ОСНОВНІ ПОНЯТТЯ ЛОГІКИ

Логіка – наука про закони мислення

Математична логіка – наука про закони *математичного* мислення

МЛ має на меті формалізацію та дослідження законів мислення математичними методами

МЛ вивчає математичні теорії в цілому, які досліджуються за допомогою логіко-математичних мов

- Стосовно *загальносвітоглядного* аспекту, поняття і методи МЛ дають обґрунтування правильності тих чи інших способів отримання істинного знання.
- Стосовно *прагматичного* аспекту, апарат МЛ належить до основних засобів моделювання різноманітних ПО, він є основою сучасних інформаційних систем.

Мета курсу – поглиблення знань з МЛ та засвоєння знань з композиційних логік. Сучасні методи розробки програмного забезпечення широко використовують логічні формалізми для доведення властивостей програм. Це робить актуальною проблему вивчення програмно-орієнтованих логічних формалізмів. Такими є ***композиційні логіки***, будовані на основі спільного для логіки й програмування композиційно-номінативного підходу

Логіка як теоретична наука зародилась в давній Греції в 6 ст. до н.е.

Поширення *софізмів* у той час вимагало точного формулювання принципів і правил логічних міркувань.

Це зумовило зародження логіки як науки про закони мислення

Софізм – нібито правильне міркування, які містить навмисну, але замасковану помилку.

- Софізм "накритий": *Чи знаєш ти цього накритого чоловіка, що спить під стіною? Не знаю. Але це ж твій батько! Отже, ти не знаєш свого батька.*
- Софізм "рогатий": *Те що ти не втратив, ти маєш. Ти не втратив роги. Отже, ти рогатий.*

Паралогізми – некоректні міркування із ненавмисним порушенням логічних законів.

Якщо через дріт пропускають електричний струм, то дріт нагрівається. Дріт нагрівається. Отже, через дріт пропускають струм.

Точну грань між софізмами і паралогізмами провести важко.

– *Усі студенти здають іспити. Микола здає іспити. Отже, Микола – студент.*

– *Ссавці населяють сушу й океани. Миші – ссавці. Отже, миші населяють сушу й океани*

Фалес Мілетський (625–547 р. до н.е.):

– діаметр ділить коло навпіл

– кути при основі рівнобедренного трикутника рівні

Принципово новим було те, що для цих тверджень пропонувалось *чисто логічне* доведення.

Піфагор (570–500 р. до н.е.): довів існування ірраціональних чисел.

Платон (427–347 р. до н.е.) розвинув загальні принципи логічних міркувань.

Арістотель (384–322 р. до н.е.) – основоположник логіки як цілісної науки:

- явно сформулював три основні закони традиційної логіки
- розробив закони логічного виведення
- запропонував аксіоматичний метод
- створив першу формально-аксіоматичну систему логіки – силогістику
- заклав основи модальної логіки.

Формальна логіка вивчає акти мислення (поняття, судження, умовиводи, доведення) тільки з огляду їх форми, логічної структури, абстрагуючись від конкретного змісту

Розвиток античної логіки після Арістотеля:

– застосування аксіоматичного методу (Евклід)

– мегаро-стоїки (Зенон, Хризіпп, Діодор, Евбулід) вивчали логіку висловлень, а не логіку імен, як Арістотель; змінні позначають висловлення, а не терміни; дослідили логічні зв'язки

Середні віки – поглиблення, деталізація, коментарі силогістики

Боецій (480-524) – комбінаторика суджень; виявив сотні умовних суджень; перші кроки до математизованої логіки

Реалісти: універсалії (загальні поняття) існують та передують існуванню одиничних понять. Ансельм Кентерберійський (1033-1109)

Томас Аквінський (1225-1274) – обґрунтування Християнської догматики на основі вчення Арістотеля

Номіналісти: універсалії – лише імена одиничних речей, не відбивають їх якостей. Росцелін (1050-1112), Абеляр (1079-1142)

Раймонд Луллій (1235-1315) – проблеми логічного слідування

Френсіс Бекон (1561-1626) – індуктивний метод

Рене Декарт (1596-1650) – дедуктивний метод

Принциповий крок в розвитку логіки – *Г. Лейбніц* (1646–1716): ідея створення універсального логічного числення

Іммануїл Кант (1724-1804) – трансцедентальна логіка

Ґеорґ Гегель (1770-1831) – діалектична логіка

Основні закони традиційної логіки

Класична МЛ опирається на основні закони традиційної логіки.

