a\}} : W \times \{a\} \to \mathbb{N}$ such that $c_{\{a\}}(w, a) = i$.

The i-th *choice cell* $S(w, a, i)$ of $w \in W$ and $a \in \mathcal{A}$ with $i \leq \Sigma(w, a)$ is the set of worlds w' such that there are an i-choice function $c_{i,w,a}$ and a completion $c_{\mathcal{A}}$ of $c_{i,w,a}$ with $w' = O(w, c_{\mathcal{A}})$.

Definition 6.21 (Translation from CL to STIT)**.** The function $f : \mathfrak{M}_{\text{CL}} \to \mathfrak{M}_{\text{STIT}}$ is given by $f(W, \Sigma, o, v) = (W', \leq, H, C, v')$ where

- $(W', \leq)$ is the tree that is obtained by unraveling $G(\mathcal{M})$,

- C is the function such that for every $a \in \mathcal{A}$ and $w' \in W'$ we have $C(a, w') = \{\{h \mid h \cap S_{w',a,0} \neq \emptyset\}, \cdots, \{h \mid h \cap S_{w',a,\Sigma(w',a)} \neq \emptyset\}\}$,[8]
- $v'(p) = \{w'/h \mid w \in v(p)\}$.[9]

The formula translation $t : \Phi_{\mathrm{CL}} \to \Phi_{\mathrm{STIT}}$ is given by

$$\begin{aligned} t(p) &= \Box p \text{ for } p \in \mathcal{P} \\ t(\neg\varphi) &= \neg t(\varphi) \\ t(\varphi_1 \vee \varphi_2) &= t(\varphi_1) \vee t(\varphi_2) \\ t([A]\varphi) &= \Diamond[A \text{ cstit}]Xt(\varphi) \end{aligned}$$

This translation is truth preserving.

Lemma 6.5. *For every $\mathcal{M} \in \mathfrak{M}_{\mathrm{CL}}$, every $w \in \mathcal{M}$ and every $\varphi \in \Phi_{\mathrm{CL}}$ we have $\mathcal{M}, w \models \varphi$ if and only if $f(\mathcal{M}), w \models t(\varphi)$.*

Sketch of proof. The model translation $f : \mathfrak{M}_{\mathrm{CL}} \to \mathfrak{M}_{\mathrm{STIT}}$ copies all structure from a CL model $\mathcal{M}$ to directly corresponding structures in $f(\mathcal{M})$.

Let us fix any $\mathcal{M} = (W, \Sigma, o, v) \in \mathfrak{M}_{\mathrm{CL}}$, any $w \in \mathcal{M}$ and any $\varphi \in \Phi_{\mathrm{CL}}$. Let $\mathcal{M}' = (W', \leq, H, C, v') = f(\mathcal{M})$. For ease of notation, let us assume that $\mathcal{M}$ is tree-like, so $W = W'$. We can immediately copy v as well; even though v' is a function from $\mathcal{P}$ to $W \times H$ instead of to W, we can just assign the same valuation to all histories in the same world by taking $v'(p) = \{w'/h \mid w \in v(p)\}$.

The structure we then still have to copy from $\mathcal{M}$ to $\mathcal{M}'$ is the choice structure Σ, o. This choice structure has to be represented in C, of course. The way the two kinds of models model choices is different: Σ, o uses a function that assigns to each combination of choices $(i_1, \cdots, i_k)$ an outcome world, whereas C gives each agent a partition of the histories going through the current world. But this difference in representation is only superficial.

Given Σ, o we can easily find a corresponding partition of the possible successor worlds. For every choice $j \leq \Sigma(w, a)$ we assign to a a set $S_{w,a,l}$ of worlds w' such that there is some collective choice where a chooses l and w' is the outcome. The set $\{S_{w,a,0}, \cdots, S_{w,a,\Sigma(w,a)}\}$ is then a partition of the possible successor worlds. From this partition of the successor worlds, we can then obtain a partition of the histories through the current world by selecting those histories that contain some element of $S_{w,a,j}$. This is indeed how we defined C, by taking $C(a, w) = \{\{h \mid h \cap S_{w,a,0} \neq \emptyset\}, \cdots, \{h \mid h \cap S_{w,a,\Sigma(w,a)} \neq \emptyset\}\}$. The way we constructed these sets also immediately guarantees that the intersection of any collective choice is nonempty, and in fact a singleton.

All that remains then is to properly translate formulas of the form $[A]\varphi$. This formula holds if and only if there is some collective choice for A that guarantees that the next state satisfies φ. The "there is a collective choice" part can be translated into STIT as $\Diamond$, A guaranteeing something can be translated as $[A \text{ cstit}]$, something holding in the next state can be translated as X and, finally, φ being this something can be translated as $t(\varphi)$. □

Note the representation of Σ, o by C could also be done the other way around; given any C (with the extra assumption of determinism) we can find corresponding

[8] Note that we are abusing notation here by ignoring the difference between $G(\mathcal{M}) = (W, E)$ and the unraveling $(W', \leq)$ thereof.

[9] Again, some abuse of notation.

Σ, o. So we can translate from deterministic STIT models to CL models. This suggests that in addition to being truth-preserving the translation is also validity preserving.

And in fact [Broersen et al., 2006a] shows that the translation does indeed preserve validity, as long as we take the extra assumption that every STIT model is deterministic. If we allow nondeterministic STIT models, the translation does not preserve validity: in that case $[\mathcal{A}]p \vee [\mathcal{A}]\neg p$ is valid but its translation $\Diamond[\mathcal{A}\ \text{cstit}]X\Box p \vee \Diamond[\mathcal{A}\ \text{cstit}]X\neg\Box p$ is not.

The variant of STIT used in [Broersen et al., 2006b] is more complicated than the one described above, because it has to be able to simulate the $\langle\langle A\rangle\rangle G\varphi$ and $\langle\langle A\rangle\rangle\varphi_1 U\varphi_2$ operators. In order to do this, the STIT variant in question contains extra connectives G, U and $[A\ \text{scstit}]$ where $[A\ \text{scstit}]$ acts like $[A\ \text{cstit}]$ except that it quantifies over *strategies* instead of single choices. The translation from ATL to this more complicated variant of STIT given in [Broersen et al., 2006b] is truth-preserving and (assuming determinism in STIT) validity preserving.

6.8 Conditions on Translations

Now we know what desiderata expressivity$_g$ must satisfy in order to fulfill its purpose. But that does not immediately tell us how to find a definition that satisfies all desiderata. In this section we start to look for such a definition by introducing a number of properties that translations can have.

We want to capture the pattern that the prototypical results have in common and that makes them interesting. Considering that all these results are truth preserving translations we should base this pattern on the existence of truth preserving translations. However, the existence of the trivial translations shows that it is not sufficient to demand a truth preserving translation. We should therefore put additional constraints on the translations between the logics. Our definition of expressivity$_g$ then looks as follows.

> A logic $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ is at least as expressive$_g$ as a logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ if there is a truth preserving translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_1$ that also has the properties $\{X_1, \cdots, X_n\}$.

The remaining task is to find the additional properties $X_1, \cdots, X_n$. In this section we introduce a number of properties that could possibly be used as such additional constraints. Unfortunately the conditions that "work" are rather complicated. If we were to immediately introduce these conditions, they would therefore probably seem somewhat ad hoc. So instead of only introducing those conditions that are used in the definition of expressivity$_g$, we introduce several non-chosen conditions that provide context to the chosen conditions, thereby hopefully explaining why the chosen conditions are appropriate.

