Completeness over Kripke models of a cyclic proof system for game logic

E.S. Hoexum

Master Project Mathematics

University of Groningen

July 11, 2022

First supervisor: Prof. dr. H.H. Hansen

Second supervisor: Dr. A.E. Sterk

Abstract

Game logic is a modal logic introduced by Rohit Parikh for reasoning about the outcomes that players can achieve in 2-player games. The language of game logic is usually interpreted over monotone neighbourhood models, but it can also be interpreted through Kripke semantics, which means the atomic games are interpreted as 1-player games. Parikh also gave a Hilbert-style proof system Par *for game logic and it was shown by Enqvist et al. that this system is complete over monotone neighbourhood models. For their proof, they defined the proof systems* CloM*,* CloG *and* G*, which are used as intermediate systems to connect* Par *to a cyclic sequent system* Clo*, which is known to be complete. In the process, all of these systems are shown to be complete over monotone neighbourhood models.*

As game logic can also be interpreted over Kripke models, it is natural to wonder whether the proof systems defined by Enqvist et al. can be adapted to be complete over Kripke models. In this thesis, we adapt CloG *into a new cyclic sequent system for game logic and we show that this system is complete over Kripke models. We do this by giving a validity-preserving translation from game logic to the modal µ-calculus and then defining a transformation from* Clo *to our new system.*

Keywords: *game logic, modal µ-calculus, completeness, proof systems, sequent systems, cyclic systems*

Contents

1. Introduction

1.1. Background

In 1985, Parikh [1] introduced game logic as a modal logic for reasoning about the outcomes that can be achieved in two-player games. The two players in these games are conventionally referred to as *Angel* and *Demon*. The games and formulas in the game logic language are constructed by applying certain operators to atomic games and propositional atoms. One operator that is essential to game logic's ability to describe two-player games is the dual operator $(-)^d$, which indicates the two players switching roles. This means that any moves or strategies that can be applied by Angel in γ are moves or strategies that can be applied by Demon in γ^d, and vice versa. The modal formula $\langle\gamma\rangle\varphi$ indicates that Angel has a strategy in the game γ to achieve an outcome where φ is true. Along with the language and semantics for game logic, Parikh [1] also provided a Hilbert-style proof system Par for the language and sketched a way to translate game logic into the bimodal normal μ-calculus. Later, Pauly [2] gave a translation that showed that game logic can also be seen as a fragment of the monotone μ-calculus.

Semantically, game logic is usually interpreted over *game models*, which were also introduced by Parikh in [1] and are essentially monotone neighbourhood models. These models consist of a set of states, a valuation that defines which propositions are true at each state and an effectivity function E_g for every atomic game g, which assigns to each state a set of neighbourhoods. Effectivity functions E_γ for non-atomic games γ are constructed inductively alongside the notion of truth in a game model. The intuition behind these effectivity function is that $X \in E_\gamma(s)$ means that Angel can achieve the set X when playing the game γ at the state s. Since Kripke models can be seen as a special case of neighbourhood models, it is also possible to interpret game logic over Kripke models. When we do this, all the atomic games are interpreted as 1-player games. It was shown by Pauly [2] that game logic interpreted over Kripke models can be translated directly into the normal μ-calculus, further deepening the connection between game logic and the μ-calculus.

A good property for a proof system to have is completeness with respect to a certain semantics, which means that, whenever a formula is considered valid over that semantics, the formula is derivable in the proof system. The completeness of the proof system Par for game logic introduced by Parikh was an open question for a long time. This was until Enqvist et al. [3] were able to prove the completeness of Par over game models by connecting it to a cyclic sequent system Clo for the normal modal μ-calculus, which was introduced and shown to be complete over Kripke models in [4]. This proof started with taking a game logic formula that was valid over game models and translating it into a valid normal μ-calculus formula. Then, by the completeness of Clo, there is a Clo-derivation of this translated formula and, trough a series of transformations to other proof systems, this derivation is turned into into a Par-derivation for the original formula, thus proving the completeness of Par. The proof systems that are used and the transformations that are performed in [3] are summarised below.

$$\mathsf{Par} \longleftarrow \mathsf{G} \longleftarrow \mathsf{CloG} \longleftarrow \mathsf{CloM} \longleftarrow \mathsf{Clo}$$

Of these proof systems, the systems Clo, CloM and CloG are called cyclic proof systems, since they each contain a closure rule, which is used to detect repeated unfoldings of fixpoints.

1.2. Motivation

Since it is known that game logic can be interpreted over Kripke models and the system Clo is complete over Kripke models, it is natural to wonder if we can adapt the system Par into a similar proof system that is complete over Kripke models. This would also involve defining new proof

systems that are adapted from G and CloG in such a way that they can be shown to be complete over Kripke models by a sequence of transformations similar to the one from [3]. In the original proof, the system CloM was defined to be a monotone analogue of Clo and facilitated switching from Kripke semantics to neighbourhood semantics. However, if we we work with Kripke semantics throughout the entire proof, there is no need for such a system, so the transformation can go directly from Clo to a new proof system that is analogous to CloG.

So, a clear first step in this process is defining a sequent system for game logic that is adapted from the system CloG in such a way that it is complete over Kripke models instead of game models. Once such a proof system has been defined, it could be shown to be complete over Kripke models by finding a way to transform Clo-derivations into derivations from this new system.

1.3. Approach and main results

The aim of this thesis is to adapt the existing proof system CloG into a new proof system for game logic, which we call CloG_K, and prove its completeness over Kripke models. This system is the same as CloG, aside from the fact that it uses the normal modal rule instead of the monotone modal rule. It works over game logic formulas that are annotated with names for formulas of the form $\langle \gamma^\times \rangle \varphi$, which we call *greatest fixpoint formulas*. The names are used to keep track of the unfoldings of greatest fixpoint formulas and together with the closure rule they allow us to detect when a fixpoint formula is unfolded repeatedly, which closes a branch of a proof tree. In order to apply conditions on the closure rule, we use the order $\preccurlyeq$ on the set F of *fixpoint formulas*, which are game logic formulas of the form $\langle \gamma^\times \rangle \varphi$ or $\langle \gamma^* \rangle \varphi$. This order is analogous to the subsumption order on fixpoint variables in the modal μ-calculus.

With the system CloG_K defined, we give a translation $(-)^\sharp$ from normal form game logic formulas to modal μ-calculus formulas and show that it is validity-preserving. For a game logic formula φ, the fixpoint formulas in φ will correspond directly to fixpoint variables in the resulting modal μ-calculus formula $\varphi^\sharp$ and the subsumption order on the fixpoint variables in $\varphi^\sharp$ will be reflected into the order $\preccurlyeq$ on the fixpoint formulas in φ.

Finally, we show how a Clo-derivation for a translated formula $\varphi^\sharp$ can be transformed into a CloG_K-derivation for the formula φ. We do this by showing that every inference rule in the system Clo can be replaced by some sequence of applications of inference rules from CloG_K.

This allows us to prove that CloG_K is complete over Kripke models by the following argument. Say we have a game logic formula φ that is valid over Kripke models, then we know that $\varphi^\sharp$ is also valid over Kripke models, since $(-)^\sharp$ is validity-preserving. Then, by the completeness of Clo, there is a Clo-derivation of $\varphi^\sharp$ and we can then transform it into a CloG_K-derivation for φ.

1.4. Outline

This thesis is structured as follows. In section 2, we review the syntax of modal logic and two semantics that are standardly used to interpret it, namely Kripke and neighbourhood semantics. In section 3, we review the syntax of the modal μ-calculus, along with other useful concepts, and we take a closer look at the system Clo for the modal μ-calculus from [4]. In section 4, we review the syntax and semantics of game logic and we introduce the system CloG_K for game logic. In section 5, we show how Clo-derivations can be transformed into CloG_K-derivations, along with giving the translation $(-)^\sharp$ from game logic formulas into modal μ-calculus formulas, and subsequently show that this proves the completeness of CloG_K over Kripke models. Finally, in section 6, we conclude the thesis and discuss possible future work.

2. Modal logic

2.1. Kripke and neighbourhood models

In this thesis, we will work with proof systems for game logic and the modal μ-calculus. In order to understand these systems, we will first need to define these languages and the different semantics we will interpret them over. In this section, we start with defining the syntax and semantics of modal logic, which is necessary in order to define the syntax and semantics of the modal μ-calculus in subsection 3.1. The definitions given in this section can be found in [5], [6] and [7]. The language of modal logic is the language of propositional logic with the addition of two *modalities*, $\Diamond$ and $\Box$.

Definition 2.1. Given a set Φ_0 of atomic propositions, the language $\mathcal{L}^{\mathsf{ML}}(\Phi_0)$ of *modal logic* is given by the following grammar.

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box\varphi, \quad p \in \Phi_0.$$

When the set Φ_0 is fixed, but arbitrary, we will write $\mathcal{L}^{\mathsf{ML}}$.

The other modality $\Diamond$ and the commonly used connective $\vee$ are not included in the grammar, since they can be defined in terms of the other logical symbols as follows.

$$\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi), \qquad \Diamond\varphi := \neg\Box\neg\varphi.$$

Some other symbols that are commonly used are the connectives $\rightarrow$ and $\leftrightarrow$, which can be defined as follows.

$$\varphi \rightarrow \psi := \neg\varphi \vee \psi, \qquad \varphi \leftrightarrow \psi := (\varphi \wedge \psi) \vee (\neg\varphi \wedge \neg\psi).$$

It is sometimes useful to specify that a formula is in *negation normal form*, meaning that the negation operator only occurs in front of atomic propositions. This condition forces us to add the $\vee$ and $\Diamond$ operators to the grammar.

Definition 2.2. Given a set Φ_0 of atomic propositions, the language $\mathcal{L}^{\mathsf{ML}}_{\mathsf{NF}}(\Phi_0)$ of *negation normal form basic modal logic* is given by the following grammar.

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box\varphi \mid \Diamond\varphi, \quad p \in \Phi_0$$

When the set Φ_0 is fixed, but arbitrary, we will write $\mathcal{L}^{\mathsf{ML}}_{\mathsf{NF}}$.

With the syntax defined, we can now give meaning to the formulas of modal logic with semantics, for example the semantics of Kripke models.

Definition 2.3. Given a set Φ_0 of atomic propositions, a *Kripke model* $\mathcal{K} = \langle S, R, V\rangle$ consists of

- a set of *states* S,
- a *relation* $R \subseteq S \times S$ that defines which states can be accessed from each state, and
- a *valuation* $V : \Phi_0 \rightarrow \mathcal{P}(S)$ that defines at which states each $p \in \Phi_0$ is true.

The class of all Kripke models is denoted $\mathtt{K}$.

We can use the relation R to define $R(s) = \{s' \in S \mid sRs'\}$, which is the set of states that are accessible from a given state s. This means we can also see the relation as a map $R : S \rightarrow \mathcal{P}(S)$ that maps a state s to the set $R(s)$. These Kripke models give us a way to check whether modal logic formulas are true in certain states.

Definition 2.4. In a Kripke model $\mathcal{K} = \langle S, R, V \rangle$, the *truth* of $\varphi \in \mathcal{L}^{\mathsf{ML}}$ at $s \in S$ is defined inductively as follows.

$$\begin{array}{lcll} \mathcal{K}, s \models p & \Leftrightarrow & s \in V(p) & (\text{for } p \in \Phi_0) \\ \mathcal{K}, s \models \neg\varphi & \Leftrightarrow & \mathcal{K}, s \not\models \varphi & \\ \mathcal{K}, s \models \varphi \wedge \psi & \Leftrightarrow & \mathcal{K}, s \models \varphi \text{ and } \mathcal{K}, s \models \psi & \\ \mathcal{K}, s \models \Box\varphi & \Leftrightarrow & \mathcal{K}, s' \models \varphi \text{ for all } s' \in S \text{ such that } sRs' & \end{array}$$

The truth assignments of $\vee$, $\Diamond$ $\rightarrow$ and $\leftrightarrow$ can be found by writing them in terms of the other operators using their definition, but we can also give a direct truth-definition for them as follows.

$$\begin{array}{lcl} \mathcal{K}, s \models \varphi \vee \psi & \Leftrightarrow & \mathcal{K}, s \models \varphi \text{ or } \mathcal{K}, s \models \psi \\ \mathcal{K}, s \models \Diamond\varphi & \Leftrightarrow & \mathcal{K}, s \models \varphi \text{ for some } s' \in S \text{ such that } sRs' \\ \mathcal{K}, s \models \varphi \rightarrow \psi & \Leftrightarrow & \mathcal{K}, s \models \varphi \Rightarrow \mathcal{K}, s \models \psi \\ \mathcal{K}, s \models \varphi \leftrightarrow \psi & \Leftrightarrow & \mathcal{K}, s \models \varphi \Leftrightarrow \mathcal{K}, s \models \psi \end{array}$$

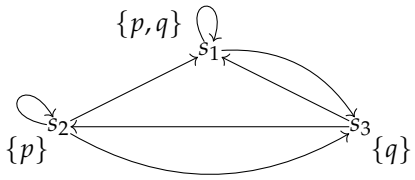
With the truth-definition given, we can define validity with respect to Kripke semantics; this will become important later for the definitions of soundness and completeness in the various proof systems.

Definition 2.5. A formula $\varphi \in \mathcal{L}^{\mathsf{ML}}$ is said to be

- *valid on a Kripke model* $\mathcal{K}$, denoted $\mathcal{K} \models \varphi$, if $\mathcal{K}, s \models \varphi$ for all $s \in S$.
- *valid over Kripke models*, denoted $\models_{\mathsf{K}} \varphi$, if $\mathcal{K} \models \varphi$ for all Kripke models $\mathcal{K}$.

In order to give better intuition for the definitions of truth and validity we provide a short example below.

Example 2.6. For $\Phi_0 = \{p, q\}$, take the Kripke model $\mathcal{K} = \langle S, R, V \rangle$, where $S = \{s_1, s_2, s_3\}$, $R(s_1) = \{s_1, s_3\}, R(s_2) = S, R(s_3) = \{s_1, s_2\}, V(p) = \{s_1, s_2\}$ and $V(q) = \{s_1, s_3\}$. This model is represented visually in the figure below.



Now, let's look at the truth of a few formulas.

1. We have $\mathcal{K}, s_3 \models q$, since $s_3 \in V(q)$.
2. We have $\mathcal{K}, s_3 \models \Box p$, since $\mathcal{K}, s \models p$ for all $s \in R(s_3) = \{s_1, s_2\}$.
3. We have $\mathcal{K}, s_2 \not\models \Box q$, since $s_2 R s_2$ and $\mathcal{K}, s_2 \not\models q$.
4. We have $\mathcal{K}, s_1 \models \Diamond\neg p$, since $s_1 R s_3$ and $\mathcal{K}, s_3 \models \neg p$.
5. We have $\mathcal{K} \models \Diamond q$, since $\mathcal{K}, s_1 \models q$ and sRs_1 for all $s \in S$.
6. We have $\mathcal{K}, s_2 \models \Box\Diamond q$, since $\mathcal{K}, s \models \Diamond q$ for all $s \in R(s_2) = S$.

Instead of using a relation to indicate how the states are related to each other, we could also define a *neighbourhood function* $N : S \rightarrow \mathcal{P}(\mathcal{P}(S))$. Instead of one set, such a function assigns to a state s a set $N(s)$ of neighbourhoods. This gives rise to a different kind of model.

Definition 2.7. Given a set Φ_0 of atomic propositions, a *neighbourhood model* $\mathcal{N} = \langle S, N, V \rangle$ consists of

- a set of states S,
- a neighbourhood function $N : S \to \mathcal{P}(\mathcal{P}(S))$ and
- a valuation $V : \Phi_0 \to \mathcal{P}(S)$.

The class of all neighbourhood models is denoted by $\mathtt{N}$.

In this thesis, we will mostly be working with the class of *monotone neighbourhood models*.

Definition 2.8. Given two partially ordered sets $\mathbb{P} = (P, \leq)$ and $\mathbb{P}' = (P', \leq')$, a map $f : P \to P'$ is considered *monotone* if, for all $x, y \in P$, we have

$$x \leq y \quad \Rightarrow \quad f(x) \leq' f(y)$$

Since $(\mathcal{P}(S), \subseteq)$ is a partially ordered set, if we interpret a neighbourhood function as a map $N : \mathcal{P}(S) \to \mathcal{P}(S)$ by setting $N(X) := \{s \in S \mid X \in N(s)\}$, we can apply the definition above and obtain a definition of monotone neighbourhood functions and models.

Definition 2.9. A neighbourhood function $N : S \to \mathcal{P}(\mathcal{P}(S))$ is *monotone* if, for all $s \in S$ and $X, Y \subseteq S$ such that $X \subseteq Y$, we have

$$X \in N(s) \quad \Rightarrow \quad Y \in N(s).$$

We say a neighbourhood model is *monotone* when its neighbourhood function is monotone. The class of all monotone neighbourhood models is denoted by $\mathtt{N}_{\mathtt{mon}}$.

A neighbourhood model is similar to a Kripke model, so most of the truth-definition is the same. However, to define truth for the box-modality, we first need to define the truth-set.

Definition 2.10. Given a language $\mathcal{L}$ and a model $\mathcal{M}$ defined on $\mathcal{L}$, the *truth-set* of $\varphi \in \mathcal{L}$ in $\mathcal{M}$ is

$$[\![\varphi]\!]_{\mathcal{M}} = \{s \in S \mid \mathcal{M}, s \models \varphi\},$$

which contains all the states at which φ is true in the model $\mathcal{M}$.

Definition 2.11. In a neighbourhood model $\mathcal{N} = \langle S, N, V \rangle$, the truth of $\varphi \in \mathcal{L}^{\mathsf{ML}}$ at $s \in S$ is defined inductively as follows.

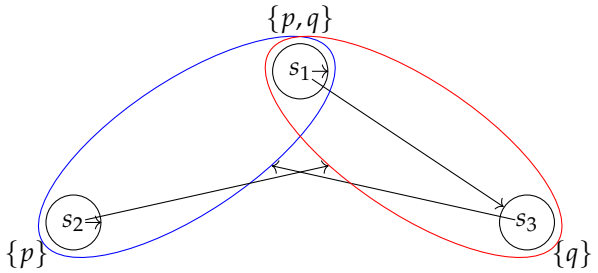
$$\begin{array}{lll}
\mathcal{N}, s \models p & \Leftrightarrow \quad s \in V(p) & (\text{for } p \in \Phi_0) \\
\mathcal{N}, s \models \neg\varphi & \Leftrightarrow \quad \mathcal{N}, s \not\models \varphi & \\
\mathcal{N}, s \models \varphi \wedge \psi & \Leftrightarrow \quad \mathcal{N}, s \models \varphi \text{ and } \mathcal{N}, s \models \psi & \\
\mathcal{N}, s \models \Box\varphi & \Leftrightarrow \quad [\![\varphi]\!]_{\mathcal{N}} \in N(s). &
\end{array}$$

Again, we could find the truth assignments of $\vee$ and $\Diamond$ by writing out their definitions, but a direct truth-definition is more convenient.

$$\begin{array}{ll}
\mathcal{N}, s \models \varphi \vee \psi & \Leftrightarrow \quad \mathcal{N}, s \models \varphi \text{ or } \mathcal{N}, s \models \psi \\
\mathcal{N}, s \models \Diamond\varphi & \Leftrightarrow \quad [\![\varphi]\!]^{C}_{\mathcal{N}} \notin N(s).
\end{array}$$

The definition of validity for neighbourhood semantics similar to the one given for Kripke semantics in Definition 2.5. We use $\models_{\mathtt{N}}$ to denote validity over neighbourhood models and $\models_{\mathtt{N}_{\mathtt{mon}}}$ for validity over monotone neighbourhood models. Below we provide an example of a neighbourhood model and some true or valid formulas.

Example 2.12. For $\Phi_0 = \{p, q\}$, take $\mathcal{N} = \langle S, N, V\rangle$, where $S = \{s_1, s_2, s_3\}, V(p) = \{s_1, s_2\}, V(q) = \{s_1, s_3\}, N(s_1) = \{\{s_1\}, \{s_3\}\}, N(s_2) = \{\{s_2\}, \{s_1, s_3\}\}$ and $N(s_3) = \{\{s_1, s_2\}\}$. This model is represented visually in the figure below.



Now, let's look at the truth of a few formulas.

1. We have $\mathcal{N}, s_1 \models p$, since $s_1 \in V(p)$.
2. We have $\mathcal{N}, s_3 \models \Box p$, since $[\![p]\!]_{\mathcal{N}} = \{s_1, s_2\} \in N(s_3)$.
3. We have $\mathcal{N}, s_2 \models \Box q$, since $[\![q]\!]_{\mathcal{N}} = \{s_1, s_3\} \in N(s_2)$.
4. We have $\mathcal{N}, s_1 \models \Box\neg p$, since $[\![\neg p]\!]_{\mathcal{N}} = \{s_3\} \in N(s_1)$.
5. We have $\mathcal{N}, s_2 \models \Box\neg q$, since $[\![\neg q]\!]_{\mathcal{N}} = \{s_2\} \in N(s_2)$.
6. We have $\mathcal{N}, s_1 \models \Diamond q$ and $\mathcal{N}, s_3 \models \Diamond q$, since $[\![q]\!]^C_{\mathcal{N}} = \{s_2\} \notin N(s_1) \cup N(s_3)$.
7. We have $\mathcal{N}, s_1 \models \Box\Box p$, since $[\![\Box p]\!]_{\mathcal{N}} = \{s_3\} \in N(s_1)$.
8. We have $\mathcal{N}, s_2 \models \Box\Diamond q$, since $[\![\Box q]\!]_{\mathcal{N}} = \{s_1, s_3\} \in N(s_2)$.
9. We have $\mathcal{N} \models \Diamond\neg(p \vee q)$, since $[\![\neg(p \vee q)]\!]^C_{\mathcal{N}} = \emptyset^C = S \notin N(s_1) \cup N(s_2) \cup N(s_3)$.

