Змістовий модуль 1. ПРОПОЗИЦІЙНА ЛОГІКА

Тема 1. Основні означення пропозиційної логіки

План лекції.

- Вступ
- Основні означення пропозиційної логіки
- Пріоритет логічних операцій
- Структура формули
- Формальна граматика для формул пропозиційної логіки
- Інтерпретація формул пропозиційної логіки. Еквівалентність формул
- Загальнозначущість і суперечність у пропозиційній логіці
- Приклади застосування пропозиційної логіки
- Семантичні таблиці.

ВСТУП

Що ж таке логіка?

Філософська логіка – це наука про закони й форми відображення в мисленні розвитку об'єктивного світу, про закономірності пізнання істини. Головною задачею філософської логіки є формулювання законів і принципів, дотримання яких необхідно для отримання істинних умовиводів.

Формальна логіка – це наука, яка вивчає форми мислення (поняття, міркування, умовиводи, доведення) з боку їхньої логічної структури, тобто відволікаючись від конкретного змісту думок і відокремлюючи лише загальний спосіб зв'язку частин цього змісту. Підвалини формальної логіки заклав Аристотель.

Математична логіка – це область знання, у якій формальна логіка вивчається математичними методами. Головними задачами математичної логіки є: 1) побудова формально-логічних (аксіоматичних) числень; 2) вивчення зв'язку логічних числень з тими змістовними областями знань, які слугують їх інтерпретаціями і моделями.

Уже в давнину робилися спроби строгого подання математичних фактів, які намагалися вивести з небагатьох вихідних тверджень – аксіом. Яскравим прикладом цього є геометрія Евкліда. Вивчалися й закони правильного виводу наслідків із вихідних посилань (логіка Аристотеля).

Перші спроби створити строгі математичні теорії стосуються робіт Буля, Фреге, Пеано, інших математиків. Дослідники мали на меті закласти такі основи математики, виокремити такі аксіоми й правила виведення, за допомогою яких можна було б формально довести будь-яке змістовно істинне математичне твердження. Подібні твердження можна було б доводити автоматично. Ця *програма* Гільберта не вдалася у повній мірі: австрійський математик Курт Ґедель довів, що всяка достатньо багата формальна система (наприклад, формальна арифметика) не повна. Це означає, що в такій системі знайдуться змістовно істинні твердження, які є недоказовими в цій системі.

Найважливіше застосування логіки в комп'ютерних науках полягає у виконуванні таких завдань, які були б названі інтелектуальними, якби їх виконували люди. Прикладами таких задач *є програмування, створення систем, які відповідають на питання, і доведення теорем.*

Друга половина 60-х років минулого століття в царині штучного інтелекту вирізнялась особливим посиленням уваги до машинного доведення теорем. Основи машинного доведення теорем були закладені Ербраном (Herbrand J.) ще у 1930 р. Його метод був нереалізовуваний практично до винаходу електронних обчислювальних машин (сучасна назва – комп'ютери). Тільки після основної статті Дж. А. Робінсона (Robinson J.A.) [28], опублікованій 1965 року, у якій запропоновано *метод резолюцій*, був зроблений важливий

поступ до створення програм, які реалізують доведення теорем. Після 1965 року було запропоновано багато удосконалень методу резолюцій.

Разом з прогресом у техніці машинного доведення теорем відбувався й прогрес у застосуванні цієї техніки в різних задачах штучного інтелекту. Першим таким застосуванням було створення дедуктивних систем, які відповідають на питання; після цього з'явилися застосування до задач синтезу та аналізу програм.

Логіка особливо важлива для підготовки спеціалістів з комп'ютерних наук, оскільки це математична основа програмного забезпечення: вона використовується для формалізації семантики мов програмування та специфікації програм, а також перевірки коректності програм.

У пропонованій книзі розглянуто застосування математичної логіки до розв'язування інтелектуально складних проблем. Мається на увазі, що апарат математичної логіки використано для подання задач і пошуку їх розв'язків. Окрім того, у книзі розглянуто елементи некласичної логіки та показано, як некласична логіка може бути застосована у комп'ютерних науках, зокрема, для задач синтезу і аналізу програм, застосування нечіткого виведення тощо.

Нижче наведено два простих приклади з метою показати, як математичну логіку можна використати для подання задач.

Приклад В1. Припустімо, що є такі твердження.

A_1: Якщо жарко й волого, то буде дощ.

A_2: Якщо волого, то жарко.

A_3: Зараз волого.

Питання B: Чи буде дощ?

Наведені твердження записано українською мовою. Для їхнього подання мовою логічних формул ми використаємо символи p, q та r:

p : жарко;

q : волого:

r : буде дощ.

Нам ще потрібно декілька логічних зв'язок. У цьому випадку ми використаємо $\wedge$ для подання «і», та $\rightarrow$ для «якщо…, то». Тоді три наведені раніше твердження подаватимуться так:

A_1: $(p \wedge q) \rightarrow r$;

A_2: $q \rightarrow p$;

A_3: q.