A first and very naive approach would be to let the extra condition on (t, f) be that they are not equal to one of the trivial translations. But that would be quite ad hoc, and it would almost certainly leave us with other pairs (t, f) that satisfy the revised condition but that should not show one logic to be at least as expressive$_g$ as another.

So instead of explicitly excluding the trivial translations we should come up with one or more reasonable conditions that exclude the trivial translations among others. The following conditions all address something that is wrong with the trivial trans-

lations, and look as if they might even *characterize* what is wrong with the trivial translations.

The first two conditions are about tautologies and entailments. Both concepts are important in logic, and the trivial translations do not preserve them. Let $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ and $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ be logics.

Definition 6.22 (Validity preserving)**.** A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *validity preserving* if for every $\varphi \in \Phi_1$ we have that φ is valid if and only if $t(\varphi)$ is valid.

Definition 6.23 (Entailment preserving/Conservative)**.** A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *entailment preserving* (also called *conservative*) if for every $\Gamma \subseteq \Phi_1$ and $\varphi \in \Phi_1$ we have that Γ entails φ if and only if $t(\Gamma)$ entails $t(\varphi)$.

Conservative translations have been studied in several publications; notable examples include [Feitosa and D'Ottaviano, 2001] and [Jeřábek, 2012]. In these publications they were studied in the context of logics $(\Phi, \vdash)$, not $(\Phi, \mathfrak{M}, \models)$, though. For that reason, and because "entailment preserving" fits better in our (very descriptive) naming scheme for conditions, we will call conservative translations *entailment preserving* here.

Validity and entailment preserving translations are in some ways very similar to truth preserving translations, except that they are more appropriate for different kinds of logic. Recall that we considered three possible definitions for a logic: pairs (Φ, T) where $T \subseteq \Phi$ is a set of tautologies, pairs $(\Phi, \vdash)$ where $\vdash \subseteq \wp(\Phi) \times \Phi$ is an entailment relation and triples $(\Phi, \mathfrak{M}, \models)$ where $\models \subseteq \mathfrak{M} \times \Phi$ is a satisfaction relation.

Truth preserving translations are defined with logics $(\Phi, \mathfrak{M}, \models)$ in mind: the main structure in such a logic is $\models$ and that is exactly what a truth preserving logic preserves. For logics (Φ, T) the main structure is T, which is what validity preserving translations preserve. For logics $(\Phi, \vdash)$ the main structure is $\vdash$, which is preserved by entailment preserving translations.

The parallels also go further than that. As demonstrated by [Jeřábek, 2012] the mere fact that a translation is entailment preserving (and therefore also validity preserving) is not sufficient for a translation to be interesting.

The combination of truth preservation and validity preservation or entailment preservation is more restrictive than either condition on its own, so it is possible that such a combination of conditions would be sufficient condition for a translation to be interesting. Unfortunately it turns out not to be a *necessary* condition; some of the prototypical translations are neither validity nor entailment preserving[10], so the pattern that unites them all cannot contain either of those requirements.

The third condition we want to consider is related to the conditions of being validity and entailment preserving, but in a way that is not immediately obvious.

Definition 6.24 (Model invertible)**.** A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *model invertible* if f is a bijection, and $g := f^{-1}$ satisfies the following property:

$$\text{for all } (\mathcal{M}_2, w_2) \in \mathfrak{M}_2 \text{ and all } \varphi_1 \in \varphi_1 \text{ we have } g(\mathcal{M}_2, w_2) \models_1 \varphi_1 \text{ if and only if } \mathcal{M}_2, w_2 \models_2 t(\varphi_1).$$

This conditions may seem a bit strange at first, but there could be some good reasons for requiring it. These reasons center around the fact that, instead of pairing a function $t : \Phi_1 \to \Phi_2$ with a function $f : \mathfrak{M}_1 \to \mathfrak{M}_2$, we could have paired $t : \Phi_1 \to \Phi_2$

[10] Specifically, the translations from temporal logic to modal logic and from ATEL to ATL introduced in [Thomason, 1974] and [Goranko and Jamroga, 2004].

with a function $g : \mathfrak{M}_2 \to \mathfrak{M}_1$. For such (t, g) we can also give a reasonable definition of truth preservation, namely the one included in the definition above: (t, g) is truth preserving if for all $\varphi_1 \in \Phi_1$ and all $\mathcal{M}_2, w_2 \in \mathfrak{M}_2$ we have $g(\mathcal{M}_2, w_2) \models_1 \varphi_1$ if and only if $\mathcal{M}_2, w_2 \models_2 t(\varphi_1)$.

Such pairs (t, g) have advantages as well as disadvantages compared to pairs (t, f). In particular, (t, f) preserves satisfiability, whereas (t, g) preserves validity. (To see why this is the case, note that if (t, f) is truth preserving and φ_1 is satisfiable, say in $\mathcal{M}_1, w_1$, then $f(\mathcal{M}_1, w_1) \models_2 t(\varphi_1)$ so $t(\varphi_1)$ is also satisfiable. If on the other hand (t, g) is truth preserving and φ_1 is valid then for every $\mathcal{M}_2, w_2 \in \mathfrak{M}_2$ we have $g(\mathcal{M}_2, w_2) \models_1 \varphi_1$ and therefore $\mathcal{M}_2, w_2 \models_2 t(\varphi_1)$ so $t(\varphi_1)$ is also valid.) The condition of being model invertible is useful if we want to have our cake and eat it; both (t, f) and (t, g) have to exist and be truth preserving.

Unfortunately, because the existence of (t, g) implies validity preservation, we already know that model invertibility is too strong a condition to require for expressivity$_g$, because some of our prototypical translations do not satisfy it.

The first three conditions placed restrictions on the translation (t, f) as a whole; if a translation (t, f) is validity preserving, entailment preserving or model based, that is no guarantee that a translation (t, f') or (t', f) has the same property. The following conditions on the other hand are about only one of the two functions in a translation. The idea is that $f : \mathfrak{M}_1 \to \mathfrak{M}_2$ should preserve (some of) the structure of $\mathfrak{M}_1$ while $t : \Phi_1 \to \Phi_2$ should independently preserve (some of) the structure of Φ_1. The fourth condition is about f.

Definition 6.25 (Model based)**.** A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *model based* if there are two functions f_1 and f_2 such that for all $(\mathcal{M}, w) \in \mathfrak{M}_1$ we have $f(\mathcal{M}, w) = (f_1(\mathcal{M}), f_2(\mathcal{M}, w))$.

The condition of translations being model based is intended to counter one of the things that is clearly wrong with the second trivial translation, namely that it translates $\mathcal{M}, w$ and $\mathcal{M}, w'$ to completely unrelated models. The first and third trivial translations are model based so this condition cannot by itself characterize the translations that should count as evidence, but it could perhaps be part of a set of conditions that characterizes the right translations.[11] Note that, as mentioned above, this condition only restricts f; if (t, f) is model based then so is (t', f) for every $t' : \Phi_1 \to \Phi_2$.

Like the condition of being model based, the following conditions only place restrictions on one of the two functions in a translations (t, f). But unlike being model based, these conditions restrict t: they require t to preserve some of the structure of Φ_1. However, in order for us to require that t preserves the structure of Φ_1, we first have to require that Φ_1 has some kind of structure. So far the set Φ in a logic $(\Phi, \mathfrak{M}, \models)$ could be any set. From this point on, we require that Φ is a compositional language, generated by some set $\mathcal{P}$ of atoms and some set C of connectives.[12]

Let us start with a rather naive condition.