2.2. *Augmented neighbourhood models*

We can see a relation R as a neighbourhood function that associates exactly one set, $R(s)$, with each state s. Since we can see relations as a special kind of neighbourhood functions, we can see a Kripke model as a special type of neighbourhood model. In this section, we will show that the class $\mathtt{K}$ of Kripke models, in fact, corresponds directly to a specific subclass of neighbourhood models, namely the class $\mathtt{N}_{\mathtt{aug}}$ of *augmented neighbourhood models*. The definitions and results given in this section were first established by Chellas in [5].

Definition 2.13. Given a set of states S, a neighbourhood function $N : S \to \mathcal{P}(\mathcal{P}(S))$ is *augmented* if, for all $s \in S$, the set $N(s)$:

- contains its core.
$$\bigcap N(s) \in N(s)$$
- is upwards closed.
$$X \in N(s), X \subseteq Y \Rightarrow Y \in N(s)$$

A neighbourhood model is *augmented* if its neighbourhood function is augmented. We say a formula $\varphi \in \mathcal{L}^{\mathsf{ML}}$ is *valid over augmented neighbourhood models*, denoted $\models_{\mathtt{N}_{\mathtt{aug}}} \varphi$, if $\mathcal{N} \models \varphi$ for all augmented neighbourhood models $\mathcal{N}$.

In order to show the correspondence between augmented neighbourhood models and Kripke models, we will first show a way to construct an augmented neighbourhood function on S, given a relation on S, and vice versa.

Definition 2.14. Given a relation $R : S \to \mathcal{P}(S)$, the *corresponding neighbourhood function* is

$$N_R(s) = \{X \subseteq S \mid R(s) \subseteq X\}.$$

Given an augmented neighbourhood function $N : S \to \mathcal{P}(\mathcal{P}(S))$, the *corresponding relation* is

$$R_N(s) = \cap N(s).$$

Lemma 2.15. For all relations R, the corresponding neighbourhood function N_R is augmented.

Proof. Fix $s \in S$. Say $s' \in R(s)$, then $s' \in X$ for all X such that $R(s) \subseteq X$. This means that

$$s' \in \bigcap_{R(s) \subseteq X} X = \bigcap_{X \in N_R(s)} X = \bigcap N_R(s)$$

So $R(s) \subseteq \cap N_R(s)$, which means that $\cap N_R(s) \in N_R(s)$. Next, say that $X \in N_R(s)$ and $X \subseteq Y$, then

$$R(s) \subseteq X \subseteq Y \quad \Rightarrow \quad Y \in N_R(s).$$

The two conditions from Definition 2.13 are satisfied, so N_R is augmented. □

It is easily shown that the two constructions given in Definition 2.14 are each others inverse. This implies that they define a bijection between relations and augmented neighbourhood functions.

Lemma 2.16. For all relations R and augmented neighbourhood functions N, we have

$$R_{N_R}(s) = R(s) \quad \text{and} \quad N_{R_N}(s) = N(s) \qquad \forall s \in S.$$

Proof. Take an arbitrary relation R, then, for all $s \in S$, we have

$$R_{N_R}(s) = \cap N_R(s) = \bigcap \{X \subseteq S \mid R(s) \subseteq X\} = R(s).$$

Given an arbitrary augmented neighbourhood function N, we have, for all $s \in S$,

$$N_{R_N}(s) = \{X \subseteq S \mid R_N(s) \subseteq X\} = \{X \subseteq S \mid \cap N(s) \subseteq X\}.$$

Since N is augmented, we know that $\cap N(s) \in N(s)$ and that it is upwards closed. This means that $\cap N(s) \subseteq X$ implies that $X \in N(s)$, so $N_{R_N}(s) = \{X \subseteq S \mid X \in N(s)\} = N(s)$. □

We say a relation R and augmented neighbourhood function N correspond if $R = R_N$ or, equivalently, $N = N_R$. We lift this correspondence relation to Kripke models and augmented neighbourhood models by saying a Kripke model $\mathcal{K} = \langle S, R, V \rangle$ and an augmented neighbourhood model $\mathcal{N} = \langle S, N, V \rangle$ correspond whenever R and N correspond. We will now show that these classes of models are *pointwise modally equivalent*, meaning that if a formula is true at some state s in a Kripke model, then it is also true at s in the corresponding augmented neighbourhood model, and vice versa. This result was first shown by Chellas in [5, Theorem 7.9].

Theorem 2.17. For all corresponding Kripke models $\mathcal{K} = \langle S, R, V \rangle$ and augmented neighbourhood models $\mathcal{N} = \langle S, N, V \rangle$, all $s \in S$ and all $\varphi \in \mathcal{L}^{\mathsf{ML}}$, we have

$$\mathcal{K}, s \models \varphi \quad \Leftrightarrow \quad \mathcal{N}, s \models \varphi$$

Proof. Take an arbitrary Kripke model $\mathcal{K} = \langle S, R, V \rangle$, then the corresponding augmented neighbourhood model is $\mathcal{N} = \langle S, N_R, V \rangle$. We will prove the claim using structural induction on φ.
For the base step, take an arbitrary atomic proposition p. In that case

$$\mathcal{K}, s \models p \quad \Leftrightarrow \quad s \in V(p) \quad \Leftrightarrow \quad \mathcal{N}, s \models p.$$

For the inductive hypothesis, assume that we have modal logic formulas φ and ψ such that

$$\mathcal{K}, s \models \varphi \quad \Leftrightarrow \quad \mathcal{N}, s \models \varphi \qquad \text{and} \qquad \mathcal{K}, s \models \psi \quad \Leftrightarrow \quad \mathcal{N}, s \models \psi.$$

Now, for the inductive step, we prove the claim for each formula that can be constructed by applying one operator to φ or to φ and ψ.
For $\neg\varphi$ we have

$$\mathcal{K}, s \models \neg\varphi \quad \Leftrightarrow \quad \mathcal{K}, s \not\models \varphi \quad \overset{\text{(IH)}}{\Leftrightarrow} \quad \mathcal{N}, s \not\models \varphi \quad \Leftrightarrow \quad \mathcal{N}, s \models \neg\varphi.$$

For $\varphi \wedge \psi$, we have

$$\begin{aligned} \mathcal{K}, s \models \varphi \wedge \psi \quad &\Leftrightarrow \quad \mathcal{K}, s \models \varphi \text{ and } \mathcal{K}, s \models \psi \\ \text{(IH)} \Leftrightarrow \quad &\mathcal{N}, s \models \varphi \text{ and } \mathcal{N}, s \models \psi \\ \Leftrightarrow \quad &\mathcal{N}, s \models \varphi \wedge \psi \end{aligned}$$

For $\Box\varphi$, we have

$$\begin{aligned} \mathcal{K}, s \models \Box\varphi \quad &\Leftrightarrow \quad \mathcal{K}, s \models \varphi \text{ for all } s' \in S \text{ s.t. } sRs' \\ \text{(IH)} \Leftrightarrow \quad &\mathcal{N}, s \models \varphi \text{ for all } s' \in S \text{ s.t. } sRs' \\ \Leftrightarrow \quad &R(s) \subseteq [\![\varphi]\!]_{\mathcal{N}} \\ \Leftrightarrow \quad &[\![\varphi]\!]_{\mathcal{N}} \in N_R(s) \\ \Leftrightarrow \quad &\mathcal{N}, s \models \Box\varphi \end{aligned}$$

We have now proven by induction that for all modal logic formulas φ, we have

$$\mathcal{K}, s \models \varphi \quad \Leftrightarrow \quad \mathcal{N}, s \models \varphi.$$

□

From this result, it easily follows that these classes of models have the same valid formulas.

Theorem 2.18. For all modal formulas φ,

$$\models_{\mathtt{K}} \varphi \quad \Leftrightarrow \quad \models_{\mathtt{N_{aug}}} \varphi$$

Proof. We will prove this claim by contraposition, so we will show that, for all $\varphi \in \mathcal{L}^{\mathsf{ML}}$,

$$\not\models_{\mathtt{K}} \varphi \quad \Leftrightarrow \quad \not\models_{\mathtt{N_{aug}}} \varphi.$$

First, assume we have $\varphi \in \mathcal{L}^{\mathsf{ML}}$ such that $\not\models_{\mathtt{K}} \varphi$. This means that there exists a Kripke model $\mathcal{K} = \langle S, R, V \rangle$ and state $s \in S$ such that $\mathcal{K}, s \not\models \varphi$. Let $\mathcal{N}$ be the augmented neighbourhood model that corresponds to $\mathcal{K}$, then, by Theorem 2.17, we have $\mathcal{N}, s \not\models \varphi$, which implies that $\not\models_{\mathtt{N_{aug}}} \varphi$.

Next, assume we have $\varphi \in \mathcal{L}^{\mathsf{ML}}$ such that $\not\models_{\mathtt{N}_{\mathtt{aug}}} \varphi$. Then, there is some augmented neighbourhood model $\mathcal{N} = \langle S, N, V \rangle$ and state $s \in S$ such that $\mathcal{N}, s \not\models \varphi$. Let $\mathcal{K}$ be the Kripke model that corresponds to $\mathcal{N}$, then, by Theorem 2.17, we know that $\mathcal{K}, s \not\models \varphi$, which means that $\not\models_{\mathtt{K}} \varphi$.

□

So far, we have been working with models that use only one relation or neighbourhood function, but, further on in this thesis, we will work with Kripke models and neighbourhood models that use a set of functions. We say such a neighbourhood model is augmented if all of its neighbourhood functions are augmented. The correspondence relation defined in this section can easily be expanded to these Kripke models and augmented neighbourhood models with multiple functions and thus, modal equivalence can be shown for the class of Kripke models and the class of augmented neighbourhood models with the same number of functions.

2.3. Normal vs. monotone modal logic

Proof systems are systems that are used to formalize logical arguments. These proof systems are defined for some language and consist of a list of *axioms*, which are formulas that are assumed to always hold, and a list of *inference rules*, which are of the form

$$\frac{A_1, \ldots, A_n}{A},$$

where $A_1, \ldots, A_n$ are the premises and A is the conclusion. One example of a class of proof systems is the class of Hilbert systems, which were first introduced by David Hilbert.

Definition 2.19. A *Hilbert system* H defined for the language $\mathcal{L}$ consists of a set of axioms, which are formulas from $\mathcal{L}$, and a set of inference rules, whose premises and conclusion are formulas from $\mathcal{L}$. A *derivation* in a Hilbert system H is a finite sequence of formulas from $\mathcal{L}$, where each formula is either an axiom in H or the conclusion of an inference rule of H, whose premises occurred earlier in the sequence. A formula $\varphi \in \mathcal{L}$ has an H-*derivation*, denoted $\vdash_{\mathsf{H}} \varphi$, if there is a derivation in H with φ as its last line.

To improve readability, each formula in an H-derivation is numbered and annotated with the axiom it is an instance of, or the inference rule it follows from along with the numbers of the premises. An example of a Hilbert system is the system Par for game logic given by Parikh in [1].

Another class of proof systems is the class of sequent systems. Derivations in these systems are not a list of formulas, like in a Hilbert system, but a tree, where each node is labelled with a *sequent* $\Delta = A_1, \ldots, A_n$, which is a finite set of formulas. An inference rule in a sequent system connects the nodes with its premises to the node with its conclusion, where the premise nodes are called *successors* of the conclusion node. Inference rules may have multiple premises, which leads to the derivations having the shape of a tree, which is built upwards from one *root sequent*. A *leaf* is a node with no successors and a *branch* is a path from the root to a leaf.

Definition 2.20. A *sequent system* (or *sequent calculus*) S defined for the language $\mathcal{L}$ consists of a set of axioms, which are formulas from $\mathcal{L}$, and a set of inference rules, whose premises and conclusion are formulas from $\mathcal{L}$. A *derivation* in a sequent system S is a tree of sequent from $\mathcal{L}$, where each sequent is either an axiom in S or follows from the formula(s) directly above it by some inference rule from S. A formula $\varphi \in \mathcal{L}$ has an S-*derivation*, denoted $\vdash_{\mathsf{S}} \varphi$, if there is a derivation in S with φ as its root.

Within a derivation in a sequent system, a sequent $\Delta = A_1, \ldots, A_n$ is interpreted as the disjunction of all the A_i's. Some examples of sequent systems are Clo from [4] and CloG from [3].

Further on in this thesis, we introduce a sequent system called $\mathsf{CloG}_{\mathsf{K}}$, which is created by replacing one of the inference rules of CloG.

There are some properties that are desirable for a system to have, relative to certain semantics. For a system S and a class of models C defined for the same language $\mathcal{L}$, we would like to know that if there is a derivation for a formula in S, then that formula is also valid over the semantics of C. Otherwise, we would be able to derive formulas that are not valid, which means that the system is not useful in combination with this semantics. This property is known as *soundness*.

Definition 2.21 (Soundness)**.** Given a language $\mathcal{L}$, a class C of models defined for $\mathcal{L}$ and a proof system S defined for $\mathcal{L}$, S is *sound with respect to* C if for all $\varphi \in \mathcal{L}$

$$\vdash_{\mathsf{S}} \varphi \quad \Rightarrow \quad \models_{\mathsf{C}} \varphi.$$

The converse is also useful to know, namely that for any formula that is valid in the class of models C, there exists a derivation in the system S. If this is the case, we say the system S is *complete* w.r.t the class of models C.

Definition 2.22 (Completeness)**.** Given a language $\mathcal{L}$, a class C of models defined for $\mathcal{L}$ and a proof system S defined for $\mathcal{L}$, S is *complete with respect to* C if, for all $\varphi \in \mathcal{L}$

$$\models_{\mathsf{C}} \varphi \quad \Rightarrow \quad \vdash_{\mathsf{S}} \varphi.$$

Within the class of Hilbert systems for modal logic, some sub-classes have been defined. Two that are of interest for our thesis are the systems that are called *normal* and those that are called *monotone*. These types of systems can be identified by certain axioms and inferences, but a system does not necessarily need to have these axioms and inferences to be considered *normal* or *monotone*.

Definition 2.23. Given a Hilbert system defined for $\mathcal{L}$, let $T = \{A \in \mathcal{L} \mid \vdash_{\mathsf{H}} A\}$. We say H

- *contains* the axiom A, if T contains all instances of A.
- *is closed under the inference rule* IR if, whenever T contains the premises of IR, it also contains the conclusion of IR.

Definition 2.24. A Hilbert system H for the language $\mathcal{L}^{\mathsf{ML}}$ is *normal* iff it contains all propositional tautologies, the axiom

$$\text{K.} \quad \Box(A \to B) \to (\Box A \to \Box B)$$

and is closed under the inference rules

$$\text{Nec.} \quad \frac{A}{\Box A} \qquad \text{and} \qquad \text{MP} \quad \frac{A \to B, A}{B}.$$

The last inference rule in this definition is also known as *modus ponens*. The *minimal normal system* K is the Hilbert system that has all propositional tautologies, the axiom K and the inference rules Nec and MP. This system is known to be both sound and complete with respect to the class K of Kripke models [5, Theorem 5.12].

Definition 2.25. A Hilbert system H for the modal logic language $\mathcal{L}^{\mathsf{ML}}$ is *monotone* iff it contains all propositional tautologies, the axiom

$$\text{M.} \quad \Box(A \wedge B) \to \Box A$$

and is closed under the inference rule MP.

The *minimal monotone system* M is the Hilbert system that has all propositional tautologies, the axiom M and the inference rule MP. This system is known to be both sound and complete with respect to the class $\mathtt{N}_{\mathtt{mon}}$ of monotone neighbourhood models [5, Theorem 9.8].

So we have seen that the minimal normal system K is compatible with Kripke semantics and the minimal monotone system M is compatible with monotone neighbourhood semantics. The opposite is not true, however, as K is not sound over the class $\mathtt{N}_{\mathtt{mon}}$ of monotone neighbourhood models. This can be easily shown by an example of a formula that is provable in K, but not valid over monotone neighbourhood models.

Example 2.26. Take $\varphi = \Box(p \to q) \to (\Box p \to \Box q)$. This is an instance of the axiom K, so $\vdash_{\mathsf{K}} \varphi$. Take $\mathcal{N} = \langle S, N, V\rangle$, where $S = \{s_1, s_2, s_3\}$, $V(p) = \{s_1, s_2\}$, $V(q) = \{s_2\}$, $N(s_1) = \{\{s_1, s_2\}, \{s_2, s_3\}, S\}$ and $N(s_2) = N(s_3) = S$. The neighbourhood function N satisfies Definition 2.9, so this is a monotone neighbourhood model. We will show that, in this model, φ is not true at s_1.

We have $\mathcal{N}, s_2 \models p$ and $\mathcal{N}, s_2 \models q$, so we can say $\mathcal{N}, s_2 \models p \Rightarrow \mathcal{N}, s_2 \models q$. This means that $\mathcal{N}, s_2 \models p \to q$. Also, $\mathcal{N}, s_3 \not\models p$ and $\mathcal{N}, s_3 \not\models q$, so $\mathcal{N}, s_3 \models p \to q$. However, $\mathcal{N}, s_1 \models p$ and $\mathcal{N}, s_1 \not\models q$, so $\mathcal{N}, s_1 \not\models p \to q$. This gives us $[\![p \to q]\!]_{\mathcal{N}} = \{s_2, s_3\} \in N(s_1)$, so $\mathcal{N}, s_1 \models \Box(p \to q)$.

Also, $[\![q]\!]_{\mathcal{N}} = \{s_2\} \notin N(s_1)$, so $\mathcal{N}, s_1 \not\models \Box q$, but since $[\![p]\!]_{\mathcal{N}} = \{s_1, s_2\} \in N(s_1)$, we do have $\mathcal{N}, s_1 \models \Box p$. This means that $\mathcal{N}, s_1 \not\models \Box p \to \Box q$, so $\mathcal{N}, s_1 \not\models \Box(p \to q) \to (\Box p \to \Box q)$. So there is a monotone neighbourhood model $\mathcal{N}$ and state s_1 such that $\mathcal{N}, s_1 \not\models \varphi$, so $\not\models_{\mathsf{N}_{\mathsf{mon}}} \varphi$.

Similarly, the system M is not complete over the class $\mathtt{K}$ of Kripke models, which we can again show via an example of a formula that is valid over Kripke models, but does not have a M-derivation. However, M *is* sound over Kripke models, which aligns with the fact that the class of Kripke models has the same valid formulas as a subclass of neighbourhood models, which we saw in Theorem 2.18. Below, we give an example of a formula that is valid over Kripke models, but has no derivation in M in order to show that M is indeed not complete over $\mathtt{K}$.

Example 2.27. Take $\varphi = (\Box p \wedge \Box q) \to \Box(p \wedge q)$. To show that this formula is valid over Kripke models, take an arbitrary Kripke model $\mathcal{K} = \langle S, R, V\rangle$ and $s \in S$ and assume that $\mathcal{K}, s \models \Box p \wedge \Box q$. This means that $\mathcal{K}, s \models \Box p$ and $\mathcal{K}, s \models \Box q$, so, for all $s' \in S$ s.t. sRs', we have $\mathcal{K}, s' \models p$ and $\mathcal{K}, s' \models q$. This means that $\mathcal{K}, s' \models p \wedge q$ for all such $s' \in S$, so $\mathcal{K}, s \models \Box(p \wedge q)$. This shows that

$$\mathcal{K}, s \models \Box p \wedge \Box q \quad \Rightarrow \quad \mathcal{K}, s \models \Box(p \wedge q),$$

so $\mathcal{K}, s \models \Box(p \wedge q) \to \Box p \wedge \Box q$ for all Kripke models $\mathcal{K}$ and states s. So, we have $\models_{\mathtt{K}} \varphi$.

Now, take $\mathcal{N} = \langle S, N, V\rangle$, where $S = \{s_1, s_2\}$, $V(p) = \{s_1\}$, $V(q) = \{s_2\}$, $N(s_1) = \{\{s_1\}, \{s_2\}, S\}$ and $N(s_2) = S$. Since N satisfies Definition 2.9, $\mathcal{N}$ is monotone. We have $[\![p]\!]_{\mathcal{N}} = \{s_1\} \in N(s_1)$ and $[\![q]\!]_{\mathcal{N}} = \{s_2\} \in N(s_1)$, so $\mathcal{N}, s_1 \models \Box p$ and $\mathcal{N}, s_1 \models \Box q$. This means that $\mathcal{N}, s_1 \models \Box p \wedge \Box q$. However, $[\![p \wedge q]\!]_{\mathcal{N}} = \emptyset \notin N(s_1)$, so $\mathcal{N}, s_1 \not\models \Box(p \wedge q)$. This shows that $\mathcal{N}, s_1 \not\models (\Box p \wedge \Box q) \to \Box(p \wedge q)$, so $\not\models_{\mathtt{N}_{\mathtt{mon}}} \varphi$. Since M is sound w.r.t. $\mathtt{N}_{\mathtt{mon}}$, this means that $\not\vdash_{\mathsf{M}} \varphi$.