- Закон тотожності
- Закон несуперечливості
- Закон виключеного третього
- Закон достатньої підстави

Закон тотожності в загальному вигляді:

В процесі одного і того ж міркування використовувані поняття не повинні змінюватись

Це засвідчує фундаментальну роль закону тотожності для організації мислення людини

В спрощеному вигляді: *А суть А*

Математичне формулювання: $A \leftrightarrow A$.

Закон несуперечливості :

Обидва твердження А та ¬А не можуть виконуватися одночасно

Інколи називають законом суперечливості (*lex contradictionis*).

Математичне формулювання: $\neg(A \,\&\, \neg A)$.

Закон виключеного третього:

Обидва твердження A та $\neg A$ не можуть заперечуватися одночасно

Сильна форма (*tertium non datur*)

Одне з тверджень A чи $\neg A$ істинне

Математичне формулювання: $A \vee \neg A$.

Закон достатньої підстави:

Жодне твердження не може прийматися без достатніх підстав

Прийняття ЗДП відокремлює логіку точних наук від змістовної логіки.

ЗДП не має математичного формулювання.

Дж.Буль (1815–1864) – перша система *математичної логіки* (алгебра логіки), вона базується на строгій логіко-математичній мові.

***Математична логіка* – це формальна логіка, що використовує математичні методи.**

Розвиток МЛ в 19 ст.: *А. де Морган* (1806–1873), *Е.Шредер* (1845–1902), *П.Порецький* (1846–1907), *Ч.Пірс* (1839–1914).

Ґ.Фреґе (1848–1925): ввів поняття предикату і кванторів, розвинув логіко-математичні мови і теорію їх смислу

Дж.Пеано (1858–1932): виклад цілих розділів математики на мові МЛ та аксіоматизація арифметики

Ґ.Фреґе, *Б.Рассел* (1872–1970): спроба зведення всієї математики до логіки. Привела до створення багатого логічного апарату, оформлення МЛ як повноцінного розділу математики.

На межі 19–20 ст. відкриті ***парадокси*** в теорії множин.

Парадокс – це логічно правильне міркування, яке веде до взаємовиключних висновків

Парадокси *не порушують* законів формальної логіки!

Парадокс брехуна: *Я брешу*.
Парадокс критянина (парадокс Епіменіда) – форма парадоксу брехуна.
Згаданий в "*Посланні до Тита св. Апостола Павла*".

Сказав один з них, власний пророк їхній:
Крітяни завжди брехливі, звірі погані, черева ледачі

Парадокс Буралі-Форті (1897)
Парадокс Кантора (1899)

Парадокс Рассела (1906).

Множина *нормальна*, якщо вона не є елементом самої себе.

Ненормальні множини:
– множина усіх абстрактних понять є абстрактним поняттям
– множина усіх множин.

$\boldsymbol{S}$ – множина усіх нормальних множин: $\boldsymbol{S} = \{A \mid A \notin A\}$

$A \in \boldsymbol{S} \Leftrightarrow A \notin A$. Звідси $\boldsymbol{S} \in \boldsymbol{S} \Leftrightarrow \boldsymbol{S} \notin \boldsymbol{S}$!

Парадокс цирульника. В одному містечку цирульник мусить голити всіх тих і тільки тих чоловіків містечка, хто не голиться сам.
Чи може цирульник голити себе?

Парадокс Беррі.

Деякі речення є визначеннями натуральних чисел:

"парне просте число", "сто в двадцятій степені "

Існують натуральні числа (їх множина нескінченна), які не можуть бути визначені реченнями української мови, що мають менше ста букв.

