Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ”

Кафедра математики факультету інформатики

Кваліфікаційна робота

Освітній ступінь – бакалавр

на тему: **“Топологічні та теоретико-графові властивості моделей Кріпке в модальній логіці”**

Виконав студент 4-го року навчання
освітньої програми “Прикладна математика”,
спеціальності 113 Прикладна математика
Кошмак Ілля Олександрович

Керівник курсової роботи:
к. ф.-м. н. *Козеренко С.О.*
Рецензент:
к. ф.-м. н. *Пилявська О.С.*

Кваліфікаційна робота захищена
з оцінкою _____
Секретар ЕК _____
(підпис)
“____”_____ 2022 р.

Київ – 2022

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ”

Кафедра математики факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри математики,
проф., д.ф-м.н., *Б. В. Олійник*

_____

(підпис)
“____”_____ 2021

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на кваліфікаційну роботу
студенту 4-го курсу , факультету інформатики
Кошмаку Іллі Олександровичу

Тема: Топологічні та теоретико-графові властивості моделей Кріпке в модальній логіці.
Вихідні дані: Програмна реалізація алгоритму перевірки істинності формул для заданих моделей Кріпке; програмна реалізація алгоритму перевірки p-морфізму шкал та моделей Кріпке; програмна реалізація алгоритму перевірки на сильну зв’язність за допомогою модальної формули а також загальнозначимості формули Чорча-Россера.

Зміст ТЧ до кваліфікаційної роботи:

- Зміст
- Календарний план виконання робіт
- Анотація
- Вступ
- Опис класичної логіки
- Опис модальної логіки та моделей Кріпке
- Розробка алгоритму перевірки істинності формул для заданих моделей Кріпке
- Розробка алгоритму перевірки p-морфізму шкал та моделей Кріпке
- Розробка алгоритму перевірки на сильну зв'язність за допомогою модальної формули а також загальнозначимості формули Чорча-Россера
- Висновки
- Список використанної літератури

Дата видачі "____"_____ 2021 р. Керівник _____

(підпис)

Завдання отримав _____

(підпис)

Зміст

Тема: Топологічні та теоретико-графові властивості моделей Кріпке в модальній логіці.

Календарний план виконання роботи:

Номер	Назва етапу кваліфікаційної	Термін виконання етапу	Примітка
1.	Отримання теми кваліфікаційної роботи.	Жовтень 2021	
2.	Ознайомлення з темою кваліфікаційної.	Листопад 2021	
3.	Розробка плану та структури роботи. Робота з науковою літературою.	Грудень 2021	
4.	Опис основних понять класичної математичної логіки.	Січень 2022	
5.	Опис основних понять модальної логіки та моделей Кріпке. Дослідження відношень в модальній логіці.	Березень 2022	
6.	Написання алгоритму перевірки істинності формул в модальній логіці	Квітень 2022	
7.	Написання алгоритму перевірки на p-морфізм моделей та шкал.	Травень 2022	
8.	Написання алгоритму перевірки на сильну зв'язність, формулу Чорча-Россера. Оформлення текстових результатів. Попередній аналіз кваліфікаційної роботи. Виправлення помилок.	Червень 2022	

1 Антоація

В кваліфікаційній роботі розглянуто базові поняття класичної математичної логіки. Розглянуті модальні оператори $\Box$ та $\Diamond$, шкали та Моделі Кріпке. Досліджені теоретико-графові відношення в моделях Кріпке. Також, розроблена низка алгоритмів на шкалах та моделях Кріпке.

Ключові слова: модальна логіка, модель Кріпке, p-морфізм.

2 Вступ

Аристотель виділив деякі прості закономірності в людських міркуваннях, а Лейбніц мріяв звести міркування до обчислень. Однак, як математична дисципліна, логіка з'явилася відносно недавно: піонерами в XIX столітті були Больцано, Буль, Кантор, Пеано, Ч. С. Пірс та Е. Шрьодер. Їхня робота започатковує булеву алгебру, теорію множин, пропозиційну логіки, логіку предикатів, введення символічних позначень для логічних операцій [4].

У період 1900-1950 рр. важливі нові ідеї прийшли від Рассела, Хаусдорфа, Гільберта, Лусіна, Гьоделя, Тарського, Черча, Кліні, Тьюринга та Гентцена. Вони довели перші теореми математичної логіки, причому теореми Гьоделя мали значний вплив. У Гільберта, Лусіна, Тарського та Черча було багато учнів і співробітників, які становили значну частину того і наступного покоління в математичній логіці [4].

Окрім класичної математичної логіки, також виділяють: багатозначну логіку [5], темпоральну логіку, модальну логіку, нечітку логіку та ін. В цій роботі буде розглянуто класичну та модальну логіки.

Модальна логіка є гілкою некласичної логіки, яка вивчає твердження, які доповнені додатковими якостями, такими як доказованість, необхідність, можливість і тому подібні [1]. Вона народилася на початку XX століття як галузь логіки, яка застосовувалася до аналізу філософських уявлень і проблем. Сьогодні ж модальна логіка знаходиться на перехресті багатьох академічних дисциплін (філософія, символічні системи, лінгвістика, інформатика та ін.) [3].

В математичній логіці можна виділити 3 основних розділи, в контексті яких існує і модальна логіка [1]:

1. *Теорія доведень* вивчає властивості і структуру логічних виводів, доказованість в певних формальних теоріях.

2. *Теорія моделей* вивчає семантику логічних формул і їх математичну інтерпретацію.

3. *Теорія алгоритмів* ставить питання про обчислюваність за скінчену кількість кроків і про витрати ресурсів за виконання тої чи іншої процедури.

В першій частині роботи розглянуто базові поняття математичної логіки: висловлювання, інтерпретація, логічні оператори, синтаксис і семантика, формули, тавтології, моделі; наведено приклади висловлювань, використання логічних операторів, формул.

В другій частині роботи було розглянуто модальні оператори $\Box$ і $\Diamond$, синтаксис модальної логіки, моделі Кріпке, загальнозначимість в шкалах, логіку шкали; було показано, як в модальній логіці виражаються відношення рефлексивності, транзитивності, симетричності та ін.; наведено лему про невиразність іррефлексивності в модальній логіці.

В третій частині описано процес створення програмної реалізації алгоритму інтерпретації формул модальної логіки для заданих моделей Кріпке на мові Python[6], що і було основною метою даної роботи. Також в цій частині описаний алгоритм перевірки, чи є відображення p-морфізмом для заданих шкал або моделей Кріпке; а також перевірка моеделей на сильну зв'язність та загальнозначимість формули Чорча-Россера за допомогою перевірки загальнозначимості відповідних модальних формул.

3 Класична логіка

3.1 Базові поняття

Метематична логіка – математична дисципліна, яка, в широкому сенсі, вивчає математичні моделі щодо істинності або хибності первних міркувань. Будуючи відповідні моделі, вважають, що ці міркування будуються з *висловлювань*.

Означення 3.1. Висловлювання – це твердження, якому за певних обставин можуть бути надані значення істини або хиби. Таке співставлення називають *інтерпретацією*.

Означення 3.2. Двохелементна множина T = {"істина" , "хиба"} називається *областю істинності*. Також використовуються наступні позначення:

- T = {"True" , "False"};
- T = {1, 0}.

Інтерпретацією висловлювання ϕ називається відображення $t : \phi \to \{0, 1\}$. Для цього відображення повинно виконуватись наступне:

- $t(\top) = 1, \quad t(\bot) = 0;$
- $t(\neg p) = 1 - t(p);$
- $t(p \vee q) = max(t(p), t(q)), \quad t(p \wedge q) = min(t(p), t(q)).$

Означення 3.3. Просте висловлювання – це висловлювання, яке не можна подати у вигляді сполучення більш коротких висловлювань.