Another inference rule that can be used to define when a Hilbert system is normal is the *normal modal rule*.

$$\frac{(A_1 \wedge \cdots \wedge A_n) \to A}{(\Box A_1 \wedge \cdots \wedge \Box A_n) \to \Box A}\ \text{mod}$$

It turns out that a Hilbert system is normal if and only if it is closed under the inference rules mod and MP and it contains all propositional tautologies. This is in fact how the notion of a normal system is defined in [5]. Similarly, we can also give a different definition of monotone Hilbert systems based on the *monotone modal rule*.

$$\frac{A \to B}{\Box A \to \Box B}\ \text{mod}_m$$

A Hilbert system is *monotone* if and only if it is closed under mod_m and MP and contains all propositional tautologies. The monotone modal rule is a special case of the normal modal rule and their connection to normal and monotone Hilbert systems implies that the normal modal rule is valid over Kripke models and the monotone modal rule is valid in monotone neighbourhood models. To understand the change we make to the proof system CloG in order to obtain the new proof system CloG_K, we need to look at the variants of these rules for sequent systems. Below, the normal modal rule for sequent systems is given on the left and the monotone one on the right.

$$\frac{A_1, \ldots, A_n, A}{\Diamond A_1, \ldots, \Diamond A_n, \Box A} \qquad \frac{A, B}{\Diamond A, \Box B}$$

These rules describe the same logical inferences as the modal rules for Hilbert systems. This can be seen for the monotone rule by plugging in $\neg A$ for A and observing that the sequent $\neg A, B$ is interpreted as $\neg A \vee B$, which is equivalent to $A \to B$. This means the premises of both monotone rules are equivalent and, by a similar reasoning, the conclusions are as well. Similarly, both normal modal rules are also equivalent. This implies that, for sequent systems, the normal modal rule is also valid over Kripke models and the monotone modal rule is valid over monotone neighbourhood models. We will use this connection in subsection 4.3 when we adapt the proof system CloG into a new proof system. By replacing the monotone modal rule in CloG with a normal modal rule, we create the proof system CloG_K that is designed to be complete over Kripke models.

3. The modal μ-calculus

In this chapter, we will review the syntax and semantics of the model μ-calculus and various other useful definitions, which can also be found in [8].

3.1. Fixpoints

In the modal μ-calculus, we work with two fixpoint operators, so before we can define its semantics, we will first need to give the definitions of greatest and least fixpoints. For any set S, we know that the partially ordered set $(\mathcal{P}(S), \subseteq)$ has

- S as its supremum, as $X \subseteq S$ for all $X \in \mathcal{P}(S)$, and
- $\emptyset$ as its infimum, as $\emptyset \subseteq X$ for all $X \in \mathcal{P}(S)$.

Furthermore, if we take $P \subseteq \mathcal{P}(S)$, then we know that

- $\bigcup P = \bigcup_{X \in P} X$ is the smallest set that has all sets in P as a subset, so it is the supremum of P.
- $\bigcap P = \bigcap_{X \in P} X$ is the largest set that is a subset of all sets in P, so it is the infimum of P.

As every subset of $\mathcal{P}(S)$ has a supremum and infimum, it is a complete lattice w.r.t. $\subseteq$. If we have a monotone function on a complete lattice, we can apply the following fixed point theorems from [9] and [10] respectively.

Theorem 3.1 (Knaster-Tarski Theorem)**.** If $(L, \preccurlyeq)$ is a complete lattice and $F : L \to L$ is a monotone function, then the set of fixpoints of F in L is also a complete lattice w.r.t. $\preccurlyeq$.

Theorem 3.2 (Kleene's fixpoint theorem)**.** If $(L, \preccurlyeq)$ is a non-empty complete lattice and $F : L \to L$ is monotone, then F has a *least fixed point* $\mathsf{lfp}.F$ (or $\mathsf{lfp}\ X.F(X)$), which is the supremum of

$$\bot \preccurlyeq F(\bot) \preccurlyeq \cdots \preccurlyeq F^n(\bot) \preccurlyeq \ldots,$$

where $\bot$ is the infimum of L.

By inverting the partial order in the last theorem, we find that F also has a *greatest fixed point* $\mathsf{gfp}.F$ (or $\mathsf{gfp}\ X.F(X)$), which is the infimum of the chain beginning with $\top$, the supremum of L.

$$\top \succcurlyeq F(\top) \succcurlyeq \cdots \succcurlyeq F^n(\top) \succcurlyeq \ldots$$

3.2. Syntax

Poly-modal logic consists of the language of modal logic with the addition of a set of actions and, instead of the general modalities $\Box$ and $\Diamond$, two modalities $\Box_d$ and $\Diamond_d$ for each action d. The modal μ-calculus adds to poly-modal logic the least and greatest fixpoint operators μ and ν. In this thesis, we will only work with modal μ-calculus formulas in negation normal form.

Definition 3.3. Given a set $\Phi_0 = \{p, q, x, y, \ldots\}$ of propositional variables and a set $D = \{d, e, \ldots\}$ of actions, the language $\mathcal{L}^{\mu}_{\mathsf{NF}}(\Phi_0, D)$ of *negation normal form (nnf) modal μ-calculus formulas* is given by the following grammar.

$$A ::= p \mid \neg p \mid (A \wedge A) \mid (A \vee A) \mid \Box_d A \mid \Diamond_d A \mid \mu x.A \mid \nu x.A,$$

where $p, x \in \Phi_0$ and $d \in D$. Propositional variables that are preceded by a fixpoint operator μ or ν are called *fixpoint variables* and denoted by x, y, z. We add the condition that for any fixpoint formula $\mu x.A$ or $\nu x.A$, the formula A must be *positive in x*, meaning that every occurrence of x

in A is not of the form $\neg x$. From now on, when a fixpoint operator is applied to a formula, this condition is implicit. The scope of the fixpoint operators extends as far as possible to the right. When Φ_0 and D are fixed, but arbitrary, we write $\mathcal{L}^{\mu}_{\mathsf{NF}}$.

With this grammar established, we can now also define the concept of a subformula.

Definition 3.4. The set $Sfor_0(A)$ of *direct subformulas* of $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ is defined as follows.

$$\begin{array}{lll} Sfor_0(A) & := \emptyset & \text{if } A \text{ is an atomic proposition} \\ Sfor_0(A \odot B) & := \{A, B\} & \text{where } \odot \in \{\wedge, \vee\} \\ Sfor_0(\heartsuit A) & := \{A\} & \text{where } \heartsuit \in \{\Box_d, \Diamond_d \mid d \in D\} \\ Sfor_0(\eta x.A) & := \{A\} & \text{where } \eta \in \{\mu, \nu\}. \end{array}$$

We use the notation $A \lhd_0 B$ if $A \in Sfor_0(B)$. The set $Sfor(B)$ of *subformulas* of B is the smallest set, which contains B and is closed under taking direct subformulas. We write $A \trianglelefteq B$ if $A \in Sfor(B)$.

When studying fixpoints, it is useful to differentiate between *free* and *bound variables* in a formula. When a variable x in a formula A is within the scope of a fixpoint operator of the form ηx, then this variable x is considered bound, otherwise it is free.

Definition 3.5. Given a formula $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$, the sets of *free* and *bound variables*, denoted $FV(A)$ and $BV(A)$ respectively, are defined inductively as follows.

$$\begin{array}{llll} FV(p) & := \{p\} & BV(p) & := \emptyset \\ FV(\neg p) & := \{p\} & BV(\neg p) & := \emptyset \\ FV(A \vee B) & := FV(A) \cup FV(B) & BV(A \vee B) & := BV(A) \cup BV(B) \\ FV(A \wedge B) & := FV(A) \cup FV(B) & BV(A \wedge B) & := BV(A) \cup BV(B) \\ FV(\Diamond_d A) & := FV(A) & BV(\Diamond_d A) & := BV(A) \\ FV(\Box_d A) & := FV(A) & BV(\Box_d A) & := BV(A) \\ FV(\eta x.A) & := FV(A) \backslash \{x\} & BV(\eta x.A) & := BV(A) \cup \{x\} \end{array}$$

We will need to define the unfolding of a fixpoint formula in order to understand some of the inferences in our proof systems. However, before we are able to do this, we need to define the notion of substitution and, for that, we first need the following definition.

Definition 3.6. Given $A, B \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ and $x \in \Phi_0$, we say that A *is free for* x *in* B if B is positive in x and whenever x occurs in a subformula of B of the form $\eta y.C$, where $y \in FV(A)$, it is in the scope of a fixpoint operator $\lambda x.D$ in B.

In order to give a better idea of how this works, we give an example below.

Example 3.7. Take $B = (\nu z.y \wedge z) \vee \mu x.\neg p \vee y \vee \nu y.q \wedge \Box_d(x \vee y)$ and $A = y \vee z$. Firstly, B is positive in x, y and z. The variable y occurs in the subformula $\nu z.y \wedge z$, a fixpoint formula for $z \in FV(A)$. This occurrence is not in the scope of a fixpoint operator ηy in B, so A is not free for y in B. The variable x occurs only in $\nu y.q \wedge \Box_d(x \vee y)$, a fixpoint formula for $y \in FV(A)$. This occurrence of x is in the scope of μx in B, so A is free for x in B.

When a formula A is free for z in another formula B, every instance of z in B could be replaced by the formula A and the resulting formula would still be a well-formed formula.

Definition 3.8. Let $A, B, C \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ and let z be a propositional variable. Assume A is free for z in B and C, then we inductively define *substitution* $[A/z]$ as follows.

$$
\begin{aligned}
B[A/z] &:= \begin{cases} A & \text{if } B = z \\ B & \text{if } B \text{ is atomic but } B \neq z \end{cases} \\
(\heartsuit B)[A/z] &:= \heartsuit B[A/z], && \text{where } \heartsuit \in \{\Diamond_d, \Box_d \mid d \in D\} \\
(B \odot C)[A/z] &:= B[A/z] \odot C[A/z], && \text{where } \odot \in \{\vee, \wedge\} \\
(\eta x.B)[A/z] &:= \eta x.B[A/z], && \text{where } \eta \in \{\mu, \nu\}
\end{aligned}
$$

Definition 3.9. The *unfolding* of a fixpoint formula $\eta x.A$ is $A[\eta x.A/x]$.

Unfoldings of fixpoint formulas are essential to the way certain inferences in our proof systems are defined. When we give these inference rules further on in this thesis, we will simplify the notation for substitution somewhat. If we have a fixpoint formula $\eta x.A$, the formula A is considered a "function" of x and we write $A(B)$ for some formula B that is free for x in A as shorthand for $A[B/x]$. In particular, we will write $A(\eta x.A)$ to denote the unfolding of $\eta x.A$.

3.3. Kripke semantics for the modal μ-calculus

Now that we have defined the syntax for formulas in the modal μ-calculus, we would now like to assign meaning to them. Just as with modal logic, we can do this through Kripke semantics.

Definition 3.10. Given a set Φ_0 of atomic propositions and a set D of actions, a *Kripke-*$\mathcal{L}^{\mu}_{\mathsf{NF}}(\Phi_0, D)$ *model* $\mathcal{K}_{\Phi_0,D} = \langle S, R, V \rangle$ consists of:

- a set S of states,
- a set $R = \{R_d \subseteq S \times S \mid d \in D\}$ of relations, one for each action d
- a valuation $V : \Phi_0 \to \mathcal{P}(S)$

When the language $\mathcal{L}^{\mu}_{\mathsf{NF}}(\Phi_0, D)$ is fixed, but arbitrary, we write $\mathcal{K}$ and we refer to these models simply as Kripke models. We denote by $\mathtt{K}^{\Phi_0,D}$ the class of all Kripke models for $\mathcal{L}^{\mu}_{\mathsf{NF}}(\Phi_0, D)$.

Using these models, we can give a truth-definition for the modal μ-calculus. However, before we can do this, we will first define an operator that can be applied to an existing valuation.

Definition 3.11. Given a Kripke model $\mathcal{K} = \langle S, R, V \rangle$ and $X \subseteq S$, we define $V[x \mapsto X]$ as

$$
V[x \mapsto X](y) := \begin{cases} V(y) & \text{if } y \neq x \\ X & \text{if } y = x \end{cases}
$$

The Kripke model $\mathcal{K}[x \mapsto X]$ is given by $\langle S, R, V[x \mapsto X] \rangle$.

With this operator defined, we can now give the truth-definition for modal μ-calculus formulas. When giving the semantics for modal logic, we started with the definition of the operator $\models$ and used this to define the truth-set of a formula. Here we will give those definitions the other way around; we first give an inductive definition of the truth-set of a modal μ-calculus formula A in the context of a Kripke model $\mathcal{K}$ and then go on to define the satisfaction relation $\models$.

Definition 3.12. Given $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ and a Kripke model $\mathcal{K} = \langle S, R, V \rangle$, the *truth-set* $[\![A]\!]_{\mathcal{K}}$ of A in $\mathcal{K}$

is defined inductively as follows.

$$\begin{aligned} [\![p]\!]_{\mathcal{K}} &= V(p) & [\![\Diamond_d A]\!]_{\mathcal{K}} &= \{s \in S \mid R_d(s) \cap [\![A]\!]_{\mathcal{K}} \neq \emptyset\} \\ [\![\neg p]\!]_{\mathcal{K}} &= S \backslash V(p) & [\![\Box_d A]\!]_{\mathcal{K}} &= \{s \in S \mid R_d(s) \subseteq [\![A]\!]_{\mathcal{K}}\} \\ [\![A \wedge B]\!]_{\mathcal{K}} &= [\![A]\!]_{\mathcal{K}} \cap [\![B]\!]_{\mathcal{K}} & [\![\mu x.A]\!]_{\mathcal{K}} &= \mathsf{lfp}.A_x^{\mathcal{K}} \\ [\![A \vee B]\!]_{\mathcal{K}} &= [\![A]\!]_{\mathcal{K}} \cup [\![B]\!]_{\mathcal{K}} & [\![\nu x.A]\!]_{\mathcal{K}} &= \mathsf{gfp}.A_x^{\mathcal{K}} \end{aligned}$$

For $x \in \Phi_0$, the map $A_x^{\mathcal{K}} : \mathcal{P}(S) \to \mathcal{P}(S)$ is given by $A_x^{\mathcal{K}}(X) = [\![A]\!]_{\mathcal{K}[x \mapsto X]}$

For the truth-sets of fixpoints formulas given by this definition to be well-defined, we need to make sure that $A_x^{\mathcal{K}}$ is monotone. If it is, it follows from Theorem 3.2 that the fixpoints $\mathsf{lfp}.A_x^{\mathcal{K}}$ and $\mathsf{gfp}.A_x^{\mathcal{K}}$ exist. We prove its monotonicity below, with the added condition that A is positive in x. We can add this condition, because the way that $A_x^{\mathcal{K}}$ is used in the definition, means that x is always a fixpoint variable of a formula of the form $\eta x.A$ and, since we are working in the language of normal form formulas, A must be positive in x. In order to give the proof for the monotonicity of $A_x^{\mathcal{K}}$, we first show the following lemma.

Lemma 3.13. Given complete lattices $\mathbb{C} = (C, \leq_C)$ and $\mathbb{D} = (D, \leq_D)$ and a monotone map $f : C \times D \to C$, the map $g : D \to C$ given by

$$g(d) := \mathsf{lfp}\ x.f(x,d)$$

is monotone.

Proof. Say we have $d_1, d_2 \in D$ s.t. $d_1 \leq_D d_2$. We define the map f^n by setting $f^1(c,d) = f(c,d)$ and

$$f^n(c,d) = f(f^{n-1}(c,d),d)$$

for all $n \geq 2$. The map f is monotone and the composition of monotone maps is monotone, so we know by induction that $f^n : C \times D \to C$ is monotone for all $n \geq 1$. We denote by $\bot_C$ the bottom element of $\mathbb{C}$. Since $d_1 \leq_D d_2$, we have $(\bot_C, d_1) \leq_{C \times D} (\bot_C, d_2)$, where $\leq_{C \times D}$ is the usual product order on $C \times D$. Since f^n is monotone, this means that

$$f^n(\bot_C, d_1) \leq_C f^n(\bot_C, d_2) \qquad \forall n \geq 1.$$

So $f^n(\bot_C, d_1)$ is a lower bound for $f^n(\bot_C, d_2)$ for all $n \geq 1$, which means it is also a lower bound for the supremum of $f^n(\bot_C, d_2)$ over $n \geq 1$, so, by Theorem 3.2, we have

$$f^n(\bot_C, d_1) \leq_C \sup_{n \geq 1}\{f^n(\bot_C, d_2)\} = \mu x.f(x, d_2) = g(d_2) \qquad \forall n \geq 1.$$

Now we see that $g(d_2)$ is an upper bound for $f^n(\bot_C, d_1)$ for all $n \geq 1$. This means that it is also an upper bound for the supremum of $f^n(\bot_C, d_1)$ over $n \geq 1$, so

$$g(d_1) = \mu x.f(x, d_1) = \sup_{n \geq 1}\{f^n(\bot_C, d_1)\} \leq_C g(d_2).$$

We have shown that $d_1 \leq_D d_2$ implies $g(d_1) \leq_C g(d_2)$, so g is monotone. □

We can give a similar proof for a variant of this lemma, where we define $g(d) := \mathsf{gfp}\ x.f(x,d)$. We will now prove the monotonicity of $A_x^{\mathcal{K}}$ by structural induction, where we use Lemma 3.13 in the inductive step.

Theorem 3.14. For all $\mathcal{K} = \langle S, R, V \rangle$, all $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ and all $x \in \Phi_0$, where A is positive in x, the map $A^{\mathcal{K}}_x$ is monotone.

Proof. We will prove this theorem by structural induction on $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$. For the base case, take some $p \in \Phi_0$ and $\mathcal{K}$ and assume that $X \subseteq Y$. If $p \neq x$, we have

$$p^{\mathcal{K}}_x(X) = [\![p]\!]_{\mathcal{K}[x \mapsto X]} = V[x \mapsto X](p) = V[x \mapsto Y](p) = [\![p]\!]_{\mathcal{K}[x \mapsto Y]} = p^{\mathcal{K}}_x(Y),$$

$$(\neg p)^{\mathcal{K}}_x(X) = [\![\neg p]\!]_{\mathcal{K}[x \mapsto X]} = S \backslash V[x \mapsto X](p) = S \backslash V(p) = S \backslash V[x \mapsto Y](p) = [\![\neg p]\!]_{\mathcal{K}[x \mapsto Y]} = (\neg p)^{\mathcal{K}}_x(Y).$$

In the case where $p = x$, we have

$$x^{\mathcal{K}}_x(X) = [\![x]\!]_{\mathcal{K}[x \mapsto X]} = V[x \mapsto X](x) = X \subseteq Y = V[x \mapsto Y](x) = [\![x]\!]_{\mathcal{K}[x \mapsto Y]} = x^{\mathcal{K}}_x(Y).$$

We know that A is positive in x, so we do not need to consider the case for $\neg x$. So, we have shown that $p^{\mathcal{K}}_x$ and $(\neg p)^{\mathcal{K}}_x$ are monotone for all $p, x \in \Phi_0$ and models $\mathcal{K}$ such that A is positive in x.

For the induction hypothesis, assume that we have two formulas $B, C \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ such that $B^{\mathcal{K}}_x$ and $C^{\mathcal{K}}_x$ are monotone for all $x \in \Phi_0$ and all models $\mathcal{K}$ such that A is positive in x.

For the inductive step we will prove monotonicity for each formula that can be constructed by applying one operator to B or B and C. Assume that $X \subseteq Y$ and take $x \in \Phi_0$ and $\mathcal{K}$ arbitrary.

For $B \wedge C$, we have

$$\begin{aligned}(B \wedge C)^{\mathcal{K}}_x(X) = [\![B \wedge C]\!]_{\mathcal{K}[x \mapsto X]} = [\![B]\!]_{\mathcal{K}[x \mapsto X]} \cap [\![C]\!]_{\mathcal{K}[x \mapsto X]} &= B^{\mathcal{K}}_x(X) \cap C^{\mathcal{K}}_x(X) \\ &\overset{\text{(IH)}}{\subseteq} B^{\mathcal{K}}_x(Y) \cap C^{\mathcal{K}}_x(Y) \\ &= [\![B]\!]_{\mathcal{K}[x \mapsto Y]} \cap [\![C]\!]_{\mathcal{K}[x \mapsto Y]} \\ &= [\![B \wedge C]\!]_{\mathcal{K}[x \mapsto Y]} \\ &= (B \wedge C)^{\mathcal{K}}_x(Y).\end{aligned}$$

So $(B \wedge C)^{\mathcal{K}}_x$ is monotone. The case for $B \vee C$ is similar.