Серед них є найменше, його можна визначити реченням

"*Найменше натуральне число, яке не можна визначити реченням української мови, що має менше ста букв*".

Але ж тут 98 букв!

Парадокс Греллінга.

Автологічні прикметники: мають ту ж властивість, яку називають ("багатоскладовий", "український").

Гетерологічні прикметники: не мають тієї властивості, яку називають ("колючий", "солоний", "зелений").

Прикметник "*гетерологічний*": автологічний чи гетерологічний?

Шляхи виходу із кризи математики:

Л.Брауер (1881–1966) висунув *інтуїціоністську програму:*
відмова від актуальної нескінченності та закону виключеного третього;
в математиці допустимі тільки конструктивні доведення.

Д.Гільберт (1862–1943) висунув програму обгрунтування математики на базі МЛ: побудова формально-аксіоматичних моделей основних розділів математики та подальше доведення їх несуперечливості надійними, інтуїтивно переконливими (фінітними) засобами

Несуперечливість: неможливість одночасного виведення деякого твердження та його заперечення

Математичні теорії стають предметом вивчення нової математичної науки – теорії доведень, або (Д.Гільберт) метаматематики.

Розробка Д.Гільбертом та його учнями теорії доведень на базі розвинутої Ґ. Фреґе та Б.Расселом логічної мови означала становлення МЛ як самостійної математичної дисципліни.

Теореми *К. Ґьоделя* (1906–1978) **про неповноту** засвідчують принципову обмеженість формально-аксіоматичного методу.

К. Ґьодель показав: Кожна достатньо багата несуперечлива формальна теорія неповна, тобто існують записані мовою теорії твердження, які не можна ні довести, ні спростувати. При цьому несуперечливість такої теорії не може бути доведена засобами самої теорії (приклад такої теорії: *формальна арифметика*)

Визначне досягнення МЛ – розробка і дослідження формальних моделей алгоритму та алгоритмічно обчислюваної функції.

Алгоритм – скінченна множина точно визначених правил для чисто механічного розв'язку задач певного класу.

Характерні властивості алгоритму: Фінітність, Масовість, Дискретність, Елементарність, Результативність, Детермінованість

Теорія алгоритмів – розділ математики, що вивчає загальні властивості алгоритмів.

Усвідомлення *неможливості* існування алгоритмів розв'язку низки масових проблем ⇒ необхідність *математичного* уточнення поняття алгоритму.

Після сформування поняття алгоритму як нової та окремої сутності першочерговою стала проблема знаходження адекватних *формальних моделей* алгоритму та їх дослідження:

– моделі для первісного поняття алгоритму (МТ, регістрові машини, НАМ)
– моделі для похідного поняття АОФ (λ-означувані функції, ЧРФ)

Кожна з відомих моделей задає (з точн. до кодування) один і той же клас функцій

Тому є всі підстави вважати, що кожна з таких моделей дає строге математичне уточнення інтуїтивного поняття АОФ.

Таке твердження стосовно АОФ та строго визначеного класу ЧРФ – *теза Чорча*

Ідея Г.Лейбніца створення універсального логічного числення втілена в *теорії доведень*. Це розділ МЛ, який вивчає поняття доведення в математиці та його застосування.

Алгоритмічна нерозв'язність проблеми істинності формул логіки предикатів 1-го порядку робить *в принципі неможливим існування універсальної процедури пошуку доведень*.

Проте, існують алгоритми, які дозволяють знайти доведення формули, якщо вона істинна. Якщо ж формула не істинна, такі алгоритми можуть роботу не завершувати.

Дуже важливе застосування МЛ – *автоматизація пошуку доведень* (задачі подання знань і роботи з ними в БД і БЗ, задачі логічного програмування і дедуктивних баз даних).

Методи пошуку доведень:

– секвенційні числення, запропоновані Ґ.Ґенценом (1909–1945)

– метод семантичних таблиць (Е.Бет)

– метод натурального виведення (Ґ.Ґенцен)

– метод резолюцій (Дж.А.Робінсон, 1969).

Предикати, висловлення

Основним поняттям логіки з семантичної точки зору є поняття ***предиката***.

