Modal and Relevance Logics for Qualitative Spatial Reasoning

Pranab Kumar Ghosh

Supervisor:
Dr. Michael Winter

Submitted in partial fulfilment
of the requirements for the degree of
Master of Science

Department of Computer Science
Brock University
St. Catharines, Ontario

Abstract

Qualitative Spatial Reasoning (QSR) is an alternative technique to represent spatial relations without using numbers. Regions and their relationships are used as qualitative terms. Mostly peer qualitative spatial reasonings has two aspect: (a) the first aspect is based on inclusion and it focuses on the ”part-of” relationship. This aspect is mathematically covered by mereology. (b) the second aspect focuses on topological nature, i.e., whether they are in ”contact” without having a common part. Mereotopology is a mathematical theory that covers these two aspects.

The theoretical aspect of this thesis is to use classical propositional logic with non-classical relevance logic to obtain a logic capable of reasoning about Boolean algebras i.e., the mereological aspect of QSR. Then, we extended the logic further by adding modal logic operators in order to reason about topological contact i.e., the topological aspect of QSR. Thus, we name this logic Modal Relevance Logic (MRL). We have provided a natural deduction system for this logic by defining inference rules for the operators and constants used in our (MRL) logic and shown that our system is correct. Furthermore, we have used the functional programming language and interactive theorem prover Coq to implement the definitions and natural deduction rules in order to provide an interactive system for reasoning in the logic.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Michael Winter for his continuous support, patience, motivation, and immense knowledge throughout this research. His guidance helped me in all the time of the research and writing of this thesis. I could not have imagined finishing this research without his help.

Besides my supervisor, I would like to thank the rest of the thesis committee: Dr. Brian Ross, and Dr. Beatrice Ombuki-Berman, and external examiner Dr. Torsten Hahman to read through my thesis and for their insightful comments and suggestions. I am appreciative to the Department of Computer Science of Brock University for the excellent education and environment.

I am profoundly grateful to my parents, brother, sister, wife Baishakhy and son Siddharth, who always believe in me and encouraged to reach this goal.

Last but not the least, I wish to thank my late grandfather Manik Lal Ghosh, his words still inspires me today. I dedicate this thesis to him.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Applications of Qualitative Spatial Reasoning (QSR) include validating spatial problems in everyday life situations. For example, an apartment design specification indicates which rooms should be to close to others, or a city design specification states that essential public services such as bus terminal, hospitals should be closer to residential area whereas factories should be far away.

In mathematics and computer science, this type of problem is an application of QSR. In artificial intelligence, QSR deals with qualitative features of spatial entities. Qualitative reasoning is an alternative technique that represents spatial relations without using numbers. Regions and their relationships are used as qualitative terms. Among others, the most basic relationships between regions are the “part-of” and the “connection” (or “contact”) relation.

The relationship between regions has two aspects. The first aspect is based on inclusion and it focuses on the “part-of” relationship. This aspect is mathematically covered by mereology. The second aspect focuses on topological nature, i.e., whether they are in “contact” without having a common part. Mereotopology is a mathematical theory that covers these two aspects. We will use the theory of Boolean contact algebras (BCAs) as the concrete mathematical theory of mereotopology. A BCA is a Boolean algebra with a binary contact relation C. The order of the Boolean algebra provides the part-of relationship between regions and the contact relation C the topological relationship between them.

In this thesis, our aim is to introduce a modal and relevance logic to reason about spatial entities. At first, we would like to represent the logic and its basic features in theory and then implement the logic using type theory.

Our logic is a combination of modal and relevance logic. Modal logic is a class of logics extending propositional logic. It adds new operators that provide access to a restricted version of quantification. Those operators can be used to describe the relationship of the elements. Relevance logic is a non-classical logic that was developed to represent the feature of the implication and was ignored in classical propositional logic. In this thesis, we will use two relevance operators, one implication based on the sum (or union) operation of regions and one negation based on the complement of a region. Consequently, the relevance logic part will cover the part-of aspect of mereotopology. First, we will concentrate on the relevance portion of the logic and present a set of axioms that is equivalent to fact that the frame provides the structure of a Boolean algebra. Then we will introduce a modal operator based on the contact relation together with a set of appropriate axioms forcing each frame to be a BCA. In addition, we will provide a natural deduction system for our logic that we will prove to be correct.

This thesis is not the first attempt to use modal logic for Qualitative Spatial Reasoning [8]. The fundamental difference between the approach taken in [8] and this thesis is the models considered. In our logic models are Boolean contact algebras but in [8] models are general topological spaces.

In order to perform the verification of our proofs in natural deduction, we will use the functional programming language and interactive theorem prover Coq [21]. One of the key advantages of Coq is programming and verification can be done in the same language. Also, Coq allows users to write customized tactics in Coqs Ltac tactics language. This thesis is not the first attempt to use Coq to reason about modal logic, our implementation of modal logic is similar to [2].

The remainder of the thesis is structured as follows. Prior to discussing our implementation, in Chapter 2 we will introduce the Boolean algebra, Boolean contact algebra, and its basic properties as described in [13]. Then we will discuss the syntax and semantics of propositional, modal and relevance logic. We will also define and prove the Boolean algebra and Boolean contact algebra axioms in our enhanced modal relevance logic. In Chapter 3 we will describe our implementation in natural deduction calculus. Then in Chapter 5, we will discuss some features of Coq which are relevant to our implementation. A discussion of our implementation follows in Chapter 6. Finally, in Chapter 7 we will present our conclusion and future work.

Chapter 2

Mathematical Preliminaries

In this chapter we want to provide the mathematical preliminaries that are required for this thesis. We will focus on topology, Boolean algebras, Boolean contact algebras, propositional, relevance and modal logic.

2.1 Topology

Topology is a mathematical theory that represents the relationships between the spatial entities of a space. In this theory spatial entities are represented as certain sets of points. More details on topology and topological space can be found in [6, 7, 17, 19]. Topology and topological space is formally defined as follows:

Definition 1. (Topology) *Let X be a non-empty set and $\tau \subseteq \mathcal{P}(X)$, i.e., a set of subsets of X. Then τ is called a topology on X if it satisfies the following conditions:*

1. *The set X and empty set $\emptyset$ are in τ,*

2. *The union of any subset of τ is in τ and*

3. *The intersection of any finitely many sets of τ is in τ.*

The pair $\langle X, \tau \rangle$ with τ a topology on X is called a topological space.

Example 1. *Suppose that $X = \{a, b, c\}$ and $\tau = \{\emptyset, \{a\}, \{a, b\}\}$. Then τ is the topology of X since it satisfies all the properties of Definition 1. Thus, $\langle X, \tau \rangle$ is the topological space.*

Given a topological space we can define open and closed sets as follows:

Definition 2. (Open and Closed set) *Let $\langle X, \tau \rangle$ be a topological space and $A \subseteq X$. Then we say that A is open if $A \in \tau$ and that A is closed if $X \setminus A \in \tau$, i.e. if the complement of A is open.*

Example 2. *In the topological space from the previous example the open sets are the sets $\emptyset, \ldots, X$ and the closed sets are the sets $\emptyset, \ldots, X$. As this example shows then empty set and X are both open and closed. Such a set is usually called clopen. In general, since the $\emptyset$ is open, X is closed, and, hence, clopen. A similar argument shows that also the $\emptyset$ is always clopen.*

An important notion in topology is the interior and the closure of a set. They are defined as follows:

Definition 3. (Interior and Closure) *Let x be a subset of X in a topological space $\langle X, T \rangle$, then the interior $Int(x)$ of x is the largest open set contained in x, i.e., $Int(x) - \bigcup\{y \mid y$ open and $y \subseteq x\}$. The closure of $Cl(x)$ is $Cl(x) - \bigcap\{y \mid y$ closed and $x \subseteq y\}$*

In mereotopology we concentrate on so-called regions. Regions are specific subsets of a topological space. A very common approach is to define regions as the regular closed subsets. Alternatively, and completely equivalent, one could also use the regular open subsets.

Definition 4. (Regular Open and Closed set) *Let $x \subseteq X$ in a topological space $\langle X, T \rangle$, then X is called regular open iff[1] x is equal to the interior of its closure, i.e., $x = Int(Cl(x))$ and X is called regular closed iff x is equal to the closure of its interior, i.e., $x = Cl(Int(x))$.*

Example 3. *Consider a Euclidean plane $\mathbb{R}^2$ with regular topology in Figure 2.1. Then the interior is regular open and the circle including the border is regular closed. Any single point is closed but not regular closed since its interior is empty.*

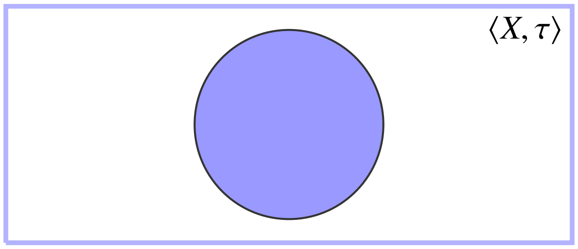

Figure 2.1: Regular Open and Closed Set

From now on we will use the notation $RegCL(X)$ and $RegOP(X)$ to denote the set of regular closed and regular open set respectively.

[1] We use the abbreviation iff for if and only if.

2.2 Boolean Algebras

A Boolean algebra is a certain kind of ordered or lattice structure. It generalizes the well known operations on sets and truth values. More details on Boolean algebra can be found in [14, 15]. We will be using the same sets of notation as shown in the Lemma 3. Boolean algebra formally defined as follows:

Definition 5. (Boolean algebra) *A Boolean algebra (BA) is a structure $\mathcal{B} = \langle B, +, \cdot, ^*, 0, 1 \rangle$ with a set B, two binary operators + and · on B, a unary operator * on B and two elements* $0, 1 \in B$ *satisfying the following axioms for all* $x, y, z \in B$*:*

Commutativity	$x + y = y + x$	$x \cdot y = y \cdot x$
Identity	$x + 0 = x$	$x \cdot 1 = x$
Distributivity	$x + y \cdot z = (x + y) \cdot (x + z)$	$x \cdot (y + z) = x \cdot y + x \cdot z$
Complements	$x + x^* = 1$	$x \cdot x^* = 0$

Example 4. *Suppose that* $A = \{a, b, c, d\}$*. Then the powerset* $\mathcal{P}(A)$ *is a complete BA under set inclusion according to Lemma 3.*

It is worth mentioning that the axioms above are sufficient to prove other properties usually required for Boolean algebras. In the following lemma, we have summarized other basic properties of BA. Proofs are available in [9] as well as in the Coq code of this thesis.

Lemma 1. *Let* $\mathcal{B} = \langle B, +, \cdot, ^*, 0, 1 \rangle$ *be a BA. Then the following axioms holds for all* $x, y, z \in B$*:*

UId_1	*If* $x + o = x$ *for all* $x,$ *then* $o = 0$
UId_2	*If* $x \cdot i = x$ *for all* $x,$ *then* $i = 1$
Idm_1	$x + x = x$
Idm_2	$x \cdot x = x$
Bnd_1	$x + 1 = 1$
Bnd_2	$x \cdot 0 = 0$
Abs_1	$x + (x \cdot y) = x$
Abs_2	$x \cdot (x + y) = x$
UNg	*If* $x + xn = 1$ *and* $x \cdot xn = 0,$ *then* $xn = x^*$
DNg	$x^{**} = x$
A_1	$x + (x^* + y) = 1$

A_2	$x \cdot (x^* \cdot y) = 0$
B_1	$(x + y) + (x^* \cdot y^*) = 1$
B_2	$(x \cdot y) \cdot (x^* + y^*) = 0$
C_1	$(x + y) \cdot (x^* \cdot y^*) = 0$
C_2	$(x \cdot y) + (x^* + y^*) = 1$
DMg_1	$(x + y)^* = x^* \cdot y^*$
DMg_2	$(x \cdot y)^* = x^* + y^*$
D_1	$(x + (y + z)) + x^* = 1$
D_2	$(x \cdot (y \cdot z)) \cdot x^* = 0$
E_1	$y \cdot (x + (y + z)) = y$
E_2	$y + (x \cdot (y \cdot z)) = y$
F_1	$(x + (y + z)) + y^* = 1$
F_2	$(x \cdot (y \cdot z)) \cdot y^* = 0$
G_1	$(x + (y + z)) + z^* = 1$
G_2	$(x \cdot (y \cdot z)) \cdot z^* = 0$
H_1	$((x + y) + z)^* \cdot x = 0$
H_2	$((x \cdot y) \cdot z)^* + x = 1$
I_1	$((x + y) + z)^* \cdot y = 0$
I_2	$((x \cdot y) \cdot z)^* + y = 1$
J_1	$((x + y) + z)^* \cdot z = 0$
J_2	$((x \cdot y) \cdot z)^* + z = 1$
K_1	$(x + (y + z)) + ((x + y) + z)^* = 1$
K_2	$(x \cdot (y \cdot z)) \cdot ((x \cdot y) \cdot z)^* = 0$
L_1	$(x + (y + z)) \cdot ((x + y) + z)^* = 0$
L_2	$(x \cdot (y \cdot z)) + ((x \cdot y) \cdot z)^* = 1$
Ass_1	$x + (y + z) = (x + y) + z$
Ass_2	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$

In the following lemma, we are going to state and proof another property of Boolean algebra.

Lemma 2. *$x = y$ iff $x^* = y^*$ for all $x, y \in B$.*

Proof.

$\Rightarrow$ Proof is trivial.

$\Leftarrow$ Assume that $x^* = y^*$. Then we have $x^{**} = y^{**}$. From DNg we conclude $x = y$.

□

2.3 Boolean Contact Algebras

As mentioned in the introduction a Boolean algebra can be used to model the mereological aspect of regions. In this section we want to concentrate on the topological aspect, i.e., a contact relation C.

Definition 6. (Boolean contact algebra) *[13] A binary relation C on a Boolean algebra $\mathcal{B}$ is called contact relation if it satisfies the following axioms:*

Null disconnected (C_0)	$xCy \Rightarrow x, y \neq 0$
Reflexivity (C_1)	$x \neq 0 \Rightarrow \ xCx$
Symmetry (C_2)	$xCy \Leftrightarrow \ yCx$
Compatibility (C_3)	xCy *and* $y \leq z \Rightarrow \ xCz$
Summation (C_4)	$xC(y+z) \Rightarrow \ xCy$ *or* xCz

Therefore Boolean contact algebra (BCA) is a structure of $\langle B, C, +, \cdot, ^*, 0, 1\rangle$.

The axioms C_0 - C_2 are trivial, thus we are going to explain only C_3 and C_4 with examples. **Compatibility (C_3):** For any three non-empty regions x, y and z, if region x is in contact with y and $y \leq z$, then x is in contact with z as well. This is called “contact relation axiom C_3” as well as “compatibility” axiom.

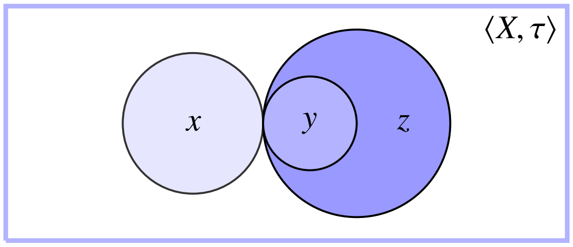

Figure 2.2: Compatibility Contact Relation

Example 5. *Consider a Euclidean plane $\mathbb{R}^2$ with regular topology and three non-empty regions x, y and z with xCy and $y \leq z$ in Figure 2.2. Therefore, we conclude region x is in contact with z as well.*

Summation (C_4): Let any three non-empty regions x, y and z, If region x is in contact with $(y + z)$, then either x is in contact with y or x is in contact with z. This is called "contact relation axiom C_4" as well as "summation" axiom.

Example 6. *Consider a Euclidean plane $\mathbb{R}^2$ with regular topology and three non-empty regions x, y and z with $xC(y + z)$ in Figure 2.3. In this case region x is in contact with region y. But in Figure 2.4 region x is in contact with region z.*

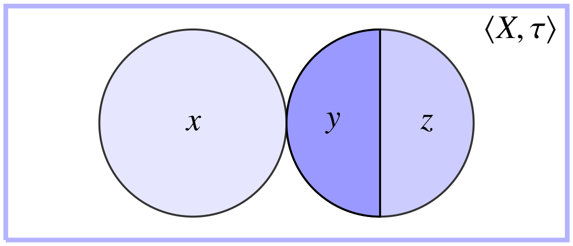

Figure 2.3: Summation (Scenario 1) Contact Relation

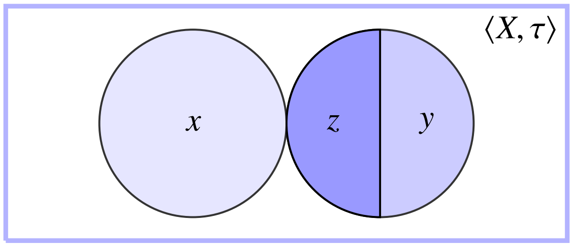

Figure 2.4: Summation (Scenario 2) Contact Relation

In the following lemma, we want to show that $RegCL(X)$ is a Boolean algebra under set inclusion.

Lemma 3. *[12] Let $RegCL(X)$ bet the set of regular closed sets of $\langle X, \tau \rangle$, then $RegCL(X)$ with the operations below is a Boolean algebra.*

1. $x + y = x \cup y$,

2. $x \cdot y = Cl(Int(x \cap y))$,

3. $x^* = Cl(X \setminus x)$,

4. $0 = \emptyset$ *and*

5. $1 = X$.

From Lemma 3, we have $x \cdot y \subseteq x \cap y$. Let us consider the scenario where $x \cdot y = \emptyset$ but $x \cap y \neq \emptyset$, i.e., x and y do not share a common region but do have common points (Figure 2.5). Using this scenario we are going to formally define contact relation C.

Definition 7. (Contact relation) *[12] Let $x, y \in RegCL(X)$ in a topological space $\langle X, \tau \rangle$. Then the contact relation C on $RegCL(X)$ is defined as:*

$$xCy \iff x \cap y \neq \emptyset.$$

Example 7. *Consider a Euclidean plane $\mathbb{R}^2$ with regular topology in Figure 2.5, regions $x, y \in RegCl(X)$ are externally connected i.e., the intersection of x and y has exactly one point. Please note that $x \cdot y = \emptyset$ since their intersection is not regular closed. Thus, x and y are in contact, i.e. xCy, but $x \cdot y = \emptyset$.*

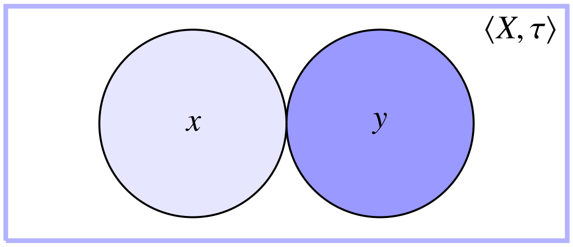

Figure 2.5: Region x and y are Externally Contacted on a Topological Space

2.4 Propositional Logic (PL)

Propositional logic was developed to constitute the relationships between declarative atomic sentences. As well, it is required that those atomic sentences are propositions i.e., they are either valid or false. To illustrate, let assume the sentences as follows:

1. Bangladesh is in South Asia.

2. “Bangladesh is a neighbour of India,” and “Bangladesh is in South Asia”.

3. What is the time now?

The first sentence has only one atomic proposition and it is either true or false. The second sentence can be separated into two atomic propositions, and they are either true or false. But the third sentence is not a declarative sentence, therefore, it is not a proposition. Classical propositional logic uses logical connectives such as “and”, “or” and “implication” to join propositions together to evaluate the truth-value of the declarative sentence. We are going to start with propositional logic syntax and semantics. More details on propositional logic can be found in [16].

2.4.1 Syntax

The syntax of propositional logic consists of propositional variables, propositional operators, and constants. We formally define the propositional logic syntax as follows:

Definition 8. (Propositional logic syntax) *Let P be a set of propositional variables, then $Prop$ be the set of propositional formulas is recursively defined by the following rules:*

(PropL.1) Each propositional variable $p \in P$ is a propositional formula, i.e., $P \subseteq Prop$,

(PropL.2) $\bot$ is a propositional formula, i.e., $\bot \in Prop$ and

(PropL.3) If $\varphi, \psi \in Prop$, then $\varphi \to \psi \in Prop$.

We will be using the same set of the operators and rules (PropL.1)-(PropL.3) in an enhancement of the logics in this thesis. In such a context the rules from above have to be used just with a different set of formulas instead of Prop. This will also be applied to other definitions of syntactic or semantic rules defined later.

It is worth mentioning that operator mentioned in (PropL.1)-(PropL.3) are sufficient to represent all propositional logic formulas, and the other optional operators are used for simplicity of representation. Thus, we are going to introduce them as abbreviations as follows:

$\neg\varphi$	:=	$\varphi \to \bot$	PLAbbr1
$\varphi \wedge \psi$	:=	$\neg(\varphi \to \neg\psi)$	PLAbbr2
$\varphi \vee \psi$	:=	$\neg\varphi \to \psi$	PLAbbr3
$\varphi \leftrightarrow \psi$	:=	$\varphi \to \psi \wedge \psi \to \varphi$	PLAbbr4
$\top$	:=	$\neg\bot$	PLAbbr5

In the next section of semantics, the semantics definitions of those abbreviations will be given in a lemma.

2.4.2 Semantics

Validity evaluation process of propositional logic required replacing those propositions with its actual value (either true or false) and this is called truth assignment. Then in the next step, we want to evaluate the validity of the propositional formulas considering the semantics of the logical operators.

According to the standard literature such as [16], propositional logic semantics does not consider the universe, model, and frame. As we mention in the above, we want to reuse the definition of propositional logic semantics as well in enhancement of the logics later of this thesis which includes modal and relevance logic. Therefore, our propositional logic semantics includes universe W, model $\mathcal{M}$, and frame $\mathcal{F}$. Now we are going formally them as follows:

Definition 9. (Propositional logic frame) *A propositional logic frame (PL-frame) $\mathcal{F}$ consist of a non-empty set universe W.*

The elements of the universe set W are the states or worlds.

Definition 10. (Propositional logic model) *A propositional logic model $\mathcal{M} = \langle \mathcal{F}, v \rangle$ is a pair where v is valuation function such that $v : P \to \mathcal{P}(W)$.*

We will be using the same definition of the propositional logic model for the other logics introduced later in this thesis only W will be replaced by respective logic universe W. For example, for $x \in W$, then the possible subset of W is $\{\emptyset, \{x\}\}$. Then $v : P \to \mathcal{P}(W)$ is true at $\{x\}$ and false otherwise. Thus, we are ensuring that propositional logic semantics remains same with this new structure and it can be reused for other logics.

Definition 11. (Propositional logic semantics) *Let $\mathcal{M}$ be a model, $x \in W$ be a state, $p \in P$ true at x and $\varphi \in Prop$. We will write $\mathcal{M}, x \models \varphi$ i.e., φ is true (or satisfied) in $\mathcal{M}$ at x. Furthermore, φ is called true in $\mathcal{M}$ (written $\mathcal{M} \models \varphi$) iff $\mathcal{M}, x \models \varphi$ for all $x \in W$ (though W is a singleton for propositional logic). We will also write $\models \varphi$ i.e., φ is valid (or true, or a tautology) in all models. Therefore, the satisfaction relation $\mathcal{M}, x \models \varphi$ is recursively defined by:*

(SemPropL.1) $\mathcal{M}, x \models p \Leftrightarrow x \in v(p)$,

(SemPropL.2) $\mathcal{M}, x \not\models \bot$ *and*

(SemPropL.3) $\mathcal{M}, x \models \varphi \to \psi \Leftrightarrow \mathcal{M}, x \models \psi$ *whenever* $\mathcal{M}, x \models \varphi$.

It is worth to mention that $\mathcal{M} \models \varphi$ and $\models \varphi$ will be used later in enhancement of the logics in this thesis with the definition similar to the one above. In the following lemma, we want to state the semantics of the abbreviations mentioned in the syntax section. Proofs are available in the Coq code of this thesis.

Lemma 4. *Let $\mathcal{M}$ be a model, and $x \in W$ be a state. Then we have:*

1. $\mathcal{M}, x \models \neg\varphi \Leftrightarrow \mathcal{M}, x \not\models \varphi$,
2. $\mathcal{M}, x \models \varphi \wedge \psi \Leftrightarrow \mathcal{M}, x \models \psi$ *and* $\mathcal{M}, x \models \varphi$,
3. $\mathcal{M}, x \models \varphi \vee \psi \Leftrightarrow \mathcal{M}, x \models \psi$ *or* $\mathcal{M}, x \models \varphi$,
4. $\mathcal{M}, x \models \varphi \leftrightarrow \psi \Leftrightarrow \mathcal{M}, x \models \varphi \to \psi$ *and* $\mathcal{M}, x \models \psi \to \varphi$ *and*
5. $\mathcal{M}, x \models \top$.

2.5 Modal Logic (ML)

Modal logic is an extension of propositional logic by introducing new operators □ and ◇. We are going to start with modal logic syntax and semantics and more details on modal logic can be found in [5, 16].

2.5.1 Syntax

The syntax of modal logic consists of propositional logic syntax and modal operator that provide access to a restricted version of quantification. The formal definition of modal logic syntax is as follows:

Definition 12. (Modal logic syntax) *The set Mod of modal logic formulas is recursively defined by (PropL.1)-(PropL.3) and :*

(ModL.1) If $\varphi \in Mod$, then $[R]\varphi \in Mod$.

It is worth mentioning that operators mentioned above are sufficient to represent all modal logic formulas. But the other optional operator is used for simplicity of representation. Thus, we are going to introduce them as an abbreviation as follows:

$$\langle R\rangle\varphi \quad := \quad \neg[R3]\neg\varphi \qquad \text{MLAbbr1}$$

In the next section of semantics, the semantics definitions of this abbreviation will be given in a lemma.

2.5.2 Semantics

We are going to formally define the semantics of modal logic as follows:

Definition 13. (Modal logic frame) *A modal logic ML-frame $\mathcal{F} = \langle W, R\rangle$ is a structure such that:*

1. *W is a non-empty set, called the universe and*

2. *R is a binary relation on W, i.e., $R \subseteq W \times W$.*

We will use the usual notation Rxy or xRy to denote $\{x, y\} \in R$, i.e., that x and y are in relation R. The definition of the modal logic model $\mathcal{M}$ is similar to Definition 10.

Definition 14. (Modal logic semantics) *Let $\mathcal{M}$ be a model, and $x, y \in W$ be a state, then the satisfaction relation $\mathcal{M}, x \models \varphi$ and $\mathcal{M}, y \models \varphi$ is recursively defined by (SemPropL.1)-(SemPropL.3) and :*

(SemModL.1) $\mathcal{M}, x \models [R]\varphi \Leftrightarrow \forall y \in W$ *iff* Rxy, *then* $\mathcal{M}, y \models \varphi$.

In the following lemma, we want to state the semantics of the abbreviation mentioned in the syntax section. The proof is available in any standard literature of modal logic [5, 16].

Lemma 5. *Let $\mathcal{M}$ be a model, and $x, y \in W$ be the states. Then we have:*

1. $\mathcal{M}, x \models \langle R\rangle\varphi \Leftrightarrow \exists y \in W$ *with* Rxy *and* $\mathcal{M}, y \models \varphi$.

2.6 Relevance Logic (RL)

[1, 11, 18] Relevance logic is non-classical logic that was developed to represent the feature of the implication and was ignored in classical propositional logic. Assume two atomic sentences “Today is Monday”, and “two and two is four”. If those two sentences are used to form an implication →, then this statement is true. But whether or not “Today is Monday” seems in no way relevant to whether “two and two are four”. On the other hand, if we consider the true statement “if $x = 1$, then $x + 1 = 2$”, then the information of the assumption “$x = 1$” is needed to conclude “$x + 1 = 2$”, i.e., the assumption is relevant to the conclusion. Relevance logic focuses on implications of the latter form, i.e., it requires the assumption is relevant to the conclusion in a valid implication.

2.6.1 Syntax

To avoid conflict with "classical logic implication" (→) symbol, we will use $\twoheadrightarrow$ to denote "relevance logic implication" and $\backsim$ is used to represent "relevance logic not" to avoid conflict with "classical logic not" ($\neg$). In the following definitions, we define the syntax and semantics of relevance logic formally.

Definition 15. (Relevance logic syntax) *The set RL of relevance logic formulas is recursively defined by the rules (PropL.1) and*

(RelL.1) If $\varphi, \psi \in RL$, then $\varphi \twoheadrightarrow \psi \in RL$ and

(RelL.2) If $\varphi \in RL$, then $\backsim \varphi \in RL$.

2.6.2 Semantics

To evaluate the validity of a relevance logic formula, we will require ternary relation R on W, and the unary function * takes elements to elements of W. The usual notation $Rxyz$ to denote x,y and z are in relation R. [8] Urquhart's interpretation of $Rxyz$ is based on a fusion ($\circ$) operator, i.e., he uses $Rxyz$ iff $z = x \circ y$. His motivation is that the information of the implication is the information that has to be added to the information needed for φ in order to conclude ψ.

In order to reason about Boolean algebras we will use a different interpretation of the relevance implication. Our motivation is to split an element x into two components y and z, i.e., $x = f(y, z)$ for a suitable function f. In other words, we want to define a relevance implication such that x consists of the information from y and z. We may visualize our version of relevance logic implication in Figure 2.6.

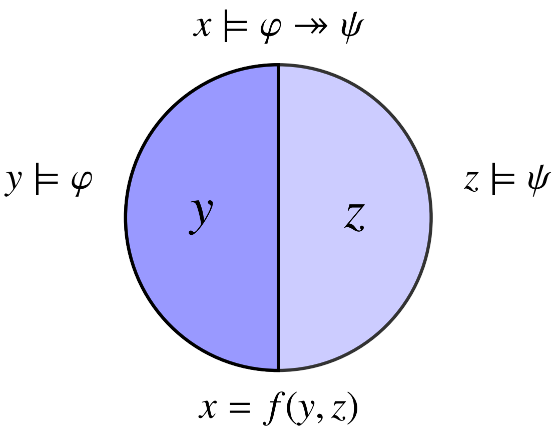

Figure 2.6: Relevance Implication

In the following definitions, we will give a formal definition of relevance logic RL-frame $\mathcal{F}$, model $\mathcal{M}$ and semantics.

Definition 16. (Relevance logic frame) *A relevance logic RL-frame* $\mathcal{F} = \langle W, f, g\rangle$ *is a structure such that:*

1. *W is a set of elements, called the universe,*
2. *f is a binary function between elements on W and*
3. *g is a unary function taking elements to elements.*

Again, the definition of relevance logic model is similar to Definition 10.

Definition 17. (Relevance logic semantics) *Let* $\mathcal{M}$ *be a model, and* $x, y, z \in W$ *be a state. Then the satisfaction relation* $\mathcal{M}, x \models \varphi$ *is recursively defined by (SemPropL.1) and :*

(SemRelL.1) $\mathcal{M}, x \models \varphi \twoheadrightarrow \psi \Leftrightarrow \forall y, z($ *if* $x = f(y, z)$ *and* $\mathcal{M}, y \models \varphi$*, then* $\mathcal{M}, z \models \psi)$ *and*

(SemRelL.2) $\mathcal{M}, x \models\backsim \varphi \Leftrightarrow \mathcal{M}, g(x) \not\models \varphi$.

2.7 Propositional Relevance Logic (PRL)

In this section we want to introduce propositional relevance logic, combining the properties from propositional and relevance logic as defined above, without involving any new operator. But we will bring in few abbreviations in this logic for simplicity of representation. Later on, we will provide the semantics definitions of those abbreviations in some lemmas. Now we are formally going defining the syntax and semantics of this logic.

2.7.1 Syntax

The syntax definition of propositional relevance logic is defined as follows:

Definition 18. (Propositional relevance logic syntax) *The set PRL of propositional relevance logic formulas is recursively defined by the rules (PropL.1) -(PropL.3) and (RelL.1) - (RelL.2).*

In addition to the abbreviation defined before we will use the following:

$$\begin{array}{lcll} \varphi ⩓ \psi & := & \neg(\varphi \twoheadrightarrow \neg\psi) & \text{PRLAbbr1} \\ \varphi ⩔ \psi & := & \neg\varphi \twoheadrightarrow \psi & \text{PRLAbbr2} \end{array}$$

$N\varphi$	:=	$\backsim \neg\varphi$	PRLAbbr3
$\varphi \multimap \psi$	:=	$N(N\varphi \twoheadrightarrow N\psi)$	PRLAbbr4
$\varphi \mathbin{\overset{\circ}{\wedge}} \psi$	:=	$\neg(\varphi \multimap \neg\psi)$	PRLAbbr5
$\varphi \mathbin{\underset{\circ}{\vee}} \psi$	:=	$\neg\varphi \multimap \psi$	PRLAbbr6

In the next section, will provide lemmas that show establish the semantics of the abbreviations defined above.

2.7.2 Semantics

In order to evaluate the validity of propositional relevance logic formula, we will require a universe, frame, and model. We are going to formally define them as follows:

Definition 19. (Propositional relevance logic frame and model) *The definition of PRL-frames and models are similar to the Definition of RL-frames and models.*

Now we are going define the semantics of the propositional relevance logic as follows:

Definition 20. (Propositional relevance logic semantics) *Let $\mathcal{M}$ be a model. Then the satisfaction relation is recursively defined by (SemPropL.1) - (SemPropL.3) and (SemRelL.1) - (SemRelL.2).*

In the following lemma, we will give an obvious proof of the fact that $g(g(x)) = x$ is equivalent to the axiom schema $\varphi \leftrightarrow \backsim\backsim \varphi$ for all $x \in W$ in a PRL-frame. Please note that the axiom schemas represent infinitely many formulas by substituting concrete formulas for the formula variables $\varphi, \psi, \ldots$. Consequently, we say that an axiom schema is valid iff all instantiations of the schema by concrete formulas is valid. Furthermore, we will often call schemas simply formulas.

Lemma 6. *The formula $\varphi \leftrightarrow \backsim\backsim \varphi$ is true in a PRL-frame $\mathcal{F}$, iff $g(g(x)) = x$ for all $x \in W$.*

Proof. Let $\mathcal{M}$ be a model and $x \in W$. Then we have:

$$\begin{aligned} & \mathcal{M}, x \models \backsim\backsim \varphi & \\ & \Leftrightarrow \mathcal{M}, g(x) \not\models \backsim \varphi & \text{by Definition (SemRelL.2)} \\ & \Leftrightarrow \mathcal{M}, g(g(x)) \models \varphi & \text{by Definition (SemRelL.2)} \end{aligned}$$

- $\Rightarrow$ Assume the formula schema is valid in $\mathcal{F}$. Then the formula $p \leftrightarrow \backsim\backsim p$ is valid in the model $\mathcal{M}$ based on $\mathcal{F}$ with $v(p) = \{x\}$. From the computation above we conclude $g(g(x)) = x$.