Приклад 3.4. Прикладом висловлювання може бути наступна фраза: "Київ – столиця України" або "НСК двух взаємно простих чисел дорівнює їх добутку".

Оскільки виключно прості висловлювання не дозволяють оперувати складними логічними конструкціями, ми будемо використовувати пропозиційну

логіку, яка вивчає схеми істинності або хибності первних логічних конструкцій, незалежно від наповнення простих тверджень. Іншими словами, ми будемо розглядати лише істинно-функціональні комбінації, утворені за допомогою логічних операцій, в яких істинність чи хибність нового речення визначається істинністю чи хибністю його складових речень (простих висловлювань).

Означення 3.5. Складене (непросте) висловлювання – висловлювання, яке будується за допомогою комбінацій простих висловлювань і логічних операцій.

Приклад 3.6. Прикладом може слугувати наступна схема: "Якщо ϕ, то ϕ або ψ"

Прості висловлювання будуть позначатись малими латинськими літерами a, b, c, ..., а складені – грецькими літерами ϕ, ψ, θ....

3.2 Логічні операції

Логічні операції використовуються як спосіб формально описати правило інтерпретації логічних (пропозиційних) сполучників.

Однією з найпростіших логічних операцій, є *заперечення*. В математичній мові заперечення позначається символом "$\neg$" , і для деякого речення a визначається через наступну схему істинності:

a	$\neg a$
1	0
0	1

Також інтуітивно зрозумілою є операція *кон'юнкції*, або логічне «і», яке позначається символом "$\wedge$". Для деяких тверджень a та b, таблиця істинності для їх кон'юнкції виглядатиме наступним чином:

a	b	$a \wedge b$
1	1	1
1	0	0
0	1	0
0	0	0

В природній мові є два способи розуміння «або»: інклюзивне і ексклюзивне. Інклюзивний спосіб розуміння «a або b» для деяких тверджень a та b означає «a або b, або a і b». Саме інклюзивне «або» є операцією *диз'юнкції*, яке позначається символом "$\vee$", для якої таблиця істинності матиме наступний вигляд:

a	b	$a \vee b$
1	1	1
1	0	1
0	1	1
0	0	0

Другий спосіб розуміння – ексклюзивне (виключне) «або», в якому «a або b» для деяких тверджень a та b означає «або a, або b». Ця операція позначається символом "$\oplus$", і таблиця істинності виглядає наступним чином:

a	b	$a \oplus b$
1	1	0
1	0	1
0	1	1
0	0	0

Наступною важливою логічною операцією є *імплікація*, або «логічний наслідок», який в природній мові застосовується у вигляді конструкцій типу «якщо a, то b» для деяких тверджень a та b; позначається символом "$\rightarrow$". Тут інтуітивного розуміння вже не вистачає, для того, щоб зрозуміти цю операцію. Так, «якщо a, то b» хибне, якщо *антецедент* a істинний, а *консеквент* b хибний, але в інших випадках це не так очевидно, оскільки в природній мові ми звикли мати певний зв'язок (зазвичай причинно-наслідковий) між умовою і наслідком. Зазначимо, що «якщо a, то b» є хибним тоді і тільки тоді, коли a є істинним, а b хибним.Тому таблиця істинності матиме наступний вигляд:

a	b	$a \rightarrow b$
1	1	1
1	0	0
0	1	1
0	0	1

Також визначимо операцію *еквівалентності*, яка в природній мові може звучати як «a тоді, і тільки тоді, коли b». Ця операція позначається символом "$\leftrightarrow$" і має наступну таблицю істинності:

a	b	$a \leftrightarrow b$
1	1	1
1	0	0
0	1	0
0	0	1

3.3 Синтаксис і семантика

Логічний формалізм починається з мови, системи шаблонів, що лежать в основі певної практики спілкування та міркування. Ці моделі є формальними, вони підкреслюють основні риси описаного явища, а також пропонують аналогії в різних ситуаціях.

Для визначення синтаксису класичної логіки зафіксуємо деяку *пропозиційну мову* $\mathfrak{L}$, алфавіт якої складається з:

- Констант $\top$ (True) і $\bot$ (False)
- Пропозиційних змінних p_0, p_1, p_2, ...;
- Пропозиційних логічних операцій: $\wedge$ (кон'юнкція), $\vee$ (диз'юнкція), $\rightarrow$ (імплікація), $\neg$ (заперечення);
- Знаків пунктуації "(" та ")";
- Та формул з $\mathfrak{L}$, які визначаються індуктивно:
  - Всі пропозиційні змінні з $\mathfrak{L}$ та логічні константи є атомарними формулами в мові $\mathfrak{L}$;
  - Якщо ϕ є формулою в $\mathfrak{L}$, то і її заперечення $\neg\phi$ теж є формулою в $\mathfrak{L}$;
  - Якщо ϕ та ψ є формулами в $\mathfrak{L}$, то $(\phi \wedge \psi)$, $(\phi \vee \psi)$ і $(\phi \rightarrow \psi)$ теж є формулами в $\mathfrak{L}$;

– Послідовність символів з алфавіту $\mathfrak{L}$ є формулою тоді, і тільки тоді, коли це випливає з трьох попередніх пунктів.

Приклад 3.7. Наступна послідовність символів є формулою:

$$((p_0 \wedge \top) \rightarrow ((\neg p_0 \vee p_1)))$$

Ми не використовували операції імплікації та виключного «але» для визначення формул мови $\mathfrak{L}$, тому що вони можуть бути визначені за допомогою інших логічних операцій (також, не обов'язкові):

- $\phi \leftrightarrow \psi = (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$;
- $\phi \oplus \psi = (\phi \vee \psi) \wedge \neg(\phi \wedge \psi)$.

Також, варто зазначити, що константа $\top$ і заперечення теж не є обов'язковими для визначення мови $\mathfrak{L}$, але зазначені у визначенні для кращого інтуітивного розуміння; вони можуть бути визначені наступним чином:

- $\neg\phi = \phi \rightarrow \bot$;
- $\top = \bot \rightarrow \bot$.

Позначимо набір з усіх змінних мови $\mathfrak{L}$ через $Var(\mathfrak{L})$, а набір з усіх формул – через $For(\mathfrak{L})$. Всі формули, яки використовувались у побудові формули ϕ, в тому числі сама формула ϕ називаються підформулами ϕ і позначаються за допомогою $Sub(\phi)$; набір всіх змінних, які використовувались у формулі ϕ позначаються через $Var(\phi)$. Ми будемо використовувати форму запису $\phi(q_0, q_1, q_2, ...)$ щоб показати, що $Var(\phi) \subseteq \{q_0, q_1, q_2, ...\}$.

Означення 3.8. Кажуть, що ϕ це *тавтологія* (записують $\models \phi$) якщо $t(\phi) = 1$ для всіх $t : For(\mathfrak{L}) \rightarrow \{0, 1\}$. При цьому, кажуть, що ϕ *виконувана*, якщо $t(\phi) = 1$ для хоча б одного $t : For(\mathfrak{L}) \rightarrow \{0, 1\}$.

Приклад 3.9. Прикладом тавтології може бути вираз:

$$(p \vee \neg p)$$

Наслідок 3.10. $\top$ *це тавтологія [4].*

Семантика (правила інтерпретації) мови $\mathfrak{L}$ полягає у наступних твердженнях:

- Кожне висловлювання завжди або істинне, або хибне (але не одночасно), при цьому константа $\bot$ завжди хибна;
- Інтерпретація складних висловлювань визначається їхніми таблицями істинності.