Отже, ми переклали українські твердження мовою логічних формул.

Пізніше ми переконаємося, що коли формули A_1, A_2 і A_3 істинні, то й формула B, яка подає твердження, r – істинна. Тому ми говоримо, що *B логічно випливає* з A_1, A_2 і A_3, тобто дощ буде.

Приклад В2. Маємо такі твердження.

A_1: Конфуцій – людина.

A_2: Кожна людина смертна.

Щоб подати A_1 і A_2 необхідним є нове поняття – предикат. Ми вважатимемо, що $P(x)$ і $Q(x)$ подають «x є людиною» та «x смертний» відповідно. Ми також використаємо $\forall x$ щоб подати «для всіх x». Тоді щойно наведені твердження можна подати у вигляді таких формул

A_1: $P(\text{Конфуцій})$,

A_2: $\forall x(P(x) \rightarrow Q(x))$.

Пізніше побачимо, що з формул A_1 і A_2 можна логічно вивести формулу B, яка подає твердження $Q(\text{Конфуцій})$, яке свідчить, що Конфуцій смертний.

У цих прикладах ми, за суттю, маємо довести, що якась формула логічно випливає з інших формул. *Твердження, що формула логічно випливає з інших формул, називають* ***теоремою****. Міркування, які доводять, що теорема правильна, тобто що формула логічно випливає з інших формул, називають* ***доведенням*** *цієї теореми.* Проблема машинного доведення теорем полягає у вивченні машинних методів для знаходження доведень теорем.

Є багато задач, які зручно перетворити в задачі доведення теорем.

1. У ***системах, які відповідають на питання***, твердження можна подати логічними формулами. Тоді, щоб відповісти на питання, використовуючи дані факти, доводять, що формула, яка подає відповідь, виводиться з формул, які подають ці факти.

2. У ***задачі аналізу програм*** можна описати виконання програми формулою A, а умову, що програма завершить роботу, іншою формулою B. Тоді перевірка того, що програма закінчить роботу, є еквівалентною доведенню того, що формула B випливає з формули A.

3. У ***проблемі перетворення станів*** є множина станів і множина операторів над станами. Коли один з операторів застосовують до стану, одержують новий стан. Почавши з початкового стану, намагаються знайти послідовність операторів, які переводять початковий стан у якійсь бажаний. У цьому випадку можна описати стани й правила переходу логічними формулами. Отже, переведення початкового стану в бажаний можна розглядати як перевірку того, що формула, яка подає бажаний стан, виводиться з формули, яка подає як стани, так і правила переходу.

Оскільки багато задач можна сформулювати як задачі доведення теорем, проблема автоматизації доведень виявляється важливою в штучному інтелекті.

1.1. Основні означення пропозиційної логіки

Висловленням називають оповідальне речення, про яке можна сказати, що воно або істинне, або фальшиве, але не одне й інше водночас. Розділ логіки, який вивчає висловлення та їхні властивості, називають *пропозиційною логікою* або *логікою висловлень*.

Приклад 1.1. Наведемо приклади речень.

1. Сніг білий.
2. Київ – столиця України.
3. x+1=3.
4. Котра година?
5. Читай уважно!

Два перших речення – висловлення, решта три – ні. Третє речення набуває істинного або фальшивого значення залежно від значення змінної x, четверте та п’яте речення – не оповідальні.

Значення «істина» або «фальш», яких набуває висловлення, називають його *значенням істинності*. Значення «істина» позначають буквою T (від англ. «true»), а значення «фальш» – буквою F (від «false»). Зазначимо, що ці позначення слід розглядати як довільні символи, які легко можна замінити будь-якою іншою парою символів, як от 1 і 0, або навіть ♣ і ♠.

Пропозиційна логіка містить *зліченну* множину *пропозиційних* змінних p, q, r, s, … і п’ять *логічних операцій* (використовують також термін *пропозиційні зв’язки*): *заперечення* $\neg$ (читають «не»), кон’юнкцію $\wedge$ (читають «і»), диз’юнкцію $\vee$ (читають «або»), імплікацію $\rightarrow$ (читають «якщо…, то») та еквіваленцію $\leftrightarrow$ (читають «тоді й тільки тоді»). Пропозиційні змінні називають *атомарними формулами* чи *атомами*.

Приклад 1.2. Наведемо приклади висловлень.

1. p: «Сніг білий».
2. q: «Київ – столиця України».

Тут символи p, q – атомарні формули.

Багато речень утворюють об'єднанням одного або декількох висловлень. Отримане висловлення називають *складеним висловленням*. Побудову складених висловлень уперше розглянуто 1854 р. у книзі англійського математика Джорджа Буля (George Boole) «The Laws of Truth». Складене висловлення утворюють з існуючих висловлень застосуванням логічних операцій.

Приклад 1.3. Наведемо приклади складених висловлень.