Next, take $\Diamond_d B$ for some $d \in D$. We have

$$(\Diamond_d B)^{\mathcal{K}}_x(X) = [\![\Diamond_d B]\!]_{\mathcal{K}[x \mapsto X]} = \{s \in S \mid R_d(s) \cap [\![B]\!]_{\mathcal{K}[x \mapsto X]} \neq \emptyset\} \quad \text{and}$$

$$(\Diamond_d B)^{\mathcal{K}}_x(Y) = [\![\Diamond_d B]\!]_{\mathcal{K}[x \mapsto Y]} = \{s \in S \mid R_d(s) \cap [\![B]\!]_{\mathcal{K}[x \mapsto Y]} \neq \emptyset\}.$$

From the induction hypothesis we have $[\![B]\!]_{\mathcal{K}[x \mapsto X]} \subseteq [\![B]\!]_{\mathcal{K}[x \mapsto Y]}$, which implies that, if

$$R_d(s) \cap [\![B]\!]_{\mathcal{K}[x \mapsto X]} \neq \emptyset \quad \Rightarrow \quad R_d(s) \cap [\![B]\!]_{\mathcal{K}[x \mapsto Y]} \neq \emptyset.$$

This means that $(\Diamond_d B)^{\mathcal{K}}_x(X) \subseteq (\Diamond_d B)^{\mathcal{K}}_x(Y)$, so $(\Diamond_d B)^{\mathcal{K}}_x$ is monotone.

For $\Box_d B$, the induction hypothesis implies that, if $R_d(s) \subseteq [\![B]\!]_{\mathcal{K}[x \mapsto X]}$, then $R_d(s) \subseteq [\![B]\!]_{\mathcal{K}[x \mapsto Y]}$. This gives us

$$\begin{aligned}(\Box_d B)^{\mathcal{K}}_x(X) = [\![\Box_d B]\!]_{\mathcal{K}[x \mapsto X]} = \{s \in S \mid R_d(s) \subseteq [\![B]\!]_{\mathcal{K}[x \mapsto X]}\} &\subseteq \{s \in S \mid R_d(s) \subseteq [\![B]\!]_{\mathcal{K}[x \mapsto Y]}\} \\ &= [\![\Box_d B]\!]_{\mathcal{K}[x \mapsto Y]} = (\Box_d B)^{\mathcal{K}}_x(Y),\end{aligned}$$

which shows that $(\Box_d B)^{\mathcal{K}}_x$ is monotone.

Now take $\mu y.B$ for some $y \in \Phi_0$, then we have

$$(\mu y.B)^{\mathcal{K}}_x(X) = \text{lfp}.B^{\mathcal{K}[x\mapsto X]}_y.$$

By the induction hypothesis, we know that $B^{\mathcal{K}[x\mapsto X]}_y : \mathcal{P}(S) \to \mathcal{P}(S)$ is monotone, so this fixpoint exists. The variable y is fixed. If we also fix $\mathcal{K}$ and x, we can define $f : \mathcal{P}(S) \times \mathcal{P}(S) \to \mathcal{P}(S)$ by setting $f(X,Y) = B^{\mathcal{K}[x\mapsto X]}_y(Y)$. We also have $(\mu y.B)^{\mathcal{K}}_x : \mathcal{P}(S) \to \mathcal{P}(S)$ which is defined by

$$(\mu y.B)^{\mathcal{K}}_x(X) = \text{lfp } Y.B^{\mathcal{K}[x\mapsto X]}_y(Y) = \text{lfp } Y.f(X,Y).$$

We know from the induction hypothesis that $B^{\mathcal{K}[x\mapsto X]}_y$ is monotone, so f is monotone. Since $\mathcal{P}(S)$ is a complete lattice, we can apply Lemma 3.13, which gives that $(\mu y.B)^{\mathcal{K}}_x$ is monotone for all $x \in \Phi_0$ and all models $\mathcal{K}$.

The case for $\nu y.B$ is similar, using a variant of Lemma 3.13, where $g(d) := \text{gfp } x.f(x,d)$. So, by induction, $A^{\mathcal{K}}_x$ is monotone for all $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$, $x \in \Phi_0$ and $\mathcal{K} = \langle S, R, V\rangle$. □

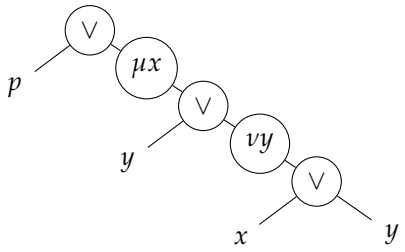
With the truth-set defined, we define the satisfaction relation $\models$ for these Kripke semantics by saying that $\mathcal{K}, s \models A$ iff $s \in [\![A]\!]_{\mathcal{K}}$. Validity is defined similarly to Definition 2.5, where we write $\models_{\mathsf{K}^{\Phi_0,D}} \varphi$ when a modal μ-calculus formula φ is *valid over Kripke-*$\mathcal{L}^{\mu}_{\mathsf{NF}}(\Phi_0, D)$ *models*.

3.4. Subsumption order

In any modal μ-calculus formula, the variables can be ordered based on the nested layers of fixpoint formulas.

Definition 3.15. Given a formula $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$, let $Var(A) = FV(A) \cup BV(A)$. For $x, y \in Var(A)$ we write $x <^{-}_{A} y$ if there is some $B \in Sfor(A)$ of the form $\eta y.C$ such that $x \in FV(B)$. The *subsumption order* $<_A$ is the transitive closure of $<^{-}_{A}$ and we write $x \leq_A y$ if either $x <_A y$ or $x = y$. We say a formula is *well-named* if $<_A$ is irreflexive.

Example 3.16. Take $A = p \vee \mu x.y \vee \nu y.x \vee y$. We provide the syntax tree for clarity.



We have $\nu y.x \vee y \in Sfor(A)$, a fixpoint formula of y in which x occurs freely, so $x <^{-}_{A} y$. However, we also have $\mu x.y \vee \nu y.x \vee y \in Sfor(A)$, which is a fixpoint formula for x in which y occurs freely, meaning $y <^{-}_{A} x$. When we construct the transitive closure, we get $x <_A x$ and $y <_A y$, so the subsumption order is not irreflexive, so this formula is not well-named.

We can fix this by renaming the variables. Since the occurrence of y in $\nu y.x \vee y \in Sfor(A)$ is not connected to the other occurrence of y, we can rename these instances of y to z. The new formula we get is $A' = p \vee \mu x.y \vee \nu z.x \vee z$. From $\nu z.x \vee z$, we get $x <^{-}_{A'} z$ and from $\mu x.y \vee \nu z.x \vee z$, we have $y <^{-}_{A'} x$, so $y <^{-}_{A'} x <^{-}_{A} z$. This means the subsumption order for A' is irreflexive.

3.5. Names and annotations

In the proof system Clo, which we introduce in the next section, the formulas are annotated with names in order to keep track of the unfoldings of fixpoint formulas. These names refer to variables; each variable $x \in \Phi_0$ has a countable set N_x of *names*, where $N_x \cap N_y = \emptyset$ if $x \neq y$. The set of all names is $N = \cup_{x \in \Phi_0} N_x$. The set N^* contains all finite sequences of names from N, also called *words*, including the *empty word* ε. Given $\mathsf{a}, \mathsf{b} \in N^*$, we say a is a *subword* of b, denoted $\mathsf{a} \sqsubseteq \mathsf{b}$, if $\mathsf{x} \in \mathsf{b}$ for all $\mathsf{x} \in \mathsf{a}$.

In order to set conditions on these names and annotations, it would be useful to extend the subsumption order to the set of names. Since the subsumption order $<_A$ is specific to a formula A, we extend it to the names of variables that occur in A. This allows us to define an order on $N_A = \cup_{x \in Var(A)} N_x$, which is the set of all names of variables in a formula A. Given a formula $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ and $\mathsf{x}, \mathsf{y} \in N_A$, we say $\mathsf{x} <_A \mathsf{y}$ ($\mathsf{x} \leq_A \mathsf{y}$) if $\mathsf{x} \in N_x$ and $\mathsf{y} \in N_y$ such that $x <_A y$ ($x \leq_A y$).

An *annotation* is a word $\mathsf{a} = \mathsf{x}_1 \dots \mathsf{x}_n$ such that for all $1 \leq i < j \leq n$, we have $\mathsf{x}_i \leq_A \mathsf{x}_j$ and $\mathsf{x}_i \neq \mathsf{x}_j$. We can also use the subsumption order to establish a relation between annotations and names. Given $\mathsf{a} \in N_A^*$ and $\mathsf{x} \in N_x$, we write $\mathsf{a} <_A \mathsf{x}$ ($\mathsf{a} \leq_A \mathsf{x}$) if for all $\mathsf{y} \in \mathsf{a}$ we have $\mathsf{y} \in N_y$ for some $y <_A x$ ($y \leq_A x$).

In [4], where the Clo system is introduced, the names inherit a global subsumption order, denoted by $<$, on all the variables in the modal μ-calculus language. Defining this global order involves renaming variables in modal μ-calculus formulas so that the subsumption orders for all modal μ-calculus formulas agree, giving a global order that can be applied to any formula. In this thesis, we will not get into this renaming process, so we will stick to our own "local" subsumption order, but we will come across this global subsumption order when we introduce the Clo system.

3.6. The proof system Clo

The system Clo is a cyclic proof system for the modal μ-calculus, which was introduced and shown to be sound and complete over Kripke models in [4]. It is defined for the language $\mathcal{L}^{\mu}_{\mathsf{NF}}$ of normal form formulas, where we write $[g]$ and $[g^d]$ instead of $\Box_g$ and $\Diamond_g$ for $g \in D$. More details about this notational change are given in subsection 5.2.

$$\frac{}{p^{\varepsilon}, (\neg p)^{\varepsilon}}\ \mathsf{Ax1} \qquad \frac{\Delta, A^{\mathsf{a}}}{[g^d]\Delta, [g]A^{\mathsf{a}}}\ \mathsf{mod}$$

$$\frac{\Delta, A^{\mathsf{a}}, B^{\mathsf{a}}}{\Delta, (A \vee B)^{\mathsf{a}}}\ \vee \qquad \frac{\Delta, A^{\mathsf{a}} \quad \Delta, B^{\mathsf{a}}}{\Delta, (A \wedge B)^{\mathsf{a}}}\ \wedge$$

$$\frac{\Delta}{\Delta, A^{\mathsf{a}}}\ \mathsf{weak} \qquad (\mathsf{a} \leq \mathsf{x}, \sigma \in \{\mu, \nu\})\ \frac{\Delta, A(\sigma x.A)^{\mathsf{a}}}{\Delta, \sigma x.A^{\mathsf{a}}}\ \sigma$$

$$(\forall i \leq k\ \mathsf{a_i} \sqsubseteq \mathsf{b_i})\ \frac{A_1^{\mathsf{a}_1}, \dots, A_k^{\mathsf{a}_k}}{A_1^{\mathsf{b}_1}, \dots, A_k^{\mathsf{b}_k}}\ \mathsf{exp}$$

$$(\mathsf{a} \leq \mathsf{x} \in N_x, \mathsf{x} \notin \Delta)\ \frac{\begin{array}{c}[\Delta, \nu x.A^{\mathsf{ax}}]^{\mathsf{x}} \\ \vdots \\ \Delta, A(\nu x.A)^{\mathsf{ax}}\end{array}}{\Delta, \nu x.A^{\mathsf{a}}}\ \nu\text{-}\mathsf{clo}_x$$

Table 1: *Axiom and rules for Clo*

The proof system Clo is a sequent system and its axiom and inference rules are given in Table 1.

Here Δ is a set of modal μ-calculus formulas, also called a *sequent*, and $[g^d]\Delta = \{[g^d]A \mid A \in \Delta\}$. As mentioned in the previous section, this system works with *annotated modal μ-calculus formulas*, which consist of a formula $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ and an annotation $\mathsf{a} \in N^*$. These annotations allow us to keep track of the unfoldings of fixpoint formulas. For some inference rules, there are conditions on these annotations, using the global subsumption order mentioned in the previous section. The most notable rule in this system is the ν-clo_x inference.

$$(\mathsf{a} \leq x \in N_x, x \notin \Delta)\ \dfrac{\begin{array}{c}[\Delta, \nu x.A^{\mathsf{ax}}]^{\mathsf{x}} \\ \vdots \\ \Delta, A(\nu x.A)^{\mathsf{ax}}\end{array}}{\Delta, \nu x.A^{\mathsf{a}}}\ \nu\text{-}\mathsf{clo}_x$$

In this rule, the sequent between brackets is an assumption that is discharged by the application of the ν-clo_x rule. For this rule to be applied, the variable x can not appear in Δ and there must be a path from the assumption in brackets to the premise $\Delta, A(\nu x.A)^{\mathsf{ax}}$. Due to the function that the annotations serve in this system, a derivation in Clo can only be considered a valid Clo-derivation of its root formula, if this formula has the empty word ε as its annotation.

Definition 3.17. A Clo-*derivation* is a finite tree of sequents consisting of annotated modal μ-calculus formulas, where each sequent is either an axiom, a sequent that is discharged by an instance of the ν-clo_x rule, or follows directly from the sequent above it by an inference rule. We say $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ *has a* Clo-*derivation*, denoted $\vdash_{\mathsf{Clo}} A$, if there is a Clo-derivation with A^{ε} as its root.

$$\frac{}{p^{\varepsilon}, (\neg p)^{\varepsilon}}\ \mathsf{Ax1} \qquad \frac{\Delta, A^{\mathsf{a}}}{[g^d]\Delta, [g]A^{\mathsf{a}}}\ \mathsf{mod}$$

$$\frac{\Delta, A^{\mathsf{a}}, B^{\mathsf{a}}}{\Delta, (A \vee B)^{\mathsf{a}}}\ \vee \qquad \frac{\Delta, A^{\mathsf{a}} \quad \Delta, B^{\mathsf{a}}}{\Delta, (A \wedge B)^{\mathsf{a}}}\ \wedge$$

$$\frac{\Delta}{\Delta, A^{\mathsf{a}}}\ \mathsf{weak} \qquad (\mathsf{a} \leq_C x, \sigma \in \{\mu, \nu\})\ \frac{\Delta, A(\sigma x.A)^{\mathsf{a}}}{\Delta, \sigma x.A^{\mathsf{a}}}\ \sigma$$

$$(\forall i \leq k\ \mathsf{a}_i \sqsubseteq \mathsf{b}_i)\ \frac{A_1^{\mathsf{a}_1}, \ldots, A_k^{\mathsf{a}_k}}{A_1^{\mathsf{b}_1}, \ldots, A_k^{\mathsf{b}_k}}\ \mathsf{exp}$$

$$(\mathsf{a} \leq_C x \in N_x, \mathsf{x} \notin \Delta)\ \dfrac{\begin{array}{c}[\Delta, \nu x.A^{\mathsf{ax}}]^{\mathsf{x}} \\ \vdots \\ \Delta, A(\nu x.A)^{\mathsf{ax}}\end{array}}{\Delta, \nu x.A^{\mathsf{a}}}\ \nu\text{-}\mathsf{clo}_x$$

Table 2: *Axiom and rules for* Clo_C

As we noted in subsection 3.6, this system uses the global order from [4] in order to apply conditions to its inference rules. However, in this thesis, we will only use the subsumption order $\leq_A$ for a specific formula A, since defining a global order would involve renaming variables, which would result in complications with our translation from game logic to the modal μ-calculus in section 5. To make sure we can work with this "local" subsumption order instead of the global one, we define the proof system Clo_C, which is parametric in a formula C and differs from Clo only in the fact that it uses the subsumption order $\leq_C$ instead of $\leq$ for its side conditions. The axiom and inference rules for Clo_C are given in Table 2. The notion of a derivation in this system is defined similarly to how it was defined for Clo.

Definition 3.18. A Clo_C*-derivation* is a finite tree of sequents consisting of annotated modal μ-calculus formulas, where each sequent is either an axiom, a sequent that is discharged by an instance of the ν-clo_x rule, or follows directly from the sequent above it by an inference rule. We say $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}$ *has a* Clo_C*-derivation*, denoted $\vdash_{\mathsf{Clo}_C} A$, if there is a Clo_C-derivation with A^{ε} as its root.

In order to use this system instead of Clo as a starting point for our proof transformation, we must first show that, whenever there is a Clo-derivation of a formula C, then there is also a Clo_C-derivation of C.

Theorem 3.19. If $\vdash_{\mathsf{Clo}} C$, then $\vdash_{\mathsf{Clo}_C} C$.

Proof. Say we have some Clo-derivation of C. Our goal is to show that if a rule can be applied in Clo, then that rule can also be applied in Clo_C. The only difference between Clo and Clo_C is in the side conditions for the rules σ and ν-clo_x, so any other rule that is applied in the Clo-derivation of C could simply be replaced with the same rule from Clo_C. The order $\leq$ used in the side conditions of Clo is a global order and thus contains the subsumption order $\leq_C$, so for $x, y \in \mathit{Var}(C)$, we have

$$\mathsf{x} \leq \mathsf{y} \quad \Rightarrow \quad \mathsf{x} \leq_C \mathsf{y} \qquad \forall \mathsf{x} \in N_x, \mathsf{y} \in N_y.$$

It is easily observed that the annotations of a Clo-derivation of C will only contains names for variables in $\mathit{Var}(C)$. So, say that we have $\mathsf{a} \leq \mathsf{x}$. Take some $\mathsf{y} \in \mathsf{a}$, then there is some $y \in \mathit{Var}(C)$ such that $\mathsf{y} \in N_y$ and $y \leq x$. Since $x, y \in \mathit{Var}(C)$, we have $x \leq_C y$. So, for all $\mathsf{y} \in \mathsf{a}$, there is some $y \in \mathit{Var}(C)$ such that $\mathsf{y} \in N_y$ and $x \leq_C y$, so $\mathsf{a} \leq_C \mathsf{x}$. So, whenever $\mathsf{a} \leq \mathsf{x}$, we also have $\mathsf{a} \leq_C \mathsf{x}$.

This means that whenever a side condition of a rule in Clo is satisfied, the side condition of the corresponding rule in Clo_C is also satisfied. So, we can construct a Clo_C-derivation of C by simply applying the same rules used in the Clo-derivation. □

Note that, though Clo is complete over Kripke models, this result does not show that the system Clo_C is complete over Kripke models. We have ensured that if a formula C can be derived in Clo, then it can also be derived in Clo_C, but this claim can not be extended to all formulas that are valid over Kripke models, since each such formula would generate its own version of Clo_C.

4. Game logic

4.1. Game logic syntax and semantics

Game logic is a modal logic designed to reason about the outcomes that can be achieved by the players, which we refer to as Angel and Demon, in 2-player games. Below we give the definition of the original language as defined by Parikh [1].

Definition 4.1. Given the set $\Phi_0 = \{p, q, \dots\}$ of atomic propositions and the set $\Gamma_0 = \{g, h, \dots\}$ of atomic games, the *game logic language* $\mathcal{L}_{\mathsf{Par}}(\Phi_0, \Gamma_0)$ is defined by the following grammar.

$$\mathcal{G}_{\mathsf{Par}} \ni \gamma ::= g \mid \gamma;\gamma \mid \gamma \sqcup \gamma \mid \gamma^* \mid \gamma^d \mid \varphi?,\ \varphi \in \mathcal{L}_{\mathsf{Par}}$$
$$\mathcal{L}_{\mathsf{Par}} \ni \varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid [\gamma]\varphi,\ \gamma \in \mathcal{G}_{\mathsf{Par}}$$

where $p \in \Phi_0$ and $g \in \Gamma_0$. When Γ_0 and Φ_0 are fixed, but arbitrary, we will simply write $\mathcal{L}_{\mathsf{Par}}$.

The game $\gamma_1 \sqcup \gamma_2$ is interpreted as Angel choosing whether to play γ_1 or γ_2 and is referred to as *choice*. The game γ^* is known as *iteration* and indicates that the game γ is repeatedly played as many times as Angel wants. The game $\varphi?$ is called *test* and it involves checking whether a formula φ holds. If it does, the game continues, but if it does not, the game ends and Angel loses. The *sequential* game $\gamma_1;\gamma_2$ simply denotes that γ_1 is played first, then γ_2. The *dual operator* $(-)^d$ is interpreted as the two players switching roles and it can be used to define demonic variants of the game operators as follows.

$$\gamma_1 \sqcap \gamma_2 := (\gamma_1^d \sqcup \gamma_2^d)^d \qquad \varphi! := ((\neg\varphi)?)^d \qquad \gamma^\times := ((\gamma^d)^*)^d.$$

Where, in the game $\gamma_1 \sqcup \gamma_2$, Angel chooses which game to play, in the game $\gamma_1 \sqcap \gamma_2$ Demon makes this decision. This game is called *demonic choice*. In the *demonic iteration* game $\gamma^\times$, Demon chooses how many times γ will be played. The game $\varphi!$, known as *demonic test*, tests whether φ holds, where Demon loses if it does not. In the game $\gamma_1;\gamma_2$, no choice is made by the players, so there is no need for a demonic variant.

The formula $[\gamma]\varphi$ denotes that Angel *has an effective strategy in game γ to achieve the outcome φ*. It is important to note here that our notation deviates from what was established by Pauly and Parikh in [11], where they use $\langle\gamma\rangle\varphi$ to denote this property. This change is made to make the interpretation of these modalities align with the interpretation of the modalities of the modal μ-calculus, which will make translating between them further on in this thesis more intuitive.