Означення 3.11. Також важливим буде визначити правло висновку для пропозиційної логіки:

Modus Ponens (MP): з p та $p \to q$ слідує q.

Означення 3.12. Тепер ми можемо визначити *класичну модель* $\mathfrak{M}$ мови $\mathfrak{L}$, яка є підмножиною множини $Var(\mathfrak{L})$ і містить лише ті атомарні формули, які є істинними в мові $\mathfrak{L}$. За індукцією на формулі ϕ ми визначаємо відношення $\mathfrak{M} \models \phi$ яке розуміється як «ϕ істинне в моделі $\mathfrak{M}$»:

- $\mathfrak{M} \models \bot$ – хибне;
- $\mathfrak{M} \models p \leftrightarrow \forall p \in Var(\mathfrak{L}) : p \in \mathfrak{M}$;
- $\mathfrak{M} \models \phi \wedge \psi \leftrightarrow (\mathfrak{M} \models \phi \wedge \mathfrak{M} \models \psi)$;
- $\mathfrak{M} \models \phi \vee \psi \leftrightarrow (\mathfrak{M} \models \phi \vee \mathfrak{M} \models \psi)$;
- $\mathfrak{M} \models \phi \to \psi \leftrightarrow (\mathfrak{M} \models \phi$ коли $\mathfrak{M} \models \psi)$

При цьому, якщо ϕ не є істинним в $\mathfrak{M}$ то пишуть $\mathfrak{M} \not\models \phi$

Означення 3.13. Модель $\mathfrak{M}$ називають *моделлю на множині формул* Γ (і позначають $\mathfrak{M} \models \Gamma$) якщо $\forall \phi \in \Gamma : \mathfrak{M} \models \phi$.

Означення 3.14. Формула ϕ називається *дійсною* якщо вона істинна у всіх моделях класичної мови $\mathfrak{L}$ (позначають $\models \phi$).

Приклад 3.15. Приклад дійсної формули:

$$p \vee (p \to \bot)$$

Означення 3.16. Нарешті, можна визначити *класичну логіку в мові* $\mathfrak{L}$ як множину $CL_{\mathfrak{L}}$ всіх дійсних формул в мові $\mathfrak{L}$:

$$CL_{\mathfrak{L}} = \{\phi \in For(\mathfrak{L}) : \models \phi\}$$

4 Модальна логіка

4.1 Синтаксис і семантика

Системи з модальностями вперше почав описувати Кларенс Люїс на початку XX-го сторіччя, який ввів поняття $\Box$ (необхідность; box) і $\Diamond$ (можливість; diamond). Тоді у цих модальностей не було семантики, вони розумілись інтуітивно [3].

Існує декілька способів розуміння $\Box$ [1]:

- $\Box\phi$ — "ϕ необхідне"(алетична модальність);
- $\Box\phi$ — $\Box$ як предикат доказованності в арифметичній теорії T (наприклад, в арифметиці Пеано);
- $\Box\phi$ — $\Box$ як внутрішність (Interior) множини в топологічному просторі (топологічна модальність);
- $\Box\phi$ — "завжди буде ϕ"(часова модальність).

Також, існує і декілька способів розуміння $\Diamond$ [1]:

- $\Diamond\phi$ — "ϕ достатнє"(алетична модальність);
- $\Diamond\phi$ — $\Diamond$ як предикат несуперечності;
- $\Diamond\phi$ — $\Diamond$ як замикання (Closure) множини в топологічному просторі (топологічна модальність);
- $\Diamond\phi$ — "колись буде ϕ"(часова модальність).

Для визначення синтаксису модальної логіки зафіксуємо деяку *пропозиційну мову* $\mathfrak{L}$, алфавіт якої складається з [1]:

- Констант $\top$ (True) і $\bot$ (False)
- Пропозиційних змінних p_0, p_1, p_2, ...;
- Пропозиційних логічних операцій: $\wedge$ (кон'юнкція), $\vee$ (диз'юнкція), $\rightarrow$ (імплікація), $\neg$ (заперечення);

- Пропозиційною модальністю $\Box$
- Знаків пунктуації "(" та ")";
- Та формул з $\mathfrak{L}$, які визначаються індуктивно:
  - Всі пропозиційні змінні з $\mathfrak{L}$ та логічні константи є атомарними формулами в мові $\mathfrak{L}$;
  - Якщо ϕ є формулою в $\mathfrak{L}$, то і її заперечення $\neg\phi$ теж є формулою в $\mathfrak{L}$;
  - Якщо ϕ та ψ є формулами в $\mathfrak{L}$, то $(\phi \wedge \psi)$, $(\phi \vee \psi)$ і $(\phi \to \psi)$ теж є формулами в $\mathfrak{L}$;
  - Якщо ϕ є формулою в $\mathfrak{L}$, то і $\Box\phi$ теж є формулою в $\mathfrak{L}$;
  - Послідовність символів з алфавіту $\mathfrak{L}$ є формулою тоді, і тільки тоді, коли це випливає з чотирьох попередніх пунктів.

Для визначення формул модальної мови $\mathfrak{L}$ не використовувалась модальність $\Diamond$ через те, що це не є необхідним, оскільки вона визначається через модальність $\Box$:

$$\Diamond\phi = \neg\Box\neg\phi$$

4.2 Модель Кріпке

Означення 4.1. *Шкала Кріпке* – це пара $\mathfrak{F} = \langle W, R\rangle$, де W – непорожня множина станів, а $R \subseteq W \times W$ – це бінарне відношення на множині W [1]. При цьому:

- Множину W називають носієм шкали (множиною можливих станів або світів);
- Відношення R називають відношенням досяжності (xRy означає, що y досяжне з x, або "x бачить y");
- Можна вважати шкалу Кріпке орієнтованим графом, в якому W – множина вершин, і R – множина ребер.

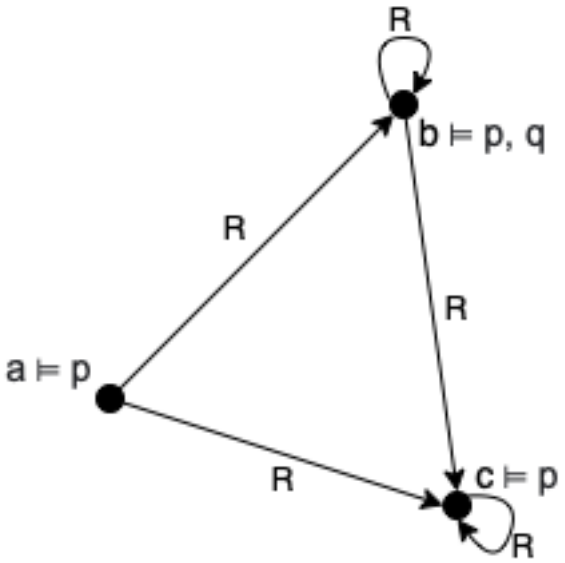
Також позначимо $R(x) = \{y \mid xRy\}$ – множину усіх досяжних із x станів.

Означення 4.2. Нехай $\mathfrak{F} = \langle W, R\rangle$ — це шкала Кріпке. Модель Кріпке — це впорядкована пара $\mathfrak{M} = \langle \mathfrak{F}, \vartheta\rangle$, де $\vartheta : \mathbb{V} \to 2^W$— це функція оцінки [1].