1. Сніг білий, і небо теж біле.
2. Якщо погода хороша, то ми їдемо відпочивати.

У наведених прикладах логічні операції – це «і» та «якщо…, то».

Приклад 1.4. Розглянемо такі висловлення: p: «Вологість велика», q: «Температура висока», r: «Ми почуваємо себе добре». Тоді речення «Якщо вологість велика та температура висока, то ми не почуваємо себе добре» можна записати складеним висловленням $((p \wedge q) \to (\neg r))$.

Висловлення подають формулою. Вивчаючи формули, розглядають два аспекти – **синтаксис** і **семантику**.

Синтаксис *– це сукупність правил, які дають змогу будувати формули та розпізнавати правильні формули серед послідовностей символів.*

Сформулюємо означення (пропозиційної) формули.

1. Всяка пропозиційна змінна є (атомарною) формулою.
2. Якщо A і B – формули, то $(\neg A)$, $(A \wedge B)$, $(A \vee B)$, $(A \to B)$, $(A \leftrightarrow B)$ – формули.
3. Ніяких інших формул, окрім породжених скінченною кількістю застосувань правил 1 або 2, немає.

Для позначення довільних формул використовують букви A, B, C, … Букви ці – не формули, а *схеми формул*, замість яких можуть бути записані конкретні формули. Такі букви ми також називатимемо «формулами», але матимемо на увазі, що це всього лише зручний через компактність спосіб мови.

Приклад 1.5. Вирази $(p \to)$, $(p \wedge)$, $(p \neg)$, $(\vee q)$ – не формули.

Часто заперечення p позначають також як $\overline{p}$. Такий спосіб запису заперечення не потребує дужок для частини формули під знаком заперечення.

Семантика *– це сукупність правил, за якими формулам надають значення істинності.* Семантику логічних операцій зручно задавати з допомогою таблиць, які містять значення істинності формул залежно від значень істинності їх атомів. Такі таблиці називають *таблицями істинності*. Семантику введених операцій у формі таблиць істинності наведено в таблиці 1.1.

Табл. 1.1

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \to q$	$p \leftrightarrow q$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

Формулу $p \to q$ називають *імплікацією*; p називають *припущенням* (також *засновком, антецедентом*) *імплікації*, а q – *висновком* (*консеквентом*) *імплікації*. Оскільки імплікацію використовують у багатьох математичних міркуваннях, то існує багато термінологічних варіантів для формули $p \to q$. Ось деякі з них: «якщо p, то q», «з p випливає q», «p тільки тоді, коли q», «p тільки, якщо q», «q, якщо p», «p достатнє для q», «q необхідне для p».

Формулу $p \leftrightarrow q$ читають «p тоді й тільки тоді, коли q» чи «p еквівалентне q» та називають *еквіваленцією* формул p і q.

Логічна операція «диз'юнкція» відповідає одному з двох способів уживання слова «або (чи)» в українській мові. Диз'юнкція істинна, якщо істинне принаймні одне з двох висловлень. Розглянемо речення «Лекції з логіки можуть відвідувати студенти, які прослухали курси математичного аналізу або дискретної математики». Його зміст полягає в тому, що лекції можуть відвідувати як студенти, які прослухали обидва курси, так і ті, хто прослухав тільки один із них. Але є й інше, *альтернативне* «*або*». Розглянемо речення «Лекції з логіки мають відвідувати студенти, які прослухали тільки один із двох курсів – математичного аналізу або дискретної математики». Зміст цього речення полягає в тому, що студенти, які прослухали обидва ці курси, уже не повинні слухати лекції з логіки. Тут використано альтернативне «або», його позначають знаком $\oplus$. Значення істинності цієї операції наведено в таблиці 1.2. У разі використання цієї логічної операції пункт 2 означення пропозиційної формули потрібно розглядати у такій редакції:

2. Якщо A і B – формули, то $(\neg A)$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \leftrightarrow B)$, $(A \oplus B)$ – формули.

Табл. 1.2

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

Конструкція «якщо p, то S» у записі «if p then S», яку використовують в алгоритмічних мовах, відрізняється за змістом від імплікації в логіці. Тут p – висловлення, а S – програмний сегмент, який складається з одного чи декількох операторів. Програмний сегмент S виконується, якщо висловлення p істинне, і не виконується, якщо воно фальшиве.

1.2. Пріоритет логічних операцій

Домовимося зовнішні дужки у формулах не писати. Для зменшення кількості дужок вкажемо, що операція $\neg$ застосовується раніше всіх інших логічних операцій (має найвищий пріоритет). Це означає, що $\neg p \wedge q$ є кон'юнкцією $\neg p$ і q, а саме, $(\neg p) \wedge q$, а не запереченням кон'юнкції p і q, а саме, не є $\neg(p \wedge q)$.