- $\Leftarrow$ Assume $g(g(x)) = x$ for all $x \in W$ with $\mathcal{M}, x \models \varphi$ based on $\mathcal{F}$. Then we have $\mathcal{M}, g(g(x)) \models \varphi$ and we from the computation above we have $\mathcal{M}, x \models \backsim\backsim \varphi$. Therefore we have $\mathcal{M}, x \models \varphi \rightarrow \backsim\backsim \varphi$ and we conclude $\mathcal{F} \models \varphi \rightarrow \backsim\backsim \varphi$.

□

Now in the following definition, we are going to define a dual operator f^d, also we will give a prove that f^d can be derived from f and g.

Definition 21. (Dual operator in PRL-frame) *Let $\mathcal{F}$ be a PRL-frame and $x, y \in W$. The dual operator f^d is defined as follows:*

$$f^d(x, y) \quad := \quad g(f(g(x), g(y)))$$

Lemma 7. *If $g(g(x)) = x$ for all $x \in W$, then $f(x, y) = g(f^d(g(x), g(y)))$ for all $x, y \in W$.*

Proof. The proof as follows:

$$\begin{array}{rll} g(f^d(g(x), g(y))) & = g(g(f(g(g(x)), g(g(y))))) & \text{by Definition} \\ & = f(x, y) & \text{assumption} \end{array}$$

□

From now on we will assume that g is involutive, i.e., $g(g(x)) = x$ for all x, or alternatively that $\varphi \leftrightarrow \backsim\backsim \varphi$, in the PRL-frame. Note that Lemma 7 can be applied in this context.

2.7.2.1 Semantics of PRL-frame Abbreviations

Now, we are going to present the semantic properties of our abbreviations PRLAbbr1 - PRLAbbr6 in the following lemmas.

Lemma 8. *If $x \in W$, then $\mathcal{M}, x \models \varphi \Wedge \psi$ iff there are $y, z \in W$ with $x = f(y, z)$ and $\mathcal{M}, y \models \varphi$ and $\mathcal{M}, z \models \psi$.*

Proof. Let $\mathcal{M}$ be a model and $x \in W$. Then we have:

$$\begin{array}{ll} \mathcal{M}, x \models \varphi \Wedge \psi & \\ \Leftrightarrow \mathcal{M}, x \models \neg(\varphi \twoheadrightarrow \neg\psi) & \text{by Definition} \\ \Leftrightarrow \mathcal{M}, x \not\models \varphi \twoheadrightarrow \neg\psi & \text{by Definition} \\ \Leftrightarrow \text{there are } y, z \in W \text{ so that it is not the case that} & \text{by Definition} \end{array}$$

if $x = f(y, z)$ and $\mathcal{M}, y \models \varphi$, then $\mathcal{M}, z \models \neg\psi$

$\Leftrightarrow$ there are $y, z \in W$ with $x = f(y, z)$ and $\mathcal{M}, y \models \varphi$
and $\mathcal{M}, z \not\models \neg\psi$

$\Leftrightarrow$ there are $y, z \in W$ with $x = f(y, z)$ and $\mathcal{M}, y \models \varphi$
and $\mathcal{M}, z \models \psi$

□

Lemma 9. *If $x \in W$, then $\mathcal{M}, x \models \varphi \doublevee \psi$ iff for all $y, z \in W$ if $x = f(y, z)$, then $\mathcal{M}, y \models \varphi$ or $\mathcal{M}, z \models \psi$.*

Proof. This can be shown similar to Lemma 8. □

Lemma 10. *If $x \in W$, then $\mathcal{M}, x \models N\varphi$ iff $\mathcal{M}, g(x) \models \varphi$ iff $\mathcal{M}, x \models \neg \backsim \varphi$*

Proof. Let $\mathcal{M}$ be a model and $x \in W$. Then we have:

$\mathcal{M}, x \models N\varphi$

$\Leftrightarrow \mathcal{M}, x \models \backsim \neg\varphi$ by Definition

$\Leftrightarrow \mathcal{M}, g(x) \models \varphi$ by Definition

$\Leftrightarrow \mathcal{M}, g(x) \not\models \neg\varphi$ by Definition

$\Leftrightarrow \mathcal{M}, x \models \neg \backsim \varphi$ by Definition

By the definition of N we obtain from the previous computation $\mathcal{M}, x \models N\varphi$ iff $\mathcal{M}, g(x) \models \varphi$. □

Lemma 11. *If $x \in W$, then $\mathcal{M}, x \models \varphi \multimap \psi$ iff $x = f^d(y, z)$ and $\mathcal{M}, y \models \varphi$ implies $\mathcal{M}, z \models \psi$ for all $y, z \in W$.*

Proof. Let $\mathcal{M}$ be a model and $x \in W$. Then we have:

$\mathcal{M}, x \models \varphi \multimap \psi$

$\Leftrightarrow \mathcal{M}, x \models N(N\varphi \twoheadrightarrow N\psi)$ by Definition

$\Leftrightarrow \mathcal{M}, g(x) \models N\varphi \twoheadrightarrow N\psi$ by Lemma 10

$\Leftrightarrow$ for all $y, z \in W$ if $g(x) = f(y, z)$ and $\mathcal{M}, y \models N\varphi$, by Definition
then $\mathcal{M}, z \models N\psi$

$\Leftrightarrow$ for all $y, z \in W$ if $g(x) = f(y, z)$ and $\mathcal{M}, g(y) \models \varphi$, by Lemma 10
then $\mathcal{M}, g(z) \models \psi$

$$
\begin{aligned}
&\Leftrightarrow \text{ for all } y, z \in W \text{ if } g(x) = g(f^d(g(y), g(z))) \text{ and} && \text{by Lemma 7}\\
&\qquad \mathcal{M}, g(y) \models \varphi, \text{ then } \mathcal{M}, g(z) \models \psi\\
&\Leftrightarrow \text{ for all } y, z \in W \text{ if } x = f^d(g(y), g(z)) \text{ and } \mathcal{M}, g(y) \models \varphi, && \text{by } g \text{ involutive}\\
&\qquad \text{then } \mathcal{M}, g(z) \models \psi\\
&\Leftrightarrow \text{ for all } y, z \in W \text{ if } x = f^d(y, z) \text{ and } \mathcal{M}, y \models \varphi, && \text{by } g \text{ involutive}\\
&\qquad \text{then } \mathcal{M}, z \models \psi
\end{aligned}
$$

□

Lemma 12. *If $x \in W$, then $\mathcal{M}, x \models \varphi \parr \psi$ iff there are $y, z \in W$ with $x = f^d(y, z)$ and $\mathcal{M}, y \models \varphi$ and $\mathcal{M}, z \models \psi$.*

Proof. This can be shown similar to Lemma 8 by using Lemma 11 and exchanging f with f^d. □

Lemma 13. *If $x \in W$, then $\mathcal{M}, x \models \varphi \rotatebox[origin=c]{180}{$\parr$} \psi$ iff for all $y, z \in W$ if $x = f^d(y, z)$, then $\mathcal{M}, y \models \varphi$ or $\mathcal{M}, z \models \psi$.*

Proof. This can be shown similar to Lemma 9 by using Lemma 11 and exchanging f with f^d. □

2.7.2.2 Additional Properties of PRL-frame

In this section, we are going to state some more additional property of PRL-frame in some lemmas.

Lemma 14. *If $x \in W$, then following statement is equivalent*

1. $\mathcal{M}, x \models \varphi \parr \psi$,
2. $\mathcal{M}, x \models \backsim (\backsim \neg\varphi \twoheadrightarrow \backsim \psi)$ *and*
3. $\mathcal{M}, x \models N(N\varphi \Wedge N\psi)$.

Proof. Let $\mathcal{M}$ be a model and $x \in W$. Then we have:

$$
\begin{aligned}
&\mathcal{M}, x \models \varphi \parr \psi\\
&\Leftrightarrow \mathcal{M}, x \models \neg(\varphi \multimap \neg\psi) && \text{by Definition}\\
&\Leftrightarrow \mathcal{M}, x \models \neg N(N\varphi \twoheadrightarrow N\neg\psi) && \text{by Definition}\\
&\Leftrightarrow \mathcal{M}, x \models \neg(N\varphi \twoheadrightarrow N\neg\psi) && \text{by Definition}
\end{aligned}
$$

$$
\begin{aligned}
&\Leftrightarrow \mathcal{M}, x \models N\neg(N\varphi \twoheadrightarrow \neg N\psi) && \text{by Lemma 10}\\
&\Leftrightarrow \mathcal{M}, x \models \backsim \neg\neg(\backsim \neg\varphi \twoheadrightarrow \neg \backsim \neg\psi) && \text{by Definition}\\
&\Leftrightarrow \mathcal{M}, x \models \backsim (\backsim \neg\varphi \twoheadrightarrow \backsim \psi) && \text{by Definition}\\
&\Leftrightarrow \mathcal{M}, x \models N\neg(N\varphi \twoheadrightarrow N\neg\psi) && \text{by Definition}\\
&\Leftrightarrow \mathcal{M}, x \models N(N\varphi \Wedge N\psi) && \text{by Definition}
\end{aligned}
$$

□

2.8 Propositional Relevance Logic with E (PRLE)

Now, we want to introduce propositional relevance logic with E, adding a new formula E to propositional relevance logic. Our motivation is that E is only true at the smallest element of a Boolean algebra. In this section, we are going to formulate Boolean algebra axioms in this new logic and prove that this logic's frame is a Boolean algebra. Now we are going to start with the formal definitions of syntax and semantics as follows:

2.8.1 Syntax

We are going to define the syntax of propositional relevance logic with E as follows:

Definition 22. (Propositional relevance logic with E syntax) *The set PRLE the set of propositional relevance logic with E formula is recursively defined by rules (PropL.1)-(PropL.3), (RelL.1)-(RelL.2) and :*

(PRLE.1) $E \in PRLE$.

For simplicity of representation, we are going to introduce a new formula U that is true in the largest element of W, in a form of abbreviation as follows:

$$U \quad := \quad NE \qquad \text{PRLEAbbr1}$$

2.8.2 Semantics

As before, in order to define the validity evaluation of propositional relevance logic with E formulas, we will require a universe, a frame, and a model. Now we are going to formally define them as follows:

Definition 23. (Propositional relevance logic with E frame) *A propositional relevance logic with E PRLE-frame $\mathcal{F} = \langle W, e, f, g\rangle$ is a structure such that:*

1. *$\langle W, f, g\rangle$ is a PRL-frame and*

2. *$e \in W$ is the smallest element.*

The definition of a model $\mathcal{M}$ for propositional relevance logic with E is similar to Definition 10.
Now we are going to formally define the semantics of propositional relevance logic with E as follows:

Definition 24. (Propositional relevance logic with E semantics) *Let $\mathcal{M}$ be a model, and $x, y, z \in W$ be a state, then the satisfaction relation $\mathcal{M}, x \models \varphi$ is recursively defined by (SemPropL.1)-(SemPropL.3), (SemRelL.1) - (SemRelL.2) and :*

(SemPRLE.1) $\mathcal{M}, x \models E \Leftrightarrow x = e$.

In the following lemma, we want to state the semantics of the abbreviation mentioned in the syntax section.

Lemma 15. *Let $\mathcal{M}$ be a model, and $x \in W$ be a state, then the satisfaction relation $\mathcal{M}, x \models U$ is recursively defined by:*

1. $\mathcal{M}, x \models U \Leftrightarrow x = g(e)$.

As mentioned before, our ultimate goal is to formulate axioms that force any PRLE-frame to be a Boolean algebra. This will be done in the next section.

2.8.3 PRLE-frame Axioms

In this section, we are going to present axioms that force any PRLE-frame to be a Boolean algebra.

Lemma 16. *The formula schema $\varphi \mathbin{\&\!\!\&} \psi \rightarrow \psi \mathbin{\&\!\!\&} \varphi$ is true in a PRLE-frame $\mathcal{F}$, iff f is commutative, i.e., $f(x, y) = f(y, x)$ for all $x, y \in W$.*

Proof.

⇒ Assume $x, y \in W$. Then by the assumption the formula $\mathcal{F} \models p \mathbin{\&\!\!\&} q \rightarrow q \mathbin{\&\!\!\&} p$ for propositional variables p and q are true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be a model with $v(p) = \{x\}$ and $v(q) = \{y\}$. Then $\mathcal{M}, f(x, y) \models p \mathbin{\&\!\!\&} q \rightarrow q \mathbin{\&\!\!\&} p$. We want to show $\mathcal{M}, f(x,y) \models p ⩕ q$, Since $\mathcal{M}, x \models p$ and $\mathcal{M}, y \models q$, then $\mathcal{M}, f(x,y) \models p ⩕ q$ (by Lemma 8). we get $\mathcal{M}, f(x,y) \models q ⩕ p$. This implies that there are x', y' with $f(x,y) = f(y',x')$ and $\mathcal{M}, y' \models q$ and $\mathcal{M}, x' \models p$ (by Lemma 8). Since $\mathcal{M}, y' \models q$ implies $y' = y$ and $\mathcal{M}, x' \models p$ implies $x' = x$ we obtain $f(x,y) = f(y,x)$.

$\Leftarrow$ Let $\mathcal{M}$ be a model and $w \in W$ so that we have to show $\mathcal{M}, w \models \varphi ⩕ \psi \rightarrow \psi ⩕ \varphi$. Assume $\mathcal{M}, w \models \varphi ⩕ \psi$ so that we have to show $\mathcal{M}, w \models \psi ⩕ \varphi$. This implies that there are x, y with $w = f(x,y)$ and $\mathcal{M}, x \models \varphi$ and $\mathcal{M}, y \models \psi$ (by Lemma 8). Since f is commutative we have there are x, y with $w = f(y,x)$ and $\mathcal{M}, x \models \varphi$ and $\mathcal{M}, y \models \psi$. By Lemma 8 we obtain $\mathcal{M}, w \models \psi ⩕ \varphi$.

□

Lemma 17. *The formula schema $\varphi ⅋ \psi \rightarrow \psi ⅋ \varphi$ is true in a PRLE-frame $\mathcal{F}$, iff f^d is commutative, i.e., $f^d(x,y) = f^d(y,x)$ for all $x, y \in W$.*

Proof. This can be shown similar to Lemma 16 by using Lemma 11 and exchanging f with f^d. □

Lemma 18. *The formula schema $\varphi \rightarrow \varphi ⩕ E$ is true in a PRLE-frame $\mathcal{F}$, iff $x = f(x,e)$ for all $x \in W$.*

Proof.

$\Rightarrow$ Assume $x \in W$. Then by assumption the formula $p \rightarrow (p ⩕ E)$ for a propositional variable p is true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be such a model with $v(p) = \{x\}$. Then $\mathcal{M}, x \models p$ and, hence, $\mathcal{M}, x \models p ⩕ E$ (by Lemma 8). This implies that there are $y, z \in W$ with $x = f(y,z)$ and $\mathcal{M}, y \models p$ and $\mathcal{M}, z \models E$. By the definition of $v(p)$ and E we have $y = x$ and $z = e$, i.e., $x = f(x,e)$.

$\Leftarrow$ Assume $x \in W$ with $\mathcal{M}, x \models \varphi$. Since $x = f(x,e)$ and $\mathcal{M}, x \models \varphi$ and $\mathcal{M}, e \models E$. We get $\mathcal{M}, x \models \varphi ⩕ E$ (by Lemma 8).

□

Please note that it is not the case that the formula schema $\varphi \leftrightarrow (\varphi ⩖ E)$ is true in a PRLE-frame $\mathcal{F}$ iff $x = f(x,e)$ for all $x \in W$. As an example consider the model based on the Boolean algebra with four elements of Figure 2.7. If we use + for f, then $x = f(x,0)$ is satisfied. In addition, Figure 2.7 shows that $\mathcal{M}, x \models \varphi$. However, if we choose $x = f(0,x)$, then $\mathcal{M}, 0 \not\models \varphi$ and $\mathcal{M}, x \not\models E$ in Figure 2.7. Because for $x = f(x,e)$, if $\mathcal{M}, x \models \varphi$, then $\mathcal{M}, x \not\models \varphi$ and $\mathcal{M}, x \not\models E$.

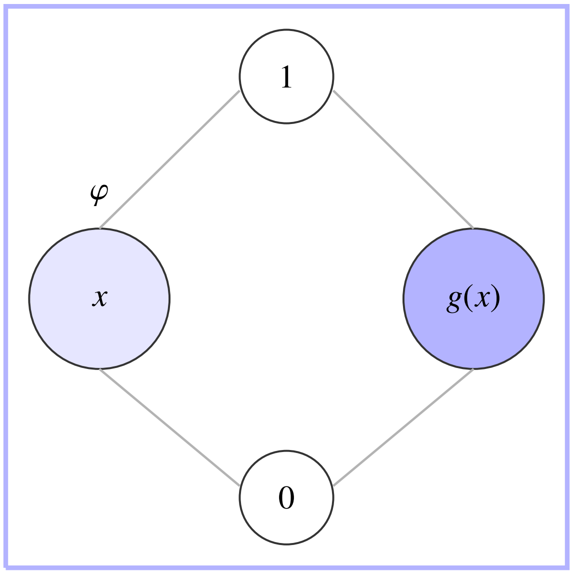

Figure 2.7: A PRLE-model on the Boolean algebra with four elements

Lemma 19. *The formula schema $\varphi \to (\varphi \mathbin{\overset{\circ}{\wedge}} U)$ is true in a PRLE-frame $\mathcal{F}$, iff $x = f^d(x, g(e))$ for all $x \in W$.*

Proof. This can be shown similar to Lemma 18 by using Lemma 11 and exchanging f with f^d. □

Please note that it is not the case that the formula schema $\varphi \to (\varphi \mathbin{\underset{\circ}{\vee}} U)$ is true in a PRLE-frame $\mathcal{F}$ iff $x = f^d(x, g(e))$ for all $x \in W$.

Lemma 20. *The formula schema $\varphi \mathbin{\overset{\circ}{\wedge}} (\psi \mathbin{⩓} \chi) \to (\varphi \mathbin{\overset{\circ}{\wedge}} \psi) \mathbin{⩓} (\varphi \mathbin{\overset{\circ}{\wedge}} \chi)$ is true in a PRLE-frame $\mathcal{F}$, iff f is distributive, i.e., $f^d(x, f(y, z)) = f(f^d(x, y), f^d(x, z))$ for all $x, y, z \in W$.*

Proof.

⇒ Assume $x, y, z \in W$. Then by the assumption the formula $\mathcal{F} \models p \mathbin{\overset{\circ}{\wedge}} (q \mathbin{⩓} r) \to (p \mathbin{\overset{\circ}{\wedge}} q) \mathbin{⩓} (p \mathbin{\overset{\circ}{\wedge}} r)$ for propositional variables p, q and r are true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be a model with $v(p) = \{x\}$, $v(q) = \{y\}$, $v(r) = \{z\}$. We want to show $\mathcal{M}, f^d(x, f(y, z)) \models p \mathbin{\overset{\circ}{\wedge}} (q \mathbin{⩓} r)$. Since $\mathcal{M}, y \models q$ and $\mathcal{M}, z \models r$ we get $\mathcal{M}, f(y, z) \models q \mathbin{⩓} r$. Furthermore, $\mathcal{M}, x \models$ p so that we conclude $\mathcal{M}, f^d(x, f(y, z)) \models p \mathbin{\overset{\circ}{\wedge}} (q \mathbin{⩓} r)$.

This completes the proof of $\mathcal{M}, f^d(x, f(y, z)) \models p \mathbin{\overset{\circ}{\wedge}} (q \mathbin{⩓} r)$ so that we conclude $\mathcal{M}, f^d(x,\ f(y, z)) \models (p \mathbin{\overset{\circ}{\wedge}} q) \mathbin{⩓} (p \mathbin{\overset{\circ}{\wedge}} r)$. This implies that there are $a, b \in W$ with $f^d(x, f(y, z)) = f(a, b)$ and $\mathcal{M}, a \models p \mathbin{\overset{\circ}{\wedge}} p$ and $\mathcal{M}, b \models p \mathbin{\overset{\circ}{\wedge}} r$. From the last property we obtain elements c, d with $b = f^d(c, d)$ and $\mathcal{M}, c \models p$ and $\mathcal{M}, d \models r$. By the definition of v we get $c = x$ and $d = z$ so that we have $f^d(x, f(y, z)) = f(a, f^d(x, z))$. Similar we obtain $a = f^d(x, y)$, i.e., we have $f^d(x, f(y, z)) = f(f^d(x, y), f^d(x, z))$.

$\Leftarrow$ Let $\mathcal{M}$ be a model and $w \in W$ so that we have to show $\mathcal{M}, w \models p \parr (q \Wedge r) \rightarrow (p \parr q) \Wedge (p \parr r)$. Therefore, assume $\mathcal{M}, w \models p \parr (q \Wedge r)$ so that we have to show $\mathcal{M}, w \models (p \parr q) \Wedge (p \parr r)$. From $\mathcal{M}, w \models p \parr (q \Wedge r)$ we obtain elements x, y, z with $w = f^d(x, f(y, z))$ and $\mathcal{M}, x \models p \parr (q \Wedge r)$. From the latter we get that $\mathcal{M}, f(f^d(x, y), f^d(x, z)) \models (p \parr p) \Wedge (p \parr r)$. Since f is distributive we have $w = f^d(x, f(y, z)) = f(f^d(x, y), f^d(x, z))$, and, hence, $\mathcal{M}, w \models (p \parr q) \Wedge (p \parr r)$.

□

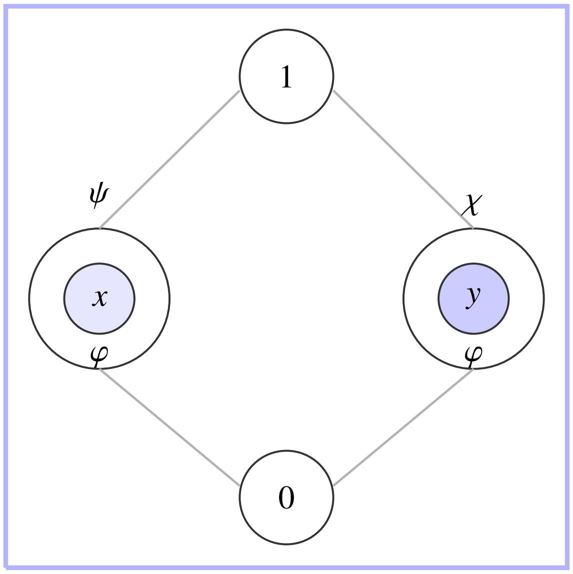

Figure 2.8: Distributive Boolean algebra Lattice (AND)

Please note that it is not the case that the opposite implication $(\varphi \parr \psi) \Wedge (\varphi \parr \chi) \rightarrow \varphi \parr (\psi \Wedge \chi)$ is true, it is described in Figure 2.8, as well as:

$$\mathcal{M}, 1 \models (\varphi \parr \psi) \Wedge (\varphi \parr \chi)$$
$$\mathcal{M}, 1 \not\models \varphi \parr (\psi \Wedge \chi)$$

Similarly, it is not true that the formulas schema $\varphi \binampersand (\psi \Vee \chi) \rightarrow (\varphi \binampersand \psi) \Vee (\varphi \binampersand \chi)$ is true iff f is distributive i.e., $f^d(x, f(y, z)) = f(f^d(x, y), f^d(x, z))$ for all $x, y, z \in W$, it is explained in Figure 2.9, as well as:

$$\mathcal{M}, 1 \models \varphi \binampersand (\psi \Vee \chi)$$
$$\mathcal{M}, 1 \not\models (\varphi \binampersand \psi) \Vee (\varphi \binampersand \chi)$$

Lemma 21. *The formula schema $\varphi \Wedge (\psi \parr \chi) \rightarrow (\varphi \Wedge \psi) \parr (\varphi \Wedge \chi)$ is true in a PRLE-frame $\mathcal{F}$, iff f^d is distributive, i.e., $f(x, f^d(y, z)) = f^d(f(x, y), f(x, z))$ for all $x, y \in W$.*

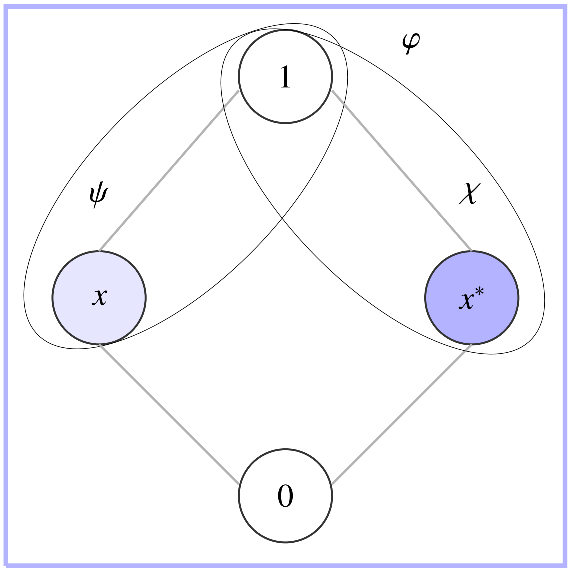

Figure 2.9: Distributive Boolean algebra Lattice (OR)

Proof. This can be shown similar to Lemma 16 by using Lemma 11 and exchanging f with f^d. □

Please note that it is not the case that the the formulas schema $\varphi \mathbin{\overset{\vee}{\circ}} (\psi \Vee \chi) \to (\varphi \mathbin{\overset{\vee}{\circ}} \psi) \Vee (\varphi \mathbin{\overset{\vee}{\circ}} \chi)$ is true iff f is distributive, i.e., $f^d(x, f(y,z)) = f(f^d(x,y), f^d(x,z))$ for all $x, y, z \in W$, it is explained in Figure 2.9.

Lemma 22. *If $g(g(x)) = x$ and $f^d(x, g(e)) = x$ for all $x \in W$, then the formula schema $\varphi \to \top \mathbin{\underset{\wedge}{\circ}} (U \wedge (\varphi \Wedge N\varphi))$ is true in a PRLE-frame $\mathcal{F}$, iff $f(x, g(x)) = g(e)$.*

Proof.

- $\Rightarrow$ Assume $x \in W$. Then by the assumption the formula $\mathcal{F} \models p \to \top \mathbin{\underset{\wedge}{\circ}} (U \wedge (p \Wedge Np))$ for propositional variable p is true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be a model with $v(p) = \{x\}$. Then we get $\mathcal{M}, x \models p \to \top \mathbin{\underset{\wedge}{\circ}} (U \wedge (p \Wedge Np))$. Since $\mathcal{M}, x \models p$ we get $\mathcal{M}, x \models \top \mathbin{\underset{\wedge}{\circ}} (U \wedge (\varphi \Wedge N\varphi))$. Then there are elements a, b with $x = f^d(a,b)$ and $\mathcal{M}, a \models \top$ and $\mathcal{M}, b \models U \wedge (\varphi \Wedge N\varphi)$. From $\mathcal{M}, b \models U$ we obtain $b = g(e)$ and from $\mathcal{M}, b \models p \Wedge Np$ we get to elements c, d with $b = f(c,d)$ and $\mathcal{M}, c \models p$ and $\mathcal{M}, d \models Np$. Using the definition of $v(p)$ get conclude that $c = x$ and $g(d) = x$. Together we obtain $g(e) = b = f(c,d) = f(c, g(g(d))) = f(x, g(x))$.

- $\Leftarrow$ Assume $w \in W$ with $\mathcal{M}, w \models \varphi$. Then $\mathcal{M}, g(w) \models N\varphi$ so that we have $\mathcal{M}, f(w, g(w)) \models \varphi \Wedge N\varphi$. Since $g(e) = f(w, g(w))$ by the assumption we obtain $\mathcal{M}, g(e) \models U \wedge (\varphi \Wedge N\varphi)$. This implies $\mathcal{M}, f^d(x, g(e)) \models \top \mathbin{\underset{\wedge}{\circ}} (U \wedge (\varphi \Wedge N\varphi))$. Finally, $f^d(x, g(e)) = x$ shows that $\mathcal{M}, x \models \top \mathbin{\underset{\wedge}{\circ}} (U \wedge (\varphi \Wedge N\varphi))$.

□

Lemma 23. *If $g(g(x)) = x$ and $f^d(x, g(e)) = x$ for all $x \in W$, then the formula schema $\varphi \to \top \Wedge (E \wedge (\varphi \parr N\varphi))$ is true in a PRLE-frame $\mathcal{F}$, iff $f^d(x, g(x)) = e$ for all $x \in W$.*

Proof. This can be shown similar to Lemma 22 by using Lemma 11 and exchanging f with f^d. □

Theorem 1. *Let W be the universe of PRLE-frame. Then W with the operations defined by*

1. $x + y = f(x, y)$,
2. $x \cdot y = f^d(x, y)$,
3. $x^* = g(x)$,
4. $0 = e$ *and*
5. $1 = g(e)$

is a Boolean algebra iff the axiom schemas

Commutativity of $\Wedge$	$\varphi \Wedge \psi \to \psi \Wedge \varphi$
Commutativity of $\Vee$	$\varphi \Vee \psi \to \psi \Vee \varphi$
Identity for $\Wedge$	$\varphi \to \varphi \Wedge E$
Identity for $\parr$	$\varphi \to \varphi \parr U$
Distributivity of $\Wedge$ *on* $\parr$	$\varphi \Wedge (\psi \parr \chi) \to (\varphi \Wedge \psi) \parr (\varphi \Wedge \chi)$
Distributivity of $\parr$ *on* $\Wedge$	$\varphi \parr (\psi \Wedge \chi) \to (\varphi \parr \psi) \Wedge (\varphi \parr \chi)$
Complements for $\Wedge$	$\varphi \to \top \parr (U \wedge (\varphi \Wedge N\varphi))$
Complements for $\parr$	$\varphi \to \top \Wedge (E \wedge (\varphi \parr N\varphi))$

are valid. We call a such frame a Boolean propositional relevance with E logic frame (BPRLE-frame).

Please note that we will use $(0, 1, +, \cdot, *)$ instead of $(e, g(e), f, f^d, g)$ for any BPRLE -frame in the remainder of this thesis.

2.8.4 Boolean Algebras in Original PRLE Language

In this section, we want to represent the axioms used in Theorem 1 without using abbreviations. Since PRLE language is a combination of propositional, relevance and propositional relevance logic with E, our minimum set of operators and atomic formulas is ($\bot$, $\neg$, $\to$, $\twoheadrightarrow$, $\backsim$, E).

2.8.4.1 Commutativity Axiom

We are going to represent the commutativity axioms from Theorem 1 in original PRLE language as follows:

1. Commutativity of $\Wedge$:

$$\begin{aligned} \varphi \Wedge \psi \to \psi \Wedge \varphi &\Leftrightarrow \neg(\varphi \to \neg\psi) \to \neg(\psi \twoheadrightarrow \neg\varphi) && \text{by Definition} \\ &\Leftrightarrow (\psi \twoheadrightarrow \neg\varphi) \to (\varphi \twoheadrightarrow \neg\psi) && \text{propositional logic} \end{aligned}$$

   We may rewrite commutativity of $\Wedge$ axiom by the following axiom:

   $(\psi \twoheadrightarrow \neg\varphi) \to (\varphi \twoheadrightarrow \neg\psi)$.

2. Commutativity of $\Vee$:

$$\varphi \Wedge \psi \to \psi \Vee \varphi \Leftrightarrow \neg\varphi \twoheadrightarrow \psi \to \neg\psi \twoheadrightarrow \varphi \qquad \text{by Definition}$$

   Similarly, we may rewrite commutativity of $\Vee$ axiom by the following axiom:

   $(\neg\varphi \twoheadrightarrow \psi) \to (\neg\psi \twoheadrightarrow \varphi)$.

2.8.4.2 Identity Axiom

Now we going to represent the identity axioms from Theorem 1 in original PRLE language as follows:

1. Identity for $\Wedge$:

$$\varphi \to \varphi \Wedge E \Leftrightarrow \varphi \to \neg(\varphi \twoheadrightarrow \neg E) \qquad \text{by Definition}$$

   We may rewrite identity for $\Wedge$ axiom by the following axiom:

   $\varphi \to \neg(\varphi \twoheadrightarrow \neg E)$.

2. Identity for $\parr$:

$$\begin{aligned}
\varphi \rightarrow \varphi \parr U &\Leftrightarrow \varphi \rightarrow\backsim (N\varphi \twoheadrightarrow\backsim U) && \text{by Definition}\\
&\Leftrightarrow \varphi \rightarrow\backsim (\backsim \neg\varphi \twoheadrightarrow\backsim U) && \text{by Definition}\\
&\Leftrightarrow \varphi \rightarrow\backsim (\backsim \neg\varphi \twoheadrightarrow\backsim NE) && \text{by Definition}\\
&\Leftrightarrow \varphi \rightarrow\backsim (\backsim \neg\varphi \twoheadrightarrow\backsim\backsim \neg E) && \text{by Definition}\\
&\Leftrightarrow \varphi \rightarrow\backsim (\backsim \neg\varphi \twoheadrightarrow \neg E) && \text{by Definition}
\end{aligned}$$

Similarly, we may rewrite identity for $\parr$ axiom by the following axiom:

$\varphi \rightarrow\backsim (\backsim \neg\varphi \twoheadrightarrow \neg E)$.

2.8.4.3 Distributivity Axiom

Now we going to represent the distributivity axioms from Theorem 1 in original PRLE language as follows:

1. Distributivity of $\Wedge$ on $\parr$:

$$\begin{aligned}
&\varphi \parr (\psi \Wedge \chi) \rightarrow (\varphi \parr \psi) \Wedge (\varphi \parr \chi)\\
&\Leftrightarrow\backsim (N\varphi \twoheadrightarrow\backsim (\psi \Wedge \chi)) \rightarrow\backsim (N\varphi \twoheadrightarrow\backsim \psi)\Wedge \backsim (N\varphi \twoheadrightarrow\backsim \chi) && \text{by Definition}\\
&\Leftrightarrow\backsim (N\varphi \twoheadrightarrow\backsim \neg(\psi \twoheadrightarrow \neg\chi)) \rightarrow && \text{by Definition}\\
&\quad \neg(\backsim (N\varphi \twoheadrightarrow\backsim \psi) \twoheadrightarrow \neg \backsim (N\varphi \twoheadrightarrow\backsim \chi))\\
&\Leftrightarrow\backsim (\backsim \neg\varphi \twoheadrightarrow\backsim \neg(\psi \twoheadrightarrow \neg\chi)) \rightarrow && \text{by Definition}\\
&\quad \neg(\backsim (\backsim \neg\varphi \twoheadrightarrow\backsim \psi) \twoheadrightarrow \neg \backsim (\backsim \neg\varphi \twoheadrightarrow\backsim \chi))\\
&\Leftrightarrow\backsim (\varphi^{'} \twoheadrightarrow\backsim \neg(\psi \twoheadrightarrow \neg\chi)) \rightarrow && \varphi^{'} :=\backsim \neg\varphi\\
&\quad \neg(\backsim (\varphi^{'} \twoheadrightarrow\backsim \psi) \twoheadrightarrow \neg \backsim (\varphi^{'} \twoheadrightarrow\backsim \chi))
\end{aligned}$$

We may rewrite distributivity of $\Wedge$ on $\parr$ axiom by the following axiom:

$\backsim (\varphi^{'} \twoheadrightarrow\backsim \neg(\psi \twoheadrightarrow \neg\chi)) \rightarrow \neg(\backsim (\varphi^{'} \twoheadrightarrow\backsim \psi) \twoheadrightarrow \neg \backsim (\varphi^{'} \twoheadrightarrow\backsim \chi))$.