[11] Note that we should allow the second function f_2 to take both $\mathcal{M}$ and w as arguments, as this allows f_2 to map "the same world" in different models to different worlds. After all, two worlds $(\mathcal{M}_1, w)$ and $(\mathcal{M}_2, w)$ have no relation to each other whatsoever, so we shouldn't require them to be translated to "the same world".

[12] The decision to define a language by a set of propositional variables together with a set of connectives does show a bias favoring propositional logics over other types of logics. This is mostly a matter of notation; conditions similar to the following ones could quite easily be defined for other types of compositional languages.

Definition 6.26 (Non-atomic)**.** A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *non-atomic* if for every non-atomic $\varphi \in \Phi_1$ the translation $t(\varphi)$ is non-atomic.

The non-atomicity condition is intended to counter one of the things that are clearly wrong with the trivial translations, namely that they translate complex formulas to propositional variables. Unfortunately, due to its simplicity, this condition is easily cheated. For example, even the third trivial translation is non-atomic, because it translates φ to $\Box p_\varphi$ instead of just p_φ. So the condition of non-atomicity is not sufficient for a truth preserving translation to be interesting.

Worse, it is not even a necessary condition for a translation to be interesting. There are several interesting translations where some (but not all) non-atomic formulas are translated to an atomic formula. For example, in [Kooi and Renne, 2011] a translation is given from Arrow Update Logic ($\mathcal{L}_{\mathrm{U}}$, see also Chapter 3) to basic modal logic. One of the clauses in that translation is $t([U]p) = p$. This translation is one of the traditional translations, so the fact that it is not non-atomic shows that non-atomicity is not a suitable condition for expressivity$_g$.

By taking a slightly more complicated condition we can make it harder, but still not impossible, to cheat.

Definition 6.27 (Subformula preserving)**.** A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *subformula preserving* if for every $\varphi_1, \varphi_2 \in \Phi_1$, if φ_1 is a subformula of φ_2 then $t(\varphi_1)$ is a subformula of $t(\varphi_2)$.

The condition of subformula preservation is a stronger variant of being non-atomic that is harder to cheat. Unfortunately, as mentioned above it can still be cheated. We could for example take $t(\Box\varphi) = p_{\Box\varphi} \wedge (\bot \to t(\varphi))$. Such a translation would be subformula preserving, but still hardly non-trivial. Additionally, like with non-atomicity, there are existing interesting translations that are not subformula preserving. The best known example of an interesting translation that is not subformula preserving is a translation from public announcement logic to basic modal logic. In that translation we have, among others, the following two clauses.

$$t([\psi]\Box_a\psi') = (t(\psi) \to \Box_a(t(\psi) \to [t(\psi)]t(\psi')))$$
$$t(\Box_a\psi') = \Box_a t(\psi').$$

Note that $t(\Box_a\psi')$ is not a subformula of $t([\psi]\Box_a\psi')$, so the translation is not subformula preserving.

By refining the concept of subformula preservation we can however get a condition that is probably sufficient (although not necessary) for a truth preserving translation to be interesting.

Definition 6.28 (Connective based)**.** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be such that Φ_i $(1 \leq i \leq 2)$ is generated by a set $\mathcal{P}_i$ of propositional variables and a set C_i of connectives. For $\circ_j \in C_i$ let r_j be the arity of $\circ_j$.

A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *connective based* if there is a function $t_0 : C_1 \to C_2$ such that for all $\circ \in C_1$ with arity r and all $\varphi_1, \cdots, \varphi_r$ we have

$$t(\circ(\varphi_1, \cdots, \varphi_r)) = t_0(\circ)(t(\varphi_1), \cdots, t(\varphi_r)).$$

This condition is quite similar—but not equivalent—to the way formulas are translated in interpretations, see for example [Verbrugge, 1993].[13]

[13] Do note that, even if being connective based would have been equivalent to the way interpretations treat formulas, an interpretation would not have been the same as a connective based truth preserving translation, because interpretations and truth preserving translations act differently on models.

The condition of being connective based rules out the trivial translations and it seems unlikely that it can be cheated like the previous two conditions. So it seems likely that a translation can be interesting because it is truth preserving and subformula preserving. But unfortunately, while the condition of being connective based disqualifies the trivial translations, it also disqualifies some of the traditional translations so it is too strong for our purpose.

For an example of an interesting translation that is not connective based, consider, once again, these two clauses in the translation from public announcement logic to basic modal logic:

$$t([\psi]\Box_a\psi') = (t(\psi) \to \Box_a(t(\psi) \to [t(\psi)]t(\psi')))$$
$$t(\Box_a\psi') = \Box_a t(\psi').$$

The translation of $t([\psi]\Box_a\varphi)$ depends critically on the second connective $\neg$, whereas a connective based translation is only allowed to depend on the first connective $[\cdot]$ in such a way.

If we want to allow translations like the one from public announcement logic to modal logic, we will have to be a bit more flexible. Instead of requiring that the translation can be done one connective at a time, we should also allow translation clauses that tell us how to translate a combination of connectives, such as $[\cdot]\neg$ in the example. If we allow an infinite number of such clauses, all we are left with is that the translation should allow a certain kind substitution of propositional variables. Before formally defining this substitution of variables let us first define some notation that allows us to use placeholder variables in formulas. We could of course use φ and ψ as variables over formulas, but we prefer to reserve those symbols as meta-variables. So instead we use $\{x_1, x_2, \cdots\}$ as placeholder variables.

Definition 6.29 ($\Phi^{\mathcal{P}'+X}$)**.** Let Φ be a set of formulas generated by a set $\mathcal{P}$ of propositional variables and a let C be a set of connectives. Furthermore, let $X := \{x_1, x_2, \cdots\}$ be a set such that $X \cap \mathcal{P} = \emptyset$ and let $\mathcal{P}' \subseteq \mathcal{P}$. Then $\Phi^{\mathcal{P}'+X}$ is the set of formulas generated by the set $\mathcal{P}' \cup \{x_1, x_2, \cdots\}$ of propositional variables and the set C of connectives.

Now we can give a definition of being substitution robust.

Definition 6.30 (Substitution Robust)**.** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be such that Φ_i $(1 \leq i \leq 2)$ is generated by a set $\mathcal{P}_i$ of propositional variables and a set C_i of connectives.

A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *substitution robust* if there is a function $t_0 : \Phi_1^{\emptyset+X} \to \Phi_2^{\mathcal{P}_2+X}$ such that for every $\varphi^X \in \Phi_1^{\emptyset+X}$ where k is the highest number such that x_k occurs in φ^X and every $p_1, \cdots, p_k \in \mathcal{P}_1$ we have

$$t(\varphi^X[x_1 \mapsto p_1, \cdots, x_k \mapsto p_k]) = t_0(\varphi)[x_1 \mapsto t(p_1), \cdots, x_k \mapsto t(p_k)]$$

Here we use the notation $x_i \mapsto p_i$ for the substitution of p_i for x_i. The idea of being substitution robust is that we should have something like $t(\varphi)[t(p) \mapsto t(q)] = t(\varphi[p \mapsto q])$. The more complicated definition is necessary to account for the fact that $t(p)$ might coincidentally occur in $t(\varphi)$ in several places, only some of which should be replaced by $t(q)$.[14]

[14] For example, suppose $\mathcal{L}_2$ uses $\bot$ as an abbreviation for $p \wedge \neg p$, and that t satisfies $t(\neg\varphi) = t(\varphi) \to \bot$, $t(p) = p$ and $t(q) = q$. Then we have $t(\neg p)[t(p) \mapsto t(q)] = q \to (q \wedge \neg q) \neq q \to (p \wedge \neg p) = t(\neg p[p \mapsto q])$.