З цим поняттям тісно пов'язане поняття ***висловлення***.

Висловлення – речення, яке можна розглядати з точки зору його істинності чи хибності.

Суб'єкт (суб'єкти) – те, про кого або про що говориться у висловленні.

Предикат виражає *властивості* суб'єкту (суб'єктів) та *відношення* між ними.

Приклад. Предикат "*x* – є простим числом" стає висловленням:

– "5 є простим числом", якщо значенням імені *x* є число 5.

– "4 є простим числом", якщо значенням *x* є число 4.

Дане – це множина пар імен суб'єктів та їх значень.

Висловлення є значенням предикату на конкретному даному.

Приклад. Застосуємо предикат "якщо *x*>*y* та *y*>*z*, то *x*>*z*"
до даного [*x*□17, *y*□11, *z*□3]
дістанемо висловлення "якщо 17>11 та 11>3, то 17>3".

Висловлення може приймати одне з двох істиннісних значень – *T* або *F*;
тому *предикат* уточнимо як *функцію*, що конкретним іменованим суб'єктам зіставляє значення *T* або *F*

Проблема програмно-орієнтованої перебудови класичної логіки

Методи МЛ доводять свою ефективність при розв'язуванні широкого кола задач інформатики та програмування. Створено багато різноманітних логічних систем, які з успіхом використовуються в програмуванні.

Такі системи зазвичай базуються на класичній логіці предикатів.

Класична логіка посідає особливе місце серед логічних формалізмів:

1. Закони логіки предикатів мають універсальний характер, виражають загальні закони мислення людини, вони істинні в усіх ПО.
2. Класична логіка всебічно досліджена, вона має широке застосування, для неї побудовано багато систем автоматизованого доведення.
3. Класична логіка є основою низки спеціальних логік (модальних, темпоральних, епістемічних, релевантних тощо).

Проте класична логіка, незважаючи на численні позитивні вартості, має низку принципових обмежень, які ускладнюють її використання. Вона недостатньо враховує структурованість, частковість інформації про ПО

Аналіз основних понять класичної логіки дозволяє зробити *висновки*:

1. При її побудові характерне превалювання синтаксичних аспектів, а для програмування набагато важливішими є семантичні. Синтактико-семантичний підхід до побудови класичної логіки виявляється в тому, що спочатку визначають мову логіки і, досить часто, відношення виведення (синтаксичні аспекти), а вже потім – інтерпретації та відношення виконуваності (семантичні аспекти).
2. У класичній логіці функції та предикати трактуються як однозначні скінченно-арні відображення, причому предикати – тотальні, а в програмуванні використовуються набагато потужніші класи часткових функцій і предикатів над іменними (номінативними) даними. При цьому базові функції та предикати класичної логіки – тотальні *n*-арні.
3. У класичній логіці пропозиційні зв'язки трактуються як тотальні *n*-арні булеві функції, квантори не мають самостійного семантичного визначення, їх зміст розкривається в процесі інтерпретації формул.

Зазначені обмеження зумовлені найперше тим, що на час формування класичної логіки програмування ще не існувало, тому логіка будувалась на базі наявного математичного апарату, який орієнтувався на використання однозначних скінченно-арних відображень.

Синтаксис завжди простіший, тому побудову починали з нього

Шляхи подолання невідповідності між номінативною природою формул та вибором як базових тотальних *n*-арних предикатів :

– перейти до класів формул, які ближчі до *n*-арних відображень;

– перейти до класів предикатів, які ближчі до номінативних формул.

Перший шлях веде до операторних термів у стилі Мальцева, але він не вирішує проблеми з некомпозиційністю.

Другий шлях веде до нових класів предикатів, які узгоджені з номінативністю формул. Такий клас мусить бути певним розвитком класу *n*-арних предикатів. Напрямок такого розвитку вже є в семантиці Тарського для формул класичної логіки – це *врахування предметних імен*.