2. Distributivity of $\parr$ on $\Wedge$:

$$\begin{aligned}
&\varphi \Wedge (\psi \parr \chi) \rightarrow (\varphi \Wedge \psi) \parr (\varphi \Wedge \chi)\\
&\Leftrightarrow \neg(\varphi \twoheadrightarrow \neg(\psi \parr \chi)) \rightarrow \neg(\varphi \twoheadrightarrow \neg\psi) \parr \neg(\varphi \twoheadrightarrow \neg\chi) && \text{by Definition}\\
&\Leftrightarrow \neg(\varphi \twoheadrightarrow \neg \backsim (N\psi \twoheadrightarrow\backsim \chi)) \rightarrow && \text{by Definition}
\end{aligned}$$

$$\begin{aligned}
&\backsim (N\neg(\varphi \twoheadrightarrow \neg\psi) \twoheadrightarrow\backsim \neg(\varphi \twoheadrightarrow \neg\chi)) && \\
\Leftrightarrow\ &\neg(\varphi \twoheadrightarrow \neg \backsim (\backsim \neg\psi \twoheadrightarrow\backsim \chi)) \rightarrow && \text{by Definition} \\
&\backsim (\backsim \neg\neg(\varphi \twoheadrightarrow \neg\psi) \twoheadrightarrow\backsim \neg(\varphi \twoheadrightarrow \neg\chi)) && \\
\Leftrightarrow\ &\neg(\varphi \twoheadrightarrow \neg \backsim (\backsim \neg\psi \twoheadrightarrow\backsim \chi)) \rightarrow && \text{propositional logic} \\
&\backsim (\backsim (\varphi \twoheadrightarrow \neg\psi) \twoheadrightarrow\backsim \neg(\varphi \twoheadrightarrow \neg\chi)) &&
\end{aligned}$$

Similarly, we may rewrite identity for $\invamp$ axiom by the following axiom:

$\neg(\varphi \twoheadrightarrow \neg \backsim (\backsim \neg\psi \twoheadrightarrow\backsim \chi)) \rightarrow\backsim (\backsim (\varphi \twoheadrightarrow \neg\psi) \twoheadrightarrow\backsim \neg(\varphi \twoheadrightarrow \neg\chi))$.

2.8.4.4 Complement Axiom

Now we going to represent the complement axioms from Theorem 1 in original PRLE language as follows:

1. Complement for $\doublewedge$:

$$\begin{aligned}
&\varphi \rightarrow \top \invamp (U \wedge (\varphi \doublewedge N\varphi)) && \\
\Leftrightarrow\ &\varphi \rightarrow N(N\top \doublewedge N(U \wedge (\varphi \doublewedge N\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow N\neg(N\top \twoheadrightarrow \neg N(U \wedge (\varphi \doublewedge N\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow N\neg(N\top \twoheadrightarrow \neg N(U \wedge \neg(\varphi \twoheadrightarrow \neg N\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow\backsim \neg\neg(\backsim \neg\top \twoheadrightarrow \neg \backsim \neg(U \wedge \neg(\varphi \twoheadrightarrow \neg \backsim \neg\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow\backsim \neg\neg(\backsim \neg\neg\bot \twoheadrightarrow \neg \backsim \neg(NE \wedge \neg(\varphi \twoheadrightarrow \neg \backsim \neg\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow\backsim (\backsim \bot \twoheadrightarrow \neg \backsim \neg(\backsim \neg E \wedge \neg(\varphi \twoheadrightarrow \neg \backsim \neg\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow\backsim (\backsim \bot \twoheadrightarrow \neg \backsim \neg\neg(\backsim \neg E \rightarrow \neg\neg(\varphi \twoheadrightarrow \neg \backsim \neg\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow\backsim (\backsim \bot \twoheadrightarrow \neg \backsim (\backsim \neg E \rightarrow (\varphi \twoheadrightarrow \neg \backsim \neg\varphi))) && \text{Prop. logic}
\end{aligned}$$

We may rewrite complement for $\doublewedge$ axiom by the following axiom:

$\varphi \rightarrow\backsim (\backsim \bot \twoheadrightarrow \neg \backsim (\backsim \neg E \rightarrow (\varphi \twoheadrightarrow \neg \backsim \neg\varphi)))$.

2. Complement for $\invamp$:

$$\begin{aligned}
&\varphi \rightarrow \top \doublewedge (E \wedge (\varphi \invamp N\varphi)) && \\
\Leftrightarrow\ &\varphi \rightarrow \neg(\top \twoheadrightarrow \neg(E \wedge (\varphi \invamp N\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow \neg(\top \twoheadrightarrow \neg(E\wedge \backsim (N\varphi \twoheadrightarrow\backsim N\varphi))) && \text{by Definition} \\
\Leftrightarrow\ &\varphi \rightarrow \neg(\top \twoheadrightarrow \neg(E\wedge \backsim (\backsim \neg\varphi \twoheadrightarrow\backsim\backsim \neg\varphi))) && \text{by Definition}
\end{aligned}$$

$\Leftrightarrow \varphi \to \neg(\top \twoheadrightarrow \neg(E \wedge \backsim (\backsim \neg\varphi \twoheadrightarrow \neg\varphi)))$	by Definition
$\Leftrightarrow \varphi \to \neg(\neg\bot \twoheadrightarrow \neg\neg(E \to\backsim \neg(\backsim \neg\varphi \twoheadrightarrow \neg\varphi)))$	by Definition
$\Leftrightarrow \varphi \to \neg(\neg\bot \twoheadrightarrow (E \to\backsim \neg(\backsim \neg\varphi \twoheadrightarrow \neg\varphi)))$	Prop. logic

Similarly, we may rewrite complement for $\overset{\circ}{\wedge}$ axiom by the following axiom:

$\varphi \to \neg(\neg\bot \twoheadrightarrow (E \to\backsim \neg(\backsim \neg\varphi \twoheadrightarrow \neg\varphi)))$.

2.9 Modal Relevance Logic (MRL)

In this section, we are going to introduce a new logic namely modal relevance logic by adding the properties of modal logic with the propositional relevance logic with E. Later in this section, we will provide axioms concerning the modal operator that forces every frame to be a Boolean contact algebra. The logics and they relationships used in this thesis are shown in Figure 2.10.

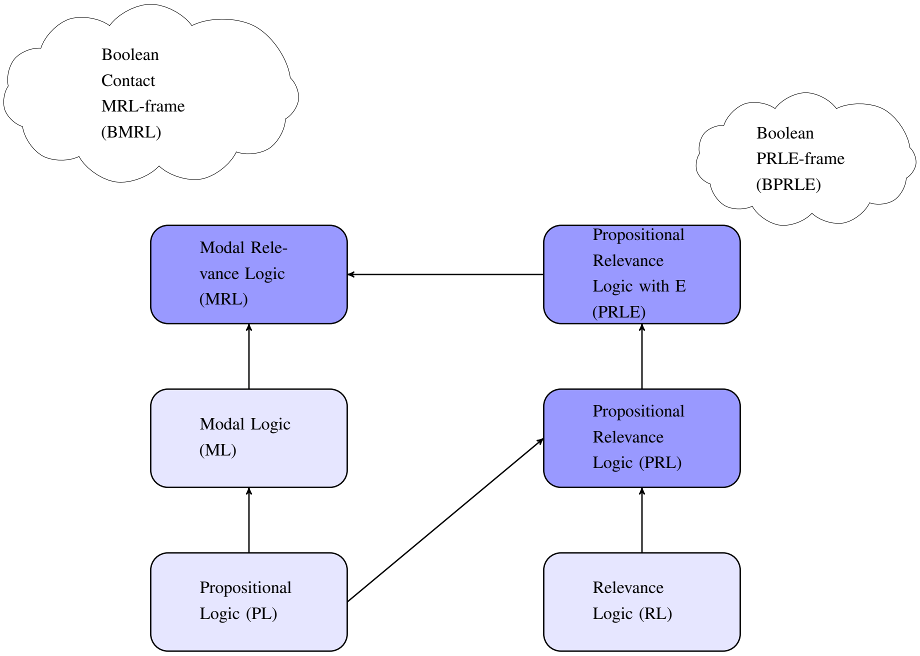

Figure 2.10: Inclusion of Logics Introduced

Now we are going to introduce the syntax and semantics of modal relevance logic in the following sections.

2.9.1 Syntax

Since we are not introducing any new operators, the syntax can be defined in terms of the rules introduced earlier. The syntax of modal relevance logic is formally defined as follows:

Definition 25. (Modal relevance logic syntax) *The set MRL of modal relevance logic formulas is recursively defined by the rules (PropL.1)- (PropL.3), (RelL.1) - (RelL.2), (PRLE.1) and (ModL.1).*

2.9.2 Semantics

Similar to the previous logics we are going to define frames and models.

Definition 26. (Modal relevance logic frame) *A modal relevance logic MRL-frame* $\mathcal{F} = \langle W, e, R, f, g \rangle$ *is a structure such that:*

(MLF.1) $\langle W, e, f, g \rangle$ *is a PRLE-frame and*

(MLF.2) R *is a binary relation on* W.

In the case that $\langle W, e, f, g \rangle$ *is a BPRLE-frame. Then we call* $\mathcal{F}$ *a BMRL-frame.*

The definition of a modal relevance logic model $\mathcal{M}$ is again similar to Definition 10. Now we are going to formally define the semantics of modal relevance logic as follows:

Definition 27. (Modal relevance logic semantics) *Let* $\mathcal{M}$ *be a model,* $x, y, z \in W$ *be a state and* $\varphi \in MRL$*. The satisfaction relation* $\mathcal{M}, x \models \varphi$ *is recursively defined by (SemPropL.1) - (SemPropL.3), (SemRelL.1)-(SemRelL.2), (SemPRLE.1) and (SemModL.1).*

2.9.3 BMRL-frame Axioms

In this section, we are going to provide axioms that are equivalent to null disconnectedness, reflexivity, symmetry, compatibility, and summation property of relation R in a BMRL-frame.

Lemma 24. *The formula* $[R]\neg E$ *is true in a BMRL-frame* $\mathcal{F}$*, iff* xRy *implies* $y \neq 0$ *for all* $x, y \in W$.

Proof. Let $\mathcal{M}$ be any model based on $\mathcal{F}$ and $x \in W$. Then $\mathcal{M}, x \models [R]\neg E$. This implies that if xRy, then $\mathcal{M}, y \models \neg E$ for all $y \in W$. Therefore, we conclude if xRy, then $y \neq 0$ for all $y \in W$. □

Lemma 25. *The formula schema $\neg E \to ([R]\varphi \to \varphi)$ is true in a BMRL-frame $\mathcal{F}$, iff xRx for all $x \in W$ with $x \neq 0$.*

Proof.

- ⇒ Let be $x \in W$ with $x \neq 0$. Assume that xRx is false. By assumption the formula $\mathcal{F} \models \neg E \to ([R]p \to p)$ for propositional variable p is true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be a model with $v(p) = W \setminus \{x\}$. Then we get $\mathcal{M}, x \models \neg E \to ([R]p \to p)$. Since $x \neq 0$ have $\mathcal{M}, x \models \neg E$, and, hence, $\mathcal{M}, x \models [R]p \to p$. Now, if xRy, then $x \neq y$ by the assumption, and, hence, $\mathcal{M}, y \models p$. This shows that $\mathcal{M}, x \models [R]p$ so that we conclude $\mathcal{M}, x \models p$, a contradiction to the definition of $v(p)$. Therefore, the assumption is wrong and xRx holds.

- ⇐ Assume $\mathcal{M}, x \models \neg E$ and xRy implies $\mathcal{M}, y \models \varphi$ for all $y \in W$. Then $x \neq 0$, therefore, xRx by the assumption. Hence $\mathcal{M}, x \models \varphi$.

□

Lemma 26. *The formula schema $\varphi \to [R]\langle R\rangle\varphi$ is true in a BMRL-frame $\mathcal{F}$, iff xRy implies yRx for all $x, y \in W$.*

Proof.

- ⇒ Assume $x, y \in W$ with xRy. Then by the assumption the formula $p \to [R]\langle R\rangle p$ for a propositional variable p is true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be such a model with $v(p) = \{x\}$. Then $\mathcal{M}, x \models p$ and, hence, $\mathcal{M}, x \models [R]\langle R\rangle p$. Since we have xRy we obtain $\mathcal{M}, y \models \langle R\rangle p$. Therefore, there is a $z \in W$ with yRz and $\mathcal{M}, z \models p$. Since $v(p) = \{x\}$ we get $z = x$ and, hence, yRx.

- ⇐ Let $\mathcal{M}$ be a model and $x \in W$ so that we have to show $\varphi \to [R]\langle R\rangle\varphi$. Therefore, assume $\mathcal{M}, x \models \varphi$ so that we have to show $\mathcal{M}, x \models [R]\langle R\rangle\varphi$. Now, assume $y \in W$ with xRy so that we have to show $\mathcal{M}, y \models \langle R\rangle\varphi$. But this is true since yRx follows from the assumption and we have $\mathcal{M}, x \models \varphi$.

□

Lemma 27. *The formula schema $[R]\varphi \to [R](\top \multimap \varphi)$ is true in a BMRL-frame $\mathcal{F}$, iff xRy and $y \leq z$ implies xRz for all $x, y, z \in W$.*

Proof.

- $\Rightarrow$ Assume $x, y, z \in W$ with xRy and $y \leq z$. Then by the assumption the formula $[R]p \rightarrow [R](\top \multimap p)$ for a propositional variable p is true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be a model with $v(p) = \{y \mid xRy\}$. Then $\mathcal{M}, x \models [R]p$, and, hence, $\mathcal{M}, x \models [R](\top \multimap p)$. Since xRy we get $\mathcal{M}, y \models \top \multimap p$. From $y \leq z$ we get $y = y + z$. Furthermore, we have $\mathcal{M}, y \models \top$ so that we conclude $\mathcal{M}, z \models p$, i.e., xRz.

- $\Leftarrow$ Let $\mathcal{M}$ be a model and $x \in W$ so that we have to show $\mathcal{M}, x \models [R]\varphi \rightarrow [R](\top \multimap p)$. Assume $\mathcal{M}, x \models [R]\varphi$,i.e., xRy implies $\mathcal{M}, y \models \varphi$ for all y, so that we have to show $\mathcal{M}, x \models [R](\top \multimap p)$. Now, assume xRy so that we have to show $\mathcal{M}, y \models \top \multimap \varphi$. Therefore, assume u, z with $y = u \cdot z$ and $\mathcal{M}, u \models \top$ so that we have to show $\mathcal{M}, z \models \varphi$. This implies $y \leq z$ from which we conclude xRz by the assumption. From $\mathcal{M}, x \models [R]\varphi$ we obtain $\mathcal{M}, z \models \varphi$.

□

Lemma 28. *If xRy implies yRx for all $x, y \in W$, then the formula schema $\varphi \rightarrow [R](\neg\langle R\rangle\varphi \twoheadrightarrow \langle R\rangle\varphi)$ is true in a BMRL-frame $\mathcal{F}$, iff $xR(y + z)$ implies xRy or xRz for all $x, y, z \in W$.*

Proof.

- $\Rightarrow$ Assume $x, y, z \in W$ with $xR(y + z)$. Then by the assumption the formula $p \rightarrow [R](\neg\langle R\rangle p \twoheadrightarrow \langle R\rangle p)$ for a propositional variable p is true in all models based on $\mathcal{F}$. Let $\mathcal{M}$ be a model with $v(p) = \{x\}$. Then $\mathcal{M}, x \models p$, and, hence, $\mathcal{M}, x \models [R](\neg\langle R\rangle p \twoheadrightarrow \langle R\rangle p)$. Since $xR(y + z)$ we obtain, $\mathcal{M}, y \models \neg\langle R\rangle p$ implies $\mathcal{M}, z \models \langle R\rangle p$, which is equivalent to $\mathcal{M}, y \models \langle R\rangle p$ or $\mathcal{M}, z \models \langle R\rangle p$. If $\mathcal{M}, y \models \langle R\rangle p$, then there is a u with yRu and $u = x$, i.e., yRx. This implies xRy by the symmetry of R. The case $\mathcal{M}, z \models \langle R\rangle p$ follows analogously.

- $\Leftarrow$ Let $\mathcal{M}$ be a model and $x \in W$ so that we have to show $\varphi \rightarrow [R](\neg\langle R\rangle\varphi \twoheadrightarrow \langle R\rangle\varphi)$. Assume $\mathcal{M}, x \models \varphi$ so that we have to show $\mathcal{M}, x \models [R](\neg\langle R\rangle\varphi \twoheadrightarrow \langle R\rangle\varphi)$. Now assume $xR(y + z)$ so that we have to show $\mathcal{M}, y \models \neg\langle R\rangle\varphi$ implies $\mathcal{M}, z \models \langle R\rangle\varphi$. This is equivalent to $\mathcal{M}, y \models \langle R\rangle\varphi$ or $\mathcal{M}, z \models \langle R\rangle\varphi$. From $xR(y + z)$ and the assumption we get xRy or xRz. If xRy, then yRx by symmetry. Therefore, $\mathcal{M}, y \models \langle R\rangle\varphi$ since $\mathcal{M}, x \models \varphi$. The case xRz follows analogously.

□

Theorem 2. *Let $\mathcal{F}$ be a BMRL-frame. Then R is a contact relation iff the following axioms:*

Null disconnected (BCAx0) $[R]\neg E$

Reflexivity (BCAx1)	$\neg E \to ([R]\varphi \to \varphi)$
Symmetry (BCAx2)	$\varphi \to [R]\langle C\rangle\varphi$
Compatibility (BCAx3)	$[R]\varphi \to [R](\top \multimap \varphi)$
Summation (BCAx4)	$\varphi \to [R](\neg\langle R\rangle\varphi \twoheadrightarrow \langle R\rangle\varphi)$

are valid in $\mathcal{F}$. We call such a frame a Boolean contact algebra frame (BCA-frame).

Please note that we will use C instead of R for any BCA-frame in the remainder of this thesis.

Chapter 3

Natural Deduction

Natural deduction is a logic calculus to derive valid formulas. It is constructed so that proofs use rules similar to the natural way of reasoning. A proof in natural deduction constructed as a tree and there are three components, premises or initial assumptions, rules of inference and conclusion.

For example, in Figure 3.1 the outer most nodes ○ denote initial assumptions, the intermediate nodes ○ denote intermediate formulas generated after applying some rules and the bottom-most node ● denote the conclusion i.e., the formula to be proven.

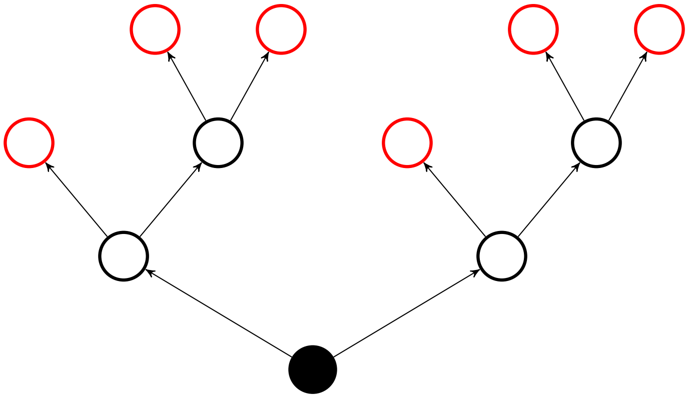

Figure 3.1: Natural Deduction Proof Tree

In our version of natural deduction used in this thesis, we consider two types of assumptions. One is an open assumption, and another one is discarded assumption. Open assumptions represent those are yet to use in the proof subtree after the rule is applied. Whereas discard assumptions represent those used by a certain rule applied in the proof tree. Then $\varphi_1, \ldots, \varphi_n \vdash \psi$ denotes the fact that there is a proof tree with open assumptions among $\varphi_1, \ldots, \varphi_n$ and conclusion ψ.

Let assume $\{\varphi_1, \ldots, \varphi_n\}$ are the set of formulas are given, and in the first step after applying a rules on the premises a new formula ψ_1 is generated as a intermediate output and it is denoted by:

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \psi_1.$$

In the next step, let assume another rule is applied and it generate another new formula ψ_2 and it is denoted by:

$$\varphi_1, \varphi_2, \ldots, \varphi_n, \psi_1 \vdash \psi_2.$$

Let assume the proof tree grows after applying few more rules and we reach to intermediate formulas ψ_3 and ψ_4 before the conclusion ψ and it is denoted by

$$\varphi_1, \varphi_2, \ldots, \varphi_n, \psi_1, \psi_2, \psi_3, \psi_4 \vdash \psi.$$

It is worth mentioning that natural deduction used in this thesis works with annotated formulas, i.e., formulas of the form φ_t where φ is an MRL-formula and t is a term. Also, note that also we call t the index of the formula φ written as φ_t. In the following definitions, we define the term environment and evaluation of term of natural deduction formally.

Definition 28. (Term) *Let $\mathcal{F}$ be a BMRL-frame, X be a set of variables. Then the set $Term$ is recursively defined by:*

1. *$x \in X$ is a term. i.e., $X \subseteq Term$,*
2. *$\alpha,\beta \in Term$, then*
   - (a) *$\alpha + \beta \in Term$, i.e., $n[\alpha + \beta] = n[\alpha] + n[\beta]$,*
   - (b) *$\alpha \cdot \beta \in Term$, i.e., $n[\alpha \cdot \beta] = n[\alpha] \cdot n[\beta]$ and*
   - (c) *$\alpha^* \in Term$ for all $\alpha \in Term$.*

It is worth mentioning that we will use $x, y, z, \ldots$ to denote variables and $\alpha, \beta, \gamma, \ldots$ to denote terms in the remainder of this thesis.

Definition 29. (Environment) *Let $\mathcal{F}$ be a BMRL-frame and X be a set of variables. An environment $n : X \to W$ is a function from X to W.*

It is worth mentioning that in our version of the natural deduction calculus we have three kinds of formulas in the proof tree: (a) annotated formulas, (b) equation of the form $\alpha = \beta$ for terms α and β and (c) contact relation $\alpha C \beta$ for terms α and β. Now, we are going to define the evalution of term as follows:

Definition 30. (Evaluation of term) *Let $\mathcal{F}$ be a BMRL-frame and X a set of variables, and n an environment. Then the evaluation $n[\alpha]$ of a term α in the environment n is recursively defined by:*

1. *$n[x] = n(x)$ for all $x \in X$,*
2. *If $\alpha, \beta \in Term$, then*
   - (a) *$n[\alpha + \beta] = n[\alpha] + n[\beta]$,*
   - (b) *$n[\alpha \cdot \beta] = n[\alpha] \cdot n[\beta]$ and*
   - (c) *$n[\alpha^*] = n[\alpha]^*$*
3. *If $\alpha, \beta \in Term$, then*
   - (a) *$n[\alpha] = n[\beta] = \alpha = \beta$ and*
   - (b) *$n[\alpha C \beta] = n[\alpha] C n[\beta]$.*

It is worth mentioning that we will call φ a formula if it is either one of the above. Also note that an annotated formula is a formula in MRL indexed by a term. Now, we are going to define the validity of annotated formulas as follow:

Definition 31. (Validity evaluation) *Let $\mathcal{M}$ be a model, $\varphi \in MRL$, $\alpha \in Term$ and Γ a set of annotated formulas. Then we have:*

$$
\begin{array}{lll}
1.\ \mathcal{M}, n & \models \varphi_\alpha & \Longleftrightarrow \mathcal{M}, n[\alpha] \models \varphi, \\
2.\ \mathcal{M}, n & \models \alpha = \beta & \Longleftrightarrow n[\alpha] = n[\beta], \\
3.\ \mathcal{M}, n & \models \alpha C \beta & \Longleftrightarrow n[\alpha] C n[\beta], \\
4.\ \mathcal{M} & \models \varphi_\alpha & \Longleftrightarrow \mathcal{M}, n \models \varphi_\alpha \textit{ for all } n, \\
5. & \models \varphi_\alpha & \Longleftrightarrow \mathcal{M} \models \varphi_\alpha \textit{ for all } \mathcal{M}, \textit{ and} \\
6.\ \Gamma & \models \varphi_\alpha & \Longleftrightarrow \textit{for all } \mathcal{M}, n \textit{ we have that } \mathcal{M}, n \models \chi_\beta \\
 & & \Longleftrightarrow \textit{for all } \chi_\beta \in \Gamma \textit{ implies } \mathcal{M}, n \models \varphi_\alpha \\
 & & \Longleftrightarrow \mathcal{M}, n \models \chi_\beta \textit{ implies } \mathcal{M}, n \models \varphi_\alpha \textit{ for all } \mathcal{M} \textit{ and } n.
\end{array}
$$

In the remainder of this thesis we will use index for all formulas except for $\perp$, because for all $\alpha, \beta \in Term$ we have

$$\mathcal{M}, n \models \perp_\alpha \Leftrightarrow \mathcal{M}, n[\alpha] \models \mathcal{M}, n[\beta] \models \perp \iff \mathcal{M}, n \models \perp_\beta$$

In the following lemma, we want to show the property of the extended environment.

Lemma 29. *Let $\alpha \in Term$, $\varphi \in BMRL$ be a formula and $\mathcal{M}$ be a model. Then we have:*

1. *If n_1 and n_2 coincide on all variables in α, then $n_1[\alpha] = n_2[\alpha]$,*
2. *If n_1 and n_2 coincide on all variables in α, then $\mathcal{M}, n_1 \models \varphi_\alpha$ iff $\mathcal{M}, n_2 \models \varphi_\alpha$.*

Proof.

(1) The proof is done by induction on the definition of a term (Definition 28).

(a) Assume $\alpha = x$ is a variable, then we get
$n_1[\alpha] = n_1(x) = n_2(x) = n_2[\alpha]$.

(b) Assume $\alpha = \beta + \gamma$ is a term, then we get

$$\begin{aligned} n_1[\alpha] &= n_1[\beta + \gamma] = n_1[\beta] + n_1[\gamma] && \text{by Definition 28} \\ &= n_2[\beta] + n_2[\gamma] = n_2[\beta + \gamma] = n_2[\alpha] && \text{by Definition 28} \end{aligned}$$

(c) Assume $\alpha = \beta \cdot \gamma$ is a term, then we get

$$\begin{aligned} n_1[\alpha] &= n_1[\beta \cdot \gamma] = n_1[\beta] \cdot n_1[\gamma] && \text{by Definition 28} \\ &= n_2[\beta] \cdot n_2[\gamma] = n_2[\beta \cdot \gamma] = n_2[\alpha] && \text{by Definition 28} \end{aligned}$$

(d) Assume $\alpha = x^*$ is a variable, then we get

$$\begin{aligned} n_1[\alpha] &= n_1(x^*) && \text{by Definition 28} \\ &= n_2(x^*) = n_2[\alpha] && \text{by Definition 28} \end{aligned}$$

(e) Assume α and β terms are equal, then we get

$$\begin{aligned} n_1[\alpha] &= n_1[\beta] && \text{by Definition 28} \\ &= n_2[\alpha] = n_2[\beta] && \text{by Definition 28} \end{aligned}$$

(f) Assume α and β terms are in contact, then we get
$n_1[\alpha C\beta] = n_2[\alpha C\beta]$.

(2) Using (1) this follows immediately from

$$\begin{aligned} \mathcal{M}, n_1 \models \varphi_\alpha &\iff \mathcal{M}, n_1[\alpha] \models \varphi && \text{by Definition 31} \\ &\iff \mathcal{M}, n_2[\alpha] \models \varphi && \text{by (1)} \\ & \mathcal{M}, n_2 \models \varphi_\alpha && \text{by Definition 31} \end{aligned}$$

□

3.1 Rules of Natural Deduction

Our version of natural deduction has some axioms, i.e., rules that do not have any subtrees, or equivalently, Leaves that are not assumption and do not need to be discarded. Those axioms are instantiations of the BA axioms, i.e., that equation where the variables have been instantiated with certain terms.

The inference rules of a natural deduction system are based on the logical operators described in Chapter 2. Therefore, we consider rules for the abbreviations as well. In total, we have 18 logical operators of two types. One is basic logical operators, and another one is negation style operators, they are as follows:

1. Basic logical operators are $=, \rightarrow, \wedge, \vee, \leftrightarrow, \twoheadrightarrow, \Wedge, \Vee, \multimap, \overset{\circ}{\wedge}, \underset{\circ}{\vee}, E, U, [C], \langle C\rangle$ and
2. Negation style operators are $\neg, \backsim, N$.

For each basic logical operator, there are two rules. One is introduction rule and the other one is an elimination rule. Inference rules that introduce the logical operator in the proof tree is called introduction rule, usually denoted by I. Whereas elimination rules eliminate the logical operator in the proof tree, usually denoted by E. For the negation operators there is an extra PBC (proof by contradiction) rule that is neither an introduction nor an elimination rule. Furthermore, we have five rules that correspond to BCA_0 - BCA_4 for contact that are also neither introduction nor elimination rules.

In this thesis assumptions we will denote the fact that the assumption φ_α is discarded by $[\varphi_\alpha]$. Please note that certain rules such as $\rightarrow$ I (see below) will discard certain assumptions in the proof tree, i.e., there are not considered assumption of the proof tree after the rule has been applied.

3.1.1 Rules for the Equality

The rules are listed in Table 3.1.

Table 3.1: Rules for Equality

Name	**Rule**	**Condition**
= I	$\dfrac{}{\alpha = \alpha}\ (= I)$	
= E	$\dfrac{\alpha = \beta \quad \varphi_\alpha}{\varphi_\beta}\ (= E)$	

3.1.2 Rules for the Propositional Operators

The proof rules for the propositional operators are exactly same as they are defined in any standard literature on propositional logic. However, we have used an index with the formula to fit the purpose of this thesis. Please note that all of these rules will always assume that all indices are equal, and they will never change any index. The rules are listed in Table 3.2.

Table 3.2: Rules for the Propositional Operators

Name	**Rule**	**Condition**
¬ I	$\dfrac{\begin{matrix}[\varphi_\alpha] \\ \vdots \\ \bot\end{matrix}}{(\neg\varphi)_\alpha}\ (\neg I)$	
¬ E	$\dfrac{\neg\varphi_\alpha \quad \varphi_\alpha}{\bot}\ (\neg E)$	
¬ PBC	$\dfrac{\begin{matrix}[(\neg\varphi)_\alpha] \\ \vdots \\ \bot\end{matrix}}{\varphi_\alpha}\ (\neg PBC)$	
→ I	$\dfrac{\begin{matrix}[\varphi_\alpha] \\ \vdots \\ \psi_\alpha\end{matrix}}{(\varphi \to \psi)_\alpha}\ (\to I)$	

Name	Rule	Condition
→ E	$\dfrac{(\varphi \to \psi)_\alpha \quad \varphi_\alpha}{\psi_\alpha}\ (\to E)$	
∧ I	$\dfrac{\varphi_\alpha \quad \psi_\alpha}{(\varphi \wedge \psi)_\alpha}\ (\wedge I)$	
∧ E	$\dfrac{(\varphi \wedge \psi)_\alpha}{\varphi_\alpha}\ (\wedge E1) \quad \dfrac{(\varphi \wedge \psi)_\alpha}{\psi_\alpha}\ (\wedge E2)$	
∨ I	$\dfrac{\varphi_\alpha}{(\varphi \vee \psi)_\alpha}\ (\vee I1) \quad \dfrac{\psi_\alpha}{(\varphi \vee \psi)_\alpha}\ (\vee I2)$	
∨ E	$\dfrac{(\varphi \vee \psi)_\alpha \quad \begin{array}{c}[\varphi_\alpha]\\ \vdots\\ \chi_\alpha\end{array} \quad \begin{array}{c}[\psi_\alpha]\\ \vdots\\ \chi_\alpha\end{array}}{\chi_\alpha}\ (\vee E)$	
↔ I	$\dfrac{(\varphi \to \psi)_\alpha \quad (\psi \to \varphi)_\alpha}{(\varphi \leftrightarrow \psi)_\alpha}\ (\leftrightarrow I)$	
↔ E	$\dfrac{(\varphi \leftrightarrow \psi)_\alpha}{(\varphi \to \psi)_\alpha}\ (\leftrightarrow E1) \quad \dfrac{(\varphi \leftrightarrow \psi)_\alpha}{(\psi \to \varphi)_\alpha}\ (\leftrightarrow E2)$	

3.1.3 Rules for the Modal Operators

Similarly, the rules for the modal operators are exactly same as they are defined in any standard literature of modal logic. However, our representation is slightly different. Firstly we have used an index with the formula. Secondly we have used the notation C instead of the usual notation R. Finally, instead of introducing proof boxes labelled by a world we have introduced a new variable y as an index of certain formulas. By a new variable we mean a variable $y \in X$ that does not occur in any term of an open assumption of the corresponding tree expect the place where it is explicitly mentioned. For example, in the rule $[C]I$ the variable y is not allowed to occur in any term, including α, of any open assumption of the tree with root φ_y. The rules are listed in Table 3.3.

Table 3.3: Rules for the Modal Operators

Name	**Rule**	**Condition**
$[C]$ I	$\begin{array}{c}[\alpha C y]\\ \vdots \\ \dfrac{\varphi_y}{([C]\varphi)_\alpha}\end{array}\ ([C]I)$	y is new
$[C]$ E	$\dfrac{([C]\varphi)_\alpha \quad \alpha C \beta}{\varphi_\beta}\ ([C]E)$	
$\langle C\rangle$ I	$\dfrac{\alpha C \beta \quad \varphi_\beta}{(\langle C\rangle\varphi)_\alpha}\ (\langle C\rangle I)$	
$\langle C\rangle$ E	$\dfrac{(\langle C\rangle\varphi)_\alpha \qquad \begin{array}{c}[\alpha C y] \quad [\varphi_y]\\ \vdots \\ \psi_\beta\end{array}}{\psi_\beta}\ (\langle C\rangle E)$	y is new in the right subtree and does not occur in β

3.1.4 Rules for the Basic Relevance Operators

In this section, we want to define the rules for the relevance operators based on the semantics defined in Chapter 2. The rules are listed in Table 3.4.