When playing a game we would expect that if Angel can ensure φ, then Demon should not be able to ensure $\neg\varphi$. If this were not the case, then the players could achieve contradicting outcomes at the end of the game. This property of games is called *consistency* and it can be written as

$$[\gamma]\varphi \quad \Rightarrow \quad \neg[\gamma^d]\neg\varphi.$$

The reverse statement is called *determinacy* and states that if an outcome can not be prevented by Demon, then Angel must be able to achieve it. In other words, for every achievable outcome, at least one of the players that has a strategy to achieve it, so

$$\neg[\gamma^d]\neg\varphi \quad \Rightarrow \quad [\gamma]\varphi.$$

The LHS can also be written as $\neg\langle\gamma\rangle\neg\varphi$, where $\langle\gamma\rangle\varphi$ is the demonic version of $[\gamma]\varphi$. In this thesis, we will be working with game logic formulas in normal form, which means the dual and negation operators can only be applied to atoms. This forces us to include the demonic versions of the game operators mentioned above in the grammar.

Definition 4.2. Given the set $\Phi_0 = \{p, q, \dots\}$ of atomic propositions and the set $\Gamma_0 = \{g, h, \dots\}$ of atomic games, the language $\mathcal{L}_{\mathsf{NF}}(\Gamma_0, \Phi_0)$ of *normal form game logic formulas* is defined as follows.

$$\mathcal{G}_{\mathsf{NF}} \ni \gamma ::= g \mid g^d \mid \gamma;\gamma \mid \gamma \sqcup \gamma \mid \gamma \sqcap \gamma \mid \gamma^* \mid \gamma^\times \mid \varphi? \mid \varphi!,\ \varphi \in \mathcal{L}_{\mathsf{NF}}$$
$$\mathcal{L}_{\mathsf{NF}} \ni \varphi := p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid [\gamma]\varphi \mid \langle\gamma\rangle\varphi,\ \gamma \in \mathcal{G}_{\mathsf{NF}}$$

where $p \in \Phi_0$ and $g \in \Gamma_0$. When Γ_0 and Φ_0 are fixed, but arbitrary, we will simply write $\mathcal{L}_{\mathsf{NF}}$.

With the language of game logic defined, we will now look at how to interpret the truth of formulas in this language. For this purpose we will use game models.

Definition 4.3. Given the set of atomic propositions Φ_0 and the set of atomic games Γ_0, a *game-*$\mathcal{L}_{\mathsf{NF}}(\Phi,\Gamma_0)$ *model* $\mathcal{G}_{\Phi_0,\Gamma_0} = \langle S, E, V\rangle$ consists of

- a set of *states* S,
- a set E of monotone *effectivity functions* $E_g : S \to \mathcal{P}(\mathcal{P}(S))$, one for each $g \in \Gamma_0$, and
- a *valuation* $V : \Phi_0 \to \mathcal{P}(S)$.

When the language $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ is fixed, but arbitrary, we write $\mathcal{G}$ and we refer to these models simply as game models. We denote by $\mathfrak{G}_{\Phi_0,\Gamma_0}$ the class of all game models for $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$.

The intuition behind the effectivity functions E_g for atomic games g is that it assign to a state s all the sets of states that Angel can "choose" when playing the game g at state s. After that, Demon picks at which state out of that set they will be after the game g. The notation $sE_\gamma X$ is used to express that $X \in E_\gamma(s)$. We can also interpret the effectivity functions as maps $E_\gamma : \mathcal{P}(S) \to \mathcal{P}(S)$ by taking $E_\gamma(X) = \{s \in S \mid sE_\gamma X\}$.

Definition 4.4. In a game model $\mathcal{G} = \langle S, E, V\rangle$, the truth-set of $\varphi \in \mathcal{L}_{\mathsf{NF}}$ at a state $s \in S$ and the effectivity functions for non-atomic games are defined by mutual induction as follows.

$$\begin{array}{lcll}
[\![p]\!]_{\mathcal{G}} & := & V(p) \text{ for } p \in \Phi_0 & E_{\alpha;\beta}(Y) := E_\alpha(E_\beta(Y)) \\
[\![\neg p]\!]_{\mathcal{G}} & := & S\backslash V(p) \text{ for } p \in \Phi_0 & E_{\alpha^d}(Y) := (E_\alpha(Y^C))^C \\
[\![\varphi \vee \psi]\!]_{\mathcal{G}} & := & [\![\varphi]\!]_{\mathcal{G}} \cup [\![\psi]\!]_{\mathcal{G}} & E_{\alpha\sqcup\beta}(Y) := E_\alpha(Y) \cup E_\beta(Y) \\
[\![\varphi \wedge \psi]\!]_{\mathcal{G}} & := & [\![\varphi]\!]_{\mathcal{G}} \cap [\![\psi]\!]_{\mathcal{G}} & E_{\alpha\sqcap\beta}(Y) := E_\alpha(Y) \cap E_\beta(Y) \\
[\![[\gamma]\varphi]\!]_{\mathcal{G}} & := & E_\gamma([\![\varphi]\!]_{\mathcal{G}}) & E_{\alpha^*}(Y) := \mathsf{lfp}\ X.Y \cup E_\alpha(X) \\
[\![\langle\gamma\rangle\varphi]\!]_{\mathcal{G}} & := & S\backslash E_\gamma([\![\varphi]\!]^C_{\mathcal{G}}) & E_{\alpha^\times}(Y) := \mathsf{gfp}\ X.Y \cap E_\alpha(X) \\
 & & & E_{\varphi?}(Y) := [\![\varphi]\!]_{\mathcal{G}} \cap Y \\
 & & & E_{\varphi!}(Y) := [\![\varphi]\!]_{\mathcal{G}} \cup Y
\end{array}$$

For the effectivity functions for γ^* and $\gamma^\times$ to be well-defined, we need to show that these fixpoints exists. As $(\mathcal{P}(S), \subseteq)$ is a complete lattice and the effectivity functions are functions on $\mathcal{P}(S)$, if we can show they are monotone, we can apply Tarski's fixed point theorem to show that the fixpoints lfp $X.Y \cup E_\alpha(X)$ and gfp $X.Y \cap E_\alpha(X)$ do indeed exist.

Theorem 4.5. The effectivity functions E_γ are monotone w.r.t. $\subseteq$ for all games γ.

Proof. We need to prove the theorem for all games γ, so we will use structural induction on γ. For the base case, if γ is an atomic game, then E_γ is monotone by the definition of a game model.

Next, for the induction hypothesis, we assume that α, β are games s.t. E_α, E_β are monotone and φ is an arbitrary formula.

For the inductive step, assume that we have $Y, Z \in \mathcal{P}(S)$ such that $Y \subseteq Z$, then we will prove monotonicity for the effectivity functions of every game that can be constructed by applying one operator to α or to α and β.

For $\varphi?$, since $Y \subseteq Z$, we have $E_{\varphi?}(Y) = [\![\varphi]\!]_{\mathcal{G}} \cap Y \subseteq [\![\varphi]\!]_{\mathcal{G}} \cap Z = E_{\varphi?}(Z)$, so $E_{\varphi?}$ is monotone.

For $\alpha;\beta$, since $Y \subseteq Z$, by the monotonicity of E_α and E_β, we have

$$E_\beta(Y) \subseteq E_\beta(Z) \quad \Rightarrow \quad E_\alpha(E_\beta(Y)) \subseteq E_\alpha(E_\beta(Z)) \quad \Rightarrow \quad E_{\alpha;\beta}(Y) \subseteq E_{\alpha;\beta}(Z),$$

so $E_{\alpha;\beta}$ is monotone.

For $\alpha \sqcup \beta$, we have by the monotonicity of E_α and E_β that

$$E_\alpha(Y) \subseteq E_\alpha(Z) \subseteq E_\alpha(Z) \cup E_\beta(Z) \quad \text{and} \quad E_\beta(Y) \subseteq E_\beta(Z) \subseteq E_\alpha(Z) \cup E_\beta(Z).$$

This gives us $E_{\alpha\sqcup\beta}(Y) = E_\alpha(Y) \cup E_\beta(Y) \subseteq E_\alpha(Z) \cup E_\beta(Z) = E_{\alpha\sqcup\beta}(Z)$, so $E_{\alpha\sqcup\beta}$ is monotone.

For α^d, since $Y \subseteq Z$, we have $Z^C \subseteq Y^C$. This means that, by the monotonicity of E_α, we have $E_\alpha(Z^C) \subseteq E_\alpha(Y^C)$, so

$$E_{\alpha^d}(Y) = (E_\alpha(Y^C))^C \subseteq (E_\alpha(Z^C))^C = E_{\alpha^d}(Z),$$

so E_{α^d} is monotone

For the final game, α^*, we know from the induction hypothesis that E_α is monotone. This means that for $F_Y(X) = Y \cup E_\alpha(X)$, we have

$$V \subseteq W \quad \Rightarrow \quad F_Y(V) = Y \cup E_\alpha(V) \subseteq Y \cup E_\alpha(W) = F_Y(W).$$

So $F_Y(X)$ is monotone for any set $Y \in \mathcal{P}(S)$, so by Kleene's theorem it has a least fixed point

$$LFP\ X.F_Y(X) = \sup_{n\in\mathbb{N}} F_Y^n(\emptyset).$$

We will now use induction on n to show that $F_Y^n(\emptyset) \subseteq F_Z^n(\emptyset)$ for all $n \geq 0$. For $n = 0$ we have $F_Y^n(\emptyset) = \emptyset = F_Z^n(\emptyset)$. Now, for the induction hypothesis, assume that $F_Y^k(\emptyset) \subseteq F_Z^k(\emptyset)$ for some $k \geq 0$. As E_α is monotone, we have $E_\alpha(F_Y^k(\emptyset)) \subseteq E_\alpha(F_Z^k(\emptyset))$, so

$$F_Y^{k+1}(\emptyset) = Y \cup E_\alpha(F_Y^k(\emptyset)) \subseteq Y \cup E_\alpha(F_Z^k(\emptyset)) \subseteq Z \cup E_\alpha(F_Z^k(\emptyset)) = F_Z^{k+1}(\emptyset)$$

So, by induction, we have $F_Y^n(\emptyset) \subseteq F_Z^n(\emptyset)$ for all $n \geq 0$. It follows that

$$E_{\alpha^*}(Y) = \mu X.F_Y(X) = \sup_{n\in\mathbb{N}} F_Y^n(\emptyset) \subseteq \sup_{n\in\mathbb{N}} F_Z^n(\emptyset) = \mu X.F_Z(X) = E_{\alpha^*}(Z),$$

so E_{α^*} is monotone. □

Now that we know that the truth-definition is well-defined, we can also define validity for game logic semantics. It is similar to the validity definitions for the semantics for modal logic.

Definition 4.6. A formula $\varphi \in \mathcal{L}_{\mathsf{Par}}$ is considered

- *valid on a game model* $\mathcal{G}$, denoted $\mathcal{G} \models \varphi$, if $\mathcal{G}, s \models \varphi$ for all $s \in S$.
- *valid over game models*, denoted $\models_{\mathsf{G}} \varphi$, if $\mathcal{G} \models \varphi$ for all game models $\mathcal{G}$.

4.2. Order on fixpoint formulas and names

As established in [3], in the context of game logic, formulas of the form $[\gamma^*]\varphi$ and $[\gamma^\times]\varphi$ take on the role of fixpoint formulas.

Definition 4.7. The sets of least and greatest fixpoint formulas in $\mathcal{L}_{\mathsf{NF}}$ are

$$F^* := \{[\gamma^*]\varphi \mid \gamma \in \mathcal{G}_{\mathsf{NF}}, \varphi \in \mathcal{L}_{\mathsf{NF}}\} \qquad \text{and} \qquad F^\times := \{[\gamma^\times]\varphi \mid \gamma \in \mathcal{G}_{\mathsf{NF}}, \varphi \in \mathcal{L}_{\mathsf{NF}}\}.$$

We also define the set of all fixpoint formulas as $F := F^* \cup F^\times$.

One of the systems used in [3] is CloG, a sequent system for game logic. Similarly to Clo, each sequent in this system is annotated with a sequence of names, which refer to greatest fixpoint formulas. In [3], an order $\preccurlyeq$ on these fixpoint formulas was given and used to define side conditions on the inference rules of CloG. We review this order in this section, since it is essential to keeping track of unfoldings of fixpoint formulas in the proof system CloG_K, which we define in the next section. However, before we can look at the order $\preccurlyeq$ on F, we need to define an order on games and formulas. A *term* is a game or a formula and we will now define the *subterm relation*.

Definition 4.8. The set $Ster(\sigma)$ of *direct subterms* of a term $\sigma \in \mathcal{L}_{\mathsf{NF}} \cup \mathcal{G}_{\mathsf{NF}}$ is defined as follows.

$Ster(g)$	$:= \emptyset$	if $g \in \Gamma_0$	$Ster(p)$	$:= \emptyset$	if $p \in \Phi_0$
$Ster(g^d)$	$:= \{g\}$	if $g \in \Gamma_0$	$Ster(\neg p)$	$:= \{p\}$	if $p \in \Phi_0$
$Ster(\gamma \heartsuit \delta)$	$:= \{\gamma, \delta\}$	where $\heartsuit \in \{\sqcup, \sqcap, ;\}$	$Ster(\varphi \odot \psi)$	$:= \{\varphi, \psi\}$	where $\odot \in \{\wedge, \vee\}$
$Ster(\gamma^\circ)$	$:= \{\gamma\}$	where $\circ \in \{*, \times\}$	$Ster([\gamma]\varphi)$	$:= \{\gamma, \varphi\}$	
$Ster(\varphi \wr)$	$:= \{\varphi\}$	where $\wr \in \{!, ?\}$			

We write $\sigma \lhd_0 \tau$ if $\sigma \in Ster(\tau)$. We denote by $\lhd$ the transitive closure of $\lhd_0$ and by $\trianglelefteq$ the reflexive, transitive closure of $\lhd_0$. We say σ is a *subterm* of τ if $\sigma \trianglelefteq \tau$. For two formulas $\varphi, \psi \in \mathcal{L}_{\mathsf{NF}}$, we say φ is a *subformula* of ψ if $\varphi \trianglelefteq \psi$.

We also define the notion of the closure of a game logic formula. This definition will be useful later, when we give a translation from game logic to the modal μ-calculus.

Definition 4.9. The *closure* $Cl(\varphi)$ of a formula $\varphi \in \mathcal{L}_{\mathsf{NF}}$ is the smallest subset of $\mathcal{L}_{\mathsf{NF}}$ that contains φ, is closed under subformulas and satisfies the following rules:

1. If $[\gamma;\delta]\psi \in Cl(\varphi)$, then $[\gamma][\delta]\psi \in Cl(\varphi)$.
2. If $[\gamma \sqcup \delta]\psi \in Cl(\varphi)$ or $[\gamma \sqcap \delta]\psi \in Cl(\varphi)$, then $[\gamma]\psi, [\delta]\psi \in Cl(\varphi)$.
3. If $[\gamma^*]\psi \in Cl(\varphi)$, then $\psi \vee [\gamma][\gamma^*]\psi \in Cl(\varphi)$.
4. If $[\gamma^\times]\psi \in Cl(\varphi)$, then $\psi \wedge [\gamma][\gamma^\times]\psi \in Cl(\varphi)$.
5. If $[\psi?]\chi \in Cl(\varphi)$ or $[\psi!]\chi \in Cl(\varphi)$, then $\psi \in Cl(\varphi)$.

We define set $F(\varphi) := F \cap Cl(\varphi)$.

With the subterm relation defined, we can now give the order on the set of fixpoint formulas.

Definition 4.10. The order $\prec$ on F is defined as follows for all $\varphi, \psi \in \mathcal{L}_{\mathsf{NF}}$ and $\gamma, \delta \in \mathcal{G}_{\mathsf{NF}}$

$$\begin{array}{lll} [\gamma^\circ]\varphi \prec [\delta^\dagger]\psi & \text{if} & \delta^\dagger \lhd \gamma^\circ, \\ [\gamma^\circ]\varphi \preccurlyeq [\delta^\dagger]\psi & \text{if} & \delta^\dagger \trianglelefteq \gamma^\circ, \end{array} \qquad \text{where } \circ, \dagger \in \{*, \times\}.$$

Example 4.11.

- We have $[g^* \sqcap h^\times]p \prec [h^\times]q$, since $h^\times \lhd g^* \sqcup h^\times$.
- We have $[([b^\times]p)?]q \prec [b^\times]p$, since $b^\times \lhd ([b^\times]p)?$.
- We have $[(g \sqcup h^d)^\times]q \preccurlyeq [(g \sqcup h^d)^\times](p \vee q)$, since $(g \sqcup h^d)^\times \trianglelefteq (g \sqcup h^d)^\times$.

In the next section, we define the proof system CloG_K. Just like in Clo, formulas in this system are annotated with names. However, where the names in Clo referred to variables, in CloG_K they refer to greatest fixpoint formulas. Concepts like *words* and *annotations* that were introduced in subsection 3.4 are defined similarly for these names and the order $\prec$ extends to names and annotations, just like the subsumption order $\leq_A$ does.

4.3. The proof system CloG_K

The proof system CloG is a cyclic sequent system for game logic. It was defined in [3] as the first game logic system in the transformation sequence from Clo to Par. This means it has many rules which are the same as rules in Clo, but it also includes an inference rule for every game operator, both angelic and demonic, and a game logic variant of the ν-clo_x rule.

We will adapt this proof system into a new cyclic proof system, called CloG_K. This new system is also a sequent system for the language $\mathcal{L}_{\mathsf{NF}}$ of normal form game logic formulas, but it uses the more general normal modal rule mod instead of the monotone modal rule mod_m.

$$\frac{\varphi^\mathsf{a}, \psi^\mathsf{b}}{([g^d]\varphi)^\mathsf{a}, ([g]\psi)^\mathsf{b}}\ \mathsf{mod}_m \qquad \frac{\Delta, \varphi^\mathsf{a}}{[g^d]\Delta, ([g]\varphi)^\mathsf{a}}\ \mathsf{mod}$$

As we established in subsection 2.3, this change ensures that CloG_K is designed to be valid over Kripke models. The axioms and inference rules of CloG_K are listed in Table 3. Here, Δ is a game logic sequent, g is an atomic game and $[g^d]\Delta = \{[g^d]\varphi \mid \varphi \in \Delta\}$.

Definition 4.12. A CloG_K*-derivation* is a finite tree of sequents consisting of annotated formulas from $\mathcal{L}_{\mathsf{NF}}$, where each sequent is either an axiom, a sequent that is discharged by an instance of the clo_x rule, or follows directly from the sequent above it by an inference rule. We say $\varphi \in \mathcal{L}_{\mathsf{NF}}$ *has a* CloG_K*-derivation*, denoted $\vdash_{\mathsf{CloG}_\mathsf{K}} \varphi$, if there is a CloG_K-derivation with φ^ε as its root.

$$\frac{}{p^\varepsilon, (\neg p)^\varepsilon}\ \mathsf{Ax1} \qquad \frac{\Delta, \varphi^\mathsf{a}}{[g^d]\Delta, ([g]\varphi)^\mathsf{a}}\ \mathsf{mod}$$

$$\frac{\Delta, \varphi^\mathsf{a}, \psi^\mathsf{a}}{\Delta, (\varphi \vee \psi)^\mathsf{a}}\ \vee \qquad \frac{\Delta, \varphi^\mathsf{a} \quad \Delta, \psi^\mathsf{a}}{\Delta, (\varphi \wedge \psi)^\mathsf{a}}\ \wedge$$

$$\frac{\Delta, ([\gamma]\varphi \vee [\delta]\varphi)^\mathsf{a}}{\Delta, ([\gamma \sqcup \delta]\varphi)^\mathsf{a}}\ \sqcup \qquad \frac{\Delta, ([\gamma]\varphi \wedge [\delta]\varphi)^\mathsf{a}}{\Delta, ([\gamma \sqcap \delta]\varphi)^\mathsf{a}}\ \sqcap$$

$$\frac{\Delta}{\Delta, \varphi^\mathsf{a}}\ \mathsf{weak} \qquad \frac{\Delta, \varphi^{\mathsf{ab}}}{\Delta, \varphi^{\mathsf{axb}}}\ \mathsf{exp} \qquad \frac{\Delta, ([\gamma][\delta]\varphi)^\mathsf{a}}{\Delta, ([\gamma;\delta]\varphi)^\mathsf{a}}\ ;$$

$$(\mathsf{a} \preccurlyeq [\gamma^*]\varphi)\ \frac{\Delta, (\varphi \vee [\gamma][\gamma^*]\varphi)^\mathsf{a}}{\Delta, ([\gamma^*]\varphi)^\mathsf{a}}\ * \qquad \frac{\Delta, (\varphi \wedge \psi)^\mathsf{a}}{\Delta, ([\varphi?]\psi)^\mathsf{a}}\ ?$$

$$(\mathsf{a} \preccurlyeq [\gamma^\times]\varphi)\ \frac{\Delta, (\varphi \wedge [\gamma][\gamma^\times]\varphi)^\mathsf{a}}{\Delta, ([\gamma^\times]\varphi)^\mathsf{a}}\ \times \qquad \frac{\Delta, (\varphi \vee \psi)^\mathsf{a}}{\Delta, ([\varphi!]\psi)^\mathsf{a}}\ !$$

$$\frac{[\Delta, ([\gamma^\times]\varphi)^{\mathsf{ax}}]^\mathsf{x}}{\vdots}$$

$$(\mathsf{a} \preccurlyeq \mathsf{x} \in N_{[\gamma^\times]\varphi}, \mathsf{x} \notin \Delta, \mathsf{a})\ \frac{\Delta, (\varphi \wedge [\gamma][\gamma^\times]\varphi)^{\mathsf{ax}}}{\Delta, ([\gamma^\times]\varphi)^\mathsf{a}}\ \mathsf{clo}_\mathsf{x}$$

Table 3: *Axiom and rules for* CloG_K

5. From game logic to the modal μ-calculus

5.1. Augmented game models

The effectivity functions used in game models are monotone neighbourhood functions, so a game model for $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ is, in fact, a monotone neighbourhood model for $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$, where we have one neighbourhood function for every atomic game in $g \in \Gamma_0$.