Відношення істинності $\mathfrak{M}, w \models \phi$ визначається індукцією по формулі ϕ [3]:

- $\mathfrak{M}, w \models p \Leftrightarrow w \in \vartheta(p)$
- $\mathfrak{M}, w \models \neg\phi \Leftrightarrow \mathfrak{M}, w \not\models \phi$
- $\mathfrak{M}, w \models \phi \vee \psi \Leftrightarrow \mathfrak{M}, w \models \phi$ або $\mathfrak{M}, w \models \psi$
- $\mathfrak{M}, w \models \phi \wedge \psi \Leftrightarrow \mathfrak{M}, w \models \phi$ та $\mathfrak{M}, w \models \psi$
- $\mathfrak{M}, w \models \Box\phi \Leftrightarrow \forall v\ wRv \to \mathfrak{M}, v \models \phi$
- $\mathfrak{M}, w \models \Diamond\phi \Leftrightarrow \exists u \in R(w)\ \mathfrak{M}, u \models \phi$

Приклад 4.3. Приклад моделі Кріпке:



Для даної моделі виконується $\vartheta(p) = \{a, b, c\}$ та $\vartheta(q) = \{b\}$. До того ж aRb, aRc, bRb, bRc і cRc. Тоді наступні формули істинні в даній моделі:

- $a \models \Box p, \Box p \to \Box\Box p, \Diamond p \to \Diamond\Box p$

  Доведення. Для a маємо: що $R(a) = \{b, c\} \subseteq \vartheta(p)$. Отже, дійсно $a \models \Box p$. Також, для a виконується $R(R(a)) = \{b, c\} \subseteq \vartheta(p)$. З цього маємо, що $a \models \Box p \to \Box\Box p$. Оскільки $R(a) = \{b, c\} \subseteq \vartheta(p)$, то $a \models \Diamond p$. А з того, що $R(R(a)) = \{b, c\} \subseteq \vartheta(p)$ маємо, що $a \models \Diamond\Box p \Rightarrow a \models \Diamond p \to \Diamond\Box p$. □

- $b \models \Box p \to p, \Box q \to q$

  Доведення. Оскільки $R(b) = \{b, c\} \subseteq \vartheta(p)$, то $b \models \Box p$, а p істинне в b за умовою $\Rightarrow b \models \Box p \to p$. З того, що $R(b) = \{b, c\} \subsetneq \vartheta(q)$, маємо, що $b \not\models \Box q$; але q дійсне в b за умовою $\Rightarrow b \models \Box q \to q$. □

- $c \models \Box p \to p$

  Доведення. З того, що $R(c) = \{c\} \subseteq \vartheta(p)$, маємо $c \models \Box p$; а p дійсне в c за умовою. Отже, дійсно $c \models \Box p \to p$. □

- $\mathfrak{M} \models \Box p \to \Diamond p$

  Доведення. $\mathfrak{M} \models \Box p \to \Diamond p$ означає, що $a, b, c \models \Box p \to \Diamond p$. Для a маємо, що з того, що $R(a) = \{b, c\} \subseteq \vartheta(p)$ то антецедент і консеквент істинні в a, а отже $a \models \Box p \to \Diamond p$. Для b також $R(b) = \{b, c\} \subseteq \vartheta(p) \Rightarrow b \models \Box p \to \Diamond p$. Для c має місце $R() = \{c\} \subseteq \vartheta(p) \Rightarrow c \models \Box p \to \Diamond p$. □

Означення 4.4. Формула ϕ істинна в моделі $\mathfrak{M}$, якщо для будь-якої точки $w \in W$ $\mathfrak{M}, w \models \phi$. У короткій формі $\mathfrak{M} \models \phi$ [1].

Означення 4.5. Формула ϕ загальнозначима в шкалі $\mathfrak{F}$, якщо для будь-якої оцінки ϑ, в будь-якій моделі $\mathfrak{M} = \langle \mathfrak{F}, \vartheta \rangle$ формула ϕ істинна, що еквівалентно $\mathfrak{M} \models \phi$. Записуємо як $\mathfrak{F} \models \phi$ [1].

Означення 4.6. Нехай $\mathfrak{F}$ — це шкала Кріпке. Логікою шкали $\mathfrak{F}$ називається множина формул, які є загальнозначимими в даній шкалі. Позначається $Log(\mathfrak{F}) = \{\phi \in \mathfrak{F} \mid \mathfrak{F} \models \phi\}$[3].

Означення 4.7. Нехай $\mathbb{F}$ — це клас шкал. Логікою класу шкал $\mathbb{F}$ називається множина формул, загальнозначних в кожній з цих шкал в даному класі. Позначається $Log(\mathbb{F}) = \bigcap\limits_{\mathfrak{F}\in\mathbb{F}} Log(\mathfrak{F})$ [1].

4.3 Властивості моделей Кріпке

В моделях Кріпке мають місце наступні формули:

- ***AT*** Нехай $\mathfrak{F} = \langle W, R\rangle$ – шкала Кріпке. Тоді формули $\Box p \to p$, $p \to \Diamond p \in Log(\mathfrak{F}) \Leftrightarrow R$ – рефлексивне.

  Доведення. Якщо $\mathfrak{F} = \langle W, R\rangle$ не рефлексивне, то $\neg xRx$ для $x \in W$. Покладемо $\vartheta(p) = W - \{x\}$, з чого слідує, що $x \models \Box p$ і $x \not\models p$, звідки $x \not\models \Box p \to p$. Навпаки, якщо $\mathfrak{F}$ рефлексивне, то $\mathfrak{F} \models \Box p \to p$, бо інакше існує модель $\mathfrak{M}$ для $\mathfrak{F}$ така, що $\mathfrak{M}, x \models \Box p$ і $\mathfrak{M}, x \not\models \Box p$ для $x \in W$; але з того, що xRx ми також повинні мати $\mathfrak{M}, x \models p$, що є суперечністю [1]. □

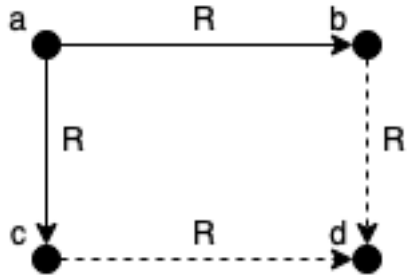
- ***A4*** Нехай $\mathfrak{F} = \langle W, R\rangle$ – шкала Кріпке. Тоді формули $\Box p \to \Box\Box p$, $\Diamond\Diamond p \to \Diamond p \in Log(\mathfrak{F}) \Leftrightarrow R$ – транзитивне.

  Доведення. Припустимо, що $\mathfrak{F} = \langle W, R\rangle$ не транзитивне. Тоді існують такі $x, y, z \in W$, що xRy, yRz та $\neg xRz$. Зафіксуємо $\vartheta(p) = W - \{z\}$. Тоді $x \models \Box p$ і $y \not\models \Box p$, а отже $x \not\models \Box\Box p$. □

- ***AB*** Нехай $\mathfrak{F} = \langle W, R\rangle$ – шкала Кріпке. Тоді формула $p \to \Box\Diamond p \in Log(\mathfrak{F}) \Leftrightarrow R$ – симетричне.