Table 3.4: Rules for the Basic Relevance Operators

Name	**Rule**	**Condition**
$\twoheadrightarrow$ I	$\begin{array}{c}[\varphi_x] \quad [\alpha = x + y]\\ \vdots \\ \dfrac{\psi_y}{(\varphi \twoheadrightarrow \psi)_\alpha}\end{array}\ (\twoheadrightarrow I)$	x, y are new
$\twoheadrightarrow$ E	$\dfrac{(\varphi \twoheadrightarrow \psi)_\alpha \quad \varphi_\beta \quad \alpha = \beta + \gamma}{\psi_\gamma}\ (\twoheadrightarrow E)$	
$\backsim$ I	$\begin{array}{c}[\varphi_\alpha]\\ \vdots \\ \dfrac{\bot}{(\backsim \varphi)_{\alpha^*}}\end{array}\ (\backsim I)$	
$\backsim$ E	$\dfrac{(\backsim \varphi)_\alpha \quad \varphi_\beta \quad \alpha = \beta^*}{\bot}\ (\backsim E)$	

Name	Rule	Condition
$\backsim$ PBC	$\dfrac{\begin{matrix}[(\backsim \varphi)_\alpha]\\ \vdots \\ \bot\end{matrix}}{\varphi_{\alpha^*}}\ (\backsim PBC)$	

3.1.5 Rules for the Derived Relevance Operators

Within our natural deduction system we want to treat the operators originally introduced as abbreviations as basic operators. Therefore, we will provide rules for them based on their semantic description from Chapter 2. The rules are listed in Table 3.5.

Table 3.5: Rules for the Derived Relevance Operators

Name	Rule	Condition
$\Wedge$ I	$\dfrac{\varphi_\beta \quad \psi_\gamma \quad \alpha = \beta + \gamma}{(\varphi \Wedge \psi)_\alpha}\ (\Wedge I)$	
$\Wedge$ E	$\dfrac{(\varphi \Wedge \psi)_\alpha \qquad \begin{matrix}[\varphi_x] \quad [\psi_y] \quad [\alpha = x + y]\\ \vdots \\ \chi_\beta\end{matrix}}{\chi_\beta}\ (\Wedge E)$	x, y are new in the right subtree
$\Vee$ I	$\dfrac{\begin{matrix}[(\neg\psi)_y] \quad [\alpha = x + y]\\ \vdots \\ \varphi_x\end{matrix}}{(\varphi \Vee \psi)_\alpha}\ (\Vee I1) \qquad \dfrac{\begin{matrix}[(\neg\varphi)_x] \quad [\alpha = x + y]\\ \vdots \\ \psi_y\end{matrix}}{(\varphi \Vee \psi)_\alpha}\ (\Vee I2)$	x,y are new
$\Vee$ E	$\dfrac{(\varphi \Vee \psi)_\alpha \quad \begin{matrix}[\varphi_\beta]\\ \vdots \\ \chi_\delta\end{matrix} \quad \begin{matrix}[\psi_\gamma]\\ \vdots \\ \chi_\delta\end{matrix} \quad \alpha = \beta + \gamma}{\chi_\delta}\ (\Vee E)$	
N I	$\dfrac{\begin{matrix}[(\neg\varphi)_\alpha]\\ \vdots \\ \bot\end{matrix}}{(N\varphi)_{\alpha^*}}\ (NI)$	
N E	$\dfrac{(N\varphi)_{\alpha^*} \quad (\neg\varphi)_\beta \quad \alpha = \beta^*}{\bot}\ (NE)$	

Name	Rule	Condition
N PBC	$\dfrac{\begin{matrix}[(N\neg\varphi)_\alpha]\\ \vdots\\ \bot\end{matrix}}{\varphi_{\alpha^*}}\ (NPBC)$	
$\multimap$ I	$\dfrac{\begin{matrix}[\varphi_x] \quad [\alpha = x\cdot y]\\ \vdots\\ \psi_y\end{matrix}}{(\varphi \multimap \psi)_\alpha}\ (\multimap I)$	x, y are new
$\multimap$ E	$\dfrac{(\varphi \multimap \psi)_\alpha \quad \varphi_\beta \quad \alpha = \beta\cdot\gamma}{\psi_\gamma}\ (\multimap E)$	
$\parr$ I	$\dfrac{\varphi_\beta \quad \psi_\gamma \quad \alpha = \beta\cdot\gamma}{(\varphi \parr \psi)_\alpha}\ (\parr I)$	
$\parr$ E	$\dfrac{(\varphi \parr \psi)_\alpha \qquad \begin{matrix}[\varphi_x] \quad [\psi_y] \quad [\alpha = x\cdot y]\\ \vdots\\ \chi_\beta\end{matrix}}{\chi_\beta}\ (\parr E)$	x, y are new in the right subtree
$\with$ I	$\dfrac{\begin{matrix}[(\neg\psi)_y] \quad [\alpha = x\cdot y]\\ \vdots\\ \varphi_x\end{matrix}}{(\varphi \with \psi)_\alpha}\ (\with I1) \qquad \dfrac{\begin{matrix}[(\neg\psi)_x] \quad [\alpha = x\cdot y]\\ \vdots\\ \varphi_y\end{matrix}}{(\varphi \with \psi)_\alpha}\ (\with I2)$	x, y are new
$\with$ E	$\dfrac{(\varphi \with \psi)_\alpha \quad \begin{matrix}[\varphi_\beta]\\ \vdots\\ \chi_\delta\end{matrix} \quad \begin{matrix}[\psi_\gamma]\\ \vdots\\ \chi_\delta\end{matrix} \quad \alpha = \beta\cdot\gamma}{\chi_\delta}\ (\with E)$	

3.1.6 Rules for the Constant E and U

Similar to the abbreviations above we want to treat E and U as basic formulas. The rules for these formulas are listed in Table 3.6.

Table 3.6: Rules for the Constant E and U

Name	Rule	Condition
E I	$\dfrac{x = x + \alpha}{E_\alpha}\ (EI)$	x is new

Name	**Rule**	**Condition**
E E	$\dfrac{E_\alpha}{\beta = \beta + \alpha}\ (EE)$	
U I	$\dfrac{x = x \cdot \alpha}{U_\alpha}\ (UI)$	x is new
U E	$\dfrac{U_\alpha}{\beta = \beta \cdot \alpha}\ (UE)$	

3.1.7 Rules for Contact

Finally, we want to define rules that forces the relation to be a contact relation. The rules are obviously based on Theorem 2. It is worth mentioning that those rules are neither introduction nor elimination rules. The rules are listed in Table 3.7.

Table 3.7: Rules for Contact

Name	**Rule**	**Condition**
BCA_0	$\dfrac{\alpha C \beta}{(\neg E)_\alpha}\ (BCA_{0\mathrm{g}})$	
BCA_1	$\dfrac{(\neg E)_\alpha}{\alpha C \alpha}\ (BCA_{1\mathrm{g}})$	
BCA_2	$\dfrac{\beta C \alpha}{\alpha C \beta}\ (BCA_{2\mathrm{g}})$	
BCA_3	$\dfrac{\alpha C \beta \quad \beta = \beta \cdot \gamma}{\alpha C \gamma}\ (BCA_{3\mathrm{g}})$	
BCA_4	$\dfrac{\alpha C(\beta + \gamma) \quad \begin{matrix}[\alpha C \beta] \\ \vdots \\ \varphi_\delta\end{matrix} \quad \begin{matrix}[\alpha C \gamma] \\ \vdots \\ \varphi_\delta\end{matrix}}{\varphi_\delta}\ (BCA_4)$	

3.2 Soundness of Natural Deduction Rules

In this section of this thesis, we want to investigate the theoretical properties of our natural deduction calculus. In particular, we are interested in soundness and completeness. Soundness of a calculus is the property that every formula that can be derived is also valid. Completeness is the opposite implication. Please note that completeness is out of the scope of the thesis and will be considered in future work. We are going to formally define soundness as follows:

Theorem 3. *(Soundness) Let $\varphi_1, \varphi_2, \dots \varphi_n$ and ψ_α to be formulas of any three types (annonated formulas, equations and contact relation). If*

$$\varphi_1, \varphi_2, \dots, \varphi_n \vdash \psi_\alpha, \text{ then}$$
$$\varphi_1, \varphi_2, \dots, \varphi_n \vDash \psi_\alpha.$$

Proof.
The proof is done by induction on the derivation $\varphi_1, \varphi_2, \dots \varphi_n \vdash \psi_\alpha$.

Base case:
In this case, the proof is simply an assumption, i.e., $\psi_\alpha \in \{\varphi_1, \varphi_2, \dots \varphi_n\}$.

Induction step:
In this case, in the proof tree, each derivation step's outcome must be the conclusion the of the natural deduction rules. Thus, we will investigate the validation of applied rules including the side conditions defined for the rules in this thesis. We will begin with distinguishing the conclusion for all three types of the formula, a) if it is of the form $\alpha = \beta$ then we will use the axioms of BA and the rules $= I$, $= E$, EE, and UE, b) Secondly, if it is of the form $\alpha C \beta$ then we will use the rules $= I$ and $= E$, c) we will do the cases for an annotated formula. In the next subsections, we are going to state the proofs of the rules mentioned in the same order they are defined above.

3.2.1 Equality Operator

= I: Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \vDash \alpha = \beta$. Since we don't have any assumptions in this case, we can conclude $\mathcal{M}, n \vDash \alpha = \beta$.

= E: In this case we have the following derivations

$$\varphi_1, \varphi_2, \dots, \varphi_n \vdash \alpha = \beta \text{ and}$$
$$\varphi_1, \varphi_2, \dots, \varphi_n \vdash \varphi_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \vDash \psi_\beta$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash \psi_\alpha$

i.e., $\mathcal{M}, n[\alpha] \models \psi$ and $\alpha = \beta$. Therefore, we can conclude $\mathcal{M}, n[\beta] \models \psi$, and, hence, $\mathcal{M}, n \models \psi_\beta$.

3.2.2 Propositional Operators

In this section, we want to discuss the soundness of the propositional logic natural deduction rules defined in the Table 3.2.

$\neg$ I: In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, \psi_\alpha \vdash \bot.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models (\neg\psi)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \neg\psi$. Assume $\mathcal{M}, n \models \psi_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \psi$. From the induction hypothesis we get $\mathcal{M}, n \models \bot$. Since the last statement is a contradiction, this implies $\mathcal{M}, n[\alpha] \not\models \psi$, and, therefore we can conclude $\mathcal{M}, n[\alpha] \models \neg\psi$, this is equivalent to $\mathcal{M}, n \models (\neg\psi)_\alpha$.

$\neg$ E: In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\neg\psi)_\alpha \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \psi_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models \bot$. We obtain from induction hypothesis $\mathcal{M}, n \models \psi_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \psi$, and $\mathcal{M}, n \models (\neg\psi)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \neg\psi$. Since the statement is a contradiction, we can conclude $\mathcal{M}, n[\alpha] \models \bot$, this is equivalent to $\mathcal{M}, n \models \bot$.

$\neg$ PBC: In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\neg\psi)_\alpha \vdash \bot.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models \psi_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \psi$. Assume $\mathcal{M}, n \models (\neg\psi)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \neg\psi$. From the induction hypothesis we get $\mathcal{M}, n \models \bot$. Since the last statement is a contradiction, we can conclude $\mathcal{M}, n[\alpha] \models \psi$, this is equivalent to $\mathcal{M}, n \models \psi_\alpha$.

$\rightarrow$ **I:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_\alpha \vdash (\psi_2)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 \rightarrow \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1 \rightarrow \psi_2$. Therefore, assume $\mathcal{M}, n[\alpha] \models \psi_1$, i.e., $\mathcal{M}, n \models (\psi_1)_\alpha$. From the induction hypothesis we obtain $\mathcal{M}, n \models (\psi_1)_\alpha$, and, hence, $\mathcal{M}, n[\alpha] \models \psi_2$.

$\rightarrow$ **E:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \rightarrow \psi_2)_\alpha \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_2$. We obtain from induction hypothesis $\mathcal{M}, n \vDash (\psi_1)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1$, and, $\mathcal{M}, n \vDash (\psi_1 \rightarrow \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 \rightarrow \psi_2$. Therefore we can conclude $\mathcal{M}, n[\alpha] \vDash \psi_2$, this is equivalent to $\mathcal{M}, n[\alpha] \vDash \psi_2$.

$\wedge$ **I:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\alpha \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_2)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 \wedge \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1 \wedge \psi_2$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\psi_1)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1$ and $\mathcal{M}, n \vDash (\psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_2$. Therefore we can conclude $\mathcal{M}, n[\alpha] \vDash \psi_1 \wedge \psi_2$, this is equivalent to $\mathcal{M}, n \vDash (\psi_1 \wedge \psi_2)_\alpha$.

$\wedge$ **E1:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \wedge \psi_2)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1$. We obtain from induction hypothesis $\mathcal{M}, n \vDash (\psi_1 \wedge \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 \wedge \psi_2$. Therefore we can conclude $\mathcal{M}, n[\alpha] \models \psi_1$, this is equivalent to $\mathcal{M}, n \models (\psi_1)_\alpha$.

$\wedge$ **E2:** Analogously to $\wedge$E1.

$\vee$ **I1:** In this case have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models (\psi_1 \vee \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \psi_1 \vee \psi_2$. We obtain from induction hypothesis $\mathcal{M}, n \models (\psi_1)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \psi_1$. Therefore we can conclude $\mathcal{M}, n[\alpha] \models \psi_1 \vee \psi_2$, this is equivalent to $\mathcal{M}, n \models (\psi_1 \vee \psi_2)_\alpha$.

$\vee$ **I2:** Analogously to $\vee$I1.

$\vee$ **E:** In this case we have the following derivations

$$\begin{aligned} \varphi_1, \varphi_2, \ldots, \varphi_n &\vdash (\psi_1 \vee \psi_2)_\alpha, \\ \varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_\alpha &\vdash \chi_\alpha \text{ and} \\ \varphi_1, \varphi_2, \ldots, \varphi_n(\psi_2)_\alpha &\vdash \chi_\alpha. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models \chi_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \chi$. We obtain from the induction hypothesis $\mathcal{M}, n \models (\psi_1 \vee \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \psi_1 \vee \psi_2$, i.e., either $\mathcal{M}, n[\alpha] \models \psi_1$ or $\mathcal{M}, n[\alpha] \models \psi_2$. If we have $\mathcal{M}, n[\alpha] \models \psi_1$ i.e., $\mathcal{M}, n \models (\psi_1)_\alpha$, then the induction hypothesis shows $\mathcal{M}, n \models \chi_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \chi$. The second case follows analogously.

$\leftrightarrow$ **I:** In this case we have the following derivations

$$\begin{aligned} \varphi_1, \varphi_2, \ldots, \varphi_n &\vdash (\psi_1 \rightarrow \psi_2)_\alpha \text{ and} \\ \varphi_1, \varphi_2, \ldots, \varphi_n &\vdash (\psi_2 \rightarrow \psi_1)_\alpha. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models (\psi_1 \leftrightarrow \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \psi_1 \leftrightarrow \psi_2$. We obtain from the induction hypothesis $\mathcal{M}, n \models (\psi_1 \rightarrow \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \psi_1 \rightarrow \psi_2$ and $\mathcal{M}, n \models (\psi_2 \rightarrow \psi_1)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \psi_2 \rightarrow \psi_1$. Therefore we can conclude $\mathcal{M}, n[\alpha] \models \psi_1 \leftrightarrow \psi_2$, this is equivalent to $\mathcal{M}, n \models (\psi_1 \leftrightarrow \psi_2)_\alpha$.

↔ **E1:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \leftrightarrow \psi_2)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 \rightarrow \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1 \rightarrow \psi_2$. We obtain from induction hypothesis $\mathcal{M}, n \vDash (\psi_1 \leftrightarrow \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 \leftrightarrow \psi_2$. Therefore we can conclude $\mathcal{M}, n[\alpha] \vDash \psi_1 \rightarrow \psi_2$, this is equivalent to $\mathcal{M}, n \vDash (\psi_1 \rightarrow \psi_2)_\alpha$.

↔ **E2:** Analogously to ↔ E1.

3.2.3 Modal Operators

Similarly, in this section, we want to discuss the soundness of the modal logic natural deduction rules defined in the Table 3.3.

$[C]$ **I:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, \alpha C y \vdash \psi_y.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash ([C]\psi)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash [C]\psi$. Therefore, assume $n[\alpha]C$ a so that we have to show $\mathcal{M}, a \models \psi$. Let $n^{'}$ be the environment defined by:

$$n'(x) = \begin{cases} n(x) & : \quad x \neq y \\ a & : \quad x = y \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

$$\begin{aligned} \mathcal{M}, n' \models \alpha C y &\Leftrightarrow n'[\alpha] C n'[y] & \\ &\Leftrightarrow n'[\alpha] C a & n'[y] = n'(y) = a \text{ by Definition} \\ &\Leftrightarrow n[\alpha] C a & \text{by Lemma 29 and} \\ & & \text{side condition of the rule,} \end{aligned}$$

i.e., $\mathcal{M}, n' \models \alpha C y$. From the induction hypothesis we obtain $\mathcal{M}, n' \models \psi_y$. We compute

$$\begin{aligned} \mathcal{M}, n^{'} \models \psi_y &\Leftrightarrow \mathcal{M}, n^{'}[y] \models \psi \\ &\Leftrightarrow \mathcal{M}, a \models \psi \end{aligned}$$

$[C]$ **E:** In this case we have the following derivations

$$\begin{aligned} &\varphi_1, \varphi_2, \dots, \varphi_n \vdash ([C]\varphi)_\alpha \text{ and} \\ &\varphi_1, \varphi_2, \dots, \varphi_n \vdash \alpha C \beta. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \models \psi_\beta$, or equivalently that $\mathcal{M}, n[\beta] \models \psi$. We obtain from the induction hypothesis $\mathcal{M}, n \models ([C]\psi)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models [C]\psi$ and $\mathcal{M}, n \models \alpha C \beta$ i.e., $n[\alpha] C n[\beta]$. Therefore by the definition of $[C]$, we can conclude $\mathcal{M}, n[\beta] \models \psi$, this is equivalent to $\mathcal{M}, n \models \psi_\beta$.

$\langle C \rangle$ **I:** In this case we have the following derivations

$$\begin{aligned} &\varphi_1, \varphi_2, \dots, \varphi_n \vdash \alpha C \beta \text{ and} \\ &\varphi_1, \varphi_2, \dots, \varphi_n \vdash \psi_\beta. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \models (\langle C \rangle \psi)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \langle C \rangle \psi$. We obtain from the induction hypothesis $\mathcal{M}, n \models \psi_\beta$ i.e., $\mathcal{M}, n[\beta] \models \psi$ and $\mathcal{M}, n \models \alpha C \beta$ i.e., $n[\alpha] C n[\beta]$. Therefore by the definition of $\langle C \rangle$, we can conclude $\mathcal{M}, n[\alpha] \models \langle C \rangle \psi$, this is equivalent to $\mathcal{M}, n \models (\langle C \rangle \psi)_\alpha$.

$\langle C \rangle$ **E:** In this case we have the following derivations

$$\begin{aligned} \varphi_1, \varphi_2, \dots, \varphi_n &\vdash (\langle C \rangle \psi)_\alpha \text{ and} \\ \varphi_1, \varphi_2, \dots, \varphi_n, \alpha C y, \psi_y &\vdash \chi_\beta. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \models \chi_\beta$, or equivalently that $\mathcal{M}, n[\beta] \models \chi$. We obtain from the induction hypothesis $\mathcal{M}, n \models (\langle C \rangle \psi)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models \langle C \rangle \psi$. Therefore by the definition of $\langle C \rangle$, there is an a with $n[\alpha] C a$ and $\mathcal{M}, a \models \psi$. Let $n^{'}$ be the environment defined by:

$$n'(x) = \begin{cases} n(x) & : \quad x \neq y \\ a & : \quad x = y \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

(1)

$$\begin{aligned} \mathcal{M}, n' \models \alpha C y &\Leftrightarrow n'[\alpha] C n'[y] & \\ &\Leftrightarrow n'[\alpha] C a & n'[y] = n'(y) = a \text{ by Definition} \\ &\Leftrightarrow n[\alpha] C a & \text{By Lemma 29 and} \\ & & \text{side condition of the rule} \end{aligned}$$

(2) $\mathcal{M}, n' \vDash \psi_y$ since $n'[y] = n'(y) = a$

By the induction hypothesis and (1) and (2) we get $\mathcal{M}, n^{'} \vDash \chi_\beta$, *i.e.*, $\mathcal{M}, n \vDash \chi_\beta$ since $n'[\beta] = n[\beta]$ by the side condition. Therefore, we have $\mathcal{M}, n[\beta] \vDash \chi$.

3.2.4 Relevance Basic Operators

Similarly, in this section, we want to discuss the soundness of the relevance logic natural deduction rules defined in the Table 3.4.

$\twoheadrightarrow$ I: In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_x, \alpha = x + y \vdash (\psi_2)_y.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 \twoheadrightarrow \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1 \twoheadrightarrow \psi_2$. Therefore, assume $n[\alpha] = a + b$ with $\mathcal{M}, a \models \psi_1$ so that we have to show $\mathcal{M}, b \models \psi_2$. Let $n^{'}$ be the environment defined by:

$$n'(z) = \begin{cases} a & : \quad z = x \\ b & : \quad z = y \\ n(z) & \text{otherwise} \end{cases}$$

By Lemma 29 and the side condition of the rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

(1) $n'[\alpha] = a + b \;= n[\alpha]$ by the side condition of the rule,

(2) $\mathcal{M}, n'[x] \vDash \psi_1$ since $n^{'}[x] = n'(x) = a$.

By the induction hypothesis and (2) we get $\mathcal{M}, b \vDash \psi_2$, i.e., $\mathcal{M}, n[y] \vDash \psi_2$ since $n'[y] = n'(y) = b$.

$\twoheadrightarrow$ **E:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \twoheadrightarrow \psi_2)_\alpha,$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\beta \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta + \gamma.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_2)_\gamma$, or equivalently that $\mathcal{M}, n[\gamma] \vDash \psi_2$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\psi_1 \twoheadrightarrow \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 \twoheadrightarrow \psi_2$, $\mathcal{M}, n \vDash (\psi_1)_\beta$ i.e., $\mathcal{M}, n[\beta] \vDash \psi_1$ and $n[\alpha] = n[\beta + \gamma] = n[\beta] + n[\gamma]$. Therefore, we can conclude $\mathcal{M}, n[\gamma] \vDash \psi_2$, this is equivalent to $\mathcal{M}, n \vDash (\psi_2)_\gamma$.

$\backsim$ **I:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, \psi_\alpha \vdash \bot.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\backsim \psi)_{\alpha^*}$, or equivalently that $\mathcal{M}, n[\alpha^*] \vDash \backsim \psi$. Assume $\mathcal{M}, n \vDash \psi_\alpha$. The induction hypothesis leads to a contradiction so that we conclude $\mathcal{M}, n \nvDash \psi_\alpha$. This implies

$$\begin{aligned} \mathcal{M}, n \nvDash \psi_\alpha &\Leftrightarrow \mathcal{M}, n[\alpha] \nvDash \psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \vDash \backsim \psi && \text{by Definition} \end{aligned}$$

Therefore we conclude $\mathcal{M}, n[\alpha^*] \vDash \backsim \psi$, this is equivalent to $\mathcal{M}, n \vDash (\backsim \psi)_{\alpha^*}$.

$\backsim$ **E:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\backsim \psi)_\alpha,$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \psi_\beta \text{ and}$$

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta^*.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n[\alpha] \vDash \bot$ or equivalently that $\mathcal{M}, n \vDash \bot$.

From 1st derivation we have:

$$\begin{aligned} \mathcal{M}, n \vDash (\backsim \psi)_\alpha &\Leftrightarrow \mathcal{M}, n[\alpha] \vDash \backsim \psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \nvDash \psi && \text{by Definition} \end{aligned}$$

From 2nd derivation we have:

$$\begin{aligned} \mathcal{M}, n \vDash \psi_\beta &\Leftrightarrow \mathcal{M}, n[\beta] \vDash \psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \vDash \psi && \text{since } \alpha = \beta^* \end{aligned}$$

Since the last two statements contradict, therefore we conclude $\mathcal{M}, n[\alpha] \vDash \bot$, this is equivalent to $\mathcal{M}, n \vDash \bot$.

$\backsim$ **PBC:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\backsim \psi)_\alpha \vdash \bot.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \psi_{\alpha^*}$, or equivalently that $\mathcal{M}, n[\alpha^*] \vDash \psi$.

Assume $\mathcal{M}, n \vDash (\backsim \psi)_\alpha$, but since the result is a contradiction this implies

$$\begin{aligned} \mathcal{M}, n \nvDash (\backsim \psi)_\alpha &\Leftrightarrow \mathcal{M}, n[\alpha] \nvDash \backsim \psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \vDash \psi && \text{by Definition} \end{aligned}$$

Therefore we can conclude $\mathcal{M}, n[\alpha^*] \vDash \psi$ this is equivalent to $\mathcal{M}, n \vDash \psi_{\alpha^*}$.

3.2.5 Relevance Derived Operators

Similarly, in this section, we want to discuss the soundness of the propositional relevance logic natural deduction rules defined in the Table 3.5.

$⩓$ **I:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\beta,$$

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\gamma \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta + \gamma.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 \mathbin{\wedge\!\!\!\wedge} \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1 \mathbin{\wedge\!\!\!\wedge} \psi_2$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\psi_1)_\beta$ i.e., $\mathcal{M}, n[\beta] \vDash \psi_1$, $\mathcal{M}, n \vDash (\psi_2)_\gamma$ i.e., $\mathcal{M}, n[\gamma] \vDash \psi_2$ and $n[\alpha] = n[\beta + \gamma] = n[\beta] + n[\gamma]$. This implies $\mathcal{M}, n[\alpha] \vDash \psi_1 \mathbin{\wedge\!\!\!\wedge} \psi_2$, which is equivalent to $\mathcal{M}, n \vDash (\psi_1 \mathbin{\wedge\!\!\!\wedge} \psi_2)_\alpha$.

$\mathbin{\wedge\!\!\!\wedge}$ **E:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \mathbin{\wedge\!\!\!\wedge} \psi_2)_\alpha \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_x, (\psi_2)_y, \alpha = x + y \vdash \chi_\beta.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \chi_\beta$, or equivalently that $\mathcal{M}, n[\beta] \vDash \chi$. From the induction hypothesis we get $\mathcal{M}, n[\alpha] \vDash \psi_1 \mathbin{\wedge\!\!\!\wedge} \psi_2$. Then there are a, b so that $n[\alpha] = a + b$ and $\mathcal{M}, a \models \psi_1$ and $\mathcal{M}, b \models \psi_2$. Let $n^{'}$ be the environment defined by:

$$n'(z) = \begin{cases} a & : \quad z = x \\ b & : \quad z = y \\ n(z) & \text{otherwise} \end{cases}$$

By Lemma 29 and the side condition of the rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

(1) $n'[\alpha] = n[\alpha] = a + b = n'[x] + n'[y]$ by the side condition of the rule

(2) $\mathcal{M}, n'[x] \vDash \psi_1$ since $n^{'}[x] = a$ and

(3) $\mathcal{M}, n'[y] \vDash \psi_2$ since $n^{'}[y] = b$.

By the induction hypothesis and (1), (2) and (3) we get $\mathcal{M}, n^{'}[\beta] \vDash \chi$ i.e., $\mathcal{M}, n \vDash \chi_\beta$ since $n'[\beta] = n[\beta]$, this is equivalent to $\mathcal{M}, n \vDash \chi_\beta$.

$\mathbin{\vee\!\!\!\vee}$ **I1:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\neg\psi_2)_y, \alpha = x + y \vdash (\psi_1)_x.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$.

We have to show that $\mathcal{M}, n \vDash (\psi_1 \Vee \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash (\psi_1 \Vee \psi_2)$. Therefore, assume that $n[\alpha] = n[a + b] = n[a] + n[b]$ with $\mathcal{M}, n[a] \vDash \psi_1$ and $\mathcal{M}, n[b] \nvDash \psi_2$. Let $n^{'}$ be the environment defined by:

$$n'(m) = \begin{cases} a & : \quad z = x \\ b & : \quad z = y \\ n(z) & \text{otherwise} \end{cases}$$

By Lemma 29 and the side condition of the rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

(1) $n'[\alpha] = n[\alpha] = a + b = n'[x] + n'[y]$ by the side condition of the rule

(2) $\mathcal{M}, n'[x] \vDash \psi_1$ since $n^{'}[x] = a$,

(3) $\mathcal{M}, n'[y] \nvDash \psi_2$ since $n^{'}[y] = b$.

By the induction hypothesis and (1), (2) and (3) we get $\mathcal{M}, n[\alpha] \vDash \psi_1 \Vee \psi_2$, i.e., either $\mathcal{M}, n[\alpha] \vDash \psi_1$ or $\mathcal{M}, n[\alpha] \vDash \psi_2$. Therefore, we can conclude $\mathcal{M}, n[\alpha] \vDash \psi_1 \Vee \psi_2$, this is equivalent to $\mathcal{M}, n \vDash (\psi_1 \Vee \psi_2)_\alpha$.

$\Vee$ **I2:** Analogously to $\Vee$I1.

$\Vee$ **E:** In this case we have the following derivations

$$\begin{aligned} \varphi_1, \varphi_2, \ldots, \varphi_n &\vdash (\psi_1 \Vee \psi_2)_\alpha, \\ \varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_\beta &\vdash \chi_\delta, \\ \varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_2)_\gamma &\vdash \chi_\delta \text{ and} \\ \varphi_1, \varphi_2, \ldots, \varphi_n &\vdash \alpha = \beta + \gamma. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \chi_\delta$, or equivalently that $\mathcal{M}, n[\delta] \vDash \chi$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\psi_1 \Vee \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 \Vee \psi_2$ with $n[\alpha] = n[\beta + \gamma] = n[\beta] + m[\gamma]$, i.e., either $\mathcal{M}, n[\beta] \vDash \psi_1$ or $\mathcal{M}, n[\gamma] \vDash \psi_2$. If we have $\mathcal{M}, n[\beta] \vDash \psi_1$ i.e., $\mathcal{M}, n \vDash (\psi_1)_\beta$, then induction hypothesis shows $\mathcal{M}, n \vDash \chi_\delta$ i.e., $\mathcal{M}, n[\delta] \vDash \chi$. The second case follows analogously.

N **I:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\neg\psi)_\alpha \vdash \bot.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models (N\psi)_{\alpha^*}$, or equivalently that $\mathcal{M}, n[\alpha^*] \models N\psi$. Therefore, assume $\mathcal{M}, n \models (\neg\psi)_\alpha$, but since the result is a contradiction this implies

$$\begin{aligned} \mathcal{M}, n \not\models (\neg\psi)_\alpha &\Leftrightarrow \mathcal{M}, n[\alpha] \not\models \neg\psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \models\backsim \neg\psi && \text{by Definition} \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \models N\psi && \text{by Definition} \end{aligned}$$

Therefore we can conclude $\mathcal{M}, n[\alpha^*] \models N\psi$ this is equivalent to $\mathcal{M}, n \models (N\psi)_{\alpha^*}$.

N **E:** In this case we have the following derivations

$$\begin{aligned} &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (N\psi)_{\alpha^*}, \\ &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\neg\psi)_\beta \text{ and} \\ &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta^*. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n[\alpha] \models \bot$ or equivalently that $\mathcal{M}, n \models \bot$.

From 1st derivation we have:

$$\begin{aligned} \mathcal{M}, n \models (N\psi)_{\alpha^*} &\Leftrightarrow \mathcal{M}, n[\alpha^*] \models N\psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \models\backsim \neg\psi && \text{by Definition} \end{aligned}$$

From 2nd derivation we have:

$$\begin{aligned} \mathcal{M}, n \models (\neg\psi)_\beta &\Leftrightarrow \mathcal{M}, n[\beta] \models \neg\psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \models \neg\psi && \text{Since } \alpha = \beta^* \\ &\Leftrightarrow \mathcal{M}, n[\alpha] \not\models\backsim \neg\psi && \text{by Definition} \end{aligned}$$

Since the last two statements contradict, therefore we can conclude $\mathcal{M}, n[\alpha] \models \bot$, this is equivalent to $\mathcal{M}, n \models \bot$.

N **PBC:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (N\neg\psi)_\alpha \vdash \bot.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$.

We have to show that $\mathcal{M}, n \models \psi_{\alpha^*}$, or equivalently that $\mathcal{M}, n[\alpha^*] \models \psi$.

Assume $\mathcal{M}, n \models (N\neg\psi)_\alpha$, but since the result is a contradiction this implies

$$\begin{aligned} \mathcal{M}, n \not\models (N\neg\psi)_\alpha &\Leftrightarrow \mathcal{M}, n[\alpha] \not\models N\neg\psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha] \not\models\backsim \neg\neg\psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha] \not\models\backsim \psi \\ &\Leftrightarrow \mathcal{M}, n[\alpha^*] \models \psi \qquad\qquad\qquad \text{by Definition} \end{aligned}$$

Therefore we conclude $\mathcal{M}, n[\alpha^*] \models \psi$, this is equivalent to $\mathcal{M}, n \models \psi_{\alpha^*}$.

$\multimap$ **I:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_x, \alpha = x \cdot y \vdash (\psi_2)_y.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \models (\psi_1 \multimap \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models \psi_1 \multimap \psi_2$. Therefore, assume $n[\alpha] = a \cdot b$ with $\mathcal{M}, a \models \psi_1$ so that we have to show $\mathcal{M}, b \models \psi_2$. Let $n^{'}$ be the environment defined by:

$$n'(z) = \begin{cases} a & : \qquad z = x \\ b & : \qquad z = y \\ n(z) & \quad \text{otherwise} \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \models \varphi_i$ iff $\mathcal{M}, n' \models \varphi_i$. Furthermore we have:

(1) $n'[\alpha] = a \cdot b \ = n[\alpha]$,

(2) $\mathcal{M}, n'[x] \models \psi_1$ since $n^{'}[x] = a$.

By the induction hypothesis and from (2) we get $\mathcal{M}, n'[y] \models \psi_2$ i.e., $\mathcal{M}, b \models \psi_2$ since $n'[y] = n(y) = b$.