Інше загальне правило пріоритету полягає в тому, що кон'юнкція має перевагу над диз'юнкцією, так що $p \wedge q \vee r$ означає $(p \wedge q) \vee r$, а не $p \wedge (q \vee r)$. Оскільки це правило може бути важко запам'ятати, ми будемо продовжувати використовувати дужки так, що порядок операцій диз'юнкції та кон'юнкції буде завжди зрозумілий.

Нарешті, прийнятим правилом є те, що операції імплікації $\rightarrow$ та еквіваленції $\leftrightarrow$ мають менший пріоритет, ніж операції кон'юнкції $\wedge$ і диз'юнкції $\vee$. Отже, $p \vee q \rightarrow r$ теж саме, що і $(p \vee q) \rightarrow r$. Ми будемо використовувати дужки також коли йдеться про порядок операцій $\rightarrow$ та $\leftrightarrow$, хоча $\rightarrow$ має пріоритет над $\leftrightarrow$.

Підсумовуючи сказане, вкажемо в порядку спадання пріоритет логічних операцій (якщо немає дужок): $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$.

Приклад 1.6. Формулу $((p \wedge q) \rightarrow (\neg r) \vee (\neg s))$ можна записати у виді $(p \wedge q) \rightarrow (\neg r \vee \neg s)$ і навіть у виді $p \wedge q \rightarrow \neg r \vee \neg s$.

Для визначення порядку виконання операцій у формулі пріоритету операцій не достатньо. Потрібно ще вказувати для однакових операцій, групуються вони зліва направо чи справа наліво. Наприклад, операції $\vee$ та $\wedge$ групуються зліва направо, а операція $\rightarrow$ – справа наліво. Тому для формули $p \wedge q \wedge r$ дужки розставляємо так: $((p \wedge q) \wedge r)$, а для формули $p \rightarrow q \rightarrow r$ розстановка дужок така: $(p \rightarrow (q \rightarrow r))$. Зазначимо, що для операцій $\vee$ та $\wedge$ порядок групування дужок не є суттєвим, але для операції $\rightarrow$ він є важливим. Справді, для формули

$p \to q \to r$ при групуванні дужок зліва направо отримаємо формулу $((p \to q) \to r)$, яка не еквівалентна попередній формулі $(p \to (q \to r))$. Проте у подальшому ми багатомісні імплікації завжди будемо записувати з дужками.

Для операції еквіваленції групування не використовують.

1.3. Структура формули

Розстановка дужок у формулі фіксує не лише порядок виконання логічних операцій, а й задає її структуру. Важливим тут є поняття головної операції у формулі та аргументів цієї головної операції. *Головною операцією* формули називають ту операцію, яка виконується **останньою**.

Пояснимо це на прикладі.

Приклад 1.7. Проаналізуємо структуру формули $(\neg p \wedge (q \leftrightarrow \neg p)) \vee \neg q$. Головною тут є диз'юнкція (для наочності головну операцію позначатимемо *). Маємо такий запис $(\neg p \wedge (q \leftrightarrow \neg p)) \vee^* \neg q$. Далі аналізуємо підформули. Підформули $(\neg p \wedge (q \leftrightarrow \neg p))$ та $\neg q$ задають перший і другий аргументи цієї операції.

Для першої підформули головною буде кон'юнкція, тобто $(\neg p \wedge^* (q \leftrightarrow \neg p))$; її аргументами є $\neg p$ і $(q \leftrightarrow \neg p)$. Далі у підформулі $(q \leftrightarrow \neg p)$ головною є операція еквіваленції: $(q \leftrightarrow^* \neg p)$; її аргументи – q і $\neg p$. Для підформул з унарною операцією заперечення $\neg q$ та $\neg p$ аргументами, очевидно, є атомарні формули q та p відповідно.

Структуру формули часто подають *деревом синтаксичного аналізу формули*. Для проаналізованої формули дерево синтаксичного аналізу подано на рис. 1.1.

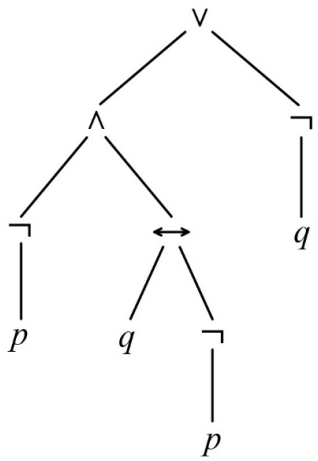


Рис. 1.1

Внутрішні вершини цього дерева позначено знаками логічних операцій, а листки – атомарними формулами, причому корінь дерева позначено головною операцією формули. Зазначимо, що обхід цього дерева зверху вниз (preorder traversal) дає польський запис розглядуваної формули $\vee \wedge \neg p \leftrightarrow q \neg p \neg q$, а обхід знизу вверх (postorder traversal) – зворотний польський запис $p \neg q p \neg \leftrightarrow \wedge q \neg \vee$.