If we only allow a finite number of clauses, we get a very different condition however, that of being finitely generated. Unfortunately there are a few important but rather complicated details related to this definition. So instead of providing the full definition here we give a "definition attempt" that omits a few of the technical details. The full definition is given in Section 6.10.

Definition Attempt I (Finitely Generated)**.** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be such that Φ_i $(1 \leq i \leq 2)$ is generated by a set $\mathcal{P}_i$ of propositional variables and a finite set C_i of connectives. A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is *finitely generated* if t can be inductively defined by a finite number of clauses of the form

$$t(\varphi^X) = \psi^X \text{ for } (x_1, \cdots, x_k) \in \Psi,$$

where the x_i only occur inside the scope of a unary operator t in ψ^X.

Note that we restrict the definition attempt to logics with a finite set of connectives. This restriction is important because if a logic has an infinite set of connectives, we cannot reasonably demand that a finite set of clauses allows us to translate them all. There are solutions to this problem, but we leave discussion of these solutions for Section 6.12.

Let us consider an example of a translation that is finitely generated. We return to what is becoming the running example in this chapter, the translation from public announcement logic to basic modal logic. But in this case let us consider the full translation.

$$\begin{aligned}
t(p) &= p && \text{for } p \in \mathcal{P}_{PAL}\\
t(\neg\varphi) &= \neg t(\varphi)\\
t(\varphi_1 \wedge \varphi_2) &= t(\varphi_1) \wedge t(\varphi_2)\\
t(\Box_a\varphi) &= \Box_a t(\varphi)\\
t([\varphi]p) &= t(\varphi) \to p && \text{for } p \in \mathcal{P}_{PAL}\\
t([\varphi_1]\neg\varphi_2) &= t(\varphi_1) \to \neg t([\varphi_1]\varphi_2)\\
t([\varphi_1](\varphi_2 \wedge \varphi_3)) &= t([\varphi_1]\varphi_2) \wedge t([\varphi_1]\varphi_3)\\
t([\varphi_1]\Box_a\varphi_2) &= t(\varphi_1) \to \Box_a t([\varphi_1]\varphi_2)\\
t([\varphi_1][\varphi_2]\varphi_3) &= t([\varphi_1 \wedge [\varphi_1]\varphi_2]\varphi_3)
\end{aligned}$$

This translation consists of a finite number of clauses, so it is a finitely generated translation. In the definition we used a slightly different notation, but we can quite easily switch between the different notations. For example, the clause $t(\Box_a\varphi) = \Box_a t(\varphi)$ could be represented by

$$t(\Box_a x_1) = \Box_a t(x_1) \text{ for } x_1 \in \Phi_{PAL}.$$

Note that $\psi^X = \Box_a t(x_1)$ does indeed satisfy the requirement that the variables x_i only occur within the scope of t. The set Ψ allows us to restrict the range of the variables x_i. For example, the clause $t([\varphi]p) = t(\varphi) \to p$ can be represented by

$$t([x_1]x_2) = t(x_1) \to t(x_2) \text{ for } (x_1, x_2) \in \Phi_{PAL} \times \mathcal{P}_{PAL}.$$

By requiring that the translation can be inductively defined, the condition of being finitely generated forces translations to respect (some of) the structure of the formulas

	Validity preserving	Entailment preserving	Model invertible	Model based	Non-atomic	Subformula preserving	Connective based	Substitution robust	Finitely generated
First trivial translation	-	-	-	+	-	-	-	-	-
Second trivial translation	-	-	-	-	-	-	-	-	-
Third trivial translation	-	-	-	+	+	-	-	-	-
First notation translation	+	+	+	+	+	+	+	+	+
Second notation translation	+	+	-	+	+	+	+	+	+
Third notation translation	+	+	-	+	+	+	+	+	+
Traditional translations	+	+	+	+	+	0	0	+	+
TL to ML	-	-	-	+	+	+	+	+	+
Classical to normal modal logic	+	+	-	+	+	+	+	+	+
ATEL to ATL	-	-	-	+	+	+	+	+	+
CL to STIT	+	+	-	+	+	+	+	+	+
ATL to STIT	+	+	-	+	+	+	+	+	+

Table 6.1: A + indicated that a translation has the relevant property, a - indicates that a translation does not have a certain property and a 0 indicates that some instances of a translation have the property and others do not. We are looking for a (set of) propertie(s) that have a - in the first three rows and a + in the other rows.

of $\mathcal{L}_1$. Their complete disrespect for the structure of Φ_1 is precisely what sets the trivial translations apart, so is seems plausible that this condition could characterize the truth preserving translations that are interesting. Indeed, it turns out that the prototypical, traditional and notation changing translations are finitely generated whereas the trivial translations are not.

In Section 6.10 we will consider this condition in more detail and give a formal definition. But for now the definition attempt given above will suffice.

6.9 Choosing the Right Conditions

We now have a number of desiderata for the definition of expressivity$_g$ as well as a number of conditions that we could place on translations in the definition of expressivity$_g$. Now we should put these things together and define expressivity$g$. So let us take a look at which translations satisfy which conditions. Information like this is best displayed in table form, so let us look at Table 6.1.

If we want a number of conditions that together characterize the translations that should count (as showing that one logic is at least as expressive as another), we cannot choose validity preservation, entailment preservation, model invertibility, subformula preservation or being connective based; for each of these conditions there are translations that should count but that do not satisfy the condition. If we want the set of conditions to disqualify the three trivial translations, we have to include either

substitution robustness or being finitely generated.

Either substitution robustness or being finitely generated would by itself partition the translations correctly. But they can also be combined with each other, or with the conditions of being model based or being non-atomic.

The conditions of non-atomicity, substitution robustness and being finitely generated all exclude the trivial translations, because those translate complex formulas to simple ones. As such it seems pointless to require translations to satisfy several of these conditions.

The condition of being model based on the other hand prevents a completely different problem, namely that different worlds in the same model get translated to completely different models by the second trivial translation. As such it is not unreasonable to require a translation to be model based in addition to being either substitution robust or finitely generated.

The main question is then which of the two conditions is better, being substitution robust or being finitely generated. Intuitively, I would say that being finitely generated is probably a better guarantee of being non-trivial than being substitution robust. Most reasonable translations seem to be both substitution robust and finitely generated, but if we look at boundary cases we can come up with a translation that is substitution robust but not finitely generated.

Example 6.9. Let $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ be such that Φ_1 is generated by an empty set of propositional variables and the set $\{\top, \neg, \vee, \Box\}$ of connectives, with $\top$ nullary, $\neg$ and $\Box$ unary, and $\vee$ binary.

Let $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ be such that Φ_2 is generated by a countable set $\mathcal{P}$ of propositional variables and an empty set of connectives.

A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ where $t(\varphi) = p_\varphi$ with $p_\varphi \in \mathcal{P}$ for each $\varphi \in \Phi_1$ is substitution robust but not finitely generated.

It seems to me that such a translation should not show that $\mathcal{L}_2$ is at least as expressive$_g$ as $\mathcal{L}_1$. Since the translation is substitution robust but not finitely generated, this means that being finitely generated does indeed seem to be the best choice of condition. The definition of expressivity$_g$ then becomes the following.

Definition 6.31 (Expressivity$_g$)**.** Let $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$, $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ be logics such that Φ_i is generated by a set $\mathcal{P}_i$ of propositional variables and a finite set C_i of connectives for $i \in \{1, 2\}$.