$\multimap$ **E:** In this case we have the following derivations

$$\begin{aligned} &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \multimap \psi_2)_\alpha, \\ &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\beta \text{ and} \\ &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta \cdot \gamma. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_2)_\gamma$, or equivalently that $\mathcal{M}, n[\gamma] \vDash \psi_2$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\psi_1 \multimap \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 \multimap \psi_2$, $\mathcal{M}, n \vDash (\psi_1)_\beta$ i.e., $\mathcal{M}, n[\beta] \vDash \psi_1$ and $n[\alpha] = n[\beta \cdot \gamma] = n[\beta] \cdot n[\gamma]$. Therefore, we can conclude $\mathcal{M}, n[\gamma] \vDash \psi_2$, this is equivalent to $\mathcal{M}, n \vDash (\psi_2)_\gamma$.

$\parr$ **I:** In this case we have the following derivations

$$\begin{aligned}
&\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\beta, \\
&\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1)_\gamma \text{ and} \\
&\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta \cdot \gamma.
\end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 ⩓ \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash \psi_1 ⩓ \psi_2$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\psi_1)_\beta$ i.e., $\mathcal{M}, n[\beta] \vDash \psi_1$, $\mathcal{M}, n \vDash (\psi_2)_\gamma$ i.e., $\mathcal{M}, n[\gamma] \vDash \psi_2$ and $n[\alpha] = n[\beta \cdot \gamma] = n[\beta] \cdot n[\gamma]$. This implies $\mathcal{M}, n[\alpha] \vDash \psi_1 ⩓ \psi_2$, which is equivalent to $\mathcal{M}, n \vDash (\psi_1 ⩓ \psi_2)_\alpha$.

$\parr$ **E:** In this case we have the following derivations

$$\begin{aligned}
\varphi_1, \varphi_2, \ldots, \varphi_n &\vdash (\psi_1 \parr \psi_2)_\alpha \text{ and} \\
\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_x, (\psi_1)_y, \alpha = x \cdot y &\vdash \chi_\beta.
\end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \chi_\beta$, or equivalently that $\mathcal{M}, n[\beta] \vDash \chi$. From the induction hypothesis we get $\mathcal{M}, n[\alpha] \vDash \psi_1 \parr \psi_2$. Then there are a, b so that $n[\alpha] = a \cdot b$ and $\mathcal{M}, a \models \psi_1$ and $\mathcal{M}, b \models \psi_2$. Let $n^{'}$ be the environment defined by:

$$n'(z) = \begin{cases} a & : \quad z = x \\ b & : \quad z = y \\ n(z) & \quad \text{otherwise} \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

(1) $n'[\alpha] == n[\alpha] = a \cdot b = n'[x] \cdot n'[y]$ by the side condition of the rule,

(2) $\mathcal{M}, n'[x] \vDash \psi_1$ since $n^{'}[x] = a$,

(3) $\mathcal{M}, n'[y] \vDash \psi_2$ since $n'[y] = b$.

By the induction hypothesis and from (1), (2) and (3) we get $\mathcal{M}, n'[\beta] \vDash \chi$ i.e., $\mathcal{M}, n[\beta] \vDash \chi$ since $n'[\beta] = n[\beta]$, this is equivalent to $\mathcal{M}, n \vDash \chi_\beta$.

$\parr$ **I1:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n, (\neg\psi_2)_y, \alpha = x \cdot y \vdash (\psi_1)_x.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash (\psi_1 \parr \psi_2)_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash (\psi_1 \parr \psi_2)$. Therefore, assume that $n[\alpha] = n[a \cdot b] = n[a] \cdot n[b]$ with $\mathcal{M}, n[a] \vDash \psi_1$, $\mathcal{M}, n[b] \nvDash \psi_2$ and $n[\alpha] = n[a \cdot b]$. Let n' be the environment defined by:

$$n'(m) = \begin{cases} a & : \quad m \in x \\ b & : \quad m \in y \\ n(m) & \text{otherwise} \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. Furthermore we have:

(1) $n'[\alpha] = n[\alpha] = a \cdot b = n'[x] \cdot n'[y]$ by the side condition of the rule

(2) $\mathcal{M}, n'[x] \vDash \psi_1$ since $n'[x] = a$,

(3) $\mathcal{M}, n'[y] \nvDash \psi_2$ since $n'[y] = b$.

By the induction hypothesis and from (1), (2) and (3) we get $\mathcal{M}, n[\alpha] \vDash \psi_1 \parr \psi_2$, i.e., either $\mathcal{M}, n[\alpha] \vDash \psi_1$ or $\mathcal{M}, n[\alpha] \vDash \psi_2$. Therefore, we can conclude $\mathcal{M}, n[\alpha] \vDash \psi_1 \parr \psi_2$, this is equivalent to $\mathcal{M}, n \vDash (\psi_1 \parr \psi_2)_\alpha$.

$\parr$ **I2:** Analogously to $\parr$I1.

$\parr$ **E:** In this case we have the following derivations

$$\begin{aligned} &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash (\psi_1 \parr \psi_2)_\alpha, \\ &\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_1)_\beta \vdash \chi_\delta, \\ &\varphi_1, \varphi_2, \ldots, \varphi_n, (\psi_2)_\gamma \vdash \chi_\delta \text{ and} \\ &\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha = \beta \cdot \gamma. \end{aligned}$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \chi_\delta$, or equivalently that $\mathcal{M}, n[\delta] \vDash \chi$. We obtain from induction hypothesis $\mathcal{M}, n \vDash (\psi_1 ⅋ \psi_2)_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash \psi_1 ⅋ \psi_2$ with $n[\alpha] = n[\beta \cdot \gamma] = n[\beta] \cdot n[\gamma]$, i.e., either $\mathcal{M}, n[\beta] \vDash \psi_1$ or $\mathcal{M}, n[\gamma] \vDash \psi_2$. If we have $\mathcal{M}, n[\beta] \vDash \psi_1$ i.e., $\mathcal{M}, n \vDash (\psi_1)_\beta$, then induction hypothesis shows $\mathcal{M}, n \vDash \chi_\delta$ i.e., $\mathcal{M}, n[\delta] \vDash \chi$. The second case follows analogously..

3.2.6 Constant E and U

Similarly, in this section, we want to discuss the soundness of the propositional relevance logic with E natural deduction rules defined in the Table 3.6.

E **I:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash x = x + \alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash E_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \vDash E$. Let n' be the environment defined by

$$n'(y) = \begin{cases} n(y) & : \quad y \neq x \\ 0 & : \quad y = x \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \vDash \varphi_i$ iff $\mathcal{M}, n' \vDash \varphi_i$. The induction hypothesis implies $n'[x] = n'[x] + n'[\alpha]$, and, hence, $n'[\alpha] = 0 + n[\alpha] = n'[x] + n'[\alpha] = n'[x] = 0$. This shows $\mathcal{M}, n^{'}[\alpha] \vDash E$, i.e., $\mathcal{M}, n[\alpha] \vDash E$ since $n'[x] = n[x]$, this is equivalent to $\mathcal{M}, n \vDash E_\alpha$.

E **E:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash E_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\beta = \beta + \alpha$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash E_\alpha$ i.e., $\mathcal{M}, n[\alpha] \vDash E$ resp. $n[\alpha] = 0$. Therefore, we can conclude $n[\beta] = n[\beta] + 0 = n[\beta] + n[\alpha] = n[\alpha + \beta]$, and, hence, $\mathcal{M}, n \models \beta = \beta + \alpha$..

U **I:** In this case we have the following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash x = x \cdot \alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \models U_\alpha$, or equivalently that $\mathcal{M}, n[\alpha] \models U$. Let $n^{'}$ be the environment defined by:

$$n'(y) = \begin{cases} n(y) & : \quad y \neq x \\ 1 & : \quad y = x \end{cases}$$

By Lemma 29 and the side condition of this rule we get $\mathcal{M}, n \models \varphi_i$ iff $\mathcal{M}, n' \models \varphi_i$. The induction hypothesis implies $n'[x] = n'[x] \cdot n'[\alpha]$, and, hence, $n'[\alpha] = 1 \cdot n[\alpha] = n'[x] \cdot n'[\alpha] = n'[x] = 1$. This shows $\mathcal{M}, n^{'}[\alpha] \models U$, i.e., $\mathcal{M}, n[\alpha] \models U$ since $n'[x] = n[x]$. Therefore, we can conclude $\mathcal{M}, n \models U_\alpha$.

U **E:** In this case we have a derivation

$$\varphi_1, \varphi_2, \dots, \varphi_n \vdash U_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\beta = \beta \cdot \alpha$. We obtain from the induction hypothesis $\mathcal{M}, n \models U_\alpha$ i.e., $\mathcal{M}, n[\alpha] \models U$ resp. $n[\alpha] = 1$. Therefore, we can conclude $n[\beta] = n[\beta] \cdot 1 = n[\beta] \cdot n[\alpha] = n[\alpha \cdot \beta]$, and, hence, $\mathcal{M}, n \models \beta = \beta \cdot \alpha$.

3.2.7 Contact Axioms

Similarly, in this section, we want to discuss the soundness of the modal relevance logic natural deduction rules defined in the Table 3.3.

$BCA_{\mathbf{0g}}$**:** In this case we have a derivation

$$\varphi_1, \varphi_2, \dots, \varphi_n \vdash \alpha C \beta.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \models \varphi_i$ for $i \in \{1, 2, \dots, n\}$. We have to show that $\mathcal{M}, n \models (\neg E)_\alpha$. We obtain from the induction hypothesis $\mathcal{M}, n \models \alpha C \beta$ i.e., $n[\alpha] C n[\beta]$. Therefore according the axiom C_0 and the definition of E we can conclude, we can conclude $\mathcal{M}, n \models (\neg E)_\alpha$.

$BCA_{\mathbf{1g}}$**:** In this case we have a derivation

$$\varphi_1, \varphi_2, \dots, \varphi_n \vdash (\neg E)_\alpha.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \alpha C \alpha$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash (\neg E)_\alpha$. Therefore according the axiom C_1 and the definition of E, we can conclude $\mathcal{M}, n \vDash \alpha C \alpha$.

BCA

$BCA_{\mathbf{2g}}$**:** In this case we have a derivation

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha C \beta.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \beta C \alpha$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash \alpha C \beta$ i.e., $n[\alpha] C n[\beta]$. Therefore according the axiom C_2, we can conclude $\mathcal{M}, n \vDash \beta C \alpha$.

$BCA_{\mathbf{3g}}$**:** In this case we have following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha C \beta \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \beta = \beta \cdot \gamma.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \alpha C \gamma$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash \alpha C \beta$ i.e., and $\mathcal{M}, n \models \beta = \beta \cdot \gamma$, i.e., $n[\alpha] C n[\beta]$ and $n[\beta] = n[\beta] \cdot n[\gamma]$. The latter is equivalent to $n[\beta] \leq n[\gamma]$. Therefore according the axiom C_3, we can conclude $\mathcal{M}, n \vDash \alpha C \gamma$.

$BCA_{\mathbf{4}}$**:** In this case we have following derivations

$$\varphi_1, \varphi_2, \ldots, \varphi_n \vdash \alpha C(\beta + \gamma),$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n, \alpha C \beta \vdash \psi_\delta \text{ and}$$
$$\varphi_1, \varphi_2, \ldots, \varphi_n, \alpha C \gamma \vdash \psi_\delta.$$

Assume $\mathcal{M}$ is a model and n be an environment so that $\mathcal{M}, n \vDash \varphi_i$ for $i \in \{1, 2, \ldots, n\}$. We have to show that $\mathcal{M}, n \vDash \psi \delta$ or equivalently that $\mathcal{M}, n[\delta] \vDash \psi$. We obtain from the induction hypothesis $\mathcal{M}, n \vDash \alpha C(\beta + \gamma)$ i.e., $n[\alpha] C n[\beta] + n[\gamma]$. From axiom C_4 we get
$n[\alpha] C n[\beta]$
$\mathcal{M}, n \models \psi_\delta$. The second case follows analogously.

□

Chapter 4

Proofs in Natural Deduction

In this chapter, we are going to provide derivations in our calculus that show that the formulas used in the abbreviations of Chapter 2 are indeed equivalent.

The abbreviations used in propositional logic and modal logic are standard, thus we are not going to prove the corresponding equivalence in this thesis. At first, we will focus on the abbreviations defined in this thesis for propositional relevance logic and propositional relevance logic with E. Then we will show that the axioms used in Chapter 2 to force a frame to be a Boolean algebra can be derived in our calculus.

Please note that in the proof tree the superscript numbers on the rules name ($\rightarrow I^1$) denote the applications of the rules sequences and the corresponding assumptions ($[\psi_z]^1$) generated by the rule will have the same superscript. Also, the Boolean algebra axioms listed in Lemma 1 are used as axioms on the proof tree.

4.1 Propositional Relevance Logic

In this section, we will be providing the natural deduction derivations for the abbreviations (PRLAbbr1 - PRLAbbr6) introduced in propositional relevance logic.

1. (PRLAbbr1) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \mathbin{\wedge\!\!\wedge} \psi \leftrightarrow \neg(\varphi \twoheadrightarrow \neg\psi)$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(\varphi \mathbin{\wedge\!\!\wedge} \psi)_z]^1 \qquad \dfrac{\dfrac{[(\varphi \twoheadrightarrow \neg\psi)_z]^2 \quad [\varphi_x]^3 \quad [z = x + y]^3}{(\neg\psi)_y}\,(\twoheadrightarrow E^5) \qquad [\psi_y]^3}{\bot}\,(\neg E^4)}{\bot}\,(\mathbin{\wedge\!\!\wedge} E^3)}{\neg(\varphi \twoheadrightarrow \neg\psi)_z}\,(\neg I^2)}{((\varphi \mathbin{\wedge\!\!\wedge} \psi) \rightarrow \neg(\varphi \twoheadrightarrow \neg\psi))_z}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the $(\mathbin{\wedge\!\!\wedge}$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{[\neg(\varphi \twoheadrightarrow \neg\psi)_z]^1 \qquad \dfrac{\dfrac{\dfrac{[\neg(\varphi \mathbin{\wedge\!\!\wedge} \psi)_z]^2 \qquad \dfrac{[\varphi_x]^4 \quad [\psi_y]^5 \quad [z = x + y]^4}{(\varphi \mathbin{\wedge\!\!\wedge} \psi)_z}\,(\mathbin{\wedge\!\!\wedge} I^7)}{\bot}\,(\neg E^6)}{(\neg\psi)_y}\,(\neg I^5)}{(\varphi \twoheadrightarrow \neg\psi)_z}\,(\twoheadrightarrow I^4)}{\bot}\,(\neg E^3)}{(\varphi \mathbin{\wedge\!\!\wedge} \psi)_z}\,(\neg PBC^2)}{(\neg(\varphi \twoheadrightarrow \neg\psi) \rightarrow (\varphi \mathbin{\wedge\!\!\wedge} \psi))_z}\,(\rightarrow I^1)$$

Also, note that the side-condition in the application of the $(\twoheadrightarrow$ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

2. (PRLAbbr2) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \Vee \psi \leftrightarrow \neg\varphi \twoheadrightarrow \psi$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi \Vee \psi)_z]^1 \quad \dfrac{[(\neg\varphi)_x]^2 \quad [\varphi_x]^4}{\bot}\,(\neg E^5) \quad \dfrac{[(\neg\psi)_y]^3 \quad [\psi_y]^4}{\bot}\,(\neg E^6) \quad [z = x + y]^2}{\bot}\,(\Vee E^4)}{\psi_y}\,(\neg PBC^3)}{(\neg\varphi \twoheadrightarrow \psi)_z}\,(\twoheadrightarrow I^2)}{(\varphi \Vee \psi \to (\neg\varphi \twoheadrightarrow \psi))_z}\,(\to I^1)$$

Please note that the side-condition in the application of the (↠ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\neg\psi)_y]^2 \quad \dfrac{[(\neg\varphi \twoheadrightarrow \psi)_z]^1 \quad [\neg\varphi_x]^3 \quad [z = x + y]^2}{\psi_y}\,(\twoheadrightarrow E^5)}{\bot}\,(\neg E^4)}{\varphi_x}\,(\neg PBC^3)}{(\varphi \Vee \psi)_z}\,(\Vee I1^2)}{(\neg\varphi \twoheadrightarrow \psi \to \varphi \Vee \psi)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\Vee$ I1) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

3. (PRLAbbr3) let $\varphi \in BMRL$, then we have $\vdash N\varphi \leftrightarrow \backsim \neg\varphi$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(N\varphi)_z]^1 \quad [(\neg\varphi)_{z^*}]^2 \quad \dfrac{DNg}{z = z^{**}}}{\bot}\,(NE^3)}{(\backsim \neg\varphi)_z}\,(\backsim I^2)}{(N\varphi \to \backsim \neg\varphi)_z}\,(\to I^1)$$

Please note that the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{[(\backsim \neg\varphi)_z]^1 \quad [(\neg\varphi)_{z^*}]^2 \quad \dfrac{DNg}{z = z^{**}}}{\bot}\,(\backsim E^3)}{(N\varphi)_z}\,(NI^2)}{(\backsim \neg\varphi \to N\varphi)_z}\,(\to I^1)$$

Also, note that the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4. (PRLAbbr4) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \multimap \psi \leftrightarrow N(N\varphi \twoheadrightarrow N\psi)$.

Proof.

$$\dfrac{\dfrac{DMg_2}{x^{**} + y^{**} = (x^* \cdot y^*)^*} \qquad \dfrac{\dfrac{DNg}{x = x^{**}} \quad \dfrac{DNg}{y = y^{**}} \quad [z^* = x + y]^4}{z^* = x^{**} + y^{**}}\,(= E)}{z^* = (x^* \cdot y^*)^*}\,(= E)$$

$$\dfrac{\dfrac{[(\neg\psi)_{y^*}]^7 \qquad \dfrac{[(\varphi \multimap \psi)_z]^1 \quad [\varphi_{x^*}]^9 \quad \dfrac{\dfrac{\vdots}{z^* = (x^* \cdot y^*)^*}}{z = x^* \cdot y^*}\,\text{(by Lemma 2)}}{\psi_{y^*}}\,(\multimap E^{11})}{\bot}\,(\neg E^{10})}{(\neg\varphi)_{x^*}}\,(\neg I^9)$$

$$\dfrac{\dfrac{\dfrac{[(\neg(N\varphi \twoheadrightarrow N\psi))_{z^*}]^2 \qquad \dfrac{\dfrac{\dfrac{[(\neg\psi)_{y^*}]^5 \qquad \dfrac{\dfrac{[(N\varphi)_x]^4 \quad \dfrac{\vdots}{(\neg\varphi)_{x^*}}\,(\neg I^9) \quad \dfrac{DNg}{x = x^{**}}}{\bot}\,(NE^8)}{\psi_{y^*}}\,(\neg PBC^7)}{\bot}\,(\neg E^6)}{(N\psi)_y}\,(NI^5)}{(N\varphi \twoheadrightarrow N\psi)_{z^*}}\,(\twoheadrightarrow I^4)}{\bot}\,(\neg E^3)}{N(N\varphi \twoheadrightarrow N\psi)_z}\,(NI^2)}{(\varphi \multimap \psi \rightarrow N(N\varphi \twoheadrightarrow N\psi))_z}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the (↠ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{DMg_1}{x^{**} \cdot y^{**} = (x^* + y^*)^*} \qquad \dfrac{\dfrac{DNg}{x = x^{**}} \quad \dfrac{DNg}{y = y^{**}} \quad [z = x \cdot y]^6}{z = x^{**} \cdot y^{**}}\,(= E)}{z = (x^* + y^*)^*}\,(= E)$$

$$\dfrac{[(N\varphi \twoheadrightarrow N\psi)_{z^*}]^4 \qquad \dfrac{\dfrac{\dfrac{\dfrac{DNg}{x^{**} = x} \quad [(\neg\psi)_{x^{**}}]^{10}}{(\neg\varphi)_x}\,(= E) \qquad [\varphi_x]^6}{\bot}\,(\neg E^{11})}{(N\varphi)_{x}*}\,(NI^{10}) \qquad \dfrac{\vdots}{z^* = x^* + y^*}\,\text{(by Lemma 2)}}{(N\psi)_{y^*}}\,(\twoheadrightarrow E^9)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[(N(N\varphi \twoheadrightarrow N\psi))_z]^1 \qquad \dfrac{\bot}{(\neg(N\varphi \twoheadrightarrow N\psi))_{z^*}}\,(\neg I^4) \qquad \dfrac{DNg}{z = z^{**}}}{\bot}\,(NE^3)}{(\varphi \multimap \psi)_z}\,(\neg PBC^2)}{(N(N\varphi \twoheadrightarrow N\psi) \to \varphi \multimap \psi)_z}\,(\to I^1)}{}$$

where the premise $\bot$ of $(\neg I^4)$ is derived by

$$\dfrac{[\neg(\varphi \multimap \psi)_z]^2 \qquad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{(N\psi)_{y^*}}\,(\twoheadrightarrow E^9) \qquad \dfrac{\dfrac{DNg}{y = y^{**}} \quad [(\neg\psi)_y]^7}{(\neg\psi)_{y^{**}}}\,(= E) \qquad \dfrac{DNg}{y^* = y^{***}}}{\dfrac{\bot}{\psi_y}\,(\neg PBC^7)}\,(NE^8)}{(\varphi \multimap \psi)_z}\,(\multimap I^6)}{}}{\bot}\,(\neg E^5)$$

Also, note that the side-condition in the application of the (⊸ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

5. (PRLAbbr5) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \parr \psi \leftrightarrow \neg(\varphi \multimap \neg\psi)$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(\varphi \parr \psi)_z]^1 \qquad \dfrac{\dfrac{[(\varphi \multimap \neg\psi)_z]^2 \quad [\varphi_x]^3 \quad [z = x \cdot y]^3}{(\neg\psi)_y}\,(\multimap E^5) \qquad [\psi_y]^3}{\bot}\,(\neg E^4)}{\bot}\,(\parr E^3)}{\neg(\varphi \multimap \neg\psi)_z}\,(\neg I^2)}{((\varphi \parr \psi) \to \neg(\varphi \multimap \neg\psi))_z}\,(\to I^1)$$

Please note that the side-condition in the application of the ($\parr$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{[\neg(\varphi \multimap \neg\psi)_z]^1 \qquad \dfrac{\dfrac{\dfrac{[\neg(\varphi \parr \psi)_z]^2 \qquad \dfrac{[\varphi_x]^4 \quad [\psi_y]^5 \quad [z = x \cdot y]^4}{(\varphi \parr \psi)_z}\,(\parr I^7)}{\bot}\,(\neg E^6)}{(\neg\psi)_y}\,(\neg I^5)}{(\varphi \multimap \neg\psi)_z}\,(\multimap I^4)}{\dfrac{\bot}{(\varphi \parr \psi)_z}\,(\neg PBC^2)}\,(\neg E^3)}{(\neg(\varphi \multimap \neg\psi) \to (\varphi \parr \psi))_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\multimap$ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

6. (PRLAbbr6) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \parr \psi \leftrightarrow \neg\varphi \multimap \psi$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi \parr \psi)_z]^1 \quad \dfrac{[(\neg\varphi)_x]^2 \quad [\varphi_x]^4}{\bot}\,(\neg E^5) \quad \dfrac{[(\neg\psi)_y]^3 \quad [\psi_y]^4}{\bot}\,(\neg E^6) \quad [z = x \cdot y]^2}{\bot}\,(\parr E^4)}{\psi_y}\,(\neg PBC^3)}{(\neg\varphi \multimap \psi)_z}\,(\multimap I^2)}{(\varphi \parr \psi \rightarrow (\neg\varphi \multimap \psi))_z}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the ($\multimap$ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\neg\psi)_y]^2 \quad \dfrac{[(\neg\varphi \multimap \psi)_z]^1 \quad [\neg\varphi_x]^3 \quad [z = x \cdot y]^2}{\psi_y}\,(\multimap E^5)}{\bot}\,(\neg E^4)}{\varphi_x}\,(\neg PBC^3)}{(\varphi \parr \psi)_z}\,(\parr I1^2)}{(\neg\varphi \multimap \psi \rightarrow \varphi \parr \psi)_z}\,(\rightarrow I^1)$$

Also, note that the side-condition in the application of the ($\parr$ I1) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4.2 Propositional Relevance Logic with E

In this section, we will be providing the natural deduction derivations of the abbreviation PRLEAbbr1 used in propositional relevance logic with E.

1. (PRLEAbbr1) let $U, E \in BMRL$, then we have $\vdash U \leftrightarrow NE$.

Proof.

$$\dfrac{\dfrac{[(\neg E)_{z^*}]^2 \qquad \dfrac{\dfrac{\dfrac{DNg}{x^{**} = x} \qquad \dfrac{\dfrac{DMg_2}{(x^* \cdot z)^* = x^{**} + z^*} \qquad \dfrac{\dfrac{[U_z]^1}{x^* = x^* \cdot z}\,(UE^5) \qquad \dfrac{}{x^{**} = x^{**}}\,(= I)}{x^{**} = (x^* \cdot z)^*}\,(= E)}{x^{**} = x^{**} + z^*}\,(= E)}{x = x + z^*}\,(= E)}{E_{z^*}}\,(EI^4)}{\dfrac{\bot}{(NE)_z}\,(NI^2)}\,(\neg E^3)}{(U \rightarrow NE)_z}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the (E I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{DNg}{x^{**} = x} \qquad \dfrac{DNg}{z^{**} = z} \qquad \dfrac{\dfrac{DMg_1}{(x^* + z^*)^* = x^{**} \cdot z^{**}} \qquad \dfrac{\dfrac{[E_{z^*}]^4}{x^* = x^* + z^*}\,(EE^6) \qquad \dfrac{}{x^{**} = x^{**}}\,(= I)}{x^{**} = (x^* + z^*)^*}\,(= E)}{x^{**} = x^{**} \cdot z^{**}}\,(= E)}{x = x \cdot z}\,(= E)}{U_z}\,(UI^5)$$

$$\dfrac{\dfrac{\dfrac{[(NE)_z]^1 \qquad \dfrac{\dfrac{[(\neg U)_z]^2 \quad \dfrac{\vdots}{U_z}\,(UI^5)}{\bot}\,(\neg E^5)}{\neg E_{z^*}}\,(\neg I^4) \qquad \dfrac{DNg}{z = z^{**}}}{\bot}\,(NE^3)}{U_z}\,(\neg PBC^2)}{(NE \to U)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the (U I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

In addition to the proof above, now we are going to natural deduction proof of the $E \leftrightarrow NU$ as well.

2. let $U, E \in BMRL$, then we have $\vdash E \leftrightarrow NU$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(\neg U)_{z^*}]^2 \qquad \dfrac{\dfrac{\dfrac{DNg}{x^{**} = x} \quad \dfrac{\dfrac{DMg_1}{(x^* + z)^* = x^{**} \cdot z^*} \quad \dfrac{\dfrac{[E_z]^1}{x^* = x^* + z}\,(EE^5) \quad \dfrac{}{x^{**} = x^{**}}\,(= I)}{x^{**} = (x^* + z)^*}\,(= E)}{x^{**} = x^{**} \cdot z^*}\,(= E)}{x = x \cdot z^*}\,(= E)}{U_{z^*}}\,(UI^4)}{\bot}\,(\neg E^3)}{(NU)_z}\,(NI^2)}{(E \to NU)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the (U I) rule is satisfied. Furthermore, the proof has no open assumption so

that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{DNg}{x^{**} = x} \quad \dfrac{DNg}{z^{**} = z} \quad \dfrac{\dfrac{DMg_2}{(x^* \cdot z^*)^* = x^{**} + z^{**}} \quad \dfrac{\dfrac{[U_{z^*}]^4}{x^* = x^* \cdot z^*}\,(UE^7) \quad \dfrac{}{x^{**} = x^{**}}\,(= I)}{x^{**} = (x^* \cdot z^*)^*}\,(= E)}{x^{**} = x^{**} + z^{**}}\,(= E)}{x = x + z}\,(= E)}{E_z}\,(EI^6)$$

$$\dfrac{\dfrac{\dfrac{[(NU)_z]^1 \quad \dfrac{\dfrac{[(\neg E)_z]^2 \quad \dfrac{\vdots}{E_z}\,(EI^6)}{\bot}\,(\neg E^5)}{\neg U_{z^*}}\,(\neg I^4) \quad \dfrac{DNg}{z = z^{**}}}{\bot}\,(NE^3)}{E_z}\,(\neg PBC^2)}{(NU \to E)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the (E I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4.3 Boolean Algebra Axioms in PRLE-frame

In this section, we are going to present the natural deduction derivations for the Boolean algebra axioms mentioned in Theorem 1. For Commutativity of ⩓, Commutativity of ⩔, Identity for ⩓ and Identity for ⅋ axioms we have shown that the axioms are valid for both side. We will start with the derivation of the Lemma 6.

1. (Lemma 6) let $\varphi \in BMRL$, then we have $\vdash \varphi \leftrightarrow \backsim\backsim \varphi$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(\backsim\varphi)_{z^*}]^2 \quad \dfrac{\dfrac{DNg}{z = z^{**}} \quad [\varphi_z]^1}{\varphi_{z^{**}}}\,(= E) \quad \dfrac{DNg}{z^* = z^{***}}}{\bot}\,(\backsim E^3)}{\backsim\backsim\varphi_z}\,(\backsim I^2)}{(\varphi \to\backsim\backsim\varphi)_z}\,(\to I^1)$$

Please note that the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{[\backsim\backsim\varphi_z]^1 \quad [\backsim\varphi_{z^*}]^2 \quad \dfrac{DNg}{z = z^{**}}}{\bot}\,(\backsim E^3)}{\varphi_z}\,(\backsim PBC^2)}{(\backsim\backsim\varphi \to \varphi)_z}\,(\to I^1)$$

Also, note that the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

2. (Commutativity of $\Wedge$) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \Wedge \psi \leftrightarrow \psi \Wedge \varphi$.

Proof.

$$\dfrac{\dfrac{[(\varphi\Wedge\psi)_z]^1 \quad \dfrac{[\psi_y]^2 \quad [\varphi_x]^2 \quad \dfrac{\dfrac{Commutativity\ of +}{x + y = y + x} \quad [z = x + y]^2}{z = y + x}\,(= E)}{(\psi\Wedge\varphi)_z}\,(\Wedge I^3)}{(\psi\Wedge\varphi)_z}\,(\Wedge E^2)}{(\varphi\Wedge\psi \to \psi\Wedge\varphi)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the ($\Wedge$ E) rule is satisfied. Furthermore, the proof has no open assumption so

that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{[(\psi ⩓ \varphi)_z]^1 \qquad \dfrac{[\varphi_x]^2 \quad [\psi_y]^2 \quad \dfrac{\dfrac{Commutativity\ of +}{y + x = x + y} \quad [z = y + x]^2}{z = x + y}\,(= E)}{(\varphi ⩓ \psi)_z}\,(⩓ I^3)}{(\varphi ⩓ \psi)_z}\,(⩓ E^2)}{(\psi ⩓ \varphi \to \varphi ⩓ \psi)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the (⩓ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

3. (Commutativity of ⩔) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi ⩔ \psi \leftrightarrow \psi ⩔ \varphi$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(\varphi ⩔ \psi)_z]^1 \quad [\varphi_x]^3 \qquad \dfrac{\dfrac{[(\neg\psi)_y]^2 \quad [\psi_y]^3}{\bot}\,(\neg E^5)}{\varphi_x}\,(\neg PBC^4) \qquad \dfrac{\dfrac{Commutativity\ of\ +}{y + x = x + y} \quad [z = y + x]^2}{z = x + y}\,(= E)}{\varphi_x}\,(⩔ E^3)}{(\psi ⩔ \varphi)_z}\,(⩔ I2^2)}{(\varphi ⩔ \psi \to \psi ⩔ \varphi)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the (⩔ I2) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{[(\psi \Vee \varphi)_z]^1 \qquad [\psi_y]^3 \qquad \dfrac{\dfrac{[(\neg\varphi)_x]^2 \quad [\varphi_x]^3}{\bot}\,(\neg E^5)}{\psi_y}\,(\neg PBC^4) \qquad \dfrac{\dfrac{\mathit{Commutativity\ of}+}{y+x=x+y} \qquad [z=y+x]^2}{z=x+y}\,(=E)}{\psi_y}\,(\Vee E^3)}{(\varphi \Vee \psi)_z}\,(\Vee I2^2)}{(\psi \Vee \varphi \rightarrow \varphi \Vee \psi)_z}\,(\rightarrow I^1)$$

Also, note that the side-condition in the application of the ($\Vee$ I2) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

□

4. (Identity for $\Wedge$) let $\varphi, E \in BMRL$, then we have $\vdash \varphi \leftrightarrow \varphi \Wedge E$.

Proof.

$$\dfrac{\dfrac{[\varphi_z]^1 \qquad \dfrac{\dfrac{\dfrac{\mathit{Complements\ for}\ \cdot}{0=z\cdot z^*} \qquad \dfrac{\dfrac{\mathit{Identity\ for}\ +}{x=x+0} \qquad \dfrac{}{x=x}\,(=I)}{x=x+0}\,(=E)}{x=x+z\cdot z^*}\,(=E)}{E_{z\cdot z^*}}\,(EI^3) \qquad \dfrac{\dfrac{\mathit{Complements\ for}\ \cdot}{0=z\cdot z^*} \qquad \dfrac{\dfrac{\mathit{Identity\ for}\ +}{z=z+0} \qquad \dfrac{}{z=z}\,(=I)}{z=z+0}\,(=E)}{z=z+z\cdot z^*}\,(=E)}{(\varphi \Wedge E)_z}\,(\Wedge I^2)}{(\varphi \rightarrow \varphi \Wedge E)_z}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the (E I) rule is satisfied. Furthermore, the proof has no open assumption so

that its conclusion is valid by the correctness theorem.

$$
\dfrac{\dfrac{\dfrac{[(\varphi \Wedge E)_z]^1 \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[E_y]^3}{x = x + y}\;(EE^5)}{x + y = x}\;(= E) \qquad \dfrac{[z = x + y]^3}{x + y = z}\;(= E)}{z = x}\;(= E)}{z^* = x^*}\;(\text{by Lemma 2}) \qquad [(\backsim \varphi)_{z^*}]^2}{(\backsim \varphi)_{x^*}}\;(= E) \qquad \dfrac{\dfrac{DNg}{x = x^{**}} \qquad [\varphi_x]^3}{\varphi_{x^{**}}}\;(= E) \qquad \dfrac{DNg}{x^* = x^{***}}}{\bot}\;(\backsim E^4)}{\bot}\;(\Wedge E^3)}{\varphi_z}\;(\backsim PBC^2)}{(\varphi \Wedge E \to \varphi)_z}\;(\to I^1)
$$

Also, note that the side-condition in the application of the ($\Wedge$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

□

5. (Identity for $\parr$) let $\varphi, U \in BMRL$, then we have $\vdash \varphi \leftrightarrow \varphi \parr U$.