In subsection 2.2, we defined the class of augmented neighbourhood models and proved that it is modally equivalent to the class of Kripke models. Within the class $\mathtt{G}$ of game models, we can define a similar subclass, namely the class $\mathtt{G}_{\mathtt{K}}$ of *augmented game models*, which are game models whose effectivity functions are augmented. Since game models are monotone neighbourhood models, we can use the correspondence relation between relations and augmented neighbourhood function defined in Definition 2.14 to define a unique corresponding Kripke model for each augmented game model and vice versa. From this, it is easy to see that the class $\mathtt{G}_{\mathtt{K}}^{\Phi_0,\Gamma_0}$ of augmented game models for $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and the class $\mathtt{K}^{\Phi_0,\Gamma_0}$ of Kripke models are modally equivalent.

When we talk about interpreting game logic "over Kripke models", we mean the language of game logic interpreted through the semantics of these augmented game models. In this context, all the atomic games are interpreted to be 1-player games, in which all decisions are made by Angel. The only way Demon interacts with games in this context is through the dual operator.

5.2. Translating game logic into the modal μ-calculus

In order to show the completeness of $\mathsf{CloG_K}$, we will transform Clo-derivations into $\mathsf{CloG_K}$-derivations. We fix the set Φ_0 of atomic propositions and the set Γ_0 of atomic games and consider $\mathsf{CloG_K}$ for $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and Clo for $\mathcal{L}^{\mu}_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$, where $\widehat{\Phi}_0 = \Phi_0 \cup \{x^{\varphi} \mid \varphi \in F\}$ for the set F of all greatest and least fixpoint formulas in $\mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$. Recall that, in the system Clo, we write $[g]$ instead of $\Box_g$ and $[g^d]$ instead of $\Diamond_g$ for an action g in order to stay closer to the syntax of game logic. Since these systems work over different languages, we will need a way of translating a game logic formula into a modal μ-calculus formula, while preserving validity.

In [3], a validity-preserving translation of game logic into the monotone μ-calculus was given. Since there is no difference in the syntax of the monotone μ-calculus and the normal modal μ-calculus, we can use their translation as a translation from game logic to the normal μ-calculus. However, since the semantics for the two languages are different, we will also need to provide a proof that the translation is still validity-preserving in this context.

Definition 5.1. The translation $(-)^\sharp : \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0) \to \mathcal{L}^{\mu}_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ is defined by mutual structural induction on formulas and games as follows.

$$
\begin{aligned}
p^\sharp &:= p & \tau^{\varphi}_{\gamma\sqcap\delta}(A) &:= \tau^{\varphi}_{\gamma}(A) \wedge \tau^{\varphi}_{\delta}(A)\\
(\neg p)^\sharp &:= \neg p & \tau^{\varphi}_{\gamma\sqcup\delta}(A) &:= \tau^{\varphi}_{\gamma}(A) \vee \tau^{\varphi}_{\delta}(A)\\
(\varphi \wedge \psi)^\sharp &:= \varphi^\sharp \wedge \psi^\sharp & \tau^{\varphi}_{\gamma^*}(A) &:= \mu x^{[\gamma^*]\varphi}.A \vee \tau^{[\gamma^*]\varphi}_{\gamma}(x^{[\gamma^*]\varphi})\\
(\varphi \vee \psi)^\sharp &:= \varphi^\sharp \vee \psi^\sharp & \tau^{\varphi}_{\gamma^\times}(A) &:= \nu x^{[\gamma^\times]\varphi}.A \wedge \tau^{[\gamma^\times]\varphi}_{\gamma}(x^{[\gamma^\times]\varphi})\\
([\gamma]\varphi)^\sharp &:= \tau^{\varphi}_{\gamma}(\varphi^\sharp) & \tau^{\varphi}_{\gamma;\delta}(A) &:= \tau^{[\delta]\varphi}_{\gamma}(\tau^{\varphi}_{\delta}(A))\\
\tau^{\varphi}_{g}(A) &:= [g]A & \tau^{\varphi}_{\psi?}(A) &:= \psi^\sharp \wedge A\\
\tau^{\varphi}_{g^d}(A) &:= [g^d]A & \tau^{\varphi}_{\psi!}(A) &:= \psi^\sharp \vee A
\end{aligned}
$$

Here the fixpoint variables that are used in the definition of $\tau^{\varphi}_{\gamma^*}$ and $\tau^{\varphi}_{\gamma^\times}$ are the unique variables $x^{[\gamma^*]\varphi}$ and $x^{[\gamma^\times]\varphi}$ associated with $[\gamma^*]\varphi, [\gamma^\times]\varphi \in F$ and are thus not elements of Φ_0. For this translation to be useful, we need it to be validity-preserving. In other words, if a formula $\varphi \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ is valid over augmented game models, then its translation $\varphi^\sharp \in \mathcal{L}^\mu_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ should be valid over Kripke models. To show this, we will define a correspondence relation between augmented game models and Kripke models, when the set of atomic propositions of the game logic language is contained in the set of atomic propositions of the modal μ-calculus. This relation is similar to the correspondence relation we introduced in subsection 2.2.

Definition 5.2. Given two sets X, Y such that $X \subseteq Y$ and an augmented game model $\mathcal{G} = \langle S, E, V\rangle$ for $\mathcal{L}_{\mathsf{NF}}(X, \Gamma_0)$, we say a Kripke-$\mathcal{L}^\mu_{\mathsf{NF}}(Y, \Gamma_0)$ model $\mathcal{K} = \langle S, R, V'\rangle$ *corresponds to* $\mathcal{G}$ if $R_g(s) := \cap E_g(s)$ for all $g \in \Gamma_0$ and $V'(p) = V(p)$ for all $p \in X$. We denote this by $\mathcal{G} \leftrightarrow \mathcal{K}$.

We will now prove that if $\varphi \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ is true at $s \in S$ in an augmented game model $\mathcal{G}$, then $\varphi^\sharp \in \mathcal{L}^\mu_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ is true at s in any corresponding Kripke model $\mathcal{K}$.

Theorem 5.3. For all $\varphi \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and all augmented game models $\mathcal{G} = \langle S, E, V\rangle$, we have

$$[\![\varphi]\!]_{\mathcal{G}} = [\![\varphi^\sharp]\!]_{\mathcal{K}}$$

for any Kripke model $\mathcal{K}$ such that $\mathcal{G} \leftrightarrow \mathcal{K}$.

Proof. We will prove this claim using structural induction on the game logic formula φ.

For the base step, take some $p \in \Phi$. We know that $p^\sharp = p$ and $(\neg p)^\sharp = \neg p$, so

$$[\![p]\!]_{\mathcal{G}} = V(p) = V'(p) = [\![p]\!]_{\mathcal{K}} = [\![p^\sharp]\!]_{\mathcal{K}} \quad \text{and} \quad [\![\neg p]\!]_{\mathcal{G}} = S \backslash V(p) = S \backslash V'(p) = [\![\neg p]\!]_{\mathcal{K}} = [\![(\neg p)^\sharp]\!]_{\mathcal{K}}$$

for any augmented game model $\mathcal{G}$ and Kripke model $\mathcal{K}$ such that $\mathcal{G} \leftrightarrow \mathcal{K}$.

For the induction hypothesis, assume that we have $\varphi, \psi \in \mathcal{L}_{\mathsf{NF}}$ such that

$$[\![\varphi]\!]_{\mathcal{G}} = [\![\varphi^\sharp]\!]_{\mathcal{K}} \quad \text{and} \quad [\![\psi]\!]_{\mathcal{G}} = [\![\psi^\sharp]\!]_{\mathcal{K}}$$

whenever $\mathcal{G} \leftrightarrow \mathcal{K}$.

For the inductive step, assume we have $\mathcal{G}$ and $\mathcal{K}$ s.t. $\mathcal{G} \leftrightarrow \mathcal{K}$. For $\varphi \wedge \psi$ and $\varphi \vee \psi$, we have

$$[\![(\varphi \wedge \psi)^\sharp]\!]_{\mathcal{K}} = [\![\varphi^\sharp \wedge \psi^\sharp]\!]_{\mathcal{K}} = [\![\varphi^\sharp]\!]_{\mathcal{K}} \cap [\![\psi^\sharp]\!]_{\mathcal{K}} = [\![\varphi]\!]_{\mathcal{G}} \cap [\![\psi]\!]_{\mathcal{G}} = [\![\varphi \wedge \psi]\!]_{\mathcal{G}},$$

$$[\![(\varphi \vee \psi)^\sharp]\!]_{\mathcal{K}} = [\![\varphi^\sharp \vee \psi^\sharp]\!]_{\mathcal{K}} = [\![\varphi^\sharp]\!]_{\mathcal{K}} \cup [\![\psi^\sharp]\!]_{\mathcal{K}} = [\![\varphi]\!]_{\mathcal{G}} \cup [\![\psi]\!]_{\mathcal{G}} = [\![\varphi \vee \psi]\!]_{\mathcal{G}}.$$

For $[\gamma]\varphi$, we need to prove that $[\![[\gamma]\varphi]\!]_{\mathcal{G}} = [\![([\gamma]\varphi)^\sharp]\!]_{\mathcal{K}}$ for all $\gamma \in \mathcal{G}_{\mathsf{NF}}$. In order to prove this, we will first prove the following claim.

Claim 1. For all $\gamma \in \mathcal{G}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$, $\zeta \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$, all $A \in \mathcal{L}^\mu_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ such that $A = \zeta^\sharp$ or $A = x^\zeta$ and all Kripke models $\mathcal{K}$, we have $[\![\tau^\zeta_\gamma(A)]\!]_{\mathcal{K}} = E_\gamma([\![A]\!]_{\mathcal{K}})$.

Proof. We will prove this claim by structural induction on the game $\gamma \in \mathcal{G}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$.

For the base case, take some $\zeta \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and Kripke model $\mathcal{K}$ and take $A \in \mathcal{L}^\mu_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ such that $A = \zeta^\sharp$ or $A = x^\zeta$. We now prove the claim for $\gamma = g$ and $\gamma = g^d$.

$$\begin{aligned}[\![\tau^\zeta_g(A)]\!]_{\mathcal{K}} \overset{5.1}{=} [\![[g]A]\!]_{\mathcal{K}} \overset{3.12}{=} \{s \in S \mid R_g(s) \subseteq [\![A]\!]_{\mathcal{K}}\} &\overset{5.2}{=} \{s \in S \mid \cap E_g(s) \subseteq [\![A]\!]_{\mathcal{K}}\} \\ &\overset{2.13}{=} \{s \in S \mid [\![A]\!]_{\mathcal{K}} \in E_g(s)\} \\ &= E_g([\![A]\!]_{\mathcal{K}})\end{aligned}$$

$$\begin{aligned}
[\![\tau^{\zeta}_{g^d}(A)]\!]_{\mathcal{K}} \overset{5.1}{=} [\![[g^d]A]\!]_{\mathcal{K}} \overset{3.12}{=} \{s \in S \mid R_g(s) \cap [\![A]\!]_{\mathcal{K}} \neq \emptyset\} &\overset{5.2}{=} \{s \in S \mid \big(\cap E_g(s)\big) \cap [\![A]\!]_{\mathcal{K}} \neq \emptyset\} \\
&= \{s \in S \mid \exists x \in \cap E_g(s) \text{ s.t. } x \in [\![A]\!]_{\mathcal{K}}\} \\
&= \{s \in S \mid \exists x \in \cap E_g(s) \text{ s.t. } x \notin [\![A]\!]^{C}_{\mathcal{K}}\} \\
&= \{s \in S \mid \cap E_g(s) \not\subseteq [\![A]\!]^{C}_{\mathcal{K}}\} \\
&= \{s \in S \mid [\![A]\!]^{C}_{\mathcal{K}} \notin E_g(s)\} \\
&= (\{s \in S \mid [\![A]\!]^{C}_{\mathcal{K}} \in E_g(s)\})^{C} \\
&= (E_g([\![A]\!]^{C}_{\mathcal{K}}))^{C} \\
&\overset{4.4}{=} E_{g^d}([\![A]\!]_{\mathcal{K}})
\end{aligned}$$

So the claim holds for $\gamma = g$ and $\gamma = g^d$ for any $\zeta \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$, any Kripke model $\mathcal{K}$ and any $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ such that $A = \zeta^{\sharp}$ or $A = x^{\zeta}$.

For the induction hypothesis, assume we have $\gamma, \delta \in \mathcal{G}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ such that

$$[\![\tau^{\zeta}_{\gamma}(A)]\!]_{\mathcal{K}} = E_{\gamma}([\![A]\!]_{\mathcal{K}}) \qquad \text{and} \qquad [\![\tau^{\zeta}_{\delta}(A)]\!]_{\mathcal{K}} = E_{\delta}([\![A]\!]_{\mathcal{K}}),$$

for all $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ such that $A = \zeta^{\sharp}$ or $A = x^{\zeta}$, $\zeta \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and all Kripke models $\mathcal{K}$.

For the inductive step, take some $\zeta \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$, Kripke model $\mathcal{K}$ and $A \in \mathcal{L}^{\mu}_{\mathsf{NF}}(\widehat{\Phi}_0, \Gamma_0)$ such that $A = \zeta^{\sharp}$ or $A = x^{\zeta}$. For the game $\gamma \sqcap \delta$, we have

$$[\![\tau^{\zeta}_{\gamma\sqcap\delta}(A)]\!]_{\mathcal{K}} \overset{5.1}{=} [\![\tau^{\zeta}_{\gamma}(A) \wedge \tau^{\zeta}_{\delta}(A)]\!]_{\mathcal{K}} = [\![\tau^{\zeta}_{\gamma}(A)]\!]_{\mathcal{K}} \cap [\![\tau^{\zeta}_{\delta}(A)]\!]_{\mathcal{K}} \overset{(\mathrm{IH})}{=} E_{\gamma}([\![A]\!]_{\mathcal{K}}) \cap E_{\delta}([\![A]\!]_{\mathcal{K}}) \overset{4.4}{=} E_{\gamma\sqcap\delta}([\![A]\!]_{\mathcal{K}})$$

The case for $\gamma \sqcup \delta$ is similar.

For γ^*, we have

$$\begin{aligned}
[\![\tau^{\zeta}_{\gamma^*}(A)]\!]_{\mathcal{K}} &\overset{5.1}{=} [\![\mu x^{[\gamma^*]\zeta}.A \vee \tau^{[\gamma^*]\zeta}_{\gamma}(x^{[\gamma^*]\zeta})]\!]_{\mathcal{K}} \\
&\overset{3.12}{=} \mathsf{lfp}.\big(A \vee \tau^{[\gamma^*]\zeta}_{\gamma}(x^{[\gamma^*]\zeta})\big)^{\mathcal{K}}_{x^{[\gamma^*]\zeta}} \\
&\overset{3.12}{=} \mathsf{lfp}\ X.[\![A \vee \tau^{[\gamma^*]\zeta}_{\gamma}(x^{[\gamma^*]\zeta})]\!]_{\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]} \\
&\overset{3.12}{=} \mathsf{lfp}\ X.[\![A]\!]_{\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]} \cup [\![\tau^{[\gamma^*]\zeta}_{\gamma}(x^{[\gamma^*]\zeta})]\!]_{\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]}
\end{aligned}$$

We know that A is either the variable x^{ζ} or the translation of ζ. In the first case, $x^{[\gamma^*]\zeta}$ clearly does not appear in A and in the second case it also does not, since otherwise $[\gamma^*]\zeta$ would be a subterm of ζ. Since $x^{[\gamma]\zeta}$ does not occur in A, changing which set the valuation assigns to $x^{[\gamma]\zeta}$ does not affect the truth-set of A, so $[\![A]\!]_{\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]} = [\![A]\!]_{\mathcal{K}}$, which gives us

$$[\![\tau^{\zeta}_{\gamma^*}(A)]\!]_{\mathcal{K}} = \mathsf{lfp}\ X.[\![A]\!]_{\mathcal{K}} \cup [\![\tau^{[\gamma^*]\zeta}_{\gamma}(x^{[\gamma^*]\zeta})]\!]_{\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]}.$$

Here we can apply the induction hypothesis for the game γ, the modal μ-calculus formula $x^{[\gamma^*]\zeta}$, the game logic formula $[\gamma^*]\zeta$ and the Kripke model $\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]$, which gives

$$\begin{aligned}
[\![\tau^{\zeta}_{\gamma^*}(A)]\!]_{\mathcal{K}} = \mathsf{lfp}\ X.[\![A]\!]_{\mathcal{K}} \cup E_{\gamma}([\![x^{[\gamma^*]\zeta}]\!]_{\mathcal{K}[x^{[\gamma^*]\zeta} \mapsto X]}) = \mathsf{lfp}\ X.[\![A]\!]_{\mathcal{K}} \cup E_{\gamma}(X) \\
\overset{4.4}{=} E_{\gamma^*}([\![A]\!]_{\mathcal{K}}).
\end{aligned}$$

The case for $\gamma^\times$ is similar.

Take the game $\gamma;\delta$, then

$$[\![\tau^\zeta_{\gamma;\delta}(A)]\!]_\mathcal{K} \overset{5.1}{=} [\![\tau^{[\delta]\zeta}_\gamma(\tau^\zeta_\delta(A))]\!]_\mathcal{K} \overset{\bigstar}{=} E_\gamma([\![\tau^\zeta_\delta(A)]\!]_\mathcal{K}) \overset{\heartsuit}{=} E_\gamma(E_\delta([\![A]\!]_\mathcal{K})) \overset{4.4}{=} E_{\gamma;\delta}([\![A]\!]_\mathcal{K}).$$

At $\bigstar$ we apply the induction hypothesis for γ with the modal μ-calculus formula $\tau^\zeta_\delta(A)$, the game logic formula $[\delta]\zeta$ (note that $([\delta]\zeta)^\sharp = \tau^\zeta_\delta(\zeta^\sharp)$) and the Kripke model $\mathcal{K}$. At $\heartsuit$ we apply the induction hypothesis for δ with the modal μ-calculus formula A, the game logic formula ζ and the Kripke model $\mathcal{K}$.

For the game $\psi?$, we get

$$[\![\tau^\zeta_{\psi?}(A)]\!]_\mathcal{K} \overset{5.1}{=} [\![\psi^\sharp \wedge A]\!]_\mathcal{K} \overset{3.12}{=} [\![\psi^\sharp]\!]_\mathcal{K} \cap [\![A]\!]_\mathcal{K} \overset{\text{(IH)}}{=} [\![\psi]\!]_\mathcal{G} \cap [\![A]\!]_\mathcal{K} \overset{4.4}{=} E_{\psi?}([\![A]\!]_\mathcal{K})$$

The case for $\psi!$ is similar. ■

We have now shown by induction that the claim holds, so we can now return to our original induction proof. If we plug in $\zeta = \varphi$ and $A = \varphi^\sharp$ into our claim, we get

$$[\![([\gamma]\varphi)^\sharp]\!]_\mathcal{K} \overset{5.1}{=} [\![\tau^\varphi_\gamma(\varphi^\sharp)]\!]_\mathcal{K} \overset{Claim\ 1}{=} E_\gamma([\![\varphi^\sharp]\!]_\mathcal{K}) \overset{\text{(IH)}}{=} E_\gamma([\![\varphi]\!]_\mathcal{G}) \overset{4.4}{=} [\![[\gamma]\varphi]\!]_\mathcal{G}$$

for all games $\gamma \in \mathcal{G}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and all $\mathcal{G}$ and $\mathcal{K}$ such that $\mathcal{G} \leftrightarrow \mathcal{K}$. This statements concludes our original induction proof, so we have now shown that

$$[\![\varphi]\!]_\mathcal{G} = [\![\varphi^\sharp]\!]_\mathcal{K}$$

for all $\varphi \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and all augmented game models $\mathcal{G}$ and Kripke models $\mathcal{K}$ s.t. $\mathcal{G} \leftrightarrow \mathcal{K}$. □

By creating unique variables $x^{[\gamma^\circ]\varphi}$ for each fixpoint formula $[\gamma^\circ]\varphi$ that is translated, the translation $(-)^\sharp$ ensures that the order on the fixpoint formulas F is the same as the subsumption order on the variables that correspond to these fixpoints in the translated formula.