1.4. Формальна граматика для формул пропозиційної логіки

Формули пропозиційної логіки можна визначити також як рядки, які генеруються контекстно вільною граматикою. Нижче подано продукції цієї граматики у формі Бекуса – Наура (BNF), разом із пріоритетами логічних операцій, від найвищого до найнижчого.

$\langle sentense \rangle ::= \langle atomic\_sentence \rangle \mid \langle complex\_sentence \rangle$
$\langle atomic\_sentence \rangle ::= \mid p \mid q \mid r \mid \ldots$
$\langle complex\_sentence \rangle ::= (\langle sentense \rangle) \mid \neg \langle sentense \rangle \mid \langle sentense \rangle \wedge \langle sentense \rangle$
$\mid \langle sentense \rangle \vee \langle sentense \rangle \mid \langle sentense \rangle \to \langle sentense \rangle$
$\mid (\langle sentense \rangle \leftrightarrow \langle sentense \rangle)$

1.5. Інтерпретація формул пропозиційної логіки. Еквівалентність формул

Для знаходження значення істинності складеного висловлення потрібно надати значення істинності всім атомам, які містить відповідна пропозиційна формула.

Нехай A – пропозиційна формула, і p_1, p_2, …, p_n – її атоми. Тоді *інтерпретацією* формули A називають таке приписування значень істинності атомам p_1, p_2, …, p_n, при якому кожному p_i приписано або T, або F (але не обидва разом).

Говорять, що формула *A істинна при деякій інтерпретації*, тоді й тільки тоді, коли A одержує значення T при цій інтерпретації; у протилежному випадку говорять, що A *фальшива* (або *хибна*) *при цій інтерпретації*.

Для обчислення значень істинності формули, яка зображає складене висловлення, потрібно знаходити значення логічних операцій, визначених таблицею 1.1. Послідовність обчислень задають парами дужок (чи враховують пріоритет операцій). Якщо формула має n атомів, то є 2^n способів надати значення істинності її атомам, тобто така формула має 2^n інтерпретацій, а всі її значення можна звести в таблицю істинності з 2^n рядками. Формулу, яка містить n атомів, називають *n-місною*.

Іноді, якщо p_1, p_2, …, p_n – всі атоми, що зустрічаються в деякій формулі, буває зручно представити інтерпретацію множиною $\{m_1, m_2, \ldots, m_n\}$, де m_i – це або p_i, або $\neg p_i$. Наприклад, множина $\{p, \neg q, \neg r, s\}$ представляє інтерпретацію, у якій атомам p, q, r і s відповідно приписані значення T, F, F і T. Іншими словами, якщо в множину, яка представляє інтерпретацію таким способом, входить сам атом p, то йому приписують значення T; якщо ж у цю множину входить $\neg p$, то p приписують значення F.

Формулу A називають *виконуваною*, якщо існує принаймні одна інтерпретація $\boldsymbol{I}$, у якій A набуває значення T. У такому разі говорять, що $\boldsymbol{I}$ *задовольняє* A, або що формула A *виконується* в інтерпретації $\boldsymbol{I}$. З іншого боку, якщо формула A набуває значення F при інтерпретації $\boldsymbol{I}$, то говорять, що $\boldsymbol{I}$ *спростовує* A, або A *спростовується* інтерпретацією $\boldsymbol{I}$. Коли інтерпретація $\boldsymbol{I}$ задовольняє формулу A, то $\boldsymbol{I}$ називають також *моделлю* A.

Приклад 1.8 Розглянемо формулу $A = (p \wedge q) \rightarrow (p \leftrightarrow \neg r)$. Значення істинності при всіх восьми інтерпретаціях формули A наведено в таблиці 1.3. Отже, ця формула спростовується при інтерпретації $\{p, q, r\}$, а при всіх інших інтерпретаціях виконується.

Табл. 1.3

p	q	r	$\neg r$	$p \wedge q$	$p \leftrightarrow \neg r$	$(p \wedge q) \rightarrow (p \leftrightarrow \neg r)$
T	T	T	F	T	F	F
T	T	F	T	T	T	T
T	F	T	F	F	F	T
T	F	F	T	F	T	T
F	T	T	F	F	T	T
F	T	F	T	F	F	T
F	F	T	F	F	T	T
F	F	F	T	F	F	T

Формули A і B називають *еквівалентними*, або *рівносильними, тотожними* (позначають $A \equiv B$, або $A \Leftrightarrow B$), тоді й тільки тоді, коли значення істинності A і B збігаються при кожній інтерпретації A і B.

Приклад 1.9. За допомогою таблиці істинності доведемо, що $p \rightarrow q \equiv \neg p \vee q$. Результат розв'язування задачі наведено в таблиці 1.4.

Табл. 1.4

p	q	$p \rightarrow q$	$\neg p$	$\neg p \vee q$	$(p \rightarrow q) \leftrightarrow (\neg p \vee q)$
T	T	T	F	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

Приклад 1.10. За допомогою таблиці істинності доведемо, що $p \to q$ не еквівалентно $q \to p$. Результат розв'язування задачі наведено в таблиці 1.5.