Proof.

$$\dfrac{\dfrac{[\varphi_z]^1 \qquad \dfrac{\dfrac{\dfrac{\mathit{Complements\ for\ +}}{1 = z + z^*} \qquad \dfrac{\dfrac{\mathit{Identity\ for\ \cdot}}{x = x \cdot 1} \qquad \dfrac{}{x = x}\,(=I)}{x = x \cdot 1}\,(=E)}{x = x \cdot (z + z^*)}\,(=E)}{U_{z+z^*}}\,(UI^3) \qquad \dfrac{\dfrac{\mathit{Complements\ for\ +}}{1 = z + z^*} \qquad \dfrac{\dfrac{\mathit{Identity\ for\ \cdot}}{z = z \cdot 1} \qquad \dfrac{}{z = z}\,(=I)}{z = z \cdot 1}\,(=E)}{z = z \cdot (z + z^*)}\,(=E)}{(\varphi \parr U)_z}\,(\parr I^2)}{(\varphi \to \varphi \parr U)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the (U I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{[(\varphi \parr U)_z]^1 \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[U_y]^3}{x = x \cdot y}\,(UE^5)}{x \cdot y = x}\,(=E) \qquad \dfrac{[z = x \cdot y]^3}{x \cdot y = z}\,(=E)}{z = x}\,(=E)}{z^* = x^*}\,(\text{by Lemma 2}) \qquad [(\backsim \varphi)_{z^*}]^2}{(\backsim \varphi)_{x^*}}\,(=E) \qquad \dfrac{\dfrac{DNg}{x = x^{**}} \qquad [\varphi_x]^3}{\varphi_{x^{**}}}\,(=E) \qquad \dfrac{DNg}{x^* = x^{***}}}{\bot}\,(\backsim E^4)}{\bot}\,(\parr E^3)}{\varphi_z}\,(\backsim PBC^2)}{(\varphi \parr U \to \varphi)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\parr$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

□

6. (Distributivity of $\curlywedge$ on $\parr$) let $\varphi, \psi, \chi \in BMRL$, then we have $\vdash \varphi \curlywedge (\psi \parr \chi) \rightarrow (\varphi \curlywedge \psi) \parr (\varphi \curlywedge \chi)$.

Proof.

$$\dfrac{\dfrac{Distributivity\ of\ +\ over\ \cdot}{x + (y \cdot z) = (x + y) \cdot (x + z)} \qquad \dfrac{[q = y \cdot z]^3 \quad [p = x + q]^2}{p = x + (y \cdot z)}\,(= E)}{p = (x + y) \cdot (x + z)}\,(= E)$$

$$\dfrac{\dfrac{[\varphi_x]^2 \quad [\psi_y]^3 \quad \dfrac{}{x + y = x + y}\,(= I)}{(\varphi \curlywedge \psi)_{x+y}}\,(\curlywedge I^5) \quad \dfrac{[\varphi_x]^2 \quad [\chi_z]^3 \quad \dfrac{}{x + z = x + z}\,(= I)}{(\varphi \curlywedge \chi)_{x+z}}\,(\curlywedge I^6) \quad \dfrac{\vdots}{p = (x + y) \cdot (x + z)}\,(= E)}{((\varphi \curlywedge \psi) \parr (\varphi \curlywedge \chi))_p}\,(\curlywedge I^4)$$

$$\dfrac{\dfrac{[(\varphi \curlywedge (\psi \parr \chi))_p]^1 \qquad \dfrac{[(\psi \parr \chi)_q]^3 \quad \dfrac{\vdots}{((\varphi \curlywedge \psi) \parr (\varphi \curlywedge \chi))_p}\,(\curlywedge I^4)}{((\varphi \curlywedge \psi) \parr (\varphi \curlywedge \chi))_p}\,(\parr E^3)}{((\varphi \curlywedge \psi) \parr (\varphi \curlywedge \chi))_p}\,(\curlywedge E^2)}{(\varphi \curlywedge (\psi \parr \chi) \rightarrow (\varphi \curlywedge \psi) \parr (\varphi \curlywedge \chi))_p}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the ($\curlywedge$ E and $\parr$ E) rules are satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

7. (Distributivity of $\parr$ on $\curlywedge$) let $\varphi, \psi, \chi \in BMRL$, then we have $\vdash \varphi \parr (\psi \curlywedge \chi) \rightarrow (\varphi \parr \psi) \curlywedge (\varphi \parr \chi)$.

Proof.

$$\dfrac{\dfrac{Distributivity\ of\ +\ over\ \cdot}{x \cdot (y + z) = (x \cdot y) + (x \cdot z)} \qquad \dfrac{[q = y + z]^3 \quad [p = x \cdot q]^2}{p = x \cdot (y + z)}\,(= E)}{p = (x \cdot y) + (x \cdot z)}\,(= E)$$

$$\dfrac{\dfrac{[\varphi_x]^2 \quad [\psi_y]^3 \quad \dfrac{}{x+y=x+y}(=I)}{(\varphi \parr \psi)_{x\cdot y}}(\parr I^5) \quad \dfrac{[\varphi_x]^2 \quad [\chi_z]^3 \quad \dfrac{}{x+z=x+z}(=I)}{(\varphi \parr \chi)_{x\cdot z}}(\parr I^6) \quad \dfrac{\vdots}{p=(x\cdot y)+(x\cdot z)}(=E)}{((\varphi \parr \psi) \Wedge (\varphi \parr \chi))_p}(\Wedge I^4)$$

$$\dfrac{\dfrac{[(\varphi \parr (\psi \Wedge \chi))_p]^1 \quad \dfrac{[(\psi \Wedge \chi)_q]^3 \quad \dfrac{\vdots}{((\varphi \parr \psi) \Wedge (\varphi \parr \chi))_p}(\Wedge I^4)}{((\varphi \parr \psi) \Wedge (\varphi \parr \chi))_p}(\Wedge E^3)}{((\varphi \parr \psi) \Wedge (\varphi \parr \chi))_p}(\parr E^2)}{(\varphi \parr (\psi \Wedge \chi) \to (\varphi \parr \psi) \Wedge (\varphi \parr \chi))_p}(\to I^1)$$

Please note that the side-condition in the application of the ($\parr$ E and $\Wedge$ E) rules are satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

8. (Complements for $\Wedge$) let $\varphi, E \in BMRL$, then we have $\vdash \varphi \to \top \parr (U \wedge (\varphi \Wedge N\varphi))$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{Complements\ for\ +}{1=z+z^*} \quad \dfrac{\dfrac{Identity\ for\ \cdot}{x=x\cdot 1} \quad \dfrac{}{x=x}(=I)}{x=x\cdot 1}(=E)}{x=x\cdot(z+z^*)}(=E)}{U_{z+z^*}}(UI^4) \qquad \dfrac{[\varphi_z]^1 \quad \dfrac{\dfrac{[\varphi_z]^1 \quad \dfrac{\dfrac{DNg}{z^{**}=z} \quad [(\neg\varphi)_{z^{**}}]^6}{(\neg\varphi)_z}(=E)}{\bot}(\neg E^7)}{(N\varphi)_{z^*}}(NI^6) \quad \dfrac{}{z+z^*=z+z^*}(=I)}{(\varphi \Wedge N\varphi)_{z+z^*}}(\Wedge I^5)}{(U \wedge (\varphi \Wedge N\varphi))_{z+z^*}}(\wedge I^3)$$

$$\dfrac{\dfrac{\vdots}{(U \wedge (\varphi \mathbin{\wedge\!\!\!\wedge} N\varphi))_{z+z^*}}\;(\wedge I^3) \qquad \dfrac{\dfrac{\mathit{Complements\ for} +}{1 = z + z^*} \quad \dfrac{\dfrac{\mathit{Identity\ for} \cdot}{z = 1 \cdot z} \quad \dfrac{}{z = z}\;(=I)}{z = 1 \cdot z}\;(=E)}{z = (z + z^*) \cdot z}\;(=E)}{\dfrac{(\top \parr (U \wedge (\varphi \mathbin{\wedge\!\!\!\wedge} N\varphi)))_z}{(\varphi \to \top \parr (U \wedge (\varphi \mathbin{\wedge\!\!\!\wedge} N\varphi)))_z}\;(\to I^1)}\;(\parr I^2)$$

Please note that the side-condition in the application of the (U I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

9. (Complements for $\parr$) let $\varphi, U \in BMRL$, then we have $\vdash \varphi \to \top \mathbin{\wedge\!\!\!\wedge} (E \wedge (\varphi \parr N\varphi))$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathit{Complements\ for} \cdot}{0 = z \cdot z^*} \quad \dfrac{\dfrac{\mathit{Identity\ for} +}{x = x + 0} \quad \dfrac{}{x = x}\;(=I)}{x = x + 0}\;(=E)}{x = x + (z \cdot z^*)}\;(=E)}{E_{z \cdot z^*}}\;(EI^4) \qquad \dfrac{[\varphi_z]^1 \quad \dfrac{\dfrac{[\varphi_z]^1 \quad \dfrac{\dfrac{DNg}{z^{**} = z} \quad [(\neg\varphi)_{z^{**}}]^6}{(\neg\varphi)_z}\;(=E)}{\bot}\;(\neg E^7)}{(N\varphi)_{z^*}}\;(NI^6) \quad \dfrac{}{z \cdot z^* = z \cdot z^*}\;(=I)}{(\varphi \parr N\varphi)_{z \cdot z^*}}\;(\parr I^5)}{(E \wedge (\varphi \parr N\varphi))_{z \cdot z^*}}\;(\wedge I^3)$$

$$\dfrac{\dfrac{\vdots}{(E \wedge (\varphi \parr N\varphi))_{z \cdot z^*}}\;(\wedge I^3) \qquad \dfrac{\dfrac{\mathit{Complements\ for} \cdot}{0 = z \cdot z^*} \quad \dfrac{\dfrac{\mathit{Identity\ for} +}{z = z + 0} \quad \dfrac{}{z = z}\;(=I)}{z = z + 0}\;(=E)}{z = z + (z \cdot z^*)}\;(=E)}{\dfrac{(\top \mathbin{\wedge\!\!\!\wedge} (E \wedge (\varphi \parr N\varphi)))_z}{(\varphi \to \top \mathbin{\wedge\!\!\!\wedge} (E \wedge (\varphi \parr N\varphi)))_z}\;(\to I^1)}\;(\mathbin{\wedge\!\!\!\wedge} I^2)$$

Please note that the side-condition in the application of the (E I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4.4 Additional Boolean Algebra Axioms in PRLE-frame

In this section, we are going to present some additional properties such as Commutativity of $\parr$, $\invamp$, and Associativity for $\Wedge$, $\Vee$, $\parr$, $\invamp$ of Boolean algebra as follows:

1. (Commutativity of $\parr$) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \parr \psi \leftrightarrow \psi \parr \varphi$.

Proof.

$$\dfrac{\dfrac{[(\varphi \parr \psi)_z]^1 \qquad \dfrac{[\psi_y]^2 \quad [\varphi_x]^2 \qquad \dfrac{\dfrac{Commutativity\ of\ \cdot}{x \cdot y = y \cdot x} \qquad [z = x \cdot y]^2}{z = y \cdot x}\,(=E)}{(\psi \parr \varphi)_z}\,(\parr I^3)}{(\psi \parr \varphi)_z}\,(\parr E^2)}{(\varphi \parr \psi \to \psi \parr \varphi)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the ($\parr$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{[(\psi \parr \varphi)_z]^1 \qquad \dfrac{[\varphi_x]^2 \quad [\psi_y]^2 \qquad \dfrac{\dfrac{Commutativity\ of\ \cdot}{y \cdot x = x \cdot y} \qquad [z = y \cdot x]^2}{z = x \cdot y}\,(=E)}{(\varphi \parr \psi)_z}\,(\parr I^3)}{(\varphi \parr \psi)_z}\,(\parr E^2)}{(\psi \parr \varphi \to \varphi \parr \psi)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the (ᛟ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

2. (Commutativity of ɤ) let $\varphi, \psi \in BMRL$, then we have $\vdash \varphi \text{ ɤ } \psi \leftrightarrow \psi \text{ ɤ } \varphi$.

Proof.

$$\dfrac{\dfrac{\dfrac{[(\varphi \text{ ɤ } \psi)_z]^1 \quad [\varphi_y]^3 \quad \dfrac{\dfrac{[(\neg\psi)_x]^2 \quad [\psi_x]^3}{\bot}\,(\neg E^5)}{\varphi_y}\,(\neg PBC^4) \quad \dfrac{\dfrac{\mathit{Commutativity\ of\ \cdot}}{x \cdot y = y \cdot x} \quad [z = x \cdot y]^2}{z = y \cdot x}\,(= E)}{\varphi_y}\,(\text{ɤ}E^3)}{(\psi \text{ ɤ } \varphi)_z}\,(\text{ɤ}I2^2)}{(\varphi \text{ ɤ } \psi \to \psi \text{ ɤ } \varphi)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the (ɤ I2) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{\dfrac{[(\psi \text{ ɤ } \varphi)_z]^1 \quad [\psi_y]^3 \quad \dfrac{\dfrac{[(\neg\varphi)_x]^2 \quad [\varphi_x]^3}{\bot}\,(\neg E^5)}{\psi_y}\,(\neg PBC^4) \quad \dfrac{\dfrac{\mathit{Commutativity\ of\ \cdot}}{x \cdot y = y \cdot x} \quad [z = x \cdot y]^2}{z = y \cdot x}\,(= E)}{\psi_y}\,(\text{ɤ}E^3)}{(\varphi \text{ ɤ } \psi)_z}\,(\text{ɤ}I2^2)}{(\psi \text{ ɤ } \varphi \to \varphi \text{ ɤ } \psi)_z}\,(\to I^1)$$

Also, note that the side-condition in the application of the (ɤ I2) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

3. (Associativity for $\Wedge$) let $\varphi, \psi, \chi \in BMRL$, then we have $\vdash (\varphi \Wedge \psi) \Wedge \chi \leftrightarrow \varphi \Wedge (\psi \Wedge \chi)$.

Proof.

$$\dfrac{\dfrac{Ass_1}{(x+y)+z = x+(y+z)} \qquad \dfrac{[q = x+y]^3 \quad [p = q+z]^2}{p = (x+y)+z}\,(=E)}{p = x+(y+z)}\,(=E)$$

$$\dfrac{[((\varphi \Wedge \psi) \Wedge \chi)_p]^1 \qquad \dfrac{[(\varphi \Wedge \psi)_q]^2 \qquad \dfrac{[\varphi_x]^3 \qquad \dfrac{[\psi_y]^3 \quad [\chi_z]^2 \quad \dfrac{}{y+z = y+z}\,(=I)}{(\psi \Wedge \chi)_{y+z}}\,(\Wedge I^5) \qquad \dfrac{\vdots}{p = x+(y+z)}\,(=E)}{(\varphi \Wedge (\psi \Wedge \chi))_p}\,(\Wedge I^4)}{(\varphi \Wedge (\psi \Wedge \chi))_p}\,(\Wedge E^3)}{\dfrac{(\varphi \Wedge (\psi \Wedge \chi))_p}{((\varphi \Wedge \psi) \Wedge \chi \to \varphi \Wedge (\psi \Wedge \chi))_p}\,(\to I^1)}\,(\Wedge E^2)$$

Please note that the side-condition in the application of the ($\Wedge$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{Ass_1}{x+(y+z) = (x+y)+z} \qquad \dfrac{[q = y+z]^3 \quad [p = x+q]^2}{p = x+(y+z)}\,(=E)}{p = (x+y)+z}\,(=E)$$

$$\dfrac{\dfrac{[(\varphi \doublewedge (\psi \doublewedge \chi))_p]^1 \quad \dfrac{[(\psi \doublewedge \chi)_q]^2 \quad \dfrac{[\chi_z]^3 \quad \dfrac{[\psi_y]^3 \quad [\varphi_x]^2 \quad \dfrac{}{y + x = y + x}\,(= I)}{(\varphi \doublewedge \psi)_{x+y}}\,(\doublewedge I^5) \quad \dfrac{\vdots}{p = (x + y) + z}\,(= E)}{((\varphi \doublewedge \psi) \doublewedge \chi)_p}\,(\doublewedge I^4)}{((\varphi \doublewedge \psi) \doublewedge \chi)_p}\,(\doublewedge E^3)}{((\varphi \doublewedge \psi) \doublewedge \chi)_p}\,(\doublewedge E^2)}{(\varphi \doublewedge (\psi \doublewedge \chi) \to (\varphi \doublewedge \psi) \doublewedge \chi)_p}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\doublewedge$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4. (Associativity for $\doublevee$) let $\varphi, \psi, \chi \in BMRL$, then we have $\vdash (\varphi \doublevee \psi) \doublevee \chi \leftrightarrow \varphi \doublevee (\psi \doublevee \chi)$.

Proof.

$$\dfrac{[(\varphi \doublevee \psi)_{x+y}]^4 \quad \dfrac{\dfrac{[(\neg\varphi)_x]^2 \quad [\varphi_x]^5}{\bot}\,(\neg E^7)}{\chi_z}\,(\neg PBC^6) \quad \dfrac{\dfrac{[(\neg\psi)_y]^3 \quad [\psi_y]^5}{\bot}\,(\neg E^9)}{\chi_z}\,(\neg PBC^8) \quad \dfrac{}{x + y = x + y}\,(= I)}{\chi_z}\,(\doublevee E^5)$$

$$\dfrac{\dfrac{\dfrac{[((\varphi \doublevee \psi) \doublevee \chi)_p]^1 \quad \dfrac{\vdots}{\chi_z}\,(\doublevee E^5) \quad [\chi_z]^4 \quad \dfrac{\dfrac{Ass_1}{x + (y + z) = (x + y) + z} \quad \dfrac{[q = y + z]^3 \quad [p = x + q]^2}{p = x + (y + z)}\,(= E)}{p = (x + y) + z}\,(= E)}{\dfrac{\chi_z}{(\psi \doublevee \chi)_q}\,(\doublevee I1^3)}\,(\doublevee E^4)}{(\varphi \doublevee (\psi \doublevee \chi))_p}\,(\doublevee I1^2)}{((\varphi \doublevee \psi) \doublevee \chi \to \varphi \doublevee (\psi \doublevee \chi))_p}\,(\to I^1)$$

Please note that the side-condition in the application of the ($\doublevee$ I1) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{[(\psi \Vee \chi)_{y+z}]^4 \quad \dfrac{\dfrac{[(\neg\psi)_y]^3 \quad [\psi_y]^5}{\bot}\,(\neg E^7)}{\varphi_x}\,(\neg PBC^6) \quad \dfrac{\dfrac{[(\neg\chi)_z]^2 \quad [\chi_z]^5}{\bot}\,(\neg E^9)}{\varphi_x}\,(\neg PBC^8) \quad \dfrac{}{y+z=y+z}\,(= I)}{\varphi_x}\,(\Vee E^5)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi \Vee (\psi \Vee \chi))_p]^1 \quad \dfrac{\vdots}{\varphi_x}\,(\Vee E^5) \quad [\varphi_x]^4 \quad \dfrac{\dfrac{Ass_1}{(x+y)+z = x+(y+z)} \quad \dfrac{[q = x+y]^3 \quad [p = q+z]^2}{p = (x+y)+z}\,(= E)}{p = x+(y+z)}\,(= E)}{\varphi_x}\,(\Vee E^4)}{(\varphi \Vee \psi)_q}\,(\Vee I1^3)}{((\varphi \Vee \psi) \Vee \chi)_p}\,(\Vee I1^2)}{(\varphi \Vee (\psi \Vee \chi) \to (\varphi \Vee \psi) \Vee \chi)_p}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\Vee$ I1) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

5. (Associativity for $\parr$) let $\varphi, \psi, \chi \in BMRL$, then we have $\vdash (\varphi \parr \psi) \parr \chi \leftrightarrow \varphi \parr (\psi \parr \chi)$.

Proof.

$$\dfrac{\dfrac{Ass_2}{(x \cdot y) \cdot z = x \cdot (y \cdot z)} \quad \dfrac{[q = x \cdot y]^3 \quad [p = q \cdot z]^2}{p = (x \cdot y) \cdot z}\,(= E)}{p = x \cdot (y \cdot z)}\,(= E)$$

$$\dfrac{\dfrac{[((\varphi\parr\psi)\parr\chi)_p]^1 \qquad \dfrac{[(\varphi\parr\psi)_q]^2 \qquad \dfrac{[\varphi_x]^3 \qquad \dfrac{[\psi_y]^3 \quad [\chi_z]^2 \quad \dfrac{}{y+z=y+z}\,(=I)}{(\psi\parr\chi)_{y\cdot z}}\,(\parr I^5) \qquad \dfrac{\vdots}{p=x\cdot(y\cdot z)}\,(=E)}{(\varphi\parr(\psi\parr\chi))_p}\,(\parr I^4)}{(\varphi\parr(\psi\parr\chi))_p}\,(\parr E^3)}{(\varphi\parr(\psi\parr\chi))_p}\,(\parr E^2)}{((\varphi\parr\psi)\parr\chi\to\varphi\parr(\psi\parr\chi))_p}\,(\to I^1)$$

Please note that the side-condition in the application of the ($\parr$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{\dfrac{Ass_2}{x\cdot(y\cdot z)=(x\cdot y)\cdot z} \qquad \dfrac{[q=y\cdot z]^3 \quad [p=x\cdot q]^2}{p=x\cdot(y\cdot z)}\,(=E)}{p=(x\cdot y)\cdot z}\,(=E)$$

$$\dfrac{\dfrac{[(\varphi\parr(\psi\parr\chi))_p]^1 \qquad \dfrac{[(\psi\parr\chi)_q]^2 \qquad \dfrac{[\chi_z]^3 \qquad \dfrac{[\varphi_x]^2 \quad [\psi_y]^3 \quad \dfrac{}{y+x=y+x}\,(=I)}{(\varphi\parr\psi)_{x\cdot y}}\,(\parr I^5) \qquad \dfrac{\vdots}{p=(x\cdot y)\cdot z}\,(=E)}{((\varphi\parr\psi)\parr\chi)_p}\,(\parr I^4)}{((\varphi\parr\psi)\parr\chi)_p}\,(\parr E^3)}{((\varphi\parr\psi)\parr\chi)_p}\,(\parr E^2)}{(\varphi\parr(\psi\parr\chi)\to(\varphi\parr\psi)\parr\chi)_p}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\parr$ E) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

6. (Associativity for $\parr$) let $\varphi, \psi, \chi \in BMRL$, then we have $\vdash (\varphi \parr \psi) \parr \chi \leftrightarrow \varphi \parr (\psi \parr \chi)$.

Proof.

$$\dfrac{[(\varphi \parr \psi)_{x\cdot y}]^4 \quad \dfrac{\dfrac{[(\neg\varphi)_x]^2 \quad [\varphi_x]^5}{\bot}\,(\neg E^7)}{\chi_z}\,(\neg PBC^6) \quad \dfrac{\dfrac{[(\neg\psi)_y]^3 \quad [\psi_y]^5}{\bot}\,(\neg E^9)}{\chi_z}\,(\neg PBC^8) \quad \dfrac{}{x\cdot y = x\cdot y}\,(=I)}{\chi_z}\,(\parr E^5)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[((\varphi\parr\psi)\parr\chi)_p]^1 \quad \dfrac{\vdots}{\chi_z}\,(\parr E^5) \quad [\chi_z]^4 \quad \dfrac{\dfrac{Ass_2}{x\cdot(y\cdot z) = (x\cdot y)\cdot z} \quad \dfrac{[q = y\cdot z]^3 \quad [p = x\cdot q]^2}{p = x\cdot(y\cdot z)}\,(=E)}{p = (x\cdot y)\cdot z}\,(=E)}{\chi_z}\,(\parr E^4)}{(\psi\parr\chi)_q}\,(\parr I2^3)}{(\varphi\parr(\psi\parr\chi))_p}\,(\parr I2^2)}{((\varphi\parr\psi)\parr\chi \to \varphi\parr(\psi\parr\chi))_p}\,(\to I^1)$$

Please note that the side-condition in the application of the ($\parr$ I2) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem.

$$\dfrac{[(\psi \parr \chi)_{y\cdot z}]^4 \quad \dfrac{\dfrac{[(\neg\psi)_y]^3 \quad [\psi_y]^5}{\bot}\,(\neg E^7)}{\varphi_x}\,(\neg PBC^6) \quad \dfrac{\dfrac{[(\neg\chi)_z]^2 \quad [\chi_z]^5}{\bot}\,(\neg E^9)}{\varphi_x}\,(\neg PBC^8) \quad \dfrac{Commutativity\ of\ \cdot}{y\cdot z = y\cdot z}}{\varphi_x}\,(\parr E^5)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi\parr(\psi\parr\chi))_p]^1 \quad \dfrac{\vdots}{\varphi_x}\,(\parr E^5) \quad [\varphi_x]^4 \quad \dfrac{\dfrac{Ass_2}{(x\cdot y)\cdot z = x\cdot(y\cdot z)} \quad \dfrac{[q = x\cdot y]^3 \quad [p = q\cdot z]^2}{p = (x\cdot y)\cdot z}\,(=E)}{p = x\cdot(y\cdot z)}\,(=E)}{\varphi_x}\,(\parr E^4)}{(\varphi\parr\psi)_q}\,(\parr I1^3)}{((\varphi\parr\psi)\parr\chi)_p}\,(\parr I1^2)}{(\varphi\parr(\psi\parr\chi) \to (\varphi\parr\psi)\parr\chi)_p}\,(\to I^1)$$

Also, note that the side-condition in the application of the ($\stackrel{\vee}{\circ}$ I1) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4.5 Boolean Contact Algebra Axioms in MRL-frame

Now we are going provide the natural deduction derivations of the Boolean contact algebra axioms mentioned in Theorem 2.

1. (BCAx0) let $E \in BMRL$, then we have $[C]\neg E$.

   Proof.

$$\dfrac{\dfrac{[zCy]^1}{(\neg E)_y}\;(BCA_{0g}{}^2)}{([C]\neg E)_z}\;([C]I^1)$$

   Please note that the side-condition in the application of the ($[C]$ I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

2. (BCAx1) let $E, \varphi \in BMRL$, then we have $\neg E \to ([C]\varphi \to \varphi)$.

   Proof.

$$\dfrac{\dfrac{\dfrac{[([C]\varphi)_z]^2 \qquad \dfrac{[(\neg E)_z]^1}{zCz}\;(BCA_{1g}{}^4)}{\varphi_z}\;([C]E^3)}{([C]\varphi \to \varphi)_z}\;(\to I^2)}{(\neg E \to ([C]\varphi \to \varphi))_z}\;(\to I^1)$$

   Please note that the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

3. (BCAx2) let $\varphi \in BMRL$, then we have $\varphi \to [C]\langle C\rangle\varphi$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{[zCy]^2}{yCz}\,(BCA_{2g}{}^4)\qquad [\varphi_z]^1}{(\langle C\rangle\varphi)_z}\,(\langle C\rangle I^3)}{([C]\langle C\rangle\varphi)_z}\,([C]I^2)}{(\varphi \to [C]\langle C\rangle\varphi)_z}\,(\to I^1)$$

Please note that the side-condition in the application of the ([C] I) rule is satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

4. (BCAx3) let $\top, \varphi \in BMRL$, then we have $[C]\varphi \to [C](\top \multimap \varphi)$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{[([C]\varphi)_z]^1 \qquad \dfrac{[zCy]^2 \qquad \dfrac{\dfrac{[y = x\cdot u]^3}{x\cdot u = y}\,(=E) \qquad \dfrac{\dfrac{Ass_2}{x\cdot(u\cdot u) = x\cdot u\cdot u} \qquad \dfrac{\dfrac{Idm_2}{u = u\cdot u} \quad \dfrac{}{x\cdot u = x\cdot u}\,(=I)}{x\cdot u = x\cdot(u\cdot u)}\,(=E)}{x\cdot u = x\cdot u\cdot u}\,(=E)}{y = y\cdot u}\,(=E)}{zCu}\,(BCA_{3g}{}^5)}{\varphi_u}\,([C]E^4)}{(\top \multimap \varphi)_y}\,(\multimap I^3)}{([C](\top \multimap \varphi))_z}\,([C]I^2)}{([C]\varphi \to [C](\top \multimap \varphi))_z}\,(\to I^1)$$

Please note that the side-condition in the application of the ([C] I and $\multimap$ I) rules are satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

5. (BCAx4) let $\varphi \in BMRL$, then we have $\varphi \to [C](\neg\langle C\rangle\varphi \twoheadrightarrow \langle C\rangle\varphi)$.

Proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[y = u + v]^3 \quad [zCy]^2}{zC(u+v)}\,(= E) \qquad \dfrac{\dfrac{[(\neg\langle C\rangle\varphi)_u]^3 \qquad \dfrac{\dfrac{[zCu]^4}{uCz}\,(BCA_{2\mathrm{g}}{}^{8}) \quad [\varphi_z]^1}{(\langle C\rangle\varphi)_u}\,(\langle C\rangle I^7)}{\bot}\,(\neg E^6)}{(\langle C\rangle\varphi)_v}\,(\neg PBC^5) \qquad \dfrac{\dfrac{[zCv]^4}{vCz}\,(BCA_{2\mathrm{g}}{}^{10}) \quad [\varphi_z]^1}{(\langle C\rangle\varphi)_v}\,(\langle C\rangle I^9)}{(\langle C\rangle\varphi)_v}\,(BCA_4{}^4)}{(\neg\langle C\rangle\varphi \twoheadrightarrow \langle C\rangle\varphi)_y}\,(\twoheadrightarrow I^3)}{([C](\neg\langle C\rangle\varphi \twoheadrightarrow \langle C\rangle\varphi))_z}\,([C]I^2)}{(\varphi \rightarrow [C](\neg\langle C\rangle\varphi \twoheadrightarrow \langle C\rangle\varphi))_z}\,(\rightarrow I^1)$$

Please note that the side-condition in the application of the ($[C]$ I and ↠ I) rules are satisfied. Furthermore, the proof has no open assumption so that its conclusion is valid by the correctness theorem. □

Chapter 5

The Coq Proof Assistant

In this chapter, we will discuss the features of Coq used in this thesis. Coq is an interactive theorem prover, and a typed functional programming language. GALLINA is the specification language of Coq, based on the Calculus of Inductive Constructions (CIC). CIC is a higher order logic and typed functional programming language. Mathematical equations, theories, axioms, and examples can be proven formally in Coq. The Coq type checking algorithm validates the correctness of definitions and the proofs. Coq enriches with inbuilt tactics for mathematical proofs, advanced notations, proof search, and modular function development to design a framework. It also supports to write user-defined tactics in Ltac language. More details can be found in [10, 21].

5.1 Sort

Coq object like terms, functions, proofs, and types has its own type, also referred as *sorts*. It classifies *sorts* into three categories, i.e., *set*, *prop*, and *type*. Type hierarchical scope of those *sorts* is like the hierarchy of the universe. Coq commands *Check*, and *Print* are used to display the type of each Coq object, and its definition. In the following subsections, we provide a brief description of these three *sorts*.

5.1.1 Set

In Coq *set* is the universe of all programs, it is the sub-universe of the type universe. It also includes natural number type (nat), and Boolean types (bool).

```
Check Set.
```

```
Set
: Type
```

The natural number *nat* definition is an inductive definition in Coq, while it define *0*, but others are the successors of zero (0) denoted by S. Therefore, one denotes by $S(0)$.

```
Check nat.


Print nat.

Check 0.


Check S(0).
```

```
nat
: Set

Inductive nat : Set := O : nat | S : nat ->
    nat
0
: nat

1
: nat
```

The boolean type *bool* definition is an inductive definition in Coq, with two constants true, and false.

```
Check bool.

Print bool.
```

```
bool
: Set
Inductive bool : Set := true : bool | false
    : bool
```

It is worth mentioning that every Coq statement ends with a period.

5.1.2 Prop

In Coq, *prop* is the type of all propositions, and every proof, is a *prop*. It is the sub-universe of the type universe.

```
Check Prop.
```

```
Prop
: Type
```

For example, binary function *less than* (*lt*) takes two natural numbers, and return type *prop* i.e., the witness. A shorthand notation (<) denotes *lt* function.

```
Check lt.

Check lt 1 2.

Check 1 < 2.
```

```
lt
: nat -> nat -> Prop
1 < 2
: Prop
1 < 2
: Prop
```

5.1.3 Type

In Coq, *type* is the topmost universe that includes *set*, and *prop* sub-universes. It is the abstract type.

```
Check Type.
```

```
Type
: Type
```

For example, binary function *less than* (*lt*) return an abstract type.

```
Check (nat -> nat -> Prop).
```

```
nat -> nat -> Prop
: Type
```

It is worth mentioning that we use only *type*, and *prop* sorts in this thesis. Thus, we concentrate only on *type*, and *prop* in the remainder of this chapter.

5.2 Logical Operators

Now that we have seen some primitive features of Coq, at this point we want to concentrate on inbuilt logical operators such as conjunction (∧), disjunction (∨), not (∽), implication (→), and material equivalence (↔). Coq command *Locate* is used to display the definition of any notations.

Conjunction operator on two props is defined by the binary function *and* in Coq. A shorthand notation (∧) denote the operator.

```
Locate "/\".
```

```
Notation
"A /\ B" := and A B : type_scope (default
    interpretation)
```

The disjunction operator on two props is defined by the binary function *or* in Coq. A shorthand notation (∨) denote the operator.

```
Locate "\/".
```

```
Notation
"A \/ B" := or A B : type_scope (default
    interpretation)
```

Negation operator on a prop is defined by the unary function *not* in Coq. A shorthand notation (∽) denote the operator.

```
Locate "∽".
```

```
Notation
"∽ x":= not x : type_scope (default
    interpretation)
```

Implication operator relates two props is defined by *forall* keyword in Coq. A shorthand notation (→) denote the operator.

```
Locate "->".
```

```
Notation
"A -> B" := forall _ : A, B : type_scope
    (default interpretation)
```

Material equivalence operator relates two props is defined by *iff* keyword in Coq. A shorthand notation (↔) denote the operator.

```
Locate "<->".
```

```
Notation
"A <-> B" := iff A B : type_scope (default
    interpretation)
```

Please note we will be using these logical operators extensively in the implementation of our logic in Coq.

5.3 Classes

Coq type overloading allows defining a class object of an abstract type with its properties and then to create an instance of a particular type. It also allows creating type class hierarchies. Consider the class definition as follows:

```
Class SigName ($\alpha_1$ : $\tau_1$) ($\alpha_2$ : $\tau_2$) ... ($\alpha_N$ : $\tau_N$):= {
    $P_1$            :  $\Phi_1$;
    $P_2$            :  $\Phi_2$;
          ⋮
    $P_N$            :  $\Phi_N$
}.
```

Here $SigName$ is the class name, and it takes properties P_1, P_2, ... P_N of abstract types τ_1, τ_2, ... τ_N, and Φ_1, Φ_2, ... Φ_N are the formulas. Now to demonstrate the overloading feature we consider $SigName$ as the superclass.