  Доведення. Припустимо, що $\mathfrak{F} = \langle W, R\rangle$ не симетричне. Тоді існують такі $x, y \in W$, що xRy та $\neg yRx$. Зафіксуємо $\vartheta(p) = \{x\}$. Тоді ми матимемо $x \models p$ та $y \not\models \Diamond p$, з чого слідує, що $x \not\models \Box\Diamond p$ і $x \not\models p \to \Box\Diamond p$ [1]. □

- ***ACR*** Нехай $\mathfrak{F} = \langle W, R\rangle$ – шкала Кріпке. Тоді формула $\Diamond\Box p \to \Box\Diamond p \in Log(\mathfrak{F}) \Leftrightarrow \forall w \forall w_1, w_2 \in R(x) : R(w_1) \cap R(w_2) \neq \emptyset$ (формула Чорча-Россера)



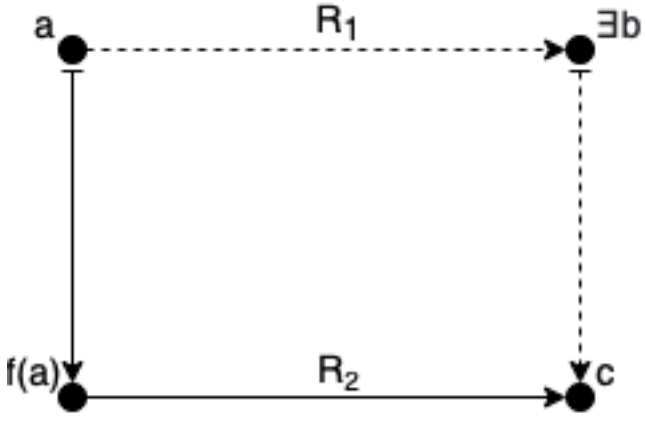
Формула Чорча-Россера

- ***AD*** Нехай $\mathfrak{F} = \langle W, R\rangle$ – шкала Кріпке. Тоді формули $\Box p \to \Diamond p, \Diamond\top \in Log(\mathfrak{F}) \Leftrightarrow R$ – серіальне.

  Доведення. Зафіксуємо $\mathfrak{M} = \langle \mathfrak{F}, \vartheta\rangle$ як модель, яка не є серіальною на шкалі $\mathfrak{F} = \langle W, R\rangle$. Тоді $x \models \Box p$ і $x \not\models \Diamond p$ для $x \in W$. З цього маємо, що x є "глухим кутом" в $\mathfrak{F}$, і якщо xRy для $y \in W$ то $y \models p$ та $y \not\models p$, що, очевидно, є суперечністю [1]. □

Означення 4.8. Також зафіксуємо *p-морфізм* – природній гомоморфізм шкал і моделей Кріпке. ρ-морфізмом шкал Кріпке називається відображення $f : \mathfrak{F}_1 \to \mathfrak{F}_2$, таке що [1]:

1. f — монотонне, $aR_1b \Rightarrow f(a)R_2f(b)$, де $a, b \in W_1$
2. f має *властивість підняття* $f(a)R_2c \Rightarrow \exists b \in W_1 \;\; aR_1b \;\; \& \;\; f(b) = c$



Властивість підняття

Означення 4.9. *Сюр'єктивний p-морфізм* це таке відображення $f : \mathfrak{F}_1 \to \mathfrak{F}_2$, що $\forall y \in \mathfrak{F}_2 \;\; \exists x \in \mathfrak{F}_1 : f(x) = y$. Позначається $f : \mathfrak{F}_1 \to \mathfrak{F}_2$ [1].

Означення 4.10. Нехай $\mathfrak{M}_1 = \langle \mathfrak{F}_1, \vartheta_1 \rangle$ та $\mathfrak{M}_2 = \langle \mathfrak{F}_2, \vartheta_2 \rangle$ — моделі Кріпке. *p-морфізмом моделей Кріпке* $f : \mathfrak{M}_1 \to \mathfrak{M}_2$ називається ρ-морфізм шкал $f : \mathfrak{F}_1 \to \mathfrak{F}_2$ за наступною умовою: $\mathfrak{M}_1, w \models p \Leftrightarrow \mathfrak{M}_2, f(w) \models p$, для всіх $w \in \mathfrak{M}_1$, де p — це змінна [1]. p-морфізми дозволяють відображати моделі та шкали, зберігаючи істинність формул.

Лема 4.11. *Про p-морфізми.*

1. *Нехай* $\mathfrak{M}_1, \mathfrak{M}_2$ – *моделі Кріпке та* $f : \mathfrak{M}_1 \to \mathfrak{M}_2$. *Тоді* $\mathfrak{M}_1, w \models \phi \Leftrightarrow \mathfrak{M}_2, f(w) \models \phi$

   Доведення. Індукція по побудові формули ϕ. Розглянемо випадок, коли $\phi = \Diamond\psi$. Покажемо, що $\mathfrak{M}_1, w \models \Diamond\psi \Leftrightarrow \mathfrak{M}_2, f(w) \models \Diamond\psi$.
   ($\Rightarrow$) Нехай $\mathfrak{M}_1, w \models \Diamond\psi$. Тоді знайдеться $u \in R_1(w)$ таке, що $\mathfrak{M}_1, u \models \psi$. По припущенні індукції, $\mathfrak{M}_2, f(u) \models \psi$. За монотонністю $wRu \Rightarrow f(w)R_2f(u)$. Тоді $\mathfrak{M}_2, f(w) \models \Diamond\psi$.
   ($\Leftarrow$) Нехай $\mathfrak{M}_2, f(w) \models \Diamond\psi$. Тоді знайдеться такий $w' \in R_w(f(w))$, що $\mathfrak{M}_2, w' \models \psi$. Оскільки $f(w)R_2w'$, то знайдеться таке $u \in W_1$, що $f(u) = w'$ та wRu. За припущенням індукції, $\mathfrak{M}_1, u \models \psi$. Тоді $\mathfrak{M}_1, w \models \Diamond\psi$ [4]. □

2. *Нехай* $f : \mathfrak{F}_1 \to \mathfrak{F}_2$, *тоді* $Log(\mathfrak{F}_1) \subseteq Log(\mathfrak{F}_2)$

   Доведення. Нехай $\mathfrak{F}_2 \not\models \phi$. Тоді знайдеться така оцінка ϑ та $y \in W$, що в моделі $\mathfrak{M}_2 = \langle \mathfrak{F}_2, \vartheta \rangle$ ми матимемо $\mathfrak{M}_2, y \not\models \phi$. Тобто, $\mathfrak{M}_2, y \models \neg\phi$. Введемо на $\mathfrak{F}_1$ оцінку $\sigma(p) = f^{-1}(\vartheta(p))$. В силу сур'єктивності f, знайдеться таке $x \in W_1$, що $f(x) = y$. Звідки $\mathfrak{M}_1, x \models \neg\phi$. З цього слідує, що $\mathfrak{M}_1, x \not\models \phi$ [4]. □

Лема 4.12. *Не існує модальної формули* ϕ, *такої, що для будь-якої шкали* $\mathfrak{F}$ *має місце* $\mathfrak{F} \models \phi \Leftrightarrow \forall x \in W \ \neg(xRx)$.

Доведення. Розглянемо шкали $\langle \mathbb{N}, < \rangle$ та $\langle \{*\}, R = \{(*, *)\} \rangle$ (рефлексивна точка). Введемо $f : \langle \mathbb{N}, < \rangle \to \langle \{*\}, R \rangle$, де $f : n \to *$. Можна побачити, що f — монотонне та сур'єктивне: якщо $n < m$, то $f(n) = *$ та $f(m) = *$, а $*R*$ ($*$ бачить сама себе за визначенням). Також f має властивість підняття. Нехай $f(n)R*$, покладемо $m := n + 1$. Тоді $n < m$ і $f(m) = *$. За леммою

про р-морфізми, $Log(\langle \mathbb{N}, < \rangle) \subseteq \langle \{*\}, R \rangle$. При цьому перша шкала іррефлексивна, а друга — ні. Що це означає? Припустимо, що існує формула, яка виражає іррефлесивність відношення. Тоді вона б належала $Log(\langle \mathbb{N}, < \rangle)$, але ми вже показали, що $Log(\langle \mathbb{N}, < \rangle) \subseteq \langle \{*\}, R \rangle$; а отже, формула, що виражає іррефлексивність, належить логіці рефлексивної точки. Це викликає суперечність [1]. □

4.4 Нормальна модальна логіка

Означення 4.13. Множина формул $\mathfrak{L}$ називається *нормальною модальною логікою*, якщо [1]:

- Всі бульові тавтології лежать у $\mathfrak{L}$;
- В $\mathfrak{L}$ лежить формула $\boldsymbol{AK}$ $\Box(p \to q) \to (\Box p \to \Box q)$;
- $\mathfrak{L}$ замкнене відносно наступних правил:
  - $\boldsymbol{MP}$ $\phi \in \mathfrak{L}$ & $\phi \to \psi \in \mathfrak{L} \Rightarrow \psi \in \mathfrak{L}$
  - $\boldsymbol{Nec}$ $\phi \in \mathfrak{L} \Rightarrow \Box\phi \in \mathfrak{L}$
  - $\boldsymbol{Sub}$ $\phi(p) \in \mathfrak{L} \Rightarrow \phi[p := \psi]$, де $\psi \in \mathfrak{L}$

Теорема 4.14. *Теорема коректності [3] Нехай $\mathfrak{F} = \langle W, R \rangle$, тоді $Log(\mathfrak{F})$ — нормальна модальна логіка.*

Доведення. Загальнозначимість тавтологій і замкнутість $Log(\mathfrak{F})$ відносно $\boldsymbol{MP}$ очевидна.

Покажемо, що в довільній шкалі формула $\boldsymbol{AK}$ є загальнозначимою.Нехай $\vartheta : \mathbb{V} \to 2^W$ — оцінка і нехай $\mathfrak{M} = \langle \mathfrak{F}, \vartheta \rangle$ — модель на шкалі $\mathfrak{F}$. Припустимо, що для будь-якого $w \in W$, якщо $\mathfrak{M}, w \models \Box(p \to q)$ і $\mathfrak{M}, w \models \Box p$. Тоді для будь-якого $v \in R(w)$, $\mathfrak{M}, v \models (p \to q)$ і $\mathfrak{M}, v \models p$. Звідси, $\mathfrak{M}, v \models q$. Тоді і $\mathfrak{M}, w \models \Box q$.

Покажемо, що $Log(\mathfrak{F})$ замкнута відносно правила Nec. Нехай $\mathfrak{F} \models \phi$. Тоді для будь-якої оцінки ϑ і для будь-якої моделі $\mathfrak{M} = \langle \mathfrak{F}, \vartheta \rangle$, для будь-якого $x \in W$, $\mathfrak{M}, x \models \phi$. Легко зрозуміти, що $\mathfrak{M}, x \models \Box\phi$.

Покажемо, що $Log(\mathfrak{F})$ замкнута відносно правила Sub. Для цього введемо оцінку формули $||\phi||_{\mathfrak{M}} = \{x \in W | \mathfrak{M}, x \models \phi\}$, де $\mathfrak{M}$ — це модель. Нехай

$\mathfrak{F} \models \phi(p)$, ψ — це модальна формула і $\mathfrak{M} = \langle \mathfrak{F}, \vartheta \rangle$ — модель. Розглянемо модель $\mathfrak{M}' = \langle \mathfrak{F}, \vartheta' \rangle$, таку що $\vartheta'(p) = ||\psi||_{\mathfrak{M}}$. Нескладною індукцією по ϕ можна встановити наступну еквівалентність $\mathfrak{M}, x \models \phi\{x := \psi\} \Leftrightarrow \mathfrak{M}', x \models \phi(p)$ □

Наслідок 4.15. *Нехай $\mathbb{F}$ — це клас шкал Кріпке, тоді $Log(\mathbb{F})$ — це нормальна модальна логіка [1].*

Означення 4.16. *Мінімальна нормальна модальна логіка K* задається наступними аксіомами і правилами виводу:

- Аксіоми класичної логіки висловлювань
- Аксіома нормальності (аксіома Кріпке) $\Box(p \to q) \to (\Box p \to \Box q)$
- Правила виводу:
  - $\frac{\phi \; \phi \to \psi}{\psi} MP$
  - $\frac{\phi}{\Box \phi} Nec$
  - $\frac{\phi(p)}{\phi[p := \psi]} Sub$

Означення 4.17. *Виводом в мінімальній нормальній модальній логіці K* називається скінчена послідовність формул, кожна з яких або аксіома, або отримана з попередніх формул за правилами MP, Nec, Sub [1].

Означення 4.18. Нормальна модальна логіка $\mathfrak{L}$ *повна за Кріпке*, якщо вона є логікою деякого класу шкал. Тобто, існує клас $\mathbb{F}$, такий, що $\mathfrak{L} = Log(\mathbb{F})$.

5 Програмна реалізація алгоритмів для моделей Кріпке

Для реалізації алгоритмів було обрано мову програмування з відкритим програмним кодом Python[6] версії 3.x.

5.1 Перевірка істинності формул

Алгортим поділейний на дві частини. Перша частина (файл *kripke.py*) відповідає за класи світів та моделей Кріпке. Для початку, було визначено клас світу, кожний з яких задається за допомогою назви ($name$) у форматі рядка (str), та таблиці істинності для змінних ($values\_dict$) у форматі словника ($dict$). Клас створюється за лопомогою спеціального методу $\_\_init\_\_(self, name : str, values\_dict : dict)$:

```
class World:
    def __init__(self, name: str, values_dict: dict):
        """
        Generate world with given truth table.
        :param name:                str : name of the world
        :param values_dict:         dict: truth table
        """
        self.name = name
        self.assert_logic_values(values_dict)
        self.values_dict = values_dict
```

Також, в класі світу додатково визначено методи:

- $\_\_getitem\_\_(self,\ item\ :\ str)$ – для доступа до значень змінних світу у форматі словника ($світ[“змінна”]$);
- $\_\_repr\_\_(self)$ – для відображення світу, при спробі вивести світ у консоль; виводить назву світу та значення всіх змінних;
- $assert\_logic\_values(self,\ values\_dict\ :\ dict)$ – перевірка на коректність даних при створенні або оновленні світу;
- $update\_truth\_table(self,\ values\_dict)$ – оновлення таблиці істинності для цього світу.

```
def __getitem__(self, item: str):
    """
    Dict way of getting a variable from the world. E.g. World_name['variable_name'].
    :param item:                str : name of the variable
    :return:                    bool: value of the variable
    """
    assert item in self.values_dict.keys(), f'There is no statement `{item}` in world {self.name}'
    return self.values_dict[item]

def __repr__(self):
    """
    The way how the world is represented in console or when printing.
    :return:                    str: string representation of the world
    """
    string = f'World {self.name} with values:'
    for key, val in self.values_dict.items():
        string += f'\n\t`{key}`: {val}'
    return string

def assert_logic_values(self, values_dict: dict):
    """
    Check if all values of variables in given truth table are bool.
    :param values_dict:         dict: truth table
    """
    for key, val in values_dict.items():
        assert type(val) == bool, f'False value `{val}` of key `{key}` in world {self.name}.' \
                                  f'\nThe value for statements has to be always `True` or `False`.'

def update_truth_table(self, values_dict):
    """
    Update truth table of the world with given truth table.
    :param values_dict:         dict: truth table
    :return:
    """
    self.assert_logic_values(values_dict)
    self.values_dict = values_dict
```

Далі було визначено клас моделі Кріпке, який задається за допомогою параметру R та оцінки ϑ; в їх ролі виступає словник ($values\_dict$) з двума словниками для таблиці істинності ($definitions$) та відношеннями ($relations$).

```
class KripkeModel:
    def __init__(self, values: dict):
        """
        Generate Kripke model using given relations table and truth tables.
        :param values:
        """
        # Check if given dict is acceptable.
        assert 'relations' in values.keys(), 'There is no `relations` key in given table!'
        assert 'definitions' in values.keys(), 'There is no `definitions` key in given table!'

        self.worlds = {name: World(name, definition) for name, definition in values['definitions'].items()}
        self.assert_relations(values['relations'])
        self.relations = values['relations']
```