The logic $\mathcal{L}_2$ is at least as *expressive$_g$* as the logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ if there is a translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ that is model based, finitely generated and truth preserving.

6.10 Formally Defining Finitely Generated

We just decided that being finitely generated is one of the conditions a translation has to satisfy in order to show that one logic is at least as expressive$_g$ as another. In some sense the concept of being finitely generated is quite simple; in most cases we "will know it when we see it". But unfortunately the fact that we can usually recognize finitely generated translations is not sufficient to base a definition on. So we need a formal definition of the concept.

Let us start by considering the example we used when introducing the definition attempt. We want t to be represented by a finite number of clauses such as the clause

$t([\varphi]\neg\psi) = t(\varphi) \to \neg t([\varphi]\psi)$ that occurs in the translation from Public Announcement Logic (PAL) to basic modal logic.

Such a clause is intended to be read with implicit universal quantification, so it holds for every PAL formula φ, ψ. In order to represent this quantification we once again need placeholder variables. Like in Section 6.8 we prefer to use φ, ψ as meta-variables so we use the symbols $\{x_1, x_2, \cdots\}$, which we assume to be distinct from any symbols in $\mathcal{L}_1$ and $\mathcal{L}_2$. These x_i are best treated as special propositional variables for which we can later substitute formulas. We use Greek letters with superscript X to denote formulas that contain one or more of these special propositional variables.

This lets us to rephrase the clause given above as $t([x_1]\neg x_2) = t(x_1) \to \neg t([x_1]x_2)$. But we need a bit more precision in our clauses. Consider for example the following translation clause, also a part of the translation from PAL to modal logic: $t([\varphi]p) = p$. This clause should also be read with universal quantification, but a restricted universal quantification. The formula φ can be any PAL formula, but p must be a propositional variable. This suggests that we should add the range of the variables x_i to our clause. The two clauses discussed so far could be represented by

$$\begin{aligned} t([x_1]\neg x_2) &= t(x_1) \to \neg t([x_1]x_2) && \text{for } (x_1, x_2) \in \Phi_{PAL} \times \Phi_{PAL} \\ t([x_1]x_2) &= t(x_2) && \text{for } (x_1, x_2) \in \Phi_{PAL} \times \mathcal{P}_{PAL} \end{aligned}$$

The general form for such clauses is the following.

$$t(\varphi^X) = \psi^X \text{ for } (x_1, \cdots, x_k) \in \Psi$$

The remaining question is then what form φ^X, ψ^X and Ψ should take. Unfortunately, it turns out that this is a rather complicated question. So let us start with a simple approximation to the answer that we can later modify. The main idea of φ^X is that it should be a formula of $\mathcal{L}_1$ with certain subformulas replaced by variables x_i. Earlier we introduced some notation for the use of placeholder variables, but here we need to use them in a slightly different way. So let us use a new definition.

Definition 6.32 ($\Phi^{\mathcal{P}',C'}$)**.** Let Φ be a set of formulas generated by a set $\mathcal{P}$ of propositional variables and a set C of connectives. Furthermore, let $\mathcal{P}'$ be a set and C' a set of connectives. Then $\Phi^{\mathcal{P}',C'}$ is the set of formulas generated by the set $\mathcal{P} \cup \mathcal{P}'$ of propositional variables and the set $C \cup C'$ of connectives.

Furthermore, let $X = \{x_1, x_2, \cdots\}$, so $\Phi^{X,C'}$ is the set of formulas generated by the set $\mathcal{P} \cup \{x_1, x_2, \cdots\}$ of propositional variables and the set $C \cup C'$ of connectives.

The extra set C' of connectives is not yet useful, but we will need it later. Our first attempt at determining the form of φ^X is then that we should have $\varphi^X \in \Phi_1^{X,\emptyset}$. The variables x_i should not take values that themselves contain other variables, so we should take $\Psi \subseteq \Phi_1^k$. That leaves only the form of ψ^X to be determined. So let us take a closer look at the first clause given above,

$$t([x_1]\neg x_2) = t(x_1) \to \neg t([x_1]x_2) \text{ for } (x_1, x_2) \in (\Phi_{PAL} \times \Phi_{PAL}).$$

Here $\psi^X = t(x_1) \to \neg t([x_1]x_2)$. The first thing to note is that ψ^X contains subformulas of the form $t(\chi^X)$, where $\chi^X \in \Phi_1^{X,\emptyset}$. But outside of the scope of t we only encounter connectives from modal logic. The easiest way to represent this is to consider subformulas of the form $t(\chi^X)$ as extra propositional variables. Let us call this set of variables T_1^C, so $T_1^C := \left\{ t(\chi^X) \mid \chi^X \in \Phi_1^{X,C} \right\}$. Then we should have $\psi^X \in \Phi_2^{T_1^\emptyset,\emptyset}$.

Remark. We took $\psi^X \in \Phi_2^{T_1,\emptyset}$. This means that variables x_i in ψ^X can only occur inside the scope of t. This might seem too restrictive. After all, consider the clause $t([\varphi_1]p) = t(\varphi_1) \to p$. We would be tempted to represent this by $t([x_1]x_2) = t(x_1) \to x_2$ for $(x_1, x_2) \in \Phi_1 \times \mathcal{P}_1$. But this is a temptation we should resist. The propositional variable p of PAL and the propositional variable p of modal logic may use the same symbol, but they are different objects. The proper way to represent $t([\varphi_1]p) = t(\varphi_1) \to p$ is therefore the clause $t([x_1]x_2) = t(x_1) \to t(x_2)$ for $(x_1, x_2) \in \Phi_1 \times \mathcal{P}_1$.

Note that there is one small problem we have to take care of. The variables x_i are not allowed to occur in ψ^X outside of the scope of t. But that means we can never truly reach the base case that grounds the inductive definition. After all, suppose we want to represent the base case $t(p) = p$. Then we write a clause $t(x_1) = \psi^X$ for $p \in \mathcal{P}_1$. On the one hand, we cannot take $\psi^X = x_1$, since x_1 may not occur outside the scope of t. But on the other hand, a clause $t(x_1) = t(x_1)$ is not very helpful. The solution is to allow a finite number of auxiliary functions $s_1, \cdots, s_n : \mathcal{P}_1 \to \mathcal{P}_2$.[15] These functions do not have to be defined inductively, but in return they may only operate on atoms. It is important to be aware that we need not have $t(p) = s_i(p)$ for some $1 \leq i \leq n$. We could for example have a clause $t(p) = \Box s_1(p)$.

So in addition to propositional variables of the form $t(\varphi^X)$ in ψ^X we should also allow propositional variables of the form $s_i(x_j)$. Let $S_1 := \{s_i(x_j) \mid 1 \leq i \leq n, j \in \mathbb{N}\}$. Then instead of requiring that $\psi^X \in \Phi_2^{T_1,\emptyset}$ we should require that $\psi^X \in \Phi_2^{T_1^\emptyset \cup S_1,\emptyset}$.