Lemma 5.4. For all $\varphi \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ and all $\psi, \zeta \in F(\varphi)$, we have $x^\psi, x^\zeta \in Var(\varphi^\sharp)$ and

$$x^\psi \leq_{\varphi^\sharp} x^\zeta \quad \Rightarrow \quad \psi \preccurlyeq \zeta$$

This result is similar to [3, Proposition 24] and since we are using the same translation $(-)^\sharp$ and orders on F and $Var(\varphi^\sharp)$, the proof for our lemma would essentially be the same as the proof given for this proposition in [3]. We will demonstrate the result in the following example.

Example 5.5. Take the normal form formula $\varphi = [((a^\times \sqcup b^*)^\times; c^\times)^*]p$. Applying $(-)^\sharp$ gives

$$\begin{aligned}
\varphi^\sharp &= \tau^p_{((a^\times \sqcup b^*)^\times; c^\times)^*}(p) \\
&= \mu x^\varphi.p \vee \tau^{[c^\times]\varphi}_{(a^\times \sqcup b^*)^\times}(\tau^\varphi_{c^\times}(x^\varphi)) \\
&= \mu x^\varphi.p \vee \tau^{[c^\times]\varphi}_{(a^\times \sqcup b^*)^\times}(\nu x^\theta.x^\varphi \wedge [c]x^\theta) && \theta = [c^\times]\varphi \\
&= \mu x^\varphi.p \vee \nu x^\zeta.(\nu x^\theta.x^\varphi \wedge [c]x^\theta) \wedge \tau^\zeta_{a^\times \sqcup b^*}(x^\zeta) && \zeta = [(a^\times \sqcup b^*)^\times]\theta \\
&= \mu x^\varphi.p \vee \nu x^\zeta.(\nu x^\theta.x^\varphi \wedge [c]x^\theta) \wedge (\tau^\zeta_{a^\times}(x^\zeta) \vee \tau^\zeta_{b^*}(x^\zeta)) \\
&= \mu x^\varphi.p \vee \nu x^\zeta.(\nu x^\theta.x^\varphi \wedge [c]x^\theta) \wedge ((\nu x^\eta.x^\zeta \wedge [a]x^\eta) \vee \mu x^\psi.x^\zeta \vee [b]x^\psi) && \eta = [a^\times]\zeta, \psi = [b^*]\zeta
\end{aligned}$$

When we apply the subsumption order $<^-_{\varphi^\sharp}$ to this formula, we get

$$x^\varphi <^-_{\varphi^\sharp} x^\theta, \quad x^\varphi <^-_{\varphi^\sharp} x^\zeta, \quad x^\zeta <^-_{\varphi^\sharp} x^\eta, \quad x^\zeta <^-_{\varphi^\sharp} x^\psi.$$

Applying the transitive closure $<_{\varphi^\sharp}$ gives the two additional pairs

$$x^\varphi <_{\varphi^\sharp} x^\eta \quad \text{and} \quad x^\varphi <_{\varphi^\sharp} x^\psi.$$

The order on fixpoint formulas gives

$$\varphi \prec \theta, \zeta, \eta, \psi \qquad \text{and} \qquad \zeta \prec \eta, \psi,$$

which aligns with the results we got from the subsumption order.

5.3. *Transforming Clo-derivations to CloG$_\mathsf{K}$-derivations*

As was shown in Theorem 3.19, any Clo-derivation of $\varphi^\sharp$ can be transformed into a Clo$_{\varphi^\sharp}$-derivation of $\varphi^\sharp$. This means that in order to prove the completeness of CloG$_\mathsf{K}$ over Kripke models, it suffices to show that a Clo$_{\varphi^\sharp}$-derivation of $\varphi^\sharp$ can be transformed into a CloG$_\mathsf{K}$-derivation of φ. In this section, we will show how this transformation can be achieved.

However, we first introduce some notation. We will use $\mathcal{A}, \mathcal{B}, \ldots$ to indicate sets of sequents. We write $\pi : \mathcal{A} \vdash_\mathsf{S} \Phi$ to say that π is an S-derivation of Φ from assumptions in the set of sequents $\mathcal{A}$. We also extend the translation $(-)^\sharp$ to annotations and (sets of) sequents as follows.

$$(\varphi^\mathsf{a})^\sharp = (\varphi^\sharp)^\mathsf{a}, \qquad \Phi^\sharp = \{(\varphi^\mathsf{a})^\sharp \mid \varphi^\mathsf{a} \in \Phi\} \quad \text{and} \quad \mathcal{A}^\sharp = \{\Phi^\sharp \mid \Phi \in \mathcal{A}\}$$

and we define the set N_φ of names for $\varphi \in F$ to be equal to N_{x^φ}.

Theorem 5.6. For all $\varphi \in \mathcal{L}_\mathsf{NF}$, if $\vdash_{\mathsf{Clo}_{\varphi^\sharp}} \varphi^\sharp$, then $\vdash_{\mathsf{CloG}_\mathsf{K}} \varphi$

Proof. We will prove the theorem by proving the following claim.

$$\begin{gathered}\text{For all } \mathsf{Clo}_{\varphi^\sharp}\text{-derivations } \pi \text{ and all game logic sequents } \Phi \text{ such that } \pi : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Phi^\sharp, \\ \text{there is a } \mathsf{CloG}_\mathsf{K}\text{-derivation } \pi' : \mathcal{B} \vdash_{\mathsf{CloG}_\mathsf{K}} \Phi, \text{ where } \mathcal{B}^\sharp = \mathcal{A}.\end{gathered} \tag{1}$$

We prove this claim by induction on the complexity of Clo$_{\varphi^\sharp}$-derivations.

For the base case, the derivation π is either an application of the axiom Ax1, which derives $\Phi^\sharp$ from no assumptions, or a derivation of the sequent $\Phi^\sharp$ from the assumption $\Phi^\sharp$. In both cases, a CloG$_\mathsf{K}$-derivation π' satisfying the claim can be found by simply replacing $\Phi^\sharp$ with Φ in π.

For the inductive hypothesis, we take some $n \geq 1$ and assume that (1) holds for all Clo$_{\varphi^\sharp}$-derivations π with n or less nodes and all game logic sequents Φ such that $\pi : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Phi^\sharp$.

For the inductive step, assume we have a Clo$_{\varphi^\sharp}$-derivation $\pi : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Phi^\sharp$ with $n+1$ nodes, then we will prove that there always exists a CloG$_\mathsf{K}$-derivation $\pi' : \mathcal{B} \vdash_{\mathsf{CloG}_\mathsf{K}} \Phi$, where $\mathcal{B}^\sharp = \mathcal{A}$. We will show this by making a case distinction based on the last rule applied in π.

Before we start this case distinction, there are some assumptions we can make. Since π has at least 2 nodes, the last applied rule can not be Ax1, so we only need to consider the inference rules. Also, we may make the *injectivity assumption*, which states that for every formula $A \in \Phi^\sharp$, there is one unique $\psi \in \Phi$ such that $\psi^\sharp = A$. If Φ did not satisfy this assumption, we could simply remove formulas until it did. This would not change $\Phi^\sharp$, so we could still use the derivation π. After finding a CloG$_\mathsf{K}$-derivation for this reduced version of Φ, we could use the weak rule to add

the removed formulas back in and obtain a $\mathsf{CloG_K}$-derivation of Φ. Similarly, we can assume that none of the formulas in Φ are of the form $[\gamma;\delta]\psi$. If a formula would be of this form, we could replace it with $[\gamma][\delta]\psi$, which has the same translation. After finding a $\mathsf{CloG_K}$-derivation for this altered Φ, we could return the formula to its original form by applying the ; rule.

If the last rule applied in π is mod, then $\Phi^\sharp = [g^d]\Delta, [g]A^{\mathsf{a}}$. By the injectivity assumption, there is one unique $\psi^{\mathsf{b}} \in \Phi$ such that $(\psi^{\mathsf{b}})^\sharp = ([g]A)^{\mathsf{a}}$ and $\Phi = \psi^{\mathsf{b}}, \Psi$, where $\Psi^\sharp = [g^d]\Delta$. We can deduce the shape of ψ by looking at the definition of the translation $(-)^\sharp$. Since it does not affect annotations, we must have $\mathsf{a} = \mathsf{b}$ and $\psi^\sharp = [g]A$. The only way the translation of ψ can have this form is if $\psi = [g]\psi_0$, where $\psi_0^\sharp = A$. We also have the game logic sequent Ψ such that $\Psi^\sharp = [g^d]\Delta$. Define $\Xi = \{\chi \mid \exists \zeta \in \Psi \text{ s.t. } \zeta = [g^d]\chi\}$, then it is clear to see that $\Xi^\sharp = \Delta$. The mod rule was applied to the premise Δ, A^{a}. If we remove this rule from π, then we have a $\mathsf{Clo}_{\varphi^\sharp}$-derivation $\pi_0 : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Delta, A^{\mathsf{a}}$, which has n nodes, and a game logic sequent $\Omega = \Xi, \psi_0^{\mathsf{a}}$ such that $\Omega^\sharp = \Delta, A^{\mathsf{a}}$. This means we can apply the induction hypothesis to get a $\mathsf{CloG_K}$-derivation $\pi_0' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Omega$, where $\mathcal{B}^\sharp = \mathcal{A}$. If we apply mod to the root sequent $\Omega = \Xi, \psi_0^{\mathsf{a}}$ of this derivation, we obtain a $\mathsf{CloG_K}$-derivation π' of the sequent $\Phi = [g^d]\Xi, [g]\psi_0^{\mathsf{a}}$ from the set of assumptions $\mathcal{B}$, where $\mathcal{B}^\sharp = \mathcal{A}$.

If the last rule applied is exp, then $\Phi^\sharp = A_1^{\mathsf{b}_1}, \ldots, A_k^{\mathsf{b}_k}$ and the exp-inference is applied to the premise $A_1^{\mathsf{a}_1}, \ldots, A_k^{\mathsf{a}_k}$, where $\mathsf{a_i} \sqsubseteq \mathsf{b_i}$ for all $i \leq k$. By the injectivity assumption and the fact that the translation does not affect annotations, we know that for every A_i there exists one unique $\psi_i \in \Phi$ s.t. $(\psi_i)^\sharp = A_i$. This means that $\Phi = \psi_1^{\mathsf{b}_1}, \ldots, \psi_k^{\mathsf{b}_k}$. Removing the last applied rule from π gives us a $\mathsf{Clo}_{\varphi^\sharp}$-derivation $\pi_0 : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} A_1^{\mathsf{a}_1}, \ldots, A_k^{\mathsf{a}_k}$ with n nodes and we also have a game logic sequent $\Psi = \psi_1^{\mathsf{a}_1}, \ldots, \psi_k^{\mathsf{a}_k}$ such that $\Psi^\sharp = A_1^{\mathsf{a}_1}, \ldots, A_k^{\mathsf{a}_k}$. This means that we can apply the induction hypothesis to get a $\mathsf{CloG_K}$-derivation $\pi_0' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Psi$, where $\mathcal{B}^\sharp = \mathcal{A}$. Since $\mathsf{a_i} \sqsubseteq \mathsf{b_i}$ for all $i \leq k$, each annotation $\mathsf{a_i}$ can be turned into $\mathsf{b_i}$ by repeatedly applying the exp rule to add every name $\mathsf{x} \in \mathsf{b_i}$ that is not in $\mathsf{a_i}$. This gives us a $\mathsf{CloG_K}$-derivation $\pi' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Phi$, where $\mathcal{B}^\sharp = \mathcal{A}$.

If the last rule applied is $\vee$, then $\Phi^\sharp = \Delta, (A_0 \vee A_1)^{\mathsf{a}}$ and the $\vee$ rule is applied to the premise $\Delta, A_0^{\mathsf{a}}, A_1^{\mathsf{a}}$. By the injectivity assumption, there is a unique $\psi^{\mathsf{a}} \in \Phi$ such that $\psi^\sharp = A_0 \vee A_1$ and $\Phi = \psi^{\mathsf{a}}, \Psi$, where $\Psi^\sharp = \Delta$. Given its translation, there are three possibilities for the shape of ψ:

(i) $\psi = \psi_0 \vee \psi_1$, where $\psi_0^\sharp = A_0$ and $\psi_1^\sharp = A_1$.

(ii) $\psi = [\gamma_0 \sqcup \gamma_1]\chi$, where $([\gamma_0]\chi)^\sharp = A_0$ and $([\gamma_1]\chi)^\sharp = A_1$.

(iii) $\psi = [\psi_0!]\psi_1$, where $\psi_0^\sharp = A_0$ and $\psi_1^\sharp = A_1$.

We can obtain a $\mathsf{Clo}_{\varphi^\sharp}$-derivation $\pi_0 : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Delta, A_0^{\mathsf{a}}, A_1^{\mathsf{a}}$ with n nodes by removing the $\vee$ rule at the root of π. In all three cases, there is a game logic sequent Ξ such that $\Xi^\sharp = \Delta, A_0^{\mathsf{a}}, A_1^{\mathsf{a}}$. In cases (i) and (iii), it is $\Xi = \Psi, \psi_0^{\mathsf{a}}, \psi_1^{\mathsf{a}}$ and in case (ii), it is $\Xi = \Psi, ([\gamma_0]\chi)^{\mathsf{a}}, ([\gamma_1]\chi)^{\mathsf{a}}$. So, in each case, we can apply the induction hypothesis to get a $\mathsf{CloG_K}$-derivation $\pi_0' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Xi$, where $\mathcal{B}^\sharp = \mathcal{A}$.
In case (i), applying the $\vee$ rule to π_0' gives $\Psi, (\psi_0 \vee \psi_1)^{\mathsf{a}} = \Psi, \psi^{\mathsf{a}} = \Phi$.
In case (ii), applying the $\vee$ rule and then the $\sqcup$ rule to π_0' gives $\Psi, ([\gamma_0 \sqcup \gamma_1]\chi)^{\mathsf{a}} = \Psi, \psi^{\mathsf{a}} = \Phi$.
In case (iii), applying the $\wedge$ rule and then the ! rule to π_0' gives $\Psi, ([\psi_0]\psi_1)^{\mathsf{a}} = \Psi, \psi^{\mathsf{a}} = \Phi$.
So, in each case, we have a $\mathsf{CloG_K}$-derivation $\pi' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Phi$, where $\mathcal{B}^\sharp = \mathcal{A}$.

If the last rule applied is $\wedge$, then $\Phi^\sharp = \Delta, (A_0 \wedge A_1)^{\mathsf{a}}$ and the $\wedge$ rule is applied to the premises Δ, A_0^{a} and Δ, A_1^{a}. By the injectivity assumption, there is a unique $\psi^{\mathsf{a}} \in \Phi$ such that $\psi^\sharp = A_0 \wedge A_1$ and $\Phi = \psi^{\mathsf{a}}, \Psi$, where $\Psi^\sharp = \Delta$. If we remove the $\wedge$ rule from π, we get two $\mathsf{Clo}_{\varphi^\sharp}$-derivations $\pi_0 : \mathcal{A}_0 \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Delta, A_0$ and $\pi_1 : \mathcal{A}_1 \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Delta, A_1$, where $\mathcal{A} = \mathcal{A}_0 \cup \mathcal{A}_1$. Given its translation, there are three possibilities for the shape of ψ:

(i) $\psi = \psi_0 \wedge \psi_1$, where $\psi_0^\sharp = A_0$ and $\psi_1^\sharp = A_1$.

(ii) $\psi = [\gamma_0 \sqcap \gamma_1]\chi$, where $([\gamma_0]\chi)^\sharp = A_0$ and $([\gamma_1]\chi)^\sharp = A_1$.

(iii) $\psi = [\psi_0?]\psi_1$, where $\psi_0^\sharp = A_0$ and $\psi_1^\sharp = A_1$.

In each case, we can find two game logic sequents Ξ_0 and Ξ_1 such that $\Xi_0^\sharp = \Delta, A_0^{\mathsf{a}}$ and $\Xi_1^\sharp = \Delta, A_1^{\mathsf{a}}$. In cases (i) and (iii), take $\Xi_i = \Psi, \psi_i^{\mathsf{a}}$ for $i = 0, 1$ and, in case (ii), take $\Xi_i = \Psi, ([\gamma_i]\chi)^{\mathsf{a}}$ for $i = 0, 1$. So,in each case, we can apply the induction hypothesis to get two $\mathsf{CloG_K}$-derivations $\pi_0' : \mathcal{B}_0 \vdash_{\mathsf{CloG_K}} \Xi_0$ and $\pi_1' : \mathcal{B}_1 \vdash_{\mathsf{CloG_K}} \Xi_1$, where $\mathcal{B}_i^\sharp = \mathcal{A}_i$ for $i = 0, 1$.
In case (i), applying the $\wedge$ rule to π_0' and π_1' gives $\Psi, (\psi_0 \wedge \psi_1)^{\mathsf{a}} = \Psi, \psi^{\mathsf{a}} = \Phi$.
In case (ii), applying the $\wedge$ rule to π_0' and π_1' and then $\sqcap$, gives $\Psi, ([\gamma_0 \sqcap \gamma_1]\chi)^{\mathsf{a}} = \Psi, \psi^{\mathsf{a}} = \Phi$.
In case (iii), applying the $\wedge$ rule to π_0' and π_1' and then the ? rule gives $\Psi, ([\psi_0?]\psi_1)^{\mathsf{a}} = \Psi, \psi^{\mathsf{a}} = \Phi$.
So, in each case, we have a $\mathsf{CloG_K}$-derivation $\pi' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Phi$, where $\mathcal{B}^\sharp = \mathcal{A}$.

If the last rule applied is σ, then $\Phi^\sharp = \Delta, (\sigma x.A)^{\mathsf{a}}$, where $\sigma \in \{\mu, \nu\}$ and $\mathsf{a} \leq_{\varphi^\sharp} \mathsf{x}$, and the σ rule is applied to the premise $\Delta, A(\sigma x.A)^{\mathsf{a}}$. By the injectivity assumption, there is a unique $\psi^{\mathsf{a}} \in \Phi$ such that $\psi^\sharp = \sigma x.A$ and $\Phi = \psi^{\mathsf{a}}, \Psi$, where $\Psi^\sharp = \Delta$.

Say $\sigma = \mu$, then we know from the definition of the translation that $\psi = [\gamma^*]\chi$ for some γ and χ such that $A(x) = \chi^\sharp \vee \tau_\gamma^\psi(x)$. Also, since the variable x is created by translating the fixpoint formula ψ, we know that $x = x^\psi$. This gives

$$\begin{aligned} A(\mu x^\psi.A(x^\psi)) = \chi^\sharp \vee \tau_\gamma^\psi(\mu x^\psi.A(x^\psi)) = \chi^\sharp \vee \tau_\gamma^\psi(\mu x^\psi.\chi^\sharp \vee \tau_\gamma^\psi(x^\psi)) &= \chi^\sharp \vee \tau_\gamma^{[\gamma^*]\chi}(\tau_{\gamma^*}^\chi(\chi^\sharp)) \\ &= \chi^\sharp \vee ([\gamma][\gamma^*]\chi)^\sharp \\ &= (\chi \vee [\gamma][\gamma^*]\chi)^\sharp \end{aligned}$$

If we remove the σ rule from π, we get a $\mathsf{Clo}_{\varphi^\sharp}$-derivation $\pi_0 : \mathcal{A} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Delta, A(\mu x^\psi.A(x^\psi))$ with n nodes. We also have the game logic sequent $\Psi, \chi \vee [\gamma][\gamma^*]\chi$, which translates to $\Delta, A(\mu x^\psi.A(x^\psi))$, so by the induction hypothesis we have a $\mathsf{CloG_K}$-derivation $\pi_0' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Psi, \chi \vee [\gamma][\gamma^*]\chi$, where $\mathcal{B}^\sharp = \mathcal{A}$. We would like to apply the $*$ rule to the root of π_0', however, to do this, we need to ensure that $\mathsf{a} \preccurlyeq [\gamma^*]\chi$. Take $\mathsf{y} \in \mathsf{a}$, then by $\mathsf{a} \leq_{\varphi^\sharp} x^\psi$, we know that $\mathsf{y} \in N_y$ such that $y \leq_{\varphi^\sharp} x^\psi$. Names are not changed between languages, so y is a name for a variable in $\widehat{\Phi}_0$, as well as a name for a fixpoint formula in F. So, there must be some $\zeta \in F$ such that $\mathsf{y} \in N_\zeta = N_{x^\zeta}$. Since $N_y \cap N_{x^\zeta} = \emptyset$ if $y \neq x^\zeta$, we must have $y = x^\zeta$. This means we have $x^\zeta \leq_{\varphi^\sharp} x^\psi$ and since the order on $Var(\varphi^\sharp)$ is reflected into the order on game logic fixpoints by the translation, we have $\zeta \preccurlyeq \psi$. So, for all $\mathsf{y} \in \mathsf{a}$, $\mathsf{y} \in N_\zeta$ such that $\zeta \preccurlyeq \psi$, which means $\mathsf{a} \preccurlyeq \psi = [\gamma^*]\chi$. The side condition is satisfied, so we can apply the $*$ rule to π_0', giving us a $\mathsf{CloG_K}$-derivation $\pi' : \mathcal{B} \vdash_{\mathsf{CloG_K}} \Phi$, where $\mathcal{B}^\sharp = \mathcal{A}$. In the case where $\sigma = \nu$, we can use similar reasoning, using the $\times$ rule instead of $*$.