Додатково в класі моделі Кріпке визначено наступні методи:

- $\_\_getitem\_\_$ – для доступу до світів у форматі словника;
- $\_\_repr\_\_$ – для відображення моделі Кріпке у консолі; виводить відношення в моделі та інформацію про кожний зі світів;
- $assert\_relations$ – метод для перевірки існування світів для заданих відношень;
- add_world – додавання нового світу у модель;
- $remove\_world$ – видалення світу з моделі (в тому числі всіх відношень, де цей світ фігурує);
- $update\_relations$ – оновлення відношень у моделі.

```
    def __getitem__(self, item: str):
        """
        Dict way of getting variable from the Kripke model. E.g. KripkeModel_name['World_name'].
        :param item:                    str  : name of the variable
        :return:                        World: World object with given name
        """
        assert item in self.worlds.keys(), f'There is no world `{item}` in this Kripke Model!'
        return self.worlds[item]

    def __repr__(self):
        """
        The way how the Kripke model is represented in console or when printing.
        :return:                    str: string representation of the Kripke model
        """
        value_to_print = "Relations:"
        for key, value in self.relations.items():
            value_to_print += f"\n\t{key}: {value}"

        value_to_print += "\nTruth tables:"
        for world in self.worlds.values():
            value_to_print += "\n" + world.__repr__()

        return value_to_print

    def assert_relations(self, relations: dict):
        """
        Check if all worlds in relations are presented in the Kripke model.
        :param relations:
        :return:
        """
        for key, value in relations.items():
            # Check if given world is presented in the Kripke model.
            assert key in self.worlds.keys(), f'There is no world {key} in this Kripke model!'
            for val in value:
                # Check if given world is presented in the Kripke model.
                assert val in self.worlds.keys(), f'There is no world {key} in this Kripke model!'


def add_world(self, world_name, truth_table: dict):
    """
    Add world to the Kripke model.
    :param world_name:              str : name of the new world
    :param truth_table:             dict: truth table for the given world
    :return:
    """
    # Check if there is no world with this name in the Kripke model.
    assert world_name not in self.worlds.keys(), f'The world with name {world_name} already in the Kripke model!'
    self.worlds[world_name] = World(world_name, truth_table)
```

```
def remove_world(self, world_name: str):
    """
    Remove world from the Kripke model. Also delete all connections from and to this world.
    :param world_name:              str: name of the world to delete
    :return:
    """
    # Check if the given world is presented in the Kripke model.
    assert world_name in self.worlds.keys(), f'There is no world {world_name} in the Kripke model!'
    del self.worlds[world_name]
    del self.relations[world_name]
    for key in self.relations.keys():
        self.relations.remove(world_name)

def update_relations(self, relations: dict):
    """
    Update relations of the worlds in the Kripke model.
    :param relations:               dict: dict of sets of all new relations in form
                                          {'world_from': {'world_to_1', 'world_to_2'}}
    """
    self.assert_relations(relations)
    self.relations = relations
```

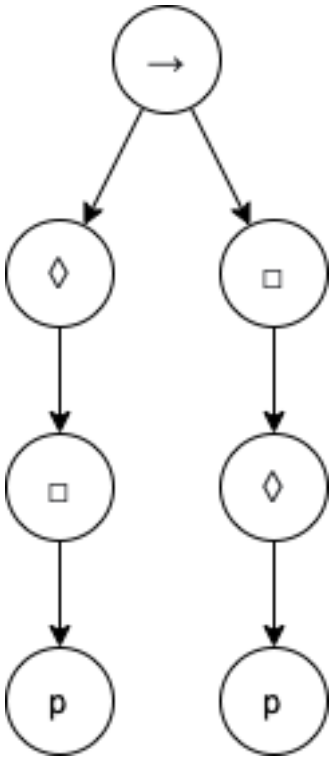
Друга частина (файл *operators.py*) містить у собі окремий клас для кожного з доступних операторів. Вони умовно поділяються на унарні (заперечення [Not], необхідність Box], достатність [Diamond]), бінарні (кон'юнкція [And], диз'юнкція [Or], виключне "але" [Xor], імплікація [Implies], еквівалентність [Iff]) та на атомарне висловлювання [Atom].

```
class Atom:
    def __init__(self, name):
        self.name = name

    def evaluate(self, km: KripkeModel, world: str):
        return km[world][self.name]

    def display(self, km: KripkeModel, world: str):
        return self.name


class And:
    def __init__(self, left, right):
        self.left = left
        self.right = right

    def evaluate(self, km: KripkeModel, world: str):
        return self.left.evaluate(km, world) and self.right.evaluate(km, world)

    def display(self, km: KripkeModel, world: str):
        return f'({self.left.display(km, world)} ∧ {self.right.display(km, world)})'
```

```
class Or:
    def __init__(self, left, right):
        self.left = left
        self.right = right

    def evaluate(self, km: KripkeModel, world: str):
        return self.left.evaluate(km, world) or self.right.evaluate(km, world)

    def display(self, km: KripkeModel, world: str):
        return f'({self.left.display(km, world)} ∨ {self.right.display(km, world)})'


class Xor:
    def __init__(self, left, right):
        self.left = left
        self.right = right

    def evaluate(self, km: KripkeModel, world: str):
        return self.left.evaluate(km, world) ^ self.right.evaluate(km, world)

    def display(self, km: KripkeModel, world: str):
        return f'({self.left.display(km, world)} ⊕ {self.right.display(km, world)})'


class Not:
    def __init__(self, inner):
        self.inner = inner

    def evaluate(self, km: KripkeModel, world: str):
        return not self.inner.evaluate(km, world)

    def display(self, km: KripkeModel, world: str):
        return f'¬{self.inner.display(km, world)}'


class Implies:
    def __init__(self, left, right):
        self.left = left
        self.right = right

    def evaluate(self, km: KripkeModel, world: str):
        return not self.left.evaluate(km, world) or self.right.evaluate(km, world)

    def display(self, km: KripkeModel, world: str):
        return f'({self.left.display(km, world)} → {self.right.display(km, world)})'
```

```python
class Iff:
    def __init__(self, left, right):
        self.left = left
        self.right = right

    def evaluate(self, km: KripkeModel, world: str):
        return (not self.left.evaluate(km, world) or self.right.evaluate(km, world)) and (
                    not self.right.evaluate(km, world) or self.left.evaluate(km, world))

    def display(self, km: KripkeModel, world: str):
        return f'({self.left.display(km, world)} ↔ {self.right.display(km, world)})'


class Box:
    def __init__(self, inner):
        self.inner = inner

    def evaluate(self, km: KripkeModel, world: str):
        values = []
        for w in km.relations[world]:
            values.append(self.inner.evaluate(km, w))
        return all(values)

    def display(self, km: KripkeModel, world: str):
        return f'□{self.inner.display(km, world)}'


class Diamond:
    def __init__(self, inner):
        self.inner = inner

    def evaluate(self, km: KripkeModel, world: str):
        values = []
        for w in km.relations[world]:
            values.append(self.inner.evaluate(km, w))
        return any(values)

    def display(self, km: KripkeModel, world: str):
        return f'◇{self.inner.display(km, world)}'
```

Кожен з операторів (окрім атомарного висловлювання) рахує значення своїх внутрішніх формул, а потім значення формули для відповідного оператора. Таким чином будується абстрактне бінарне дерево, кожний листок якого – атом. Приклад абстрактного дерева для формули Чорча-Россера ($\Diamond\Box p \to \Box\Diamond p$):