В такій семантиці використовується *означування* (оцінка) множини предметних імен (змінних) V на множині базових значень A, тобто тотальне відображення $V \to A$. A^V – множина таких відображень

Кожне таке відображення – набір іменованих значень вигляду $[v_1 \square a_1, \ldots, v_n \square a_n, \ldots]$, де $\{v_1, \ldots, v_n, \ldots\} = V$, $a_i \in A$.

Значення формул фактично задаються як значення предикатів на таких наборах: кожній формулі зіставляється тотальний предикат на A^V.

Проте таке подання не зовсім адекватне, адже предикати звичайно задаються на скінченних наборах іменованих значень.

Синтактико-семантична схема побудови логіки зумовила певну *невідповідність* між номінативною природою формул і вибором як базових тотальних *n*-арних предикатів.

Номінативна природа означає використання імен (змінних) у формулах

Цю невідповідність удалося затушувати шляхом неявного використання реномінацій і суперпозицій при переході від базових *n*-арних функцій і предикатів до скінченно-арних термальних функцій та атомарних предикатів, які вже залежать від конкретних предметних імен.

Необхідність *явного* введення номінативних функцій і предикатів з' явилась набагато пізніше завдяки розвитку програмування

Переваги трактування базових функцій та предикатів як *n*-арних – можливість точно вказувати необхідні аргументи

Недоліки: такі подання *не є композиційними* за Фреге

Композиційність означає: денотат складного виразу (терму, формули) є композицією денотатів його компонент

Приклад 1. Формула Ф: A&B, де A, B – ПС. Її денотат den(A&B) трактують як бінарну БФ $\& : Bool^2 \to Bool,$ де $Bool = \{T, F\}$

Формула Ψ: A&C, де A, C – ПС. Її денотат den(A&C) – та ж БФ $\& : Bool^2 \to Bool.$ Звідси $den(\Phi) = den(\Psi)$.

Денотат формули $\Phi \vee \Psi$: (A&B)$\vee$(A&C) – це БФ $\gamma : Bool^3 \to Bool$

Денотат формули $\Phi \vee \Phi$: (A&B)$\vee$(A&B) – це БФ $\& : Bool^2 \to Bool.$

Нехай σ – композиція побудови денотатів складної функції, що відповідає утворенню формули за допомогою $\vee$.

Тоді $den(\Phi \vee \Psi) = \sigma(den(\Phi), den(\Psi))$, $den(\Phi \vee \Phi) = \sigma(den(\Phi), den(\Phi))$.

Але $den(\Phi) = den(\Psi)$, звідки $den(\Phi \vee \Psi) = den(\Phi \vee \Phi)$ – суперечність.

Некомпозиційність істотно ускладнює роботу з функціями та предикатами. Це видно на прикладі операторного подання функцій, при використанні якого необхідно весь час узгоджувати арності функцій за допомогою спеціальних функцій-проекторів

Шляхи подолання невідповідності між номінативною природою формул та вибором як базових тотальних *n*-арних предикатів :

– перейти до класів формул, які ближчі до *n*-арних відображень;

– перейти до класів предикатів, які ближчі до номінативних формул.

Перший шлях веде до операторних термів у стилі Мальцева, але він не вирішує проблеми з некомпозиційністю.

Другий шлях веде до нових класів предикатів, які узгоджені з номінативністю формул. Такий клас мусить бути певним розвитком класу *n*-арних предикатів. Напрямок такого розвитку вже є в семантиці Тарського для формул класичної логіки – це *врахування предметних імен*.

В такій семантиці використовується *означування* (оцінка) множини предметних імен (змінних) V на множині базових значень A, тобто тотальне відображення $V \to A$. A^V – множина таких відображень

Кожне таке відображення – набір іменованих значень вигляду $[v_1 \square a_1, \ldots, v_n \square a_n, \ldots]$, де $\{v_1, \ldots, v_n, \ldots\} = V$, $a_i \in A$.

Значення формул фактично задаються як значення предикатів на таких наборах: кожній формулі зіставляється тотальний предикат на A^V.

Проте таке подання не зовсім адекватне, адже предикати звичайно задаються на скінченних наборах іменованих значень.