Recall that so far we required that $\varphi^X \in \Phi_1^{X,\emptyset}, \psi^X \in \Phi_2^{T_1^\emptyset \cup S_1,\emptyset}$ and $\Psi \subseteq \Phi^k$. This allows us to represent most finitely generated translations. So far we have only looked at single clauses in the translation from PAL to modal logic, but now let us consider the entire translation. Let us assume that PAL uses the connectives $\neg, \wedge, \Box$ and $[\cdot]$ as well as the countably infinite set $\mathcal{P}_{PAL}$ of propositional variables while modal logic uses the connectives $\neg, \wedge, \vee, \to$ and $\Box$ as well as the countably infinite set $\mathcal{P}_{ML}$ of propositional variables. Then the translation is given by

$$
\begin{aligned}
t(p) &= p && \text{for } p \in \mathcal{P}_{PAL} \\
t(\neg\varphi) &= \neg t(\varphi) \\
t(\varphi_1 \wedge \varphi_2) &= t(\varphi_1) \wedge t(\varphi_2) \\
t(\Box\varphi) &= \Box t(\varphi) \\
t([\varphi]p) &= t(\varphi) \to p && \text{for } p \in \mathcal{P}_{PAL} \\
t([\varphi_1]\neg\varphi_2) &= t(\varphi_1) \to \neg t([\varphi_1]\varphi_2) \\
t([\varphi_1](\varphi_2 \wedge \varphi_3)) &= t([\varphi_1]\varphi_2) \wedge t([\varphi_1]\varphi_3) \\
t([\varphi_1]\Box\varphi_2) &= t(\varphi_1) \to \Box t([\varphi_1]\varphi_2) \\
t([\varphi_1][\varphi_2]\varphi_3) &= t([\varphi_1 \wedge [\varphi_1]\varphi_2]\varphi_3)
\end{aligned}
$$

[15] We should allow multiple functions $s_1, \cdots, s_n$ to allow for translations that split a propositional variable into two parts. For example, $t(p) = p^+ \wedge \Box p^-$ could be represented by $t(x_1) = s_1(x_1) \wedge \Box s_2(x_1)$ for $x_1 \in \mathcal{P}_1$.

which we represent as

$$\begin{aligned}
t(x_1) &= s_1(x_1) && \text{for } x_1 \in \mathcal{P}_{PAL}\\
t(\neg x_1) &= \neg t(x_2) && \text{for } x_1 \in \Phi_{PAL}\\
t(x_1 \wedge x_2) &= t(x_1) \wedge t(x_2) && \text{for } (x_1, x_2) \in \Phi^2_{PAL}\\
t(\Box x_1) &= \Box t(x_1) && \text{for } x_1 \in \Phi_{PAL}\\
t([x_1]x_2) &= t(x_1) \to t(x_2) && \text{for } (x_1, x_2) \in \Phi_{PAL} \times \mathcal{P}_{PAL}\\
t([x_1]\neg x_2) &= t(x_1) \to \neg t([x_1]x_2) && \text{for } (x_1, x_2) \in \Phi^2_{PAL}\\
t([x_1](x_2 \wedge x_3)) &= t([x_1]x_2) \wedge t([x_1]x_3) && \text{for } (x_1, x_2, x_3) \in \Phi^3_{PAL}\\
t([x_1]\Box x_2) &= t(x_1) \to \Box t([x_1]x_2) && \text{for } (x_1, x_2) \in \Phi^2_{PAL}\\
t([x_1][x_2]x_3) &= t([x_1 \wedge [x_1]x_2]x_3) && \text{for } (x_1, x_2, x_3) \in \Phi^3_{PAL}.
\end{aligned}$$

This works perfectly fine. But now consider a logic PAL$'$ that is exactly like PAL except that it only uses the connectives $\neg, \Box$ and $[\cdot]$ (so no $\wedge$). PAL$'$ is a fragment of PAL so we would like to define a translation t' from PAL$'$ to modal logic that is the restriction of t to PAL$'$. And ideally we would like to define t' by that subset of the clauses of t that can actually occur in PAL$'$. So we would try to define t' inductively by

$$\begin{aligned}
t'(x_1) &= s_1(x_1) && \text{for } x_1 \in \mathcal{P}_{PAL'}\\
t'(\neg x_1) &= \neg t'(x_2) && \text{for } x_1 \in \Phi_{PAL'}\\
t'(\Box x_1) &= \Box t'(x_1) && \text{for } x_1 \in \Phi_{PAL'}\\
t'([x_1]x_2) &= t'(x_1) \to t'(x_2) && \text{for } (x_1, x_2) \in \Phi_{PAL'} \times \mathcal{P}_{PAL'}\\
t'([x_1]\neg x_2) &= t'(x_1) \to \neg t'([x_1]x_2) && \text{for } (x_1, x_2) \in \Phi^2_{PAL'}\\
t'([x_1]\Box x_2) &= t'(x_1) \to \Box t'([x_1]x_2) && \text{for } (x_1, x_2) \in \Phi^2_{PAL'}\\
t'([x_1][x_2]x_3) &= t'([x_1 \wedge [x_1]x_2]x_3) && \text{for } (x_1, x_2, x_3) \in \Phi^3_{PAL'},
\end{aligned}$$

where we simply removed the clauses for $\wedge$. But now we have a problem, because $[x_1 \wedge [x_1]x_2]x_3 \notin \Phi^{X,\emptyset}_{PAL'}$ and therefore the final clause is not allowed. We could try to remove the $\wedge$ connective from the clause, but there is no straightforward way to do so.

Should we then discard the clause entirely? No, we should not. While it is true that $\wedge$ is not a connective of PAL$'$, it is quite harmless to include it in a clause of the translation t', as long as we also include the clause $t'(x_1 \wedge x_2) = t'(x_1) \wedge t'(x_2)$ that allows us to remove the connective later. The connective $\wedge$ is simply an auxiliary symbol that is used during the computation of t'.

This is why, when defining $\Phi_i^{X,C'}$ we allowed an extra set C' of connectives in addition to an extra set of variables. Instead of requiring that $\varphi^X \in \Phi_1^{X,\emptyset}$, we should require that $\varphi^X \in \Phi_1^{X,C'}$ for some finite set C' of auxiliary connectives. Of course we then have to change the requirements for ψ^X and Ψ correspondingly. We should require $\psi^X \in \Phi_2^{T_1^{C'} \cup S_1,\emptyset}$ and $\Psi \subseteq (\Phi^{\emptyset,C'})^k$.

We could consider demanding that the auxiliary symbols C' are only used in intermediate steps, but that is in fact automatic because t is a function from Φ_1 to Φ_2. So we are now done; we can give the formal definition of being finitely generated.

Definition 6.33 (Finitely generated)**.** Let $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ and $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ be logics such that for $1 \leq i \leq 2$, the set Φ_i is generated by a set $\mathcal{P}_i$ of propositional variables and a finite set C_i of connectives. A translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ is finitely generated if there are a finite set C' of auxiliary connectives and a finite set $s_1, \cdots, s_n : \mathcal{P}_1 \to \mathcal{P}_2$ of functions such that t is inductively defined by a finite set of clauses of the form

$$t(\varphi^X) = \psi^X \text{ for } (x_1, \cdots, x_k) \in \Psi$$

where $\varphi^X \in \Phi_1^{X,C'}$, $\psi^X \in \Phi_2^{T_1^{C'} \cup S_1, \emptyset}$ and $\Psi \subseteq (\Phi^{\emptyset,C})^k$ for $T_1^{C'} := \{t(\chi^X) \mid \chi^X \in \Phi_1^{X,C'}\}$ and $S_1 := \{s_i(x_j) \mid 1 \leq i \leq n, j \in \mathbb{N}\}$.