Now, we want to declare a class inheriting the properties from the superclass $SigName$. The syntax of subclass *Name* definition is as follows:

```
Class Name ($\beta_1$ : $\tau_1$) ($\beta_2$ : $\tau_2$) ... ($\beta_N$ : $\tau_N$):= {
     Sig              : > SigName $\alpha_1$, $\alpha_2$, ... $\alpha_n$;
    $Q_1$              : $\Psi_1$;
    $Q_2$              : $\Psi_2$;
            ⋮
    $Q_N$              : $\Psi_N$
}.
```

Please note that, property Sig holds inherited properties of the superclass $SigName$. However, class *Name* has additional properties Q_1, Q_2, ... Q_N, and Ψ_1, Ψ_2, ... Ψ_N are the formulas.

Coq instances have not used this thesis; thus, we want to conclude this section with the syntax of instance definition. Coq syntax of instance definition for superclass, and subclass are as follows:

```
Instance ISigName : SigName $t_1$, $t_2$, ... $t_N$ := {
    $p_1$            : $\varphi_1$;
    $p_2$            : $\varphi_2$;
          ⋮
    $p_N$            : $\varphi_N$
}.
```

```
Instance IName : Name $t_1$, $t_2$, ... $t_N$ := {
     Sig              : > ISigName;
    $q_1$              : $\psi_1$;
    $q_2$              : $\psi_2$;
            ⋮
    $q_N$              : $\psi_N$
}.
```

More details on Coq classes, and instances are available in [20].

5.4 Tactic and Proof

Proofs are the mathematically proven fact, often written as lemma or a theorem in Coq. Proof steps in Coq are similar to the handwritten proof steps; thus, it is called an interactive theorem prover. Tactics are the Coq commands used to manipulate the state of a proof. The

Coq library is enriched with many inbuilt tactics. In this thesis, we have used only a few of them. The full list of tactics is available in [21].

An illustration of Coq proof, and applications of inbuilt tactics is as follows:

```
Lemma DNG : forall P : Prop, P ->
    ∽∽P.
Proof.
```

```
1 subgoal
________________________________________(1/1)
forall P : Prop, P -> ∽ ∽ P
```

Please consider the Coq code above, the left-hand side is the program window, and the right-hand side is the output window of CoqIde. The first line declare the lemma named *DNG*, and Coq immediately generates the goal on the output window. Now, we begin assuming an arbitrary assumption *P* using the tactic *intro*.

```
intro p.
```

```
1 subgoal
p : Prop
________________________________________(1/1)
p -> ∽ ∽ p
```

The new goal is an implication. Now, we use the tactic *intro* to introduce an arbitrary hypothesis $H0$.

```
intro H0.
```

```
1 subgoal
p : Prop
H0 : p
________________________________________(1/1)
∽ ∽ p
```

The goal has two *not* (∽) in front of the proposition *P*. We use the tactic *unfold* to unfold the definition of both the *not* (∽) in the goal.

```
unfold not.
```

```
1 subgoal
p : Prop
H0 : p
________________________________________(1/1)
(p -> False) -> False
```

Again the goal is an implication we use the same tactic *intro* to introduce another arbitrary hypothesis $H1$.

```
intro H1.
```

```
1 subgoal
p : Prop
H0 : p
H1 : p -> False
________________________________________(1/1)
False
```

As of now, we introduce all the propositions, and hypotheses. The current goal is the same as the conclusion of the hypothesis $H1$. Thus, we use the tactic *apply* to apply the hypothesis $H1$.

```
apply H1.
```

```
1 subgoal
p : Prop
H0 : p
H1 : p -> False
________________________________________(1/1)
p
```

The goal is the same as one of the assumptions. So we can use the tactic *trivial* to complete the proof. This will check the goal with all the assumptions as well as hypotheses.

```
trivial.
```

```
No more subgoals.
```

Finally, there is no more goal to proof. We save the proof in Coq using *Qed* tactic.

```
Qed.
```

```
DNG is defined
```

The illustration of the tactic application described so far is sufficient to understand the implementation of this thesis. But we have defined customized Ltac tactics in our implementation. In the following section, we discuss the customized Ltac tactic definition as well as its application in the proofs.

5.5 Proof with Ltac

Coq provides the flexibility to write a customized tactic using Ltac language. Usually, Ltac uses the *match* tactic to compare the pattern either with a goal or hypothesis. Among others, one of the significant advantages of the Ltac tactics is to shorten the proof steps as well as makes the proof well organized. Additionally, a tactic notation may be introduced to shorten the Ltac tactic name.

To demonstrate the proof by customized Ltac tactic, we prove the same lemma proved in the previous section, but with the customized tactics. At first, we define a tactic *Ltac_DNG*.

```
Ltac Ltac_DNG :=
match goal with
| |- ~ ~ ?P
    => let H := fresh "H" in assert (H : forall a:Prop, a -> ~ ~ a); [let a :=
        fresh "a" in intro a; let H0 := fresh "H" in intro H0;unfold not; let H1
        := fresh "H" in intro H1;apply H1;trivial|trivial];apply H;clear H
| _ => fail 1 "Goal is not a DNG formula"
end.
```

The first line declare the name of the tactic as *Ltac_DNG* and the second, and third line restrict the pattern match with the goal only. Then command *let* is used to assert a formula *H*; subsequently, we provide the proof steps for the formula *H*. At the end of the proof, *H* moved up to the hypothesis, and then we apply *H* to the goal. Finally, we use the tactic *clear* to remove the hypothesis *H* i.e., the asserted formula.

Now, we illustrate the proof using Ltac tactic *Ltac_DNG*. The first three steps remain same as above, since we want to match the goal with the defined pattern in the Ltac tactic.

```
Lemma DNG : forall P : Prop, P ->
    ~ ~ P.
Proof.
intro p.
intro H0.
```

```
1 subgoal
p : Prop
H0 : p
_______________________________________(1/1)
~ ~ p
```

The current goal matches the pattern defined in the Ltac tactic. Therefore, we use the tactic *Ltac_DNG* at this point.

```
Ltac_DNG.
```

```
1 subgoal
p : Prop
H0 : p
_______________________________________(1/1)
p
```

The goal is the same as one of the assumption. We can use the tactic *trivial* to complete the proof.

```
trivial.
```

```
No more subgoals.
```

Finally, no more goal to proof. We save the proof in Coq using *Qed* tactic.

```
Qed.
```

```
DNG is defined
```

More details on Coq Ltac tactics are available in [21].

Chapter 6

Implementation in Coq

This chapter includes the implementation of our calculus in Coq, start with the definition of the structure of Boolean algebra. Then, to prove Boolean algebra axioms, the duality principle is applied. After that, we define Boolean contact algebra properties using the class overloading feature of Coq.

The implementation of our calculus in Coq begin with the customized Ltac tactics definition of the natural deduction rules defined in Chapter 3 for the operators, and abbreviations of modal relevance logic. Then, we introduce all the abbreviations as lemmas and prove them as well. To conclude the implementation, we again declare, and prove each Boolean algebra, and Boolean contact algebra axioms as a lemma. Our complete source will be available in the online or digital appendix.

6.1 Implementation of Boolean algebra

We begin with the abstract structure of Boolean algebra, application of duality to prove the Boolean algebra axioms, and order relations on Boolean algebra in Coq.

6.1.1 Abstract Structure of Boolean algebra

To begin with, Coq class [10, 20] is used to introduce the theory of Boolean algebras. We declare the signature class with appropriate syntactical notation.

```
Class BASig (A : Type) := {
   join   :  A -> A -> A;
   meet   :  A -> A -> A;
   zero   :  A;
```

```
    one    :  A;
    comp   :  A -> A
}.
```

Here *A* represent the underlying type or a set of the algebra, *join*, and *meet* are two binary operators, *comp* is the unary operator, *zero*, and *one* are two elements of *A*. As mentioned before, we define appropriate notations for the structure.

```
Infix    "+"      :=  (join).
Infix    "*"      :=  (meet).
Notation "0"      :=  zero.
Notation "1"      :=  one.
Notation "x  ^* " := (comp x) (at level 30).
```

Finally, we conclude the definition of the theory of Boolean algebra by combining the signature of *BASig* class, and the axioms listed in the Definition 5.

```
Class BA (A : Type) := {
    sig        : > BASig A;
    join_comm  : forall x y, x + y = y + x;
    zero_ident : forall x, x + 0 = x;
    join_distr : forall x y z, x + y * z = (x + y) * (x + z);
    join_comp  : forall x, x + x^* = 1;
    meet_comm  : forall x y, x * y = y * x;
    one_ident  : forall x, x * 1 = x;
    meet_distr : forall x y z, x * (y + z) = x * y + x * z;
    meet_comp  : forall x, x * x^* = 0
}.
```

In the subsequent section, we want to focus on the proof of Boolean algebra axioms. Proofs steps in Coq are similar to the [9], but we use the duality principle.

6.1.2 Duality of Boolean algebra

Our effort is to simplify the Boolean algebra axioms proof steps by introducing the duality principle in this thesis. Our aim is to use the duality principal such that Coq compiler will proof the axioms automatically using the proven property of the other axioms that is proven if only the operators are exchanged, i.e., join operator is replaced by the meet operator or vice versa. We start with the *dualBASig* definition.

```
Definition dualBASig {A : Type} (baSig : BASig A) : BASig A := {|
   join          :=  @meet A baSig;
   meet          :=  @join A baSig;
   zero          :=  @one A baSig;
   one           :=  @zero A baSig;
   comp          :=  @comp A baSig
|}.
```

Finally, we conclude the definition of the theory of Boolean algebra duality principle combining the signature of *dualBASig*, and the axioms listed in the Definition 5.

```
Definition dualBA {A : Type} (ba : BA A) : BA A := {|
   sig          := dualBASig (@sig A ba);
   join_comm    := @meet_comm A ba;
   zero_ident   := @one_ident A ba;
   join_distr   := @meet_distr A ba;
   join_comp    := @meet_comp A ba;
   meet_comm    := @join_comm A ba;
   one_ident    := @zero_ident A ba;
   meet_distr   := @join_distr A ba;
   meet_comp    := @join_comp A ba
|}.
```

Now, to use the duality principle, we prove the *dualize* lemma.

```
Lemma dualize {P : forall (A : Type), BA A -> Prop} (Lem : forall (A : Type) (ba
    : BA A), P A ba) {A : Type} {BA : BA A} : P A (dualBA ba).
Proof.
   apply Lem.
Qed.
```

We define an Ltac tactic to apply the *dualize* lemma.

```
Ltac dual x := apply (dualize x).
```

Please note that we will omit the proofs of all the axioms mentioned in the Lemma 1. Proofs are available in the Coq code included in this thesis. To illustrate the proof steps that are applying the duality principle we only mention the proofs of *join_assoc*, and *meet_assoc* axioms.

```
Lemma join_assoc {A : Type} {ba : BA A} : forall x y z, x + (y + z) = (x + y) + z.
Proof.
   intros.
   assert (H := UNg _ _ (K1 x y z) (L1 x y z)).
   apply (f_equal comp) in H.
   rewrite <- 2?DNg in H.
   symmetry.
   trivial.
Qed.
```

Finally, we are ready to demonstrate the benefit of the dualized principle to proof the *meet_assoc* lemma. *meet_assoc* lemma is proven by appling the proven property of *join_assoc* lemma.

```
Lemma meet_assoc {A : Type} {ba : BA A} : forall x y z, x * (y * z) = (x * y) * z.
Proof.
   dual @join_assoc.
Qed.
```

6.1.3 Order Relations on Boolean algebra

Besides the operators mentioned in the previous section, we include two more order relation operators (less equal, and greater equal) in our implementation. It is worth mentioning that the less equal operator will be used to define the *BCA3* contact relation Ltac rule later on in this thesis.

We begin with the definition of *less equal* operator, and its notation.

```
Definition leBA {A : Type} {ba : BA A} : A -> A -> Prop := fun x y => x * y = x.
Infix "<=" := leBA.
```

Now, we define the *greater equal* operator, and its notation.

```
Definition geBA {A : Type} {ba : BA A} : A -> A -> Prop := fun x y => x + y = x.
Infix ">=" := geBA.
```

Finally, we want to prove the equivalence relation between *less equal*, and *greater equal* in the form of a lemma.

```
Lemma OrdConsistent {A : Type} {ba : BA A} : forall x y, x <= y <-> y >= x.
Proof.
   intros; split; intros.
   unfold geBA; unfold leBA in H.
   rewrite <- H; rewrite meet_comm; apply join_absorp.
   unfold geBA in H; unfold leBA.
   rewrite <- H; rewrite join_comm; apply meet_absorp.
Qed.
```

6.2 Implementation of Boolean contact algebra

In this section, we want to introduce the theory of Boolean contact algebra by extending the class of Boolean algebra, and adding the binary contact relation, and its axioms mentioned in the Definition 6.

```
Class BCA (A : Type) := {
   ba :> BA A;
   C  : relation A;
   c0 : forall x y,    C x y               ->   x<>0 /\ y<>0;
   c1 : forall x,      x<>0                ->   C x x;
   c2 : forall x y,    C x y              <->   C y x;
   c3 : forall x y z, (C x y /\ y <= z)    ->   C x z;
   c4 : forall x y z,  C x (y+z)           ->   (C x y \/ C x z)
}.
```

6.3 Implementation of MRL Proposition and Model

This section includes the preliminaries of our calculus implementation in Coq. The following subsections define modal relevance logic proposition, model, and evaluation function. Please note that our idea is the same as [2].

6.3.1 MRL Proposition

Propositions in logic are same as type *prop* in Coq, but our versions of propositions are not the same. Our MRL propositions (or formule) becomes a proposition on the underlying Boolean algebra.

```
Definition MRLProp := forall (A : Type) (bca : BCA A), A -> Prop.
```

6.3.2 MRL Model

In this section, we want to explain the implementation of the evaluation function, and the model for our modal relevance logic. We start with the definition of the evaluation function.

The evaluation function of modal relevance logic applies on an MRL-formula to a specific element, and becomes a proposition.

```
Definition V (p : MRLProp) (A : Type) (bca : BCA A) (x : A) : Prop := p A bca x.
```

Model of modal relevance logic is valid iff it is true for all elements, and all Boolean algebra. We define the model as a function.

```
Definition MRLValid (p : MRLProp) : Prop := forall (A : Type) (bca : BCA A) (x :
    A), V p A bca x.
```

Now, we introduce a notation for MRLValid.

```
Notation "[ p ] x" := (V p _ _ x) (at level 70).
```

We define an Ltac tactic to convert an MRL-formula into an annotated formula, and start the proof immediately.

```
Ltac start := unfold MRLValid; let A:= fresh "A" in intro A;let bca:= fresh "bca"
    in intro bca;let z:= fresh "z" in intro z.
```

6.4 Implementation of Natural Deduction Rules

This section includes the implementation of modal relevance logic operator's rules in Coq. We begin by defining the logical operators, their notations, and the Ltac tactic of the rules. It is worth mentioning that all logical operators are defined as a function on *MRLProp*, and we have given appropriate names for the Ltac tactics. Please note that we will omit the implementation of the equality rules in Table 3.1. Because inbuilt Coq tactic *trivial* is used for equality introduction rule (= I), and *rewrite* tactic is used for equality elimination rule (= E).

In the upcoming subsections, all logical operators are implemented in the same order as the natural deduction rules that are defined in Chapter 3.

6.4.1 Implementation of Propositional Operators

We will explain the implementation of propositional logic operators in the same order as they are defined in Table 3.2.

6.4.1.1 Propositional Logic *False*

The propositional logic formula *false* (⊥) is defined as follows:

```
Definition Bot : MRLProp := fun (A : Type) (bca : BCA A) x => False.
```

```
Notation  "⊥"    := (V Bot _ _ _).
```

Please note there are no rules required for $false$.

6.4.1.2 Propositional Logic *Implication*

The propositional logic operator *implication* (→) is defined as follows:

```
Definition MRL_P_Impl (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => V p A bca x -> V q A bca x.
```

```
Infix     "->"   := (MRL_P_Impl).
```

```
Ltac MRL_P_Impl_Intro :=
match goal with
| |- V (?p -> ?q) ?A ?bca ?z
     => replace (V (p -> q) A bca z) with (forall _ : V p A bca z, V q A bca z)
         by (unfold V; unfold MRL_P_Impl; trivial); let H := fresh "H" in intro H
| _  => fail 1 "Goal is not an MRL_P_Impl formula"
end.
```

```
Ltac MRL_P_Impl_Elim H0 H1 :=
match type of H0 with
| V (?p -> ?q) ?A ?bca ?z
  => match type of H1 with
  | V p A bca z
     => replace (V (p -> q) A bca z) with (forall _ : V p A bca z, V q A bca z)
         by (unfold V; unfold MRL_P_Impl in H0; trivial); let H2 := fresh "H" in
         assert (H2 := H1); apply H0 in H2;repeat assumption
  | _ => fail 2 "2nd hypothesis does not match the assumption of the first
      hypothesis"
    end
| _ => fail 1 "1st hypothesis is not an MRL_P_Impl formula"
end.
```

6.4.1.3 Propositional Logic *Not*

The propositional logic operator *not* (¬) is defined as follows:

```
Definition MRL_P_Not  (p  : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => ∽ V p A bca x.
```

```
Notation  "¬ x"  := (MRLProp x)    (at level 30).
```

```
Ltac MRL_P_Not_Intro :=
match goal with
| |- V (¬?p) ?A ?bca ?z
=> replace (V (¬p) A bca z) with (MRL_P_Not p A bca z) by (unfold V; unfold
    MRL_P_Not; trivial);let H := fresh "H" in intro H;replace False with (V Bot A
    bca z) by (unfold V; unfold Bot; trivial);repeat assumption
| _ => fail 1 "Goal is not an MRL_P_Not formula"
end.
```

```
Ltac MRL_P_Not_Elim H0 H1 :=
match type of H0 with
| V (¬?p) ?A ?bca ?z
=> match type of H1 with
| V p A bca z
=> replace (V (¬p) A bca z) with (MRL_P_Not p A bca z) by (unfold V; unfold
    MRL_P_Not in H0; trivial); let H2 := fresh "H" in assert (H2 := H1); apply H0
    in H2;repeat assumption
| _ => fail 2 "2nd hypothesis does not match the body of the first hypothesis"
end
| _ => fail 1 "1st hypothesis is not a MRL_P_Not formula"
end.
```

Now, we define the PBC rule.

```
Ltac MRL_P_PBC :=
match goal with
| |- V ?p ?A ?bca ?z
   => let H := fresh "H" in apply NNPP; intro H; replace (not ([p] z)) with
       ([¬p] z) in H by (unfold V; unfold MRL_P_Not; trivial); replace False with
       (V Bot A bca z) by (unfold V; unfold Bot; trivial);repeat assumption
```

```
end.
```

6.4.1.4 Propositional Logic *And*

The propositional logic operator *and* (∧) is defined as follows:

```
Definition MRL_P_And  (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => V p A bca x /\ V q A bca x.
```

```
Infix     "/\"    := (MRL_P_And).
```

```
Ltac MRL_P_And_Intro :=
match goal with
| |- V (?p /\ ?q) ?A ?bca ?z
=> replace (V (p /\ q) A bca z) with (and (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_And; trivial); split;repeat assumption
| _ => fail 1 "Goal is not an MRL_P_And formula"
end.
```

```
Ltac MRL_P_And_Elim_1 H :=
match type of H with
|V (?p /\ ?q) ?A ?bca ?z
=> replace (V (p /\ q) A bca z) with (and (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_And in H; trivial); let H0 := fresh "H" in let H1 := fresh
    "H" in assert (H0 := H); destruct H0 as [H0 H1]; clear H1;repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_P_And formula"
end.
```

```
Ltac MRL_P_And_Elim_2 H :=
match type of H with
| V (?p /\ ?q) ?A ?bca ?z
=> replace (V (p /\ q) A bca z) with (and (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_And in H; trivial); let H0 := fresh "H" in let H1 := fresh
    "H" in assert (H0 := H); destruct H0 as [H1 H0]; clear H1;repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_P_And formula"
end.
```

Now, we define a Ltac tactic for all eliminations as well as its notation.

```
Ltac MRL_P_And_Elim' H :=
match type of H with
| V (?p /\ ?q) ?A ?bca ?z
=> replace (V (p /\ q) A bca z) with (and (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_And in H; trivial); let H0 := fresh "H" in destruct H as [H
    H0];MRL_P_And_Elim' H; MRL_P_And_Elim' H0;repeat assumption
```

```
| _ => fail 1 "Hypothesis is not an MRL_P_And formula"
end.
```

```
Tactic Notation "MRL_P_And_Elim" hyp(H) := MRL_P_And_Elim' H.
```

6.4.1.5 Propositional Logic *Or*

The propositional logic operator *or* (∨) is defined as follows:

```
Definition MRL_P_Or  (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => forall y z, x = y * z -> V p A bca y \/ V q A bca z.
```

```
Infix     "\/"    := (MRL_P_Or).
```

```
Ltac MRL_P_Or_Intro_1 :=
match goal with
| |- V (?p \/ ?q) ?A ?bca ?z
=> replace (V (p \/ q) A bca z) with (or (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_Or; trivial); left;repeat assumption
| _ => fail 1 "Goal is not an MRL_P_Or formula"
end.
```

```
Ltac MRL_P_Or_Intro_2 :=
match goal with
| |- V (?p \/ ?q) ?A ?bca ?z
=> replace (V (p \/ q) A bca z) with (or (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_Or; trivial); right;repeat assumption
| _ => fail 1 "Goal is not an MRL_P_Or formula"
end.
```

```
Ltac MRL_P_Or_Elim H :=
match type of H with
| V (?p \/ ?q) ?A ?bca ?z
=> replace (V (p \/ q) A bca z) with (or (V p A bca z) (V q A bca z)) by (unfold
    V; unfold MRL_P_Or in H; trivial); destruct H;repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_P_Or formula"
end.
```

6.4.1.6 Propositional Logic *Equivalence*

The propositional logic operator *equivalence* (↔) is defined as follows:

```
Definition MRL_P_Equal (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A)
    x =>V p A bca x <-> V q A bca x.
```

```
Infix     "<->"     := (MRL_P_Equal).
```

```
Ltac MRL_P_Equal_Intro :=
match goal with
| [|- V (?p <-> ?q) ?A ?bca ?x]
=> replace (V (p <-> q) A bca x) with (and (V (p -> q) A bca x) (V (q -> p) A bca
    x)) by (unfold V; unfold MRL_P_Equal; split); split
| _ => fail 1 "Goal is not an MRL_P_Equal formula"
end.
```

```
Ltac MRL_P_Equal_Elim H :=
match type of H with
| V (?p <-> ?q) ?A ?bca ?x
=> replace (V (p <-> q) A bca x) with (and (V (p -> q) A bca x) (V (q -> p) A bca
    x)) in H by (unfold V; unfold MRL_P_Equal; split); let H0 := fresh "H" in
    destruct H as [H H0];repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_P_And formula"
end.
```

6.4.1.7 Propositional Logic *True*

The propositional logic formula *true* (⊤) is defined as follows:

```
Definition Top : MRLProp := fun (A : Type) (bca : BCA A) x => True.
```

```
Notation  "⊤"     := (V Top _ _ _).
```

Please note no rules required for *true*.

6.4.2 Implementation of Modal Operators

Now, we will explain the implementation of modal logic operators in the same order as they are defined in Table 3.3. Please note that our implementation idea of modal logic is same as [2].

6.4.2.1 Modal Logic *Box*

The modal logic operator *box* ([]) is defined as follows:

```
Definition MRL_K_BoxC    (p: MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => forall y, (forall _ :(C x y), (V p A bca y)).
```

```
Notation  "[C]φ":= (MRL_K_BoxC φ)  (at level 15).
```

```
Ltac MRL_K_BoxC_Intro :=
match goal with
| |- V ([C]?p) ?A ?bca ?z
=> replace (V ([C] p) A bca z) with (MRL_K_BoxC p A bca z) by (unfold V; unfold
    MRL_K_BoxC; trivial); let y:= fresh "y" in intro y; let H:= fresh "H" in
    intro H
| _ => fail 1 "Goal is not an MRL_K_BoxC formula"
end.
```

Please note that the side-condition of ([C] I) rule is automatically taken care of by using the *fresh* command that introduced a new variable.

```
Ltac MRL_K_BoxC_Elim H0 H1:=
match type of H0 with
|  V ([C]?p) ?A ?bca ?z =>
match type of H1 with
| C z ?y
=> replace (V ([C] p) A bca z) with (forall y : A, forall _ : C z y , [p] y) in
    H0 by (unfold V; unfold MRL_K_BoxC; trivial); specialize H0 with y;let H2 :=
    fresh "H" in assert (H2 := H1); apply H0 in H1; repeat assumption
| _ => fail 2 "2nd hypothesis does not match the body of the first hypothesis"
end
| _ => fail 1 "1st hypothesis is not an MRL_K_BoxS formula"
end.
```

6.4.2.2 Modal Logic *Diamond*

The modal logic operator *diamond* (⟨⟩) is defined as follows:

```
Definition MRL_K_DiamondC (p: MRLProp) : MRLProp := fun (A : Type) (bca : BCA A)
    x => exists y, and (C x y) (V p A bca y).
```

```
Notation  "<C>φ":= (MRL_K_DiamondC φ) (at level 15).
```

```
Ltac MRL_K_DiamondC_Intro x :=
match goal with
| |- V (<C>?p) ?A ?bca ?z
=> replace (V (<C> p) A bca z) with (MRL_K_DiamondC p A bca z) by (unfold V;
    unfold MRL_K_DiamondC; trivial); exists x; split
| _ => fail 1 "Goal is not an MRL_K_DiamondC formula"
end.
```

```
Ltac MRL_K_DiamondC_Elim H0 :=
match type of H0 with
```

```
| V (<C>?p) ?A ?bca ?z
=> replace (V (<C> p) A bca z) with (MRL_K_DiamondC p A bca z) in H0 by (unfold
    V; unfold MRL_K_DiamondC; trivial); let x:= fresh "x" in destruct H0 as [x
    H0]; let H1:= fresh "H" in destruct H0 as [H1 H0]; repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_K_DiamondC formula"
end.
```

Please note that the side-condition of ($\langle C \rangle$ E) rule is automatically taken care of by using the *fresh* command that introduced a new variable.

6.4.3 Implementation of Basic Relevance Operators

At this point, we will explain the implementation of relevance logic operators in the same order as they are defined in Table 3.4.

6.4.3.1 Relevance Logic *Implication*

The relevance logic operator *implication* ($\twoheadrightarrow$) is defined as follows:

```
Definition MRL_RJ_Impl (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A)
    x => forall y z, x = y + z -> V p A bca y -> V q A bca z.
```

```
Infix     "->>"     := (MRL_RJ_Impl)     (at level 60).
```

```
Ltac MRL_RJ_Impl_Intro :=
match goal with
| |- V (?p ->> ?q) ?A ?bca ?z
=> replace (V (p ->> q) A bca z) with (MRL_RJ_Impl p q A bca z) by (unfold V;
    unfold MRL_RJ_Impl; trivial); let x := fresh "x" in intro x; let y := fresh
    "y" in intro y; let H := fresh "H" in intro H; let H := fresh "H" in intro
    H;repeat assumption
| _ => fail 1 "Goal is not an MRL_RJ_Impl formula"
end.
```

Please note that the side-condition of ($\twoheadrightarrow$ I) rule is automatically taken care of by using the *fresh* command that introduced two new variables.

```
Ltac MRL_RJ_Impl_Elim H0 H1 H2:=
match type of H0 with
| V (?p ->> ?q) ?A ?bca ?z =>
match type of H2 with
| z = ?x + ?y  =>
```

```
match type of H1 with
| V p A bca x
=> replace (V (p ->> q) A bca z) with ( MRL_RJ_Impl p q A bca z) by (unfold V;
    unfold MRL_RJ_Impl in H0; trivial); let H3 := fresh "H" in assert (H3 :=
    H2);apply H0 in H2;apply H2 in H1;repeat assumption
| _ => fail 3 "3rd hypothesis does not match the assumption of the first
    hypothesis"
end
| _ => fail 2 "2nd hypothesis does not match the assumption of the first
    hypothesis"
end
| _ => fail 1 "1st hypothesis is not an MRL_RJ_Impl formula"
end.
```

6.4.3.2 Relevance Logic *Not*

The relevance logic operator *not* (∽) is defined as follows:

```
Definition MRL_R_Not  (p  : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => ∽ V p A bca (x^*).
```

```
Notation  "∽ x"  := (MRL_R_Not x).
```

```
Ltac MRL_R_Not_Intro :=
match goal with
| |- V (∽(?p)) ?A ?bca ?z
=> replace (V (∽p) A bca z) with (MRL_R_Not p A bca z) by (unfold V; unfold
    MRL_R_Not; trivial); let H := fresh "H" in intro H; replace False with (V Bot
    A bca z) by (unfold V; unfold Bot; trivial);repeat assumption
| _ => fail 1 "Goal is not an MRL_R_Not formula"
end.
```

```
Ltac MRL_R_Not_Elim H0 H1 H2 :=
match type of H0 with
| V (∽ (?p)) ?A ?bca ?z =>
match type of H1 with
| V p A bca ?z0 =>
match type of H2 with
| z=z0^*
=> replace (V (∽(p)) A bca z) with ( MRL_R_Not p A bca z) by (unfold V; unfold
    MRL_R_Not in H0; trivial); let H3 := fresh "H" in assert (H3 := H1);apply H0
    in H1; repeat assumption
| _ => fail 3 "3rd hypothesis does not match the body of the first hypothesis"
end
| _ => fail 2 "2nd hypothesis does not match the body of the first hypothesis"
end
```

```
| _ => fail 1 "1st hypothesis is not a MRL_R_Not formula"
end.
```

Now, we define the PBC rule.

```
Ltac MRL_R_Not_PBC :=
match goal with
| |- V ?p ?A ?bca ?z
=> let H := fresh "H" in apply NNPP; intro H; replace (not (V p A bca z)) with (V
    (~p) A bca (z^*)) in H by (unfold V; unfold MRL_R_Not; rewrite <- DNg;
    trivial); replace False with (V Bot A bca z) by (unfold V; unfold Bot;
    trivial);repeat assumption
end.
```

6.4.4 Implementation of Derived Relevance Operators

Now, we will explain the implementation of the derived relevance logic operators in the same order as they are defined in Table 3.5.

6.4.4.1 Derived Relevance Operator *And for Join*

The derived relevance operator *and for join* (⩕) is defined as follows:

```
Definition MRL_RJ_And (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => exists y z, x = y + z /\ V p A bca y /\ V q A bca z.
```

```
Infix     "//\\":= (MRL_RJ_And)     (at level 60).
```

```
Ltac MRL_RJ_And_Intro x y :=
match goal with
| |- V (?p //\\ ?q) ?A ?bca ?z      => replace (V (p //\\ q) A bca z) with (
    MRL_RJ_And p q A bca z) by (unfold V; unfold MRL_RJ_And; trivial); exists x;
    exists y;repeat split;repeat assumption
| _ => fail 1 "Goal is not an MRL_RJ_And formula"
end.
```

```
Ltac MRL_RJ_And_Elim H0 :=
match type of H0 with
| V (?p //\\ ?q) ?A ?bca ?z
=> replace (V (p //\\ q) A bca z) with ( MRL_RJ_And p q A bca z) by (unfold V;
    unfold MRL_RJ_And in H0; trivial); let x := fresh "x" in destruct H0 as [x
    H0]; let y := fresh "y" in destruct H0 as [y H0]; let H1 := fresh "H" in
    destruct H0 as [H1 H0]; let H2 := fresh "H" in destruct H0 as [H2 H0];repeat
    assumption
```

```
| _ => fail 1 "Hypothesis is not an MRL_RJ_And formula"
end.
```

Please note that the side-condition of (⩓ E) rule is automatically taken care of by using the *fresh* command that introduced two new variables.