Приклад 2. $V = \{x,\ y,\ z,\ ...\}$ – множина предметних змінних (імен). Предикат P над Z, визначений формулою $x<y$, можна розглядати як визначений на наборах іменованих значень $[x \square a, y \square b, z \square c, ...]$, де $a, b, c \in Z$, $x, y, z \in V$. Такі предикати назвемо *V-квазіарними.*

Для предиката P *істотними* (від яких предикат може залежати) є тільки змінні x та y, а інші змінні *неістотні*, тому P буде визначеним на всіх наборах, що містять значення змінних x та y, напр. $[x \square 5, y \square 8, z \square 4]$.

Цей набір – часткова функція $V \rightarrow Z$. Клас цих наборів позначимо ${}^{V}Z$

Такі набори (функції) назвемо номінативними (іменними) даними, тому що стосовно предикатів вони є їхніми аргументами.

Предикат P не буде визначений на даних, що не містять значень для x або y, наприклад, він невизначений на $[y \square 7, z \square 3]$.

Таким чином, семантику формул логіки доцільно задавати *частковими* предикатами типу $({}^{V}A \rightarrow Bool)$, де $Bool = \{T, F\}$

Такі номінативні предикати назвемо *V-квазіарними.*

Поняття квазіарного предиката виникає тоді, коли кількість його аргументів наперед не визначена.

Подання номінативних відображень уже композиційні за Фреге

В класичній логіці логічні зв’язки зазвичай трактуються як булеві функції. Для часткових предикатів таке трактування не зовсім прийнятне.

Приклад 3. Розгл. формулу $(x>y)\vee(x>z)$. Якщо трактувати диз’юнкцію як тотальну бінарну БФ $\vee : Bool^2 \to Bool$, то предикат, що задається цією формулою, можна отримати суперпозицією $\boldsymbol{S}^3$ у функцію $\vee$ предикатів, заданих підформулами $x>y$ та $x>z$.

Таке подання коректне, якщо предикати тотальні.

На даному $[x \square 7, y \square 5]$ виникає частковість: підформула $x>y$ істинна, але значення підформули $x>z$ невизначене з-за відсутності значення для z. Водночас значення всієї формули $(x>y)\vee(x>z)$ на цьому даному природно вважати визначеним та істинним. Але для подання предиката, заданого формулою $(x>y)\vee(x>z)$, не можна використовувати суперпозицію $\boldsymbol{S}^3$, адже її значення невизначене, якщо хоча б один аргумент невизначений

Такі труднощі долають, переходячи до тотальних тризначних предикатів і відповідних тризначних логічних зв’язок.

Перехід до тризначних відображень порушує адекватність подання багатьох властивостей часткових відображень, зокр., обчислюваності

Отже, для випадку часткових предикатів трактування логічних зв’язок як *n*-арних булевих функцій стає неадекватним.

Тому варто відмовитися від трактування логічних зв’язок як БФ. Трактуємо їх як оператори (композиції) на мн-ні часткових предикатів.

Таким чином, обмеження класичної логіки мотивують необхідність побудови нових логік, більше орієнтованих на потреби програмування. Таку побудову доцільно вести в семантико-синтаксичному стилі.

Висновки. При побудові програмно-орієнтованих логік доцільно керуватися такими положеннями:

– використовувати семантико-синтаксичний підхід до побудови логіки;

– переходити від *n*-арних до номінативних (квазіарних) предикатів при поданні денотатів формул;

– пропозиційні зв’язки і квантори трактувати як композиції квазіарних предикатів

Центральний момент побудови програмно-орієнтованих логік – *вибір класу номінативних (квазіарних) предикатів як семантичної основи* логіки.