6.11 A Non-trivial Preorder

So far we have shown that expressivity$_g$ satisfies all desiderata about which translations should or should not show a logic to be at least as expressive$_g$ as another, that is, Desiderata 6.1–6.4. What we have not shown is that expressivity$_g$ satisfies Desideratum 6.5, so that expressivity$_g$ corresponds to a non-trivial preorder. In this section we show that expressivity$_g$ satisfies Desideratum 6.5 as well. First let us show that expressivity$_g$ corresponds to a preorder.

Lemma 6.6. *Let $\mathcal{L} = (\Phi, \mathfrak{M}, \models)$ be any logic such that Φ is generated by a set $\mathcal{P}$ of propositional variables and a finite set C of connectives. Then $\mathcal{L}$ is at least as expressive$_g$ as $\mathcal{L}$.*

Proof. Let t be the identity map on Φ and f the identity map on $\mathfrak{M}$. Then it is easily seen that (t, f) is truth-preserving and model based. Left to show is that t is finitely generated. Let s be the identity map on $\mathcal{P}$. For every connective $\circ \in C$ let $r_\circ$ be the arity of $\circ$. Then the set of clauses

$$t(\circ(x_1, \cdots, x_{r_\circ})) = \circ(t(x_1), \cdots, t(x_{r_\circ})) \text{ for } (x_1, \cdots, x_{r_\circ}) \in \Phi^{r_\circ}$$

for every $\circ \in C$ together with the clause

$$t(x_1) = s(x_1) \text{ for } x_1 \in \mathcal{P}$$

inductively define t, thus completing the proof. □

Lemma 6.7. *If $\mathcal{L}_3$ is at least as expressive$_g$ as $\mathcal{L}_2$ and $\mathcal{L}_2$ is at least as expressive$_g$ as $\mathcal{L}_1$, then $\mathcal{L}_3$ is at least as expressive$_g$ as $\mathcal{L}_1$.*

Proof. Suppose $\mathcal{L}_3$ is at least as expressive$_g$ as $\mathcal{L}_2$ and $\mathcal{L}_2$ at least as expressive$_g$ as $\mathcal{L}_1$. Then there are translations (t_1, f_1) from $\mathcal{L}_1$ to $\mathcal{L}_2$ and (t_2, f_2) from $\mathcal{L}_2$ to $\mathcal{L}_3$ that are model based, finitely generated and truth-preserving. Now consider the translation $(t, f) = (t_2 \circ t_1, f_2 \circ f_1)$ from $\mathcal{L}_1$ to $\mathcal{L}_3$.

It should be immediately clear that (t, f) is model based and truth preserving. Left to show is therefore that (t, f) is finitely generated. This can also quite easily be seen however. If we take all connectives of $\mathcal{L}_2$ as well as t_1 and t_2 as auxiliary symbols we can simply take t to be inductively defined by the combination of all clauses of t_1 with all clauses of t_2. □

Of course we usually do not need to retain all clauses of t_1 and t_2. By combining certain clauses, we can reduce the number of clauses, remove the need for some auxiliary symbols and obtain a more elegant inductive definition. Note that, as mentioned when introducing Desideratum 6.5 expressivity$_g$ corresponds to a preorder because the identity map guarantees reflexivity and function composition guarantees transitivity.

Now that we know that expressivity$_g$ corresponds to a preorder and is therefore well-behaved, it is time to show that it is non-trivial. In other words, we should show that there are logics $\mathcal{L}_1$ and $\mathcal{L}_2$ such that $\mathcal{L}_2$ is *not* at least as expressive$_g$ as $\mathcal{L}_1$. But where should we look for such logics?

Suppose we have a logic $\mathcal{L}_1$ and we want to find a logic $\mathcal{L}_2$ such that $\mathcal{L}_2$ is not at least as expressive as $\mathcal{L}_1$. Then the first thing to try is to take $\mathcal{L}_2$ to be basic Modal Logic (ML). There are of course logics that are even less expressive than ML, such as propositional logic interpreted on possible worlds. But generally ML is the least expressive logic such that it is interesting to compare $\mathcal{L}_2$ to it.

Here we want to find $\mathcal{L}_1$ and $\mathcal{L}_2$ such that $\mathcal{L}_2$ is not at least as expressive$_g$ as $\mathcal{L}_1$. It would seem to make sense to try ML as $\mathcal{L}_2$. But it turns out that ML is at least as expressive$_g$ as every other logic discussed in this chapter. So the logic that is often used as a base case with very low expressivity is at least as expressive$_g$ as all those other logics.

That sounds bad, but it isn't. ML may have low expressivity, but it has very high expressivity$_g$. While ML has relatively little flexibility in its connectives it makes up for it with a huge amount of flexibility in its models. The logics discussed in this chapter are evaluated on models that consist of a set of possible worlds and some kind of structure on these worlds. But in almost all cases this structure can easily be unraveled into a relational structure.[16] ML can then be used to reason about this relational structure.

If ML has a high expressivity$_g$, we should not take it for $\mathcal{L}_2$. But we could take it for $\mathcal{L}_1$. What should we take for $\mathcal{L}_2$ then? We could try some of the logics that have been discussed in this chapter. I would for example conjecture that ATL, CL and STIT are all less expressive$_g$ than modal logic. Unfortunately, proving such a conjecture seems to be very hard. It is generally much harder to show that a translation from one logic to another does not exist than it is to show that a translation does exist. This is already true when we are working with the traditional definition of expressivity, but due to the much greater freedom given to translations in the definition of expressivity$_g$ the problem is worse here.

Considering that all we want to do here is to show that not every logic is as expressive$_g$ as basic modal logic it does not seem worthwhile to look for and write down a very complicated proof. So let us consider one more logic for which it can be quite easily shown that it is less expressive$_g$ than basic modal logic.

Recall that $\mathcal{L}_{\mathrm{m}} = (\Phi_{\mathrm{m}}, \mathfrak{M}_{\mathrm{m}}, \models)$ is basic mono-modal logic, where $\mathfrak{M}_{\mathrm{m}}$ is the unrestricted class of relational models, so the class $\mathbf{K}$ of models. Now let $\mathcal{L}_{\mathrm{S5}} = (\Phi_{\mathrm{m}}, \mathfrak{M}_{\mathrm{S5}}, \models)$ be the restriction of $\mathcal{L}_{\mathrm{m}}$ to the class $\mathbf{S5}$ of models. So $\mathfrak{M}_{\mathrm{S5}}$ is the class of triples $\mathcal{M} = (W, R, v)$ where W is a set of possible worlds, $v : \mathcal{P} \to \wp W$ is a valuation function and $R \subseteq W \times W$ is an equivalence relation on W. The logics $\mathcal{L}_{\mathrm{m}}$ and $\mathcal{L}_{\mathrm{S5}}$ have the same set of formulas, $\mathfrak{M}_{\mathrm{S5}}$ is a subclass of $\mathfrak{M}_{\mathrm{m}}$ and the satisfaction relations of the two logics agree on $\mathfrak{M}_{\mathrm{S5}}$.

[16] The only structure that cannot *easily* be unraveled into a relational structure is the neighborhood structure from Classical Modal Logic. This structure can still be simulated using a relational structure, as shown by the translation from [Thomason, 1974]. But this translation is not an easy unraveling.

Lemma 6.8. *$\mathcal{L}_{\mathrm{m}}$ is at least as expressive$_g$ as $\mathcal{L}_{\mathrm{S5}}$ and $\mathcal{L}_{\mathrm{S5}}$ is not at least as expressive$_g$ as $\mathcal{L}_{\mathrm{m}}$.*

Proof. Let t_{id} be the identity map on Φ_{m} and let $f_{\mathrm{inc}} : \mathfrak{M}_{\mathrm{S5}} \to \mathfrak{M}_{\mathrm{m}}$ be the inclusion map. Then $(t_{\mathrm{id}}, f_{\mathrm{inc}})$ is a model based, finitely generated, truth-preserving translation from $\mathcal{L}_{\mathrm{S5}}$ to $\mathcal{L}_{\mathrm{m}}$. So $\mathcal{L}_{\mathrm{m}}$ is at least as expressive$_g$ as $\mathcal{L}_{\mathrm{S5}}$.