If the last rule applied is ν-clo_x, then $\Phi^\sharp = \Delta, (\nu x.A)^{\mathsf{a}}$, the ν-clo_x rule was applied to $\Delta, A(\nu x.A)^{\mathsf{a}\mathsf{x}}$, where $\mathsf{a} \leq_{\varphi^\sharp} x$, $\mathsf{x} \in N_x$ and $\mathsf{x} \notin \Delta, \mathsf{a}$ and there is a leaf of π that is labelled with the assumption $\Omega = \Delta, (\nu x.A)^{\mathsf{a}\mathsf{x}}$. By the injectivity assumption, there is a unique $\psi^{\mathsf{a}} \in \Phi$ such that $\psi^\sharp = \nu x.A$ and $\Phi = \psi^{\mathsf{a}}, \Psi$, where $\Psi^\sharp = \Delta$. From the translation, we know that $\psi = [\gamma^\times]\chi$ for some γ and χ such that $A(x) = \chi^\sharp \wedge \tau_\gamma^\psi(x)$ and we have $x = x^\psi$. We also know that $A(\nu x.A) = (\chi \wedge [\gamma][\gamma^\times]\chi)^\sharp$.

The approach we used in the previous cases will not work here. This is because when we apply the induction hypothesis to get a $\mathsf{CloG_K}$-derivation of $\Psi, \chi \wedge [\gamma][\gamma^\times]\chi$, we have no guarantee that this derivation has $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}\mathsf{x}}$ as one of its assumptions. We do get at least one assumption which would translate to Ω and, by the definition of the translation $(-)^\sharp$, we know this assumption would have the shape $\Theta, ([\gamma^\times]\chi)^{\mathsf{a}\mathsf{x}}$, where $\Theta^\sharp = \Delta$, but we can not be sure that $\Theta = \Psi$. This means we might not be able to apply the clo_x rule in order to obtain the sequent $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$ from $\Psi, (\chi \wedge [\gamma][\gamma^\times]\chi)^{\mathsf{a}\mathsf{x}}$. So, we will need to use a different approach here.

We will first construct a derivation of $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$ that only differs from a valid CloG_K-derivation in the fact that not all assumptions associated with applications of the clo_x rule are discharged, and then transform it into a proper CloG_K-derivation. In order to obtain the building blocks that we will use to construct this proof, we will first need the observation below, where we define $\mathcal{S}$ to be the set of game logic sequents Σ such that $\Sigma^\sharp = \Delta$.

Claim 2. For all $\Sigma \in \mathcal{S}$ there exist sets $\mathcal{B}_\Sigma$ and $\mathcal{L}_\Sigma$ of game logic sequents such that

1) $\mathcal{B}_\Sigma^\sharp = \mathcal{A}$ and $\mathcal{L}_\Sigma^\sharp = \{\Delta\}$
2) For any annotation $\mathsf{b} = \mathsf{a}x_1, \dots, x_n$, where $x_1, \dots, x_k \in N_{[\gamma^\times]\chi}$, there is a CloG_K-derivation

$$\rho_\Sigma^\mathsf{b} : \mathcal{B}_\Sigma \cup \{\Theta, ([\gamma^\times]\chi)^\mathsf{b} \mid \Theta \in \mathcal{L}_\Sigma\} \vdash_{\mathsf{CloG}_\mathsf{K}} \Sigma, (\chi \wedge [\gamma][\gamma^\times]\chi)^\mathsf{b}$$

Proof of claim. Fix $\Sigma \in \mathcal{S}$. We have $\Sigma^\sharp = \Delta$, so the game logic sequent $\Sigma, (\chi \wedge [\gamma][\gamma^\times]\chi)^{\mathsf{ax}}$ translates to $\Delta, A(\nu x.A)^{\mathsf{ax}}$. If we remove the last applied rule, ν-clo_x, from π, we get a $\mathsf{Clo}_{\varphi^\sharp}$-derivation $\pi_0 : \mathcal{A} \cup \{\Omega\} \vdash_{\mathsf{Clo}_{\varphi^\sharp}} \Delta, A(\nu x.A)^{\mathsf{ax}}$. Since the ν-clo_x rule was removed, the assumption Ω is no longer discharged, so it is added to the set of assumptions for π_0. This derivation has n nodes, so by the induction hypothesis, we have a CloG_K-derivation $\pi_0' : \mathcal{B}_\Sigma \cup \mathcal{B}' \vdash_{\mathsf{CloG}_\mathsf{K}} \Sigma, (\chi \wedge [\gamma][\gamma^\times]\chi)^{\mathsf{ax}}$, where $\mathcal{B}_\Sigma^\sharp = \mathcal{A}$ and $\mathcal{B}'$ contains all the assumptions that translate to Ω. This means that every sequent in $\mathcal{B}'$ must be of the form $\Theta, \zeta^{\mathsf{ax}}$, where $\Theta^\sharp = \Delta$ and $\zeta^\sharp = \nu x.A$. Since $\zeta^\sharp$ is the same fixpoint formula with the same fixpoint variable x as $\psi^\sharp$, they must both be translations of the same game logic fixpoint formula, so $\zeta = \psi = [\gamma^\times]\chi$. So $\mathcal{B}' = \{\Theta, ([\gamma^\times]\chi)^{\mathsf{ax}} \mid \Theta \in \mathcal{L}_\Sigma\}$, where $\mathcal{L}_\Sigma$ is a set of sequents such that $\mathcal{L}^\sharp = \{\Delta\}$. We have now obtained the sets $\mathcal{B}_\Sigma$ and $\mathcal{L}_\Sigma$ that satisfy the first condition of our claim.

Say we have an annotation $\mathsf{b} = \mathsf{a}\bar{x}$, where $\bar{x} = x_1 \dots x_k$ and each x_i is a name for $[\gamma^\times]\chi$. Now, we replace every instance of x in an annotation in π_0' with $\bar{x}$. The inference rules that leave annotations unchanged are not affected by this replacement. Note that x is in the annotation of the root sequent of π_0', so there was no application of ν-clo_x, since this would have removed x from the annotations. This means that the problem this replacement brings about is that applications of the exp rule that were used to x to the annotation of a formula are now no longer valid. However, we can easily solve this problem by replacing each such use of the exp rule with a sequence of applications of the exp rule which adds each name in $\bar{x}$ one by one. By making these changes, we have turned the derivation π_0' into a derivation

$$\rho_\Sigma^\mathsf{b} : \mathcal{B}_\Sigma \cup \{\Theta, ([\gamma^\times]\chi)^\mathsf{b} \mid \Theta \in \mathcal{L}_\Sigma\} \vdash_{\mathsf{CloG}_\mathsf{K}} \Sigma, (\chi \wedge [\gamma][\gamma^\times]\chi)^\mathsf{b},$$

so the proof of the claim is finished. ■

With this claim proven, we can now use these derivations ρ_Σ^b to construct a derivation of $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$, which will be *almost* a valid CloG_K-derivation. This will be a derivation in the proof system D, which is CloG_K extended with the following inference rule.

$$(\mathsf{a} \preccurlyeq x \in N_{[\gamma^\times]\varphi}, x \notin \Delta, \mathsf{a}) \; \frac{\Delta, (\varphi \wedge [\gamma][\gamma^\times]\varphi)^{\mathsf{ax}}}{\Delta, ([\gamma^\times]\varphi)^{\mathsf{a}}} \; \mathsf{D}_x$$

This rule is the clo_x rule without the requirement that the assumptions of the form $\Delta, ([\gamma^\times]\varphi)^{\mathsf{ax}}$ are discharged. We will call a node t *dangling* if the D_x rule is applied at t. Before we start constructing our D-derivation of $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$, we will add one name x_Σ for every $\Sigma \in \mathcal{S}$ to the set N_ψ. We know from the side condition of ν-clo_x that $\mathsf{a} \leq_{\varphi^\sharp} x^\psi$. We saw in the case for the σ rule that this means that for all $y \in \mathsf{a}$ there is some $\zeta \in F$ such that $y \in N_\zeta$ and $\zeta \preccurlyeq \psi$, so we know that $\mathsf{a} \preccurlyeq x$ for any $x \in N_\psi$. Furthermore, if $\bar{x}$ contains only names from N_ψ, we have $\mathsf{a}\bar{x} \preccurlyeq x$.

We start our construction with a one-node derivation, which contains only the sequent $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$. Suppose that at any time during our construction, our derivation has an assumption of the form $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$, where $\Sigma \in \mathcal{S}$ and $\mathsf{b} = \mathsf{a}\bar{\mathsf{x}}$, where $\bar{\mathsf{x}} = \mathsf{x}_{\Sigma_1} \dots \mathsf{x}_{\Sigma_k}$ such that $\Sigma \notin \{\Sigma_1, \dots, \Sigma_k\} \subseteq \mathcal{S}$ (Note that this includes our initial sequent!). By Claim 2, there is a $\mathsf{CloG_K}$-derivation $\rho_\Sigma^{\mathsf{bx}_\Sigma}$ of $\Sigma, (\chi \wedge [\gamma][\gamma^\times]\chi)^{\mathsf{b}}$ from the set of assumptions $\mathcal{B}_\Sigma \cup \{\Theta, ([\gamma^\times]\chi)^{\mathsf{b}} \mid \Theta \in \mathcal{L}_\Sigma\}$. Through an application of $\mathsf{D}_{\mathsf{x}_\Sigma}$, we will adjoin the root of $\rho_\Sigma^{\mathsf{bx}_\Sigma}$ to every leaf that is labelled with the assumption $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$. By doing this, we replace the assumption $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$ with the set of assumptions $\mathcal{B}_\Sigma \cup \{\Theta, ([\gamma^\times]\chi)^{\mathsf{b}} \mid \Theta \in \mathcal{L}_\Sigma\}$.

Since $\mathcal{S}$ is a finite set, each branch of our constructed derivation will either end on an application of Ax1, a sequent from $\mathcal{B}_\Sigma$ for some $\Sigma \in \mathcal{S}$ or an assumption of the form $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$, which has already appeared earlier in the branch, albeit with a different annotation. When this last case happens, we cannot use the claim to obtain a derivation $\rho_\Sigma^{\mathsf{bx}_\Sigma}$, since the name x_Σ is already in the annotation b. Once all branches have been closed in one of these three ways, we have found a D-derivation ρ of $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$ from assumptions in $\{B_\Sigma \mid \Sigma \in \mathcal{S}\} \cup \{\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}} \mid \Sigma \in \mathcal{S}\}$.

We can make some observations about the D-derivation ρ that we have constructed.

1. All leaves of ρ are either labelled with an axiom, a sequent from $\mathcal{B} = \bigcup_{\Sigma \in \mathcal{S}} \mathcal{B}_\Sigma$ or a sequent of the form $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$, where $\Sigma \in \mathcal{S}$ and $\mathsf{b} = \mathsf{a}\mathsf{x}_{\Sigma_1} \dots \mathsf{x}_{\Sigma_k}$ such that $\Sigma_1 = \Psi$, $\Sigma \in \{\Sigma_1, \dots, \Sigma_k\} \subseteq \mathcal{S}$ and $\Sigma_i \neq \Sigma_j$ if $i \neq j$.
2. If a leaf l is labelled with $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$, where $\mathsf{b} = \mathsf{a}\mathsf{x}_{\Sigma_1} \dots \mathsf{x}_{\Sigma_k}$, then the path from the root of ρ to l passes through the nodes $t_1, \dots t_k$ in that order, where t_i is a dangling node or the conclusion of an application of the $\mathsf{clo}_{\mathsf{x}_{\Sigma_i}}$ rule and the name x_{Σ_i} is introduced by that rule.
3. If t is a dangling node of ρ, labelled with $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$ and l is a leaf above t labelled with $\Sigma, ([\gamma^\times]\chi)^{\mathsf{c}}$, then $\mathsf{bx}_\Sigma \sqsubseteq \mathsf{c}$.

To show that we can transform the D-derivation ρ into a $\mathsf{CloG_K}$-derivation, we will show that we can turn any D-derivation satisfying the above conditions into a D-derivation that still satisfies the conditions and has a smaller number of dangling nodes.

Let σ be a D-derivation satisfying the conditions given above. Pick a dangling node t with maximal distance to the root of σ. This ensures that there are no dangling nodes above t. Since t is a dangling node, it is labelled with $\Sigma, ([\gamma^\times]\chi)^{\mathsf{a}\bar{\mathsf{x}}}$ and its successor is labelled with $\Sigma, (\chi \wedge [\gamma][\gamma^\times]\chi)^{\mathsf{a}\bar{\mathsf{x}}\mathsf{x}_\Sigma}$ for some $\Sigma \in \mathcal{S}$ and $\bar{\mathsf{x}} = \mathsf{x}_{\Sigma_1} \dots \mathsf{x}_{\Sigma_k}$, where $\Sigma \notin \{\Sigma_1, \dots, \Sigma_k\} \subset \mathcal{S}$. Let L_t be the set of leaves above t that are labelled with a sequent of the form $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}}$.

If L_t is empty, then we can remove all occurrences of the name x_Σ above t and replace the $\mathsf{D}_{\mathsf{x}_\Sigma}$ rule applied at t with an application of the $\times$ rule.

If L_t is non-empty, then every $l \in L_t$ is labelled by a sequent of the form $\Sigma, ([\gamma^\times]\chi)^{\mathsf{b}_l}$ for some annotation b_l. From condition 3, we know that $\mathsf{b}_l = \mathsf{a}\bar{\mathsf{x}}\mathsf{x}_\Sigma\mathsf{c}_l$ for some sequence of names c_l. Clearly, for every leaf $l \in L_t$, the sequent it is labelled with could be derived from $\Sigma, ([\gamma^\times]\chi)^{\mathsf{a}\bar{\mathsf{x}}\mathsf{x}_\Sigma}$ by repeated applications of the exp rule, which add all the names in c_l. So, we attach to every leaf $l \in L_t$ such a sequence of applications of exp, replacing each leaf l with a new leaf l' that is labelled with $\Sigma, ([\gamma^\times]\chi)^{\mathsf{a}\bar{\mathsf{x}}\mathsf{x}_\Sigma}$. Once this is done, we replace the $\mathsf{D}_{\mathsf{x}_\Sigma}$ rule applied at t with the $\mathsf{clo}_{\mathsf{x}_\Sigma}$ rule and subsequently discharge the assumption $\Sigma, ([\gamma^\times]\chi)^{\mathsf{a}\bar{\mathsf{x}}\mathsf{x}_\Sigma}$ at every leaf l' with $l \in L_t$.

It is easy to see that in both of these cases, the resulting D-derivation still fulfills conditions 1, 2 and 3, the node t is no longer a dangling node and no new dangling nodes have been created. So, by these transformations, we have finally obtained a $\mathsf{CloG_K}$-derivation π' for $\Psi, ([\gamma^\times]\chi)^{\mathsf{a}}$ from the set of assumptions $\mathcal{B}$, where $\mathcal{B}^\sharp = \mathcal{A}$.

So in every case, there exists a $\mathsf{CloG_K}$-derivation π' for Φ from assumptions $\mathcal{B}$, where $\mathcal{B}^\sharp = \mathcal{A}$, so the inductive step is complete and (1) has been proven by induction. □

5.4. *Completeness of CloG_K via transformation*

Now that we have established that for any game logic formula φ and Clo-derivation of $\varphi^\sharp$ we can find a CloG_K-derivation of φ, we have everything we need to show that the system CloG_K for the game logic language $\mathcal{L}$ is complete over Kripke models.

Theorem 5.7. [Completeness of CloG_K] For all $\varphi \in \mathcal{L}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$, if $\models_{\mathtt{K}^{\Phi_0,\Gamma_0}} \varphi$, then $\vdash_{\mathsf{CloG}_\mathsf{K}} \varphi$

Proof. Say we have a game logic formula φ that is valid over Kripke-$\mathcal{L}^{\mu}_{\mathsf{NF}}(\Phi_0, \Gamma_0)$ models. By the modal equivalence of the class $\mathtt{K}^{\Phi_0,\Gamma_0}$ and the class $\mathtt{G}_\mathtt{K}^{\Phi_0,\Gamma_0}$ of augmented game models, this means that φ is also valid over augmented game models. By Theorem 5.3, the translation $(-)^\sharp$ is validity-preserving, so we know that the modal μ-calculus formula $\varphi^\sharp$ is valid over Kripke models. Since the system Clo is known to be complete over Kripke models, we know that there must be a Clo-derivation of $\varphi^\sharp$. By Theorem 3.19, there also exists a $\mathsf{Clo}_{\varphi^\sharp}$-derivation of $\varphi^\sharp$ and, by Theorem 5.6, there is a CloG_K-derivation for φ. □

6. Conclusion

In this thesis, we adapted the sequent system CloG for game logic into a new cyclic sequent system CloG_K for game logic by replacing the monotone modal rule from CloG with the normal modal rule. We then proved the completeness of CloG_K over Kripke models. This result was obtained by giving a validity-preserving translation $(-)^\sharp$ from game logic formulas into the modal μ-calculus and showing that we can transform any Clo-derivation of a translated game logic formula $\varphi^\sharp$ into a CloG_K-derivation of φ.

6.1. Discussion

The translation $(-)^\sharp$ from game logic to the modal μ-calculus we defined closely resembles the one given in [3], but it differs in the fact that, in our paper, the languages it translates between have different sets of propositional atoms. This is because the fixpoint variables x^φ for fixpoint formulas $\varphi \in F$ are unique variables that are "created" by this translation and did not exist in the set of propositional atoms for game logic. These variables being unique is essential to make sure that the translation reflects the subsumption order on the fixpoint variables into the order on fixpoint formulas in game logic

We decided not to work with the global subsumption order for fixpoint variables in the modal μ-calculus, instead creating the intermediate system Clo_C to get around using this order. This was because we expected the renaming of variables, which is necessary to define this global order, to pose a problem to the way we defined our translation $(-)^\sharp$. Since some unique variables are "created" by this translation, having to rename some instances of these variables to use the global order could have introduced complications.

The method we used to transform from Clo-derivations to CloG_K-derivations closely follows the approach used in [3] for the transformation from Clo to CloM. Our approach for finding a CloG_K-derivation that gives the same conclusion as an application of the closure rule in a Clo-derivation involves constructing a derivation, which is almost a proper CloG_K-derivation, by continuously appending derivations to leaves that are labelled with a particular kind of assumption. This derivation is then transformed step by step in to a proper CloG_K-derivation. This method and the machinery of annotated proofs could be some useful tools in working with and proving the completeness of other cyclic proof systems.

Given more time, we would have liked to also adapt the systems G from [3] and Par from [1] into systems that are complete over Kripke models and shown this completeness, but these tasks turned out to be too intensive for time-frame we had for this paper.

6.2. Future research

As outlined in the introduction, after proving the completeness over Kripke models of CloG_K, a next step could be to also adapt the systems G and Par into systems that can be shown to be complete over Kripke models by replacing their monotone modal rules with the normal version, similarly to how we adapted CloG into CloG_K. It might be possible to use a similar sequence of transformations as the one given in [3] for the proof of the soundness and completeness over game models of Par, starting at CloG_K, to prove the completeness over Kripke models of these newly defined systems.

It could also be attempted to give a transformation directly from Clo to CloG_K by renaming the variables in the modal μ-calculus to be able to use the global subsumption order. This would likely involve redefining the translation $(-)^\sharp$ such that it still reflects the this global order on fixpoint variables into the order on fixpoint formulas. However, this seems like it would be a very difficult task and there is not much new knowledge to be gained by doing it, so it is not of high priority.

References

[1] Rohit Parikh. The logic of games and its applications. In *North-Holland Mathematics Studies*, volume 102, pages 111–139. Elsevier, 1985.

[2] Marc Pauly. *Logic for social software*. Universiteit van Amsterdam, 2001.

[3] Sebastian Enqvist, Helle Hvid Hansen, Clemens Kupke, Johannes Marti, and Yde Venema. Completeness for game logic. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13. IEEE, 2019.

[4] Bahareh Afshari and Graham E Leigh. Cut-free completeness for modal mu-calculus. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12. IEEE, 2017.

[5] Brian F Chellas. *Modal logic: an introduction*. Cambridge university press, 1980.

[6] Eric Pacuit. *Neighborhood semantics for modal logic*. Springer, 2017.

[7] Patrick Blackburn, Maarten De Rijke, and Yde Venema. *Modal logic: graph. Darst*, volume 53. Cambridge University Press, 2002.

[8] Yde Venema. Lectures on the modal μ-calculus, 2008.

[9] Patrick Cousot and Radhia Cousot. Constructive versions of Tarski's fixed point theorems. *Pacific Journal of Mathematics*, 82(1):43 – 57, 1979.

[10] Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285 – 309, 1955.

[11] Marc Pauly and Rohit Parikh. Game logic - an overview. *Studia Logica*, 75(2):165–182, 2003.

[12] Christopher Worthington. Proof transformations for game logic. Bachelor thesis, Rijksuniversiteit Groningen, 2021.