Цей вибір має важливі наслідки для логіки:

– логіка стає логікою часткових (а не тотальних) предикатів;

– з'являється можливість побудови логік різного рівня абстракції (інтенсіональні аспекти логіки);

– з'являється можливість відокремити семантику від синтаксису й дослідити її окремо в межах алгебр предикатів;

– логіка стає ближчою до програмування

Основні аспекти логік, орієнтованих на дослідження програм

На базі розгляду основних конструкцій мов програмування та їх формалізації (на прикладі мови *SIPL*) можна зробити висновки:

1) формалізація здійснюється на базі різних типів алгебр даних і програмних алгебр (алгебр функцій)

2) побудова складних даних із простіших відбувається на основі відношень іменування (номінативних відношень)

3) побудова складних функцій, що задають семантику програм (програмних функцій), – за допомогою композицій (алгебраїчних операцій);

4) основні властивості програмних функцій – еквітонність і монотонність

Це означає: якщо програма завершується з якимись результатами на певному стані пам’яті, то вона завершиться й на розширеному стані, причому однакові змінні в обох результуючих станах будуть мати однакові значення. Це гарантує, що програми при збільшеній пам’яті працюватимуть так само, як і при меншій.

Ці твердження є основоположними для побудови логік, які орієнтовані на дослідження властивостей програм.

Таким чином, провідну роль при побудові програмно-орієнтованих логік відіграють алгебраїчні й композиційно-номінативні аспекти. Тому за основу побудови нових логік природно взяти *композиційно-номінативний* підхід

Принципи композиційно-номінативного підходу

КНП до побудови моделей програм і орієнтованих на них логік задає принципи визначення й дослідження формальних мов програм і логічних систем

Базовий методологічний принцип, використовуваний при експлікації понять логіки та програмування, – *принцип розвитку від абстрактного до конкретного*:

поняття досліджуваного об'єкта визначається в процесі розвитку

Розвиток починається з найабстрактніших визначень, що відбивають загальні властивості об'єкта, і поступово переходить до конкретніших визначень, які відбивають специфічні властивості об'єкта.

Рух від абстрактного до конкретного відбувається за тріадною схемою:

теза антитеза синтез

Ця схема відповідає рівням дослідження об'єкта:

– *синкретичний* рівень (об'єкт як нерозчленована єдність),

– *аналітичний* рівень (виділяються та аналізуються складові об'єкта),

– *синтетичний* рівень (об'єкт подається в єдності його складових).

Принцип розвитку дуже важливий для логіки та програмування, він веде до ієрархії визначень об'єкта на різних рівнях абстрактності й загальності

Визначення об’єкта на певному рівні абстракції (*інтенсіональний* аспект) має бути доповненим визначенням класу об’єктів, які належать цьому рівню (*екстенсіональний* аспект), при цьому обидва аспекти мають вивчатися в їх єдності. Таким чином, формулюємо загальнонауковий принцип

єдності (*інтегрованості*) *інтенсіональних й екстенсіональних аспектів*:

– поняття подаються в єдності їх інтенсіональних та екстенсіональних аспектів

– інтенсіональний аспект у цій єдності має провідну роль.

Тут інтенсіональні та екстенсіональні аспекти розуміємо згідно з логічною традицією: інтенсіонал тлумачимо як зміст поняття, екстенсіонал – як його обсяг.

Втіленням пріоритету інтенсіонального аспекту над екстенсіональним є *принцип пріоритетності семантики над синтаксисом*:

семантичний і синтаксичний аспекти об’єкта мусять спочатку вивчатись окремо, потім сумісно з пріоритетом семантичного аспекту

Основні наукові принципи: композиційності й номінативності.

Вони задають особливості КНП.

Принцип композиційності трактує засоби побудови програм (функцій, предикатів) як алгебраїчні операції.

Для логіки це означає зведення логічних зв'язок і кванторів до композицій предикатів.

Принцип номінативності вимагає використання відношень іменування для побудови семантичних моделей та опису програм (функцій, предикатів).

Наведені принципи визначають напрями й особливості експлікації основних понять логіки та програмування,

вони є основою, ядром композиційно-номінативного підходу до побудови програмних і логічних систем

Розвиток основних понять програмування і розгортання поняття програми веде до пентади основних програмних понять (програмної пентади)

Дані функція ім'я функції композиція дескрипція.

Програмні поняття формалізуються за допомогою програмних систем, які розкривають поняття програми на різних рівнях розгляду.