6.4.4.2 Derived Relevance Operator *Or for Join*

The derived relevance operator *or for join* (⩔) is defined as follows:

```
Definition MRL_RJ_Or  (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => forall y z, x = y + z -> V p A bca y \/ V q A bca z.
```

```
Infix     "\\//":= (MRL_RJ_Or)      (at level 60).
```

```
Ltac MRL_RJ_Or_Intro1:=
match goal with
| |- V (?p \\// ?q) ?A ?bca ?z      => replace (V (p \\// q) A bca z) with
    (MRL_RJ_Or p q A bca z) by (unfold V; unfold MRL_RJ_Or; trivial); unfold
    V;let x := fresh "x" in intro x; let y := fresh "y" in intro y; let H :=
    fresh "H" in intro H; let H := fresh "H" in assert (H : forall p, p <-> ∽∽p);
    [ intro; split; intro; [ intro; contradiction | apply NNPP; trivial ] |
    rewrite (H ([p] x))]; clear H; apply imply_to_or; replace (not ([p] x)) with
    ([¬p] x) by (unfold V; unfold MRL_P_Not; trivial); let H := fresh "H" in
    intro;repeat assumption
| _ => fail 1 "Goal is not an MRL_RJ_Or formula"
end.
```

```
Ltac MRL_RJ_Or_Intro2:=
match goal with
| |- V (?p \\// ?q) ?A ?bca ?z      => replace (V (p \\// q) A bca z) with
    (MRL_RJ_Or p q A bca z) by (unfold V; unfold MRL_RJ_Or; trivial); unfold
    V;let x:= fresh "x" in intro x; let y:= fresh "y" in intro y; let H := fresh
    "H" in intro H; apply or_comm; let H := fresh "H" in assert (H : forall q, q
    <-> ∽∽q); [ intro; split; intro; [ intro; contradiction | apply NNPP; trivial
    ] | rewrite (H ([q] y))]; clear H; apply imply_to_or; replace (not ([q] y))
    with ([¬q] y) by (unfold V; unfold MRL_P_Not; trivial); let H := fresh "H" in
    intro;repeat assumption
| _ => fail 1 "Goal is not an MRL_RJ_Or formula"
end.
```

Please note that the side-condition of (⩔ I) rule is automatically taken care of by using the *fresh* command that introduced two new variables.

```
Ltac MRL_RJ_Or_Elim H0 H1:=
match type of H0 with
| V (?p \\// ?q) ?A ?bca ?z      => replace (V (p \\// q) A bca z) with (MRL_RJ_Or
    p q A bca z) by (unfold V; unfold MRL_RJ_Or in H0; trivial); let H2 := fresh
    "H" in assert (H2 := H1);apply H0 in H1; destruct H1;repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_RJ_Or formula"
end.
```

6.4.4.3 Derived Relevance Operator *N*

The derived relevance operator *N* is defined as follows:

```
Definition MRL_R_NNot (p   : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => V p A bca (x^*).
```

```
Notation  "'N' x":= (MRL_R_NNot x)  (at level 30).
```

```
Ltac MRL_R_NNot_Intro :=
match goal with
| |- V (N ?p) ?A ?bca ?z
=> replace (V (N p) A bca z) with (MRL_R_NNot p A bca z) by (unfold V; unfold
    MRL_R_NNot; trivial); unfold MRL_R_NNot at 1; MRL_P_PBC;repeat assumption
| _ => fail 1 "Goal is not an MRL_R_NNot formula"
end.
```

```
Ltac MRL_R_NNot_Elim H0 H1 H2:=
match type of H0 with
| V (N ?p) ?A ?bca ?z =>
match type of H1 with
| V (¬p) A bca ?z0 =>
match type of H2 with
| z=z0^*
=> replace (V (N(p)) A bca z) with (MRL_R_NNot p A bca z) by (unfold V; unfold
    MRL_R_NNot in H0; trivial); let H2 := fresh "H" in assert (H2 := H0);apply H1
    in H0;repeat assumption
| _ => fail 3 "3rd hypothesis does not match the body of the first hypothesis"
end
| _ => fail 2 "2nd hypothesis does not match the body of the first hypothesis"
end
| _ => fail 1 "1st hypothesis is not a MRL_R_NNot formula"
end.
```

Now, we define the PBC rule.

```
Ltac MRL_R_NNot_PBC :=
match goal with
```

```
| |- V ?p ?A ?bca ?z
=> let H := fresh "H" in apply NNPP; intro H;replace (not ([p] z)) with ([N ¬ p]
    z^*) in H by (unfold V; unfold MRL_R_NNot;rewrite <- DNg; trivial); False
    with (V Bot A bca z) by (unfold V; unfold Bot; trivial);repeat assumption
end.
```

6.4.4.4 Derived Relevance Operator *Implication for Meet*

The derived relevance operator *implication for meet* (⊸) is defined as follows:

```
Definition MRL_RM_Impl (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A)
    x => forall y z, x = y * z -> V p A bca y -> V q A bca z.
```

```
Infix     "->o"     := (MRL_RM_Impl)    (at level 60).
```

```
Ltac MRL_RM_Impl_Intro :=
match goal with
| [|- V (?p ->o ?q) ?A ?bca ?z]
=> replace (V (p ->o q) A bca z) with ( MRL_RM_Impl p q A bca z) by (unfold V;
    unfold MRL_RM_Impl; trivial); let x := fresh "x" in intro x;let y := fresh
    "y" in intro y; let H := fresh "H" in intro H;let H := fresh "H" in intro
    H;repeat assumption
| _ => fail 1 "Goal is not an MRL_RM_Impl formula"
end.
```

Please note that the side-condition of (⊸ I) rule is automatically taken care of by using the *fresh* command that introduced two new variables.

```
Ltac MRL_RM_Impl_Elim H0 H1 H2:=
match type of H0 with
| V (?p ->o ?q) ?A ?bca ?z =>
match type of H2 with
| z = ?x * ?y =>
match type of H1 with
| V p A bca x
=> replace (V (p ->o q) A bca z) with ( MRL_RM_Impl p q A bca z) by (unfold V;
    unfold MRL_RM_Impl in H0; trivial); let H3 := fresh "H" in assert (H3 :=
    H2);apply H0 in H2;apply H2 in H1;repeat assumption
| _ => fail 3 "3rd hypothesis does not match the assumption of the first
    hypothesis"
end
| _ => fail 2 "2nd hypothesis does not match the assumption of the first
    hypothesis"
end
```

```
| _ => fail 1 "1st hypothesis is not an MRL_RM_Impl formula"
end.
```

6.4.4.5 Derived Relevance Operator *And for Meet*

The derived relevance operator *and for meet* ($\overset{\circ}{\wedge}$) is defined as follows:

```
Definition MRL_RM_And (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => exists y z, x = y * z /\ V p A bca y /\ V q A bca z.
```

```
Infix     "/\o"  := (MRL_RM_And)     (at level 60).
```

```
Ltac MRL_RM_And_Intro x y :=
match goal with
| |- V (?p /\o ?q) ?A ?bca ?z
=> replace (V (p /\o q) A bca z) with ( MRL_RM_And p q A bca z) by (unfold V;
    unfold MRL_RM_And; trivial); exists x; exists y;repeat split;repeat assumption
| _ => fail 1 "Goal is not an MRL_RJ_And formula"
end.
```

```
Ltac MRL_RM_And_Elim H0 :=
match type of H0 with
| V (?p /\o ?q) ?A ?bca ?z
=> replace (V (p /\o q) A bca z) with ( MRL_RM_And p q A bca z) by (unfold V;
    unfold MRL_RM_And in H0; trivial); let x := fresh "x" in destruct H0 as [x
    H0]; let y := fresh "y" in destruct H0 as [y H0]; let H1 := fresh "H" in
    destruct H0 as [H1 H0]; let H2 := fresh "H" in destruct H0 as [H2 H0];repeat
    assumption
| _ => fail 1 "Hypothesis is not an MRL_RM_And formula"
end.
```

Please note that the side-condition of ($\overset{\circ}{\wedge}$ E) rule is automatically taken care of by using the *fresh* command that introduced two new variables.

6.4.4.6 Derived Relevance Operator *Or for Meet*

The derived relevance operator *or for meet* ($\underset{\circ}{\vee}$) is defined as follows:

```
Definition MRL_RM_Or  (p q : MRLProp) : MRLProp := fun (A : Type) (bca : BCA A) x
    => forall y z, x = y * z -> V p A bca y \/ V q A bca z.
```

```
Infix     "\/o"  := (MRL_RM_Or)      (at level 60).
```

```
Ltac MRL_RM_Or_Intro1:=
match goal with
| |- V (?p \/o ?q) ?A ?bca ?z
=> replace (V (p \/o q) A bca z) with (MRL_RM_Or p q A bca z) by (unfold V;
    unfold MRL_RM_Or; trivial); unfold V;let x := fresh "x" in intro x; let y :=
    fresh "y" in intro y; let H := fresh "H" in intro H; let H := fresh "H" in
    assert (H : forall p, p <-> ~~p); [ intro; split; intro; [ intro;
    contradiction | apply NNPP; trivial ] | rewrite (H ([p] x))]; clear H; apply
    imply_to_or; replace (not ([p] x)) with ([¬p] x) by (unfold V; unfold
    MRL_P_Not; trivial); let H := fresh "H" in intro;repeat assumption
| _ => fail 1 "Goal is not an MRL_RM_Or formula"
end.
```

```
Ltac MRL_RM_Or_Intro2:=
match goal with
| |- V (?p \/o ?q) ?A ?bca ?z
=> replace (V (p \/o q) A bca z) with (MRL_RM_Or; p q A bca z) by (unfold V;
    unfold MRL_RM_Or; trivial); unfold V;let x:= fresh "x" in intro x; let y:=
    fresh "z" in intro y; let H := fresh "H" in intro H; apply or_comm; let H :=
    fresh "H" in assert (H : forall q, q <-> ~~q); [ intro; split; intro; [
    intro; contradiction | apply NNPP; trivial ] | rewrite (H ([q] y))]; clear H;
    apply imply_to_or; replace (not ([q] y)) with ([¬q] y) by (unfold V; unfold
    MRL_P_Not; trivial); let H := fresh "H" in intro;repeat assumption
| _ => fail 1 "Goal is not an MRL_RM_Or formula"
end.
```

Please note that the side-condition of ($\underset{\circ}{\vee}$ I) rule is automatically taken care of by using the *fresh* command that introduced two new variables.

```
Ltac MRL_RM_Or_Elim H0 H1:=
match type of H0 with
| V (?p \/o ?q) ?A ?bca ?z
=> replace (V (p \/o q) A bca z) with (MRL_RM_Or p q A bca z) by (unfold V;
    unfold MRL_RM_Or in H0; trivial); let H2 := fresh "H" in assert (H2 :=
    H1);apply H0 in H1;destruct H1;repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_RJ_Or formula"
end.
```

6.4.5 Implementation of Constant E and U

In this section, we will conclude the implementation of the rules definitions with the constant *E*, and *U* in the same order as they are defined in Table 3.6.

6.4.5.1 Constant *E*

The constant *E* is defined as follows:

```
Definition E : MRLProp := fun (A : Type) (bca : BCA A) x => forall α, α = α + x.
```

```
Ltac MRL_R_E_Intro :=
match goal with
| |- V E ?A ?bca ?z
=> let x:= fresh "x" in intro x;repeat assumption
| _ => fail 1 "Goal is not an MRL_R_E formula"
end.
```

Please note that the side-condition of (*E* I) rule is automatically taken care of by using the *fresh* command that introduced a new variable.

```
Ltac MRL_R_E_Elim H0 x:=
match type of H0 with
|  V E ?A ?bca ?z
=> unfold V in H0;unfold E in H0; specialize H0 with x; repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_R_E formula"
end.
```

6.4.5.2 Constant *U*

The constant *U* is defined as follows:

```
Definition U : MRLProp := fun (A : Type) (bca : BCA A) x => forall α, α = α * x.
```

```
Ltac MRL_R_U_Intro :=
match goal with
| |- V U ?A ?bca ?z
=> let x:= fresh "x" in intro x;repeat assumption
| _  => fail 1 "Goal is not an MRL_R_U formula"
end.
```

Please note that the side-condition of (*U* I) rule is automatically taken care of by using the *fresh* command that introduced a new variable.

```
Ltac MRL_R_U_Elim H0 x:=
match type of H0 with
```

```
|  V U ?A ?bca ?z
=> unfold V in H0;unfold U in H0;specialize H0 with x; repeat assumption
| _ => fail 1 "Hypothesis is not an MRL_R_U formula"
end.
```

6.4.6 Implementation of Contact

In this section, we will explain the implementation contact rules in the same order as they are defined in Table 3.7. Please note we have defined the contact axiom (BCA_0) - (BCA_3) to be applied both on goal, and the hypothesis as well.

6.4.6.1 Contact (BCA_0)

The contact rule (BCA_0) is defined as follows:

```
Ltac MRL_K_C_BCA0g z:=
match goal with
| |- V (¬ E) ?A ?bca ?x
=> let H := fresh "H" in assert (H : forall a b, (C a b) -> [¬ E] b); [let a :=
    fresh "a" in intro a;let b := fresh "b" in intro b;let H1 := fresh "H" in
    intro H1;apply c0 in H1; let H2 := fresh "H" in destruct H1 as [H2 H1];
    unfold V, MRL_P_Not, E, V;contradict H1; let H3 := fresh "H" in assert (H3 :=
    H1 0);rewrite join_comm, zero_ident in H3; symmetry;trivial|trivial];apply H
    with z;trivial;clear H
| _ => fail 1 "Goal is not an [¬ E] ?x formula"
end.
```

```
Ltac MRL_K_C_BCA0h H0:=
match type of H0 with
| C ?x ?y
=> let H := fresh "H" in assert (H : forall a b, (C a b) -> [¬ E] b); [let a :=
    fresh "a" in intro a;let b := fresh "b" in intro b;let H1 := fresh "H" in
    intro H1;apply c0 in H1; let H2 := fresh "H" in destruct H1 as [H2 H1];
    unfold V, MRL_P_Not, E, V;contradict H1; let H3 := fresh "H" in assert (H3 :=
    H1 0);rewrite join_comm, zero_ident in H3; symmetry;trivial|trivial];apply H
    in H0;trivial;clear H
| _ => fail 1 "Hypothesis is not in C ?x ?y form"
end.
```

6.4.6.2 Contact (BCA_1)

The contact rule (BCA_1) is defined as follows:

```
Ltac MRL_K_C_BCA1g :=
match goal with
| |- C ?x ?x
=> let H := fresh "H" in assert (H : forall a, [¬ E] a -> (C a a)); [let x0 :=
    fresh "x" in intro x0;let H1 := fresh "H" in intro H1;apply c1; unfold V,
    MRL_P_Not, E in H1;contradict H1;unfold V;intro;rewrite H1; rewrite
    zero_ident;trivial|trivial]; apply H;clear H
| _ => fail 1 "Goal is not in C ?x ?x form"
end.
```

```
Ltac MRL_K_C_BCA1h H0:=
match type of H0 with
| V (¬ E) ?A ?bca ?x
=> let H := fresh "H" in assert (H : forall a, [¬ E] a -> (C a a)); [let a :=
    fresh "a" in intro a;let H1 := fresh "H" in intro H1;apply c1; unfold V,
    MRL_P_Not, E in H1;contradict H1;unfold V;intro;rewrite H1; rewrite
    zero_ident;trivial|trivial]; apply H in H0;clear H
| _ => fail 1 "Hypothesis is not in [¬ E] x form"
end.
```

6.4.6.3 Contact *(BCA$_2$)*

The contact rule *(BCA$_2$)* is defined as follows:

```
Ltac MRL_K_C_BCA2g :=
match goal with
| |- C ?x ?y
=> apply c2
| _ => fail 1 "Goal is not in C ?x ?x form"
end.
```

```
Ltac MRL_K_C_BCA2h H0:=
match type of H0 with
| C ?x ?y
=> let H1:= fresh "H" in assert (H1 := H0); apply c2 in H1
| _ => fail 1 "Hypothesis is not in C ?x ?x form"
end.
```

6.4.6.4 Contact *(BCA$_3$)*

The contact rule *(BCA$_3$)* is defined as follows:

```
Ltac MRL_K_C_BCA3g y:=
match goal with
```

```
|- C ?x ?z
=> let H := fresh "H" in assert (H : forall a b c, (C a b) /\ b=b * c -> C a c);
    [let a := fresh "a" in intro a; let b := fresh "b" in intro b; let c := fresh
    "c" in intro c; let H1 := fresh "H" in intro H1;apply c3 with b;unfold
    leBA;destruct H1;split;trivial;symmetry;trivial|trivial];apply H with y;clear
    H
| _ => fail 1 "Goal is not in C ?x ?z form"
end.
```

```
Ltac MRL_K_C_BCA3h H0 H1:=
match type of H0 with
|  C ?x ?y
=> match type of H1 with
|  y = y * ?z
=> let H := fresh "H" in assert (H : forall a b c, (C a b) /\ b=b * c -> C a c);
    [let a := fresh "a" in intro a; let b := fresh "b" in intro b; let c := fresh
    "c" in intro c; let H1 := fresh "H" in intro H1;apply c3 with b;unfold
    leBA;destruct H1;split;trivial;symmetry;trivial|trivial]; assert(H2 := conj
    H0 H1); apply H in H2;clear H
| _ => fail 2 "2nd hypothesis doesn t match with the body of the 1st hypothesis"
end
| _ => fail 1 "1st hypothesis is not an MRL_K_C_BCA3 formula"
end.
```

6.4.6.5 Contact *(BCA$_4$)*

The contact rule *(BCA$_4$)* is defined as follows:

```
Ltac MRL_K_C_BCA4 H0:=
match type of H0 with
| C ?x (?y + ?z)
=> let H1 := fresh "H" in assert (H1 := H0); apply c4 in H1; repeat destruct H1
| _ => fail 1 "Hypothesis is not an MRL_K_C_BCA4 formula"
end.
```

6.5 Proofs in Coq

In this section, we present the proofs of the abbreviations, and axioms mentioned in Chapter 4. Please note that Coq proofs are a one-to-one translation of the proof trees mentioned in Chapter 4.

6.5.1 Propositional Logic

Now, we will mention the proofs of the abbreviations (PLAbbr1 - PLAbbr5). In the following lemma, we provide the proof of the abbreviation PLAbbr1.

```
Lemma PLAbbr1 (φ   : MRLProp) : MRLValid (¬ φ <-> (φ -> Bot)).
Proof.
   start.
   MRL_P_Equal_Intro.
   MRL_P_Impl_Intro.
   MRL_P_Impl_Intro.
   MRL_P_Not_Elim H H0.
   MRL_P_Impl_Intro.
   MRL_P_Not_Intro.
   MRL_P_Impl_Elim H H0.
Qed.
```

Please note that we will omit the proofs of the abbreviations (PLAbbr2 - PLAbbr5). Proofs are available in the Coq provided in this thesis.

```
Lemma PLAbbr2 (φ ψ : MRLProp) : MRLValid ((φ /\ ψ) <-> (¬(φ -> ¬ ψ))).
```

```
Lemma PLAbbr3 (φ ψ : MRLProp) : MRLValid ((φ \/ ψ) <-> (¬ φ -> ψ)).
```

```
Lemma PLAbbr4 (φ ψ : MRLProp) : MRLValid ((φ <-> ψ) <-> ((φ -> ψ) /\ (ψ -> φ))).
```

```
Lemma PLAbbr5                  : MRLValid (Top <-> ¬ Bot).
```

6.5.2 Modal Logic

Now, we will mention the proof of the abbreviation (MLAbbr1). In the following lemma, we provide the proof of the abbreviation MLAbbr1.

```
Lemma MLAbbr1 (φ  :  MRLProp ) : MRLValid (<C>φ <-> ¬[C]¬φ).
Proof.
   start.
   MRL_P_Equal_Intro.
   MRL_P_Impl_Intro.
   MRL_P_Not_Intro.
   MRL_K_DiamondC_Elim H.
   MRL_K_BoxC_Elim H0 H3.
   MRL_P_Not_Elim H3 H.
   MRL_P_Impl_Intro.
```

```
  MRL_P_PBC.
  assume ([[C] (¬ φ)] z).
  MRL_P_Not_Elim H H3.
  MRL_K_BoxC_Intro.
  MRL_P_Not_Intro.
  assume ([<C> φ] z).
  MRL_P_Not_Elim H0 H5.
  MRL_K_DiamondC_Intro y.
  assumption.
  assumption.
Qed.
```

Please note that the *assume* tactic in the above proof is similar to a *cut* in sequent calculus. The *assume* tactic adds the formula as an assumption, and at the same time it generates a proof evidence of the formula as a goal. This is used to combine the separated proof trees together.

6.5.3 Propositional Relevance Logic

At this point we will mention the proofs of the abbreviations (PRLAbbr1 - PRLAbbr6). In the following lemma, we provide the proof of the abbreviation PRLAbbr1.

```
Lemma PRLAbbra1 (φ ψ : MRLProp)  : MRLValid (φ //\\ ψ <-> ¬ (φ ->> ¬ ψ)).
Proof.
  start.
  MRL_P_Equal_Intro.
  MRL_P_Impl_Intro.
  MRL_P_Not_Intro.
  MRL_RJ_And_Elim H.
  assume ([¬ ψ] y).
  MRL_P_Not_Elim H5 H.
  MRL_RJ_Impl_Elim H0 H4 H3.
  MRL_P_Impl_Intro.
  MRL_P_PBC.
  assume ([(φ ->> ¬ ψ)] z).
  MRL_P_Not_Elim H H3.
  MRL_RJ_Impl_Intro.
  MRL_P_Not_Intro.
  assume ([(φ //\\ ψ)] z).
  MRL_P_Not_Elim H0 H6.
  MRL_RJ_And_Intro x y.
Qed.
```

Please note that we will omit the proofs of the abbreviations (PRLAbbr2 - PRLAbbr6). Proofs are available in the Coq provided in this thesis.

```
Lemma PRLAbbra2 (φ ψ : MRLProp)  : MRLValid (φ \\// ψ <-> ¬ φ ->> ψ).
```

```
Lemma PRLAbbra3 (φ : MRLProp)    : MRLValid (N φ <-> ∽ ¬ φ).
```

```
Lemma PRLAbbra4 (φ ψ : MRLProp)  : MRLValid (φ ->o ψ <-> N (N φ ->> N ψ)).
```

```
Lemma PRLAbbra5 (φ ψ : MRLProp)  : MRLValid (φ /\o ψ <-> ¬ (φ ->o ¬ ψ)).
```

```
Lemma PRLAbbra6 (φ ψ : MRLProp)  : MRLValid (φ \/o ψ <-> ¬ φ ->o ψ).
```

6.5.4 Propositional Relevance Logic with E

In the following lemma, we will mention the proof of the abbreviation (PRLEAbbr1).

```
Lemma PRLEAbbr1 : MRLValid (U <-> N E).
Proof.
    start.
    MRL_P_Equal_Intro.
    MRL_P_Impl_Intro.
    MRL_R_NNot_Intro.
    assume ([E] z^*).
    MRL_P_Not_Elim H0 H3.
    MRL_R_E_Intro.
    MRL_R_U_Elim H (x^*).
    rewrite (DNg x).
    rewrite <- DMg2.
    rewrite <- H.
    trivial.
    MRL_P_Impl_Intro.
    MRL_P_PBC.
    assume ([¬ E] (z^*)).
    assume (z=(z^*)^*).
    MRL_R_NNot_Elim H H3 H4.
    apply DNg.
    MRL_P_Not_Intro.
    assume ([U] z).
    MRL_P_Not_Elim H0 H4.
    MRL_R_U_Intro.
    rewrite (DNg x).
    rewrite (DNg z).
    rewrite <- DMg1.
    MRL_R_E_Elim H3 (x ^*).
```

```
   rewrite <- H3.
   trivial.
Qed.
```

Please note we will omit the extra abbreviation (PRLEAbbr2) proof. Proof is available in the Coq code included in this thesis.

```
Lemma PRLEAbbr2 : MRLValid ( E <-> N U).
```

6.5.5 Boolean Algebra Axioms

Now, we want to prove the Boolean algebra axioms defined in Theorem 1 in Coq. We begin with the proof of Lemma 7.

```
Lemma lemma2.7.2              (φ : MRLProp) : MRLValid (φ <-> ∽∽φ).
Proof.
   start.
   MRL_P_Equal_Intro.
   MRL_P_Impl_Intro.
   MRL_R_Not_Intro.
   assume (z^*=((z^*)^*)^*).
   rewrite DNg in H.
   MRL_R_Not_Elim H0 H H3.
   apply DNg.
   MRL_P_Impl_Intro.
   MRL_R_Not_PBC.
   assume (z=(z^*)^*).
   MRL_R_Not_Elim H H0 H3.
   apply DNg.
Qed.
```

In the following lemma, we prove the *Commutativity of* ⩓ axiom from Theorem 1.

```
Lemma Commutativity1   (φ ψ  : MRLProp) : MRLValid (φ //\\ ψ <-> ψ //\\ φ).
Proof.
   start.
   MRL_P_Equal_Intro.
   MRL_P_Impl_Intro.
   MRL_RJ_And_Elim H.
   MRL_RJ_And_Intro y x.
   rewrite join_comm in H0.
   assumption.
   MRL_P_Impl_Intro.
   MRL_RJ_And_Elim H.
   MRL_RJ_And_Intro y x.
   rewrite join_comm in H0.
   assumption.
Qed.
```

Please note that the side-condition of the rule (⩓ E) is automatically taken care of by the tactic *MRL_RJ_And_Elim*. We will omit *Commutativity2* lemma proof. It is available in the Coq code included in this thesis.

```
Lemma Commutativity2  (φ ψ   : MRLProp) : MRLValid (φ \\// ψ <-> ψ \\// φ).
```

In the following lemma, we prove the *Identity1 for* ⩓ axiom from Theorem 1.

```
Lemma Identity1       (φ     : MRLProp) : MRLValid (φ <-> φ //\\ E).
Proof.
   start.
   MRL_P_Equal_Intro.
   MRL_P_Impl_Intro.
   MRL_RJ_And_Intro z (z*(z^*)).
   rewrite meet_comp.
   rewrite zero_ident.
   trivial.
   MRL_R_E_Intro.
   rewrite meet_comp.
   rewrite zero_ident.
   trivial.
   MRL_P_Impl_Intro.
   MRL_R_Not_PBC.
   MRL_RJ_And_Elim H.
   MRL_R_E_Elim H x.
   rewrite <- H in H3.
   rewrite H3 in H0.
   assume (x^*=((x^*)^*)^*).
   rewrite DNg in H4.
   MRL_R_Not_Elim H0 H4 H5.
   apply DNg.
Qed.
```

Please note that the side-condition of the rule (⩓ E) is automatically taken care of by the tactic *MRL_RJ_And_Elim*. We will omit *Identity2* lemma proof, but it is available in the Coq code submitted in this thesis.

```
Lemma Identity2       (φ     : MRLProp) : MRLValid (φ <-> φ /\o U).
```

In the following lemma, we prove the *Distributivity1 of ⩓ on ᛟ* axiom from Theorem 1.

```
Lemma Distributivity1 (φ ψ χ : MRLProp) : MRLValid (φ //\\ (ψ /\o χ) ->
                                           (φ //\\ ψ) /\o (φ //\\ χ)).
Proof.
   start.
   MRL_P_Impl_Intro.
   MRL_RJ_And_Elim H.
   MRL_RM_And_Elim H.
   MRL_RM_And_Intro (x + x0) (x + y0).
   rewrite H4 in H0.
   rewrite join_distr in H0.
   assumption.
   MRL_RJ_And_Intro x x0.
   MRL_RJ_And_Intro x y0.
Qed.
```

Please note that the side-condition of the rules (⩓ E and ᛟ E) is automatically taken care of by the tactics *MRL_RJ_And_Elim*, and *MRL_RM_And_Elim*. We will omit *Distributivity2* lemma proof. It is available in the Coq code included in this thesis.

```
Lemma Distributivity2 (φ ψ χ : MRLProp) : MRLValid (φ /\o (ψ //\\ χ) ->
                                           (φ /\o ψ) //\\ (φ /\o χ)).
```

In the following lemma, we prove the *Complements for ⩓* axiom from Theorem 1.

```
Lemma Complement1     (φ     : MRLProp) : MRLValid (φ ->
                                           Top /\o (U /\ (φ //\\ N φ))).
Proof.
   start.
   MRL_P_Impl_Intro.
   MRL_RM_And_Intro z (z+z^*).
   rewrite join_comp.
   rewrite one_ident.
   trivial.
   MRL_R_U_Intro.
   rewrite join_comp.
   rewrite one_ident.
   trivial.
   MRL_RJ_And_Intro z (z^*).
   MRL_R_NNot_Intro.
   rewrite <- DNg in H0.
   MRL_P_Not_Elim H0 H.
Qed.
```

Please note that the side-condition of the rule (U I) is automatically taken care of by the tactic *MRL_R_U_Intro*. We will omit *Distributivity2* lemma proof. It is available in the Coq code included in this thesis.

```
Lemma Complement2    (φ      : MRLProp) : MRLValid (φ ->
                                                   Top //\\ (E /\ (φ /\o N φ))).
```

6.5.6 Additional Boolean Algebra Axioms

In this section, we want to explain the additional Boolean algebra axioms we have proven in this thesis. In the following lemma, we have proven the associativity law for the derived relevance logic *and for join* operator.

```
Lemma Extra_Associativity1    (φ ψ χ: MRLProp) : MRLValid ((φ //\\ ψ) //\\ χ
                                                        <-> φ //\\ (ψ //\\ χ)).
Proof.
     start.
     MRL_P_Equal_Intro.
     MRL_P_Impl_Intro.
     MRL_RJ_And_Elim H.
     MRL_RJ_And_Elim H3.
     rewrite H4 in H0.
     rewrite <- join_assoc in H0.
     MRL_RJ_And_Intro x0 (y0+y).
     MRL_RJ_And_Intro y0 y.
     MRL_P_Impl_Intro.
     MRL_RJ_And_Elim H.
     MRL_RJ_And_Elim H.
     rewrite H4 in H0.
     rewrite join_assoc in H0.
     MRL_RJ_And_Intro (x+x0) y0.
     MRL_RJ_And_Intro x x0.
Qed.
```

Please note we will omit other axiom's proofs. They are available in the Coq code included in this thesis.

```
Lemma Extra_Commutativity1    (φ ψ  : MRLProp) : MRLValid (φ /\o ψ <-> ψ /\o φ).
```

```
Lemma Extra_Commutativity2    (φ ψ  : MRLProp) : MRLValid (φ \/o ψ <-> ψ \/o φ).
```

```
Lemma Extra_Associativity2    (φ ψ χ: MRLProp) : MRLValid ((φ \\// ψ) \\// χ
                                                        <-> φ \\// (ψ \\// χ)).
```

```
Lemma Extra_Associativity3    (φ ψ χ: MRLProp) : MRLValid ((φ /\o ψ) /\o χ
                                                        <-> φ /\o (ψ /\o χ)).
```

```
Lemma Extra_Associativity4    (φ ψ χ: MRLProp) : MRLValid ((φ \/o ψ) \/o χ
                                                        <-> φ \/o (ψ \/o χ)).
```

6.5.7 Boolean Contact Algebra Axioms

The implementation concludes with the proof of Boolean contact algebra axioms defined in Theorem 2 in Coq. In the following lemma, we prove the *Null disconnected (BCAx0)* axiom from Theorem 2.

```
Lemma BCAx0               : MRLValid ([C]¬ E).
Proof.
    start.
    MRL_K_BoxC_Intro.
    MRL_K_C_BCA0h H.
Qed.
```

In the following lemma, we prove the *Reflexivity (BCAx1)* axiom from Theorem 2.

```
Lemma BCAx1 (φ : MRLProp) : MRLValid (¬ E -> ([C]φ -> φ)).
Proof.
    start.
    MRL_P_Impl_Intro.
    MRL_P_Impl_Intro.
    MRL_K_C_BCA1h H.
    MRL_K_BoxC_Elim H0 H.
Qed.
```

In the following lemma, we prove the *Symmetry (BCAx2)* axiom from Theorem 2.

```
Lemma BCAx2 (φ : MRLProp) : MRLValid (φ -> ([C]<C>φ)).
Proof.
    start.
    MRL_P_Impl_Intro.
    MRL_K_BoxC_Intro.
    MRL_K_DiamondC_Intro z.
    MRL_K_C_BCA2h H0.
    assumption.
    assumption.
Qed.
```

In the following lemma, we prove the *Compatibility (BCAx3)* axiom from Theorem 2.

```
Lemma BCAx3 (φ : MRLProp) : MRLValid ([C]φ -> [C] (Top ->o φ)).
Proof.
    start.
    MRL_P_Impl_Intro.
    MRL_K_BoxC_Intro.
    MRL_RM_Impl_Intro.
    assume (x * y0 = x * y0).
    rewrite (Idm2 y0) in H5 at 2.
    rewrite meet_assoc in H5.
    rewrite <- H3 in H5.
    MRL_K_C_BCA3h H0 H5.
    MRL_K_BoxC_Elim H H6.
    trivial.
Qed.
```

In the following lemma, we prove the *Summation (BCAx4)* axiom from Theorem 2.

```
Lemma BCAx4 (φ : MRLProp) : MRLValid (φ -> [C](¬ <C> φ ->> <C> φ)).
Proof.
    start.
    MRL_P_Impl_Intro.
    MRL_K_BoxC_Intro.
    MRL_RJ_Impl_Intro.
    rewrite H3 in H0.
    MRL_K_C_BCA4 H0.
    MRL_P_PBC.
    assume ([ <C> φ] x).
    MRL_P_Not_Elim H4 H7.
    MRL_K_DiamondC_Intro z.
    MRL_K_C_BCA2h H5.
    trivial.
    trivial.
    MRL_K_DiamondC_Intro z.
    MRL_K_C_BCA2h H5.
    trivial.
    trivial.
Qed.
```

Chapter 7

Conclusion and Future Work

In this thesis, we have introduced a new logic namely modal relevance logic for mereotopology. This logic is used to reason the application related to QSR. In this logic, we have the modal operators to reason about the topological aspects and relevance operators to reason about mereological aspects. For this all to be possible we have specified a set of axioms for the applications based on QSR. We have also used a natural deduction calculus for our logic for a soundness proof. Furthermore, we have implemented the calculus in functional programming language and interactive theorem prover Coq.

In future work, firstly, we want to focus on the completeness proof of our calculus using standard techniques. Secondly, we will concentrate on the decidability of our logic using the filtration theorem. Thirdly, we would like to extend our logic by introducing other axioms of the Boolean contact algebra. Furthermore, for the sake of automation, we would like to introduce automated Ltac tactics for the proof of Boolean algebra derivations.

Bibliography

[1] Anderson, A.R., Belnap, N., Dunn, J.: *Entailment: The Logic of Relevance and Necessity (Vol. 1)*. Princeton University Press (1976).

[2] Benzmüller, C., Paleo, B.W.: *Interacting with Modal Logics in the Coq Proof Assistant*. Springer, LNCS 9139, pp. 398-411 (2015).

[3] Van, B.J., Bezhanishvili, G.: *Modal Logics of Space*. Springer, pp. 217-298 (2007).

[4] Blackburn, P., Van, B.J, Wolter, F. *Handbook of Modal Logic*. Studies in Logic and Practical Reasoning 3 (2007).

[5] Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. Cambridge Tracts in Theoretical Computer Science 8 (2002).

[6] Barmak, J.A.: *Algebraic Topology of Finite Topological Spaces and Applications.* LNM 2032 (2011).

[7] Baum, J.D.: *Elements of Point Set Topology.* Dover (1991).

[8] Bennett, B.: *Modal Logics for Qualitative Spatial Reasoning.* Logic Journal of the IGPL 4(1), pp. 23-45 (1996).

[9] Boolean Algebra (structure). `https://en.wikipedia.org/wiki/Boolean_algebra_(structure)`, Last accessed 8-July-2018.

[10] Chlipala, A.: *An Introduction to Programming and Proving with Dependent Types in Coq.* Journal of Formalized Reasoning 3(2), pp. 1-93 (2010).

[11] Dunn, J.M., Restall, G.: *Relevance logic.* In Gabbay, G., Guenthner, F. (eds.): Handbook of Philosophical Logic. pp. 1-128 (2002).

[12] Düntsch, I., Winter, M.:. *Algebraization and Representation of Mereotopological Structures.* Journal on Relational Methods in Computer Science,(Vol. 1), pp. 161-180 (2004).

[13] Düntsch, I., Winter, M.: *A Representation Theorem for Boolean Contact Algebras.* TCS 347(3), pp. 498-512 (2005).

[14] Givant, S., Halmos, P.: *Introduction to Boolean Algebras.* Springer (2009).

[15] Halmos, P.: *Lectures on Boolean Algebras.* Van Nostrand (1963).

[16] Huth, M., Ryan, M.: *Logic in Computer Science: Modelling and Reasoning About Systems.* Cambridge University Press (2004).

[17] Munkres, J,R,: *Topology: A First Course.* Prentice-Hall (1974).

[18] Read, S.: *Relevant Logic: A Philosophical Examination of Inference.* Basil Blackwell (1989).

[19] Renz, J.: *Qualitative Spatial Reasoning with Topological Information.* Springer, LNAI 2293 (2002).

[20] Sozeau, M., Oury, N.: *First-class Type Classes.* Springer, LNCS 5170, pp. 278-293 (2008).

[21] The Coq Proof Assistant. `https://coq.inria.fr/`, Last accessed 8-July-2018.