Табл. 1.5

p	q	$p \to q$	$q \to p$	$(p \to q) \leftrightarrow (q \to p)$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

1.6. Загальнозначущість і суперечність у пропозиційній логіці

Формулу A пропозиційної логіки називають *загальнозначущою формулою* чи *тавтологією*, якщо вона виконується при всіх інтерпретаціях. Якщо формула A – загальнозначуща, то використовують позначення $\models A$. Формулу B, фальшиву при всіх інтерпретаціях, називають *суперечністю* чи *невиконуваною формулою*. Для суперечності B використовують позначення $\not\models B$.

Властивість еквівалентності формул A і B можна сформулювати у вигляді такого твердження.

Теорема 1.1. Формули A і B логічно еквівалентні тоді й лише тоді, коли формула $A \leftrightarrow B$ загальнозначуща, тобто $A \equiv B$ тоді й тільки тоді, коли $\models (A \leftrightarrow B)$.

Оскільки кожна формула пропозиційної логіки має скінченну кількість інтерпретацій, то завжди можна перевірити її загальнозначущість чи суперечність, знайшовши значення істинності при всіх можливих інтерпретаціях.

Приклад 1.11. Розглянемо формулу $((p \to q) \wedge p) \to q$. Атомами в цій формулі є p та q, а формула має $2^2=4$ інтерпретації. Значення істинності наведено в таблиці 1.6. Ця формула істинна при всіх інтерпретаціях, тобто є загальнозначущою.

Табл. 1.6

p	q	$(p \to q)$	$(p \to q) \wedge p$	$((p \to q) \wedge p) \to q$
T	T	T	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

Приклад 1.12. Розглянемо формулу $(p \to q) \wedge (p \wedge \neg q)$**.** З таблиці 1.7 доходимо висновку, що вона фальшива при всіх інтерпретаціях, тобто є суперечністю.

Табл. 1.7

p	q	$p \to q$	$\neg q$	$p \wedge \neg q$	$(p \to q) \wedge (p \wedge \neg q)$
T	T	T	F	F	F
T	F	F	T	T	F
F	T	T	F	F	F
F	F	T	T	F	F

Далі ми побачимо, що доведення факту загальнозначущості чи суперечності формули – дуже важлива задача.

У пропозиційній логіці через скінченність числа інтерпретацій ми способом повного перебору всіх можливих інтерпретацій завжди можемо вияснити, чи є формула загальнозначущою, чи суперечністю.

1.7. Приклади застосування пропозиційної логіки

Логіка має багато важливих застосувань у математиці, інформатиці та багатьох інших дисциплінах. Твердження в математиці, науках і на природній мові часто є неточними або

неоднозначними. Щоб зробити такі твердження точними, їх можна перекласти мовою логіки. Наприклад, логіка використовується в специфікації програмного та апаратного забезпечення, оскільки ці специфікації мають бути точними до початку розробки. Крім того, пропозиційна логіка та її правила можна використовувати для проєктування комп'ютерних схем, створення комп'ютерних програм, для перевірки коректності програм, а також для побудови експертних систем. Логіку можна використовувати для аналізу і розв'язання багатьох знайомих головоломок. Розроблено програмні системи, засновані на правилах логіки, для автоматичної побудови доказів. Ми обговоримо деякі з цих застосувань логіки в цьому розділі та в наступних розділах.

1.7.1. Переклад речень з природної мови

Є багато причин для перекладу речень української (чи іншої) природної мови у вирази, що містять пропозиційні змінні та логічні сполучники. Зокрема, речення українською (і будь-якою іншою природною мовою) часто неоднозначні. Переклад речень у складені висловлення усуває двозначність. Зауважимо, що це може передбачати створення ряду обґрунтованих припущень на основі передбачуваного значення речення. Крім того, як тільки ми переклали речення з української в логічні вирази, ми можемо аналізувати ці логічні вирази, щоб визначити їх значення істинності, ми можемо маніпулювати ними, і ми можемо використовувати правила логічного висновку (які обговорюються в підрозділі 1.15) для міркування про них.

Щоб проілюструвати процес перекладу українського речення в логічний вираз, розглянемо два приклади.

Приклад 1.13. Як це українське речення можна перекласти в логічний вираз?

«Ви можете отримати доступ до Інтернету з кампусу, лише якщо навчаєтесь за спеціальністю інформатика або не є першокурсником.»

Є багато способів перевести це речення в логічний вираз. Хоча можливо представити це речення однією пропозиційною змінною, такою як p, це не корисно для аналізу його значення або міркування з ним. Замість цього ми використаємо пропозиційні змінні для представлення кожної частини речення та визначимо відповідні логічні операції над ними. Зокрема, ми вибрали a, c і f для позначення «Ви можете отримати доступ до Інтернету з кампусу», «Ви навчаєтесь за спеціальністю інформатика» та «Ви першокурсник», відповідно. Зазначаючи, що «тільки якщо» є одним із способів висловлення умовного твердження, це речення можна подати як $a \rightarrow (c \vee \neg f)$.