Абстрактне дерево для формулы Чорча-Россера

І будується в Python наступним чином (також, метод *display* відображає в консолі саму формулу):

```
formula = Implies(Diamond(Box(Atom('p'))), Box(Diamond(Atom('p'))))
print(formula.display())
/usr/local/bin/python3.9 /Users/illya.koshmak/PycharmProjects/diploma/kripke.py
(◇□p → □◇p)
```

5.2 Приклад перевірки істинності формул

На прикладі моделі Кріпке з другої частини (с.13) побудуємо її програмну реалізацію та перевіримо деякі формули (за допомогою методу *evaluate*):

```
values = {
    'relations': {
        'A': {'B', 'C'},
        'B': {'B', 'C'},
        'C': {'C'}
    },
    'definitions': {
        'A': {'p': True, 'q': False},
        'B': {'p': True, 'q': True},
        'C': {'p': True, 'q': False}
    }
}
km = KripkeModel(values)

formula_1 = Implies(Box(Atom('p')), Box(Box(Atom('p'))))
formula_1_value = formula_1.evaluate(km, 'A')
print(f'{formula_1.display()}\t|= {formula_1_value}')

formula_2 = Implies(Diamond(Atom('p')), Diamond(Box(Atom('p'))))
formula_2_value = formula_2.evaluate(km, 'A')
print(f'{formula_2.display()}\t|= {formula_2_value}')

formula_3 = Implies(Box(Atom('q')), Atom('q'))
formula_3_value = formula_3.evaluate(km, 'B')
print(f'{formula_3.display()}\t|= {formula_3_value}')
```

Результат:

```
/usr/local/bin/python3.9 /Users/illya.koshmak/PycharmProjects/diploma/kripke.py
(□p → □□p)  |= True
(◇p → ◇□p) |= True
(□q → q)    |= True
```

5.3 Перевірка на p-морфізм

В класі моделі Кріпке є окремі реалізації алгоритму перевірки p-морфізму шкал Кріпке, та p-морфізму моделей Кріпке (вони реалізовані у методах $p\_morphism\_of\_frames$ та $p\_morphism\_of\_models$ відповідно):

```
def p_morphism_of_frames(self, other, mapping: dict):
    """
    Check if given mapping is a p-morphism of Kripke frames for two given Kripke models.
    :param other:                  KripkeModel: other Kripke model
    :param mapping:                dict       : mapping from given Kripke model to other Kripke model
    :return:                       bool       : True if it is a p-morphism of Kripke frames, False otherwise
    """
    # Check if given mapping is monotone
    for x in self.worlds.keys():
        for x_ in self.relations[x]:
            val = mapping[x_] in other.relations[mapping[x]]
            if not val:
                print(f"It's not a p-morphism of Kripke frames because f is not monotone: f({x_}) not in R(f({x}))")
                return False

    reversed_mapping = defaultdict(set)
    for key, value in mapping.items():
        reversed_mapping[value].add(key)

    # Check if given mapping has uplifting property
    for x in self.worlds.keys():
        for y in other.relations[mapping[x]]:
            val = reversed_mapping[y].intersection(self.relations[x]) != set()
            if not val:
                print(f"It's not a p-morphism of Kripke frames because f does not have a uplifting property: "
                      f"f^-1({y}) does not intersect R({x})")
                return False

    return True
def p_morphism_of_models(self, other, mapping: dict):
    """
    Check if given mapping is a p-morphism of Kripke models for two given Kripke models.
    :param other:                  KripkeModel: other Kripke model
    :param mapping:                dict       : mapping from given Kripke model to other Kripke model
    :return:                       bool       : True if it is a p-morphism of Kripke models, False otherwise
    """
    # Check if it is a p-morphism of Kripke frames
    if self.p_morphism_of_frames(other, mapping):
        # Check if truth table is the same for other Kripke models
        for w in self.worlds.keys():
            for key, val in self.worlds[w].values_dict.items():
                truth = val == other.worlds[mapping[w]][key]
                if not truth:
                    print(f"It's not a p-morphism of Kripke models because θ1({w}) != θ2(f({w}))")
                    return False
        return True

    return False
```

Також, окрім перевірки на p-морфізм, в разі якщо для данних моделей данне відображення не є p-морфізмом, метод виводить причину, через яку воно не є p-морфізмом:

```
/usr/local/bin/python3.9 /Users/illya.koshmak/PycharmProjects/diploma/kripke.py
It's not a p-morphism of Kripke frames because f is not monotone: f(C) not in
 R(f(B))
False
```

5.4 Перевірка на сильну зв'язність та формулу Чорча-Россера

Наступним алгоритмом є перевірка заданої моделі Кріпке на сильну зв'язність за допомогою перевірки загальнозничимості формули $\Box(\Box p \to q) \vee \Box(\Box q \to p)$ для заданої моделі:

```python
def is_strongly_connected(self):
    """
    Check if the Kripke model is strongly connected using modal formula.
    :return:                        bool: True if strongly connected, False otherwise
    """
    for world_name, world in self.worlds.items():
        assert len(world.values_dict.keys()) > 1, \
            'There is not enough of variables to check for strong connectivity!'
        for p in world.values_dict.keys():
            for q in world.values_dict.keys():
                if p == q:
                    continue
                atom_p, atom_q = Atom(p), Atom(q)
                # (□(□p → q) ∨ □(□q → p))
                formula = Or(Box(Implies(Box(atom_p), atom_q)), Box(Implies(Box(atom_q), atom_p)))
                print(world_name, formula.display(), formula.evaluate(self, world_name))
                if not formula.evaluate(self, world_name):
                    return False
    return True
```

А останнім алгоритмом – перевірка занальнозначимості формули Чорча-Россера $(\Diamond\Box p \to \Box\Diamond p)$ для заданої моделі Кріпке:

```python
def is_church_rosser(self):
    """
    Check if Church-Rosser formula is True the Kripke model using modal formula.
    :return:                        bool: True if strongly connected, False otherwise
    """
    for world_name, world in self.worlds.items():
        for var in world.values_dict.keys():
            atom = Atom(var)
            # (◇□p → □◇p)
            formula = Implies(Diamond(Box(atom)), Box(Diamond(atom)))
            if not formula.evaluate(self, world_name):
                return False
    return True
```

6 Висновки

В роботі було розглянуто базові поняття класичної математичної логіки: висловлювання, інтерпретація, логічні оператори, синтаксис і семантика, формули, тавтології, моделі; наведено приклади висловлювань, використання логічних операторів, формул. Також було розглянуто модальні оператори $\Box$ і $\Diamond$, синтаксис модальної логіки, моделі Кріпке, загальнозначимість в шкалах, логіку шкали; було показано, як в модальній логіці виражаються відношення рефлексивності, транзитивності, симетричності та ін.; наведено лему про невиразність ірефлексивності в модальній логіці.

В практичній частині описано процес створення програмної реалізації алгоритму інтерпретації формул модальної логіки для заданих моделей Кріпке на мові Python[6], що і було основною метою даної роботи. Також в цій частині описаний алгоритм перевірки, чи є відображення p-морфізмом для заданих шкал або моделей Кріпке; а також перевірка моеделей на сильну зв'язність та загальнозначимість формули Чорча-Россера за допомогою перевірки загальнозначимості відповідних модальних формул.

Література

[1] A. Chagrov, M. Zakharyachev, *Modal logic* (1997)

[2] E. Mendelson, *Introduction to Mathematical logic* (1997)

[3] J. van Benthen, *Modal logic for open minds* (2010)

[4] Lou van den Dries, *Mathematical Logic (Math 570) Lecture Notes* (2019)

[5] Rescher, N. «Many-Valued Logic», Mc.Graw-Hill, New York (1969).

[6] https://www.python.org