Left to show is that $\mathcal{L}_{\mathrm{S5}}$ is not at least as expressive$_g$ as $\mathcal{L}_{\mathrm{m}}$. Let (t, f) be any finitely generated translation from $\mathcal{L}_{\mathrm{m}}$ to $\mathcal{L}_{\mathrm{S5}}$. We will show that (t, f) is not truth-preserving, which implies that $\mathcal{L}_{\mathrm{S5}}$ is not at least as expressive$_g$ as $\mathcal{L}_{\mathrm{m}}$. Let T be the set of clauses associated with (t, f). Fix a propositional variable p; for $i \in \mathbb{N}$ let $\varphi_i = \Diamond^i p$ and let $\Gamma = \{\varphi_i \mid i \in \mathbb{N}\}$.

There is a finite set P of propositional variables that occur either in $t(p)$ or in T. For every $i \in \mathbb{N}$ the propositional variables that occur in $t(\varphi_i)$ must be a subset of this P, so the propositional variables that occur in $t(\Gamma)$ are also a subset of P. Now let $\Delta \subseteq \Phi_{\mathrm{m}}$ be the set of formulas that contain only propositional variables from P.

The set Δ is infinite, but since P is finite there are only a finite number of formulas in Δ that are non-equivalent in $\mathcal{L}_{\mathrm{S5}}$. But $t(\Gamma) \subseteq \Delta$ and Γ contains an infinite number of formulas that are non-equivalent (in $\mathcal{L}_{\mathrm{m}}$). So (t, f) is not truth-preserving, which proves the lemma. □

6.12 Further Generalizations

The definition of expressivity$_g$ requires the logics $\mathcal{L}_1$ and $\mathcal{L}_2$ to have a finite set of connectives. We would like to get rid of this restriction, because there are a number of interesting logics that use an infinite set of connectives. The problem is that our current definition of being finitely generated cannot handle logics with an infinite set of connectives; we simply cannot fully determine the translation of an infinite number of connectives with a finite number of clauses.

There are two ways in which we can generalize the condition of being finitely generated in order to overcome this problem. The first way is to make the condition apply not to the entire logic but to the fragments of the logic with a finite number of connectives. Suppose $\mathcal{L}_1$ is a logic with an infinite number of connectives and (t, f) is a translation from $\mathcal{L}_1$ to $\mathcal{L}_2$. Then we can say that (t, f) is finitely generated$_g$ if and only if for every fragment $\mathcal{L}_1'$ of $\mathcal{L}_1$ with a finite number of connectives, the restriction (t', f) of (t, f) to $\mathcal{L}_1'$ is finitely generated.

The second way to overcome the problem of having an infinite number of connectives is less general but perhaps more elegant. If a logic has an infinite number of connectives, it is common for all but finitely many of these connectives to contain some kind of parameter or variable. As an example let us, once again, consider PAL. In PAL we have a finite number of connectives, namely $\neg, \wedge, \{\Box_a \mid a \in \mathcal{A}\}$ and $[\cdot]$ (where $\mathcal{A}$ is a finite set of agents). But there is a different way of looking at PAL: instead of considering $[\cdot]$ to be a single binary connective, we could consider it to be an infinite number of unary connectives, $\{[\varphi] \mid \varphi \in \Phi_{PAL}\}$.[17] If we consider $[\cdot]$ as an infinite number of connectives then, strictly speaking, a translation from PAL to any other logic cannot be finitely generated.

Something similar happens with some other logics. Consider Propositional Dynamic Logic (PDL) [Fischer and Ladner, 1979], for example. In PDL there is a set $\mathcal{A}$

[17] This would make the definition of Φ_{PAL} circular, but we need not worry about that at the moment.

of *atomic programs* that can be combined in different ways to form modalities. For example, if $a, b \in A$ then $[a \cup b]$ is the nondeterministic choice between a and b, $[a;b]$ is the sequential composition of a and b, and $[(a;b)^*]$ is an arbitrary number of repetitions of $a;b$. The set Π containing all such combinations is infinite, and for every $\pi \in \Pi$ there is an associated modality $[\pi]$. PDL therefore has an infinite number of connectives, so a translation from PDL cannot be finitely generated. But we could try to treat this infinite set $\{[\pi] \mid \pi \in \Pi\}$ as one single connective $[\cdot]$, just like we normally do in PAL.[18]

There is one important difference between the connective $[\cdot]$ from PAL and the connective $[\cdot]$ from PDL, however. The PAL connective is a true binary connective that takes two formulas $\varphi_1, \varphi_2 \in \Phi_{PAL}$ and turns them into a new formula $[\varphi_1]\varphi_2$. Because our variables x_i range over formulas, we can represent this as $[x_1]x_2$ in our translation. The PDL connective $[\cdot]$ on the other hand takes one formula $\varphi \in \Phi_{PDL}$ and one program $\pi \in \Pi$, turning them into a formula $[\pi]\varphi$. But the variables x_i range only over formulas so we cannot represent this by $[x_1]x_2$. In order to represent a translation from PDL to another logic with a finite number of clauses we therefore have to allow variables to range over things other than formulas. Doing so is notationally complicated, but conceptually not very hard. By allowing the variables this greater range, we can find a generalization *finitely generated*$_g$ that allows us to compare at least some logics with infinite numbers of variables.

We have just shown that there are at least two ways in which we could generalize expressivity$_g$. So why did we not include these further generalizations in the definition of expressivity$_g$? The problem is that the prototypical cases that we studied cannot guide us in the adoption of such generalizations. We would need to look for different translations that can be considered prototypical for the generalization of expressivity$_g$. It seems quite likely that such prototypical translations exist, but studying them is outside the scope of this chapter and this thesis.

6.13 Conclusion

In this chapter we considered a number of translations between logics that were introduced in [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b]. We called these translations the prototypical translations. In order to explain why these prototypical translations are interesting while certain other translations are uninteresting we defined a generalization expressivity$_g$ of expressivity: a logic $\mathcal{L}_2 = (\Phi_2, \mathfrak{M}_2, \models_2)$ is at least as expressive$_g$ as a logic $\mathcal{L}_1 = (\Phi_1, \mathfrak{M}_1, \models_1)$ if and only if there is a translation (t, f) from $\mathcal{L}_1$ to $\mathcal{L}_2$ that is truth preserving, model based and finitely generated.

Results are often interesting (partially) because they fit a certain pattern. For example, a result can be interesting because it is a computational complexity result, a completeness result or an expressivity result. We suggest that the prototypical results might be interesting because they are expressivity$_g$ results.

We are not suggesting that expressivity$_g$ is, will be or should be a large area of research. Instead, the definition of expressivity$_g$ should be seen as a prediction: we predict that translations similar to the ones presented in [Thomason, 1974, Gasquet and Herzig, 1996, Goranko and Jamroga, 2004, Broersen et al., 2006a,b] will generally be interesting if they are truth preserving, model based and finitely generated (and are

[18] [van Eijck, 2004b] does exactly this.

therefore expressivity$_g$ results) and that they will generally be uninteresting if they do not have these properties.