Приклад 1.14. Як це українське речення можна перекласти в логічний вираз?

«Ви не можете кататися на американських гірках, якщо ваш зріст менше 122 см і вам не більше 16 років.»

Нехай q, r і s означають «Ви можете кататися на американських гірках», «Ваш зріст менше 122 см», та «Вам більше 16 років» відповідно. Тоді речення можна перекласти як $(r \wedge \neg s) \rightarrow \neg q$.

Існують інші способи представити оригінальне речення як логічний вираз, але спосіб, який ми використовуємо, має відповідати нашим потребам.

1.7.2. Технічні характеристики системи

Переклад речень природною мовою (наприклад, українською) у логічні вирази є важливою частиною специфікації апаратних і програмних систем. Системні та програмні розробники беруть вимоги, описані природною мовою, та переводять їх у точні й однозначні специфікації, які можуть використовуватись як основа для розробки системи. Приклад 3 показує, як складені висловлення можуть використовуватись у цьому процесі.

Приклад 1.15. Виразіть специфікацію «Автоматична відповідь не може бути надіслана, коли файлова система заповнена» за допомогою логічних зв'язок.

Позначимо p: «Автоматична відповідь може бути надіслана», а q: «Файлова система заповнена». Тоді $\neg p$ означає «Це не той випадок, коли автоматична відповідь може бути

надіслано», що також можна виразити як «Автоматична відповідь не може бути надіслана». Отже, наша специфікація може бути представлена умовним реченням $q \to \neg p$.

Системні специфікації мають бути *узгодженими*, тобто вони не повинні містити суперечливих вимог, які можна використати для виведення суперечності. Якщо специфікації не узгоджуються, то не немає способу розробити систему, яка б задовольняла всі специфікації.

Приклад 1.16.Визначте, чи відповідають ці системні характеристики:

«Діагностичне повідомлення зберігається в буфері або передається повторно».

«Діагностичне повідомлення не зберігається в буфері».

«Якщо діагностичне повідомлення зберігається в буфері, воно передається повторно».

Щоб визначити, чи узгоджуються ці специфікації, ми спочатку виразимо їх за допомогою логічних виразів. Нехай p позначає «Діагностичне повідомлення зберігається в буфері», q позначає «Діагностичне повідомлення передається повторно». Тоді специфікації можна записати як $p \vee q$, $\neg p$ і $p \to q$. Присвоєння істиннісних значень, яке робить усі три специфікації істинними, таке: p має бути фальшивим, щоб зробити $\neg p$ істинним. Оскільки потрібно, щоб $p \vee q$ було істинним, але p має бути хибним, то q має бути істинним. Оскільки $p \to q$ є істинним, коли p є хибним, а q є істинним, ми робимо висновок, що ці специфікації є *узгодженими*, тому що всі вони істинні, коли p хибне, а q істинне. Ми можемо прийти до того ж висновку за допомогою таблиці істинності.

Приклад 1.17. Чи системні специфікації в прикладі 2 залишаються узгодженими, якщо додана специфікація «Діагностичне повідомлення не передається повторно»?

Відповідно до міркувань у прикладі 2, три специфікації з цього прикладу істинні лише у випадку, коли p хибне, а q істинне. Проте ця нова специфікація є $\neg q$, що є хибним, коли q є істинним. Отже, ці чотири специфікації *неузгоджені*.

У подальших лекціях ми продовжимо вивчення проблем перекладу речення з природної (української) мови в логічний вираз і навпаки, а також опису засобами логіки технічних характеристик системи.

1.8. Семантичні таблиці

Розглянемо таблицю істинності для пропозиційної формули $(p \to (q \vee r)) \to (p \to q)$ (таблиця 1.8).

Табл. 1.8

p	q	r	$q \vee r$	$p \to (q \vee r)$	$p \to q$	$(p \to (q \vee r)) \to (p \to q)$
T	T	T	T	T	T	T
T	T	F	T	T	T	T
T	F	T	T	T	F	F
T	F	F	F	F	F	T
F	T	T	T	T	T	T
F	T	F	T	T	T	T
F	F	T	T	T	T	T
F	F	F	F	T	T	T

Можна побачити, що в цій таблиці багато надлишкової інформації. Таблицю можна значно скоротити, якщо скористатись такими правилами: для істинності диз'юнкції достатньо істинності одного з членів; для фальшивості кон'юнкції достатня фальшивість одного з членів; істина випливає з будь-чого; із фальші випливає будь-що. Скорочену таблицю подано в табл. 1.9.

Табл. 1.9

p	q	r	$q \vee r$	$p \to (q \vee r)$	$p \to q$	$(p \to (q \vee r)) \to (p \to q)$
F	–	–	–	T	T	T
T	T	–	T	T	T	T
T	F	F	F	F	F	T
T	F	T	T	T	F	F

При такому скороченні таблиці важливо не пропустити жодного випадку. Голландський логік Бет (E.W. Beth) запропонував у 50-х роках XX століття формальні правила, які гарантують правильність побудови. Ці правила ґрунтуються на необхідних і достатніх умовах істинності та фальшивості формул. Пояснимо це на прикладі останньої формули.

Приклад 1.18. Шукаємо всі можливі інтерпретації, коли формула $(p \to (q \vee r)) \to (p \to q)$ фальшива. Якщо таких не буде знайдено, то формула є загальнозначущою. Головною операцією у формулі, яка розглядається, є імплікація. Головну операцію позначаємо *: $(p \to (q \vee r)) \to^* (p \to q)$. Імплікація фальшива, якщо засновок істинний, а висновок фальшивий. Одержуємо таке перетворення:

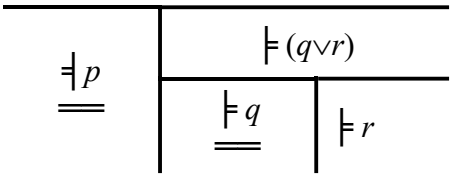
$$\nvDash (p \to (q \vee r)) \to (p \to q)$$
$$\vDash (p \to (q \vee r))$$
$$\nvDash (p \to^* q)$$

Після застосування аналогічного перетворення до третьої формули одержимо

$$\nvDash (p \to (q \vee r)) \to (p \to q)$$
$$\vDash (p \to^* (q \vee r))$$
$$\nvDash (p \to q)$$
$$\vDash p$$
$$\nvDash q$$

Тепер потрібно розглянути умови істинності імплікації (це головна операція у другій формулі). Два варіанти вичерпують всі можливості істинності для імплікації $p \to (q \vee r)$: необхідно й достатньо, щоб p було фальшивим, або $q \vee r$ істинним. Після декомпозиції імплікації аналогічно вчинимо з диз'юнкцією: тут теж два варіанти вичерпують всі можливості істинності для диз'юнкції.

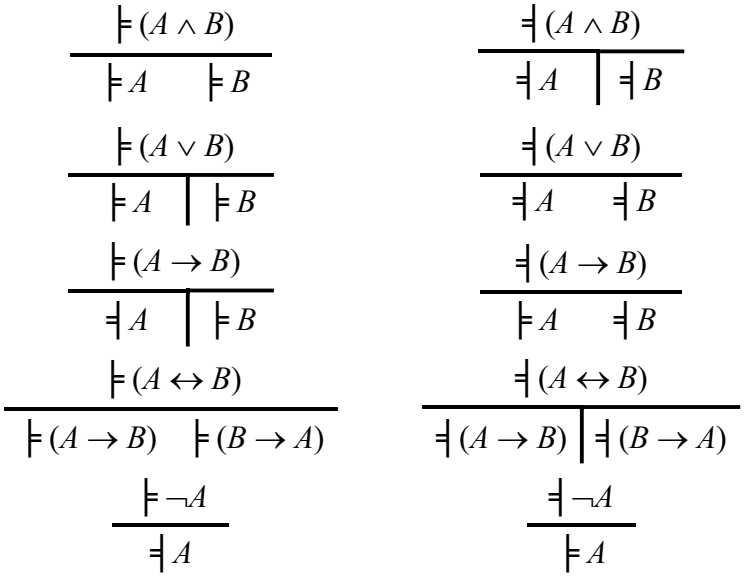
$$\nvDash (p \to (q \vee r)) \to (p \to q)$$
$$\vDash (p \to (q \vee r))$$
$$\nvDash (p \to q)$$
$$\vDash p$$
$$\nvDash q$$



Ми одержали три підтаблиці. У двох з них є суперечності ($\vDash p$, $\nvDash p$ у першій та $\nvDash q$, $\vDash q$ у другій); такі підтаблиці називають *закритими* і позначають подвійною рискою внизу. Підтаблиця, яка залишилася, дає інтерпретацію, при якій формула фальшива: (T, F, T).

Отже, запропоновано такий метод перевірки загальнозначущості формули чи побудови контрприкладу. Припускаємо, що формула є суперечністю. Будуємо семантичну таблицю. Якщо всі її підтаблиці закрилися, то припущення про фальшивість формули призвело до суперечності і, отже, формула є загальнозначущою. Інакше незакрита підтаблиця дає контрприклад.

Наведемо правила декомпозиції для зв'язок пропозиційної логіки.



Якщо в правилі нижня частина не розділена, то формули, які є результатами декомпозиції, залишаються в тій самій підтаблиці. Інакше вони розподіляються по двох нових підтаблицях, котрі далі будуються незалежно. Це означає, що *підтаблиці семантичної таблиці утворюють бінарне дерево.*

Будь-яка послідовність правильних кроків приводить до результату, але вдалий вибір порядку декомпозицій може скоротити таблицю.