У програмній пентаді виділяємо семантичний, синтаксичний, денотаційний аспекти. Кожен з них подається відповідною системою – композиційною, дескриптивною, денотаційною. У сукупності вони задають *програмну* систему.

Композиційні системи визначають засоби побудови функцій над деякою множиною даних. Вони мають вигляд (*D*, *Fn*, ***C***), де

– *D* – множина даних,

– *Fn* – множина функцій над *D*,

– ***C*** – множина композицій (операцій) над *Fn*.

КС (*D*, *Fn*, ***C***), задає дві алгебри:

– алгебру даних (*D*, *Fn*)

– алгебру функцій (програмну алгебру) (*Fn*, ***C***).

Дескриптивні системи задають дескрипції (терми, формули), що є описами функцій

Дескрипції визначаються індуктивно, використовуючи імена базових функцій та імена композицій.

Зазвичай дескрипціями є терми програмної алгебри.

Денотаційні системи задають денотати (значення) дескрипцій.

Програмна система визначається як

композиційно-номінативна система (***Cs***, ***Ds***, ***Dns***), де

Cs – композиційна, ***Ds*** – дескриптивна, ***Dns*** – денотаційна системи.

Центральним поняттям логіки є поняття предиката.

З математичного погляду предикати – це функції, значеннями яких є істиннісні (булеві) значення. Тому класи предикатів теж можна задавати композиційно-номінативними системами.

Предикатні композиційно-номінативні системи є семантичною основою різноманітних логік. Це дозволяє подавати логічні та програмні системи в єдиному стилі й вивчати їх на основі спільного підходу

Висновки. КНП задає принципи визначення й дослідження формальних мов програм і логік. Його суть полягає в такому:

– семантичний аспект мови є провідним, синтаксичний – похідним; тому спочатку визначають і досліджують семантичні аспекти, а лише потім – синтаксичні;

– семантичні аспекти уточнюють за допомогою композиційно-номінативних систем, які визначають структури даних, функцій (зокрема предикатів) і композицій; такі системи можна подавати у вигляді двох семантичних алгебр – алгебри даних і алгебри функцій (предикатів);

– дані уточнюють як номінативні, побудовані на базі відношення ім’я значення

– основні структури даних мов програмування можна подати як конкретизації номінативних даних

– властивості семантичних алгебр індукують синтаксичні аспекти мов

– побудова мов відбувається від абстрактного до конкретного, що приводить до мов різного рівня абстракції й загальності

Інтенсіональні аспекти понять математичної логіки

Математична логіка та програмування мають визначатись у розвитку своїх понять. Виокремимо основні поняття логіки й визначимо їх на різних рівнях абстракції, ураховуючи інтенсіональні аспекти в першу чергу.

Первісним для логіки є поняття ***висловлення*** (судження).

Висловлення – це речення (текст), яке можна розглядати з погляду його істинності чи хибності.

Математична логіка відрізняється від *загальної* (традиційної) логіки й логіки *пізнання* (гносеології) зовнішнім (формальним) наданням смислу висловленням. Це веде до концепції багатьох світів та інтерпретацій висловлень у них. Математично це можна задати за допомогою поняття ***предиката*** як відображення даних у істиннісні значення; висловлення стає значенням предиката на конкретних даних.

Фундаментальні аспекти логіки – *істинність висловлень* та *вивідність* (істинних) *висловлень*. Задаючи істинність і вивідність як *класи* висловлень, маємо *найабстрактніше* тлумачення логіки, що має три складових:

– *Prop* – висловлення;

– |= – істинність;

– |– – вивідність

Класова надабстрактна логіка. На початковому рівні уточнення понять логіки вважаємо, що *Prop* є класом (множиною), |= та |– є підкласами (підмножинами) *Prop*. Звідси початкове визначення логіки

Класова надабстрактна логіка – це трійка L(COA) = (*Prop*, |=, |–).

Це вже дозволяє дати початкові формулювання проблем, притаманних логікам.

1) *Несуперечливість істинності*:

клас істинних висловлень – власний підклас усіх висловлень: $|= \subset Prop$

2) *Несуперечливість вивідності*: $|- \subset Prop$

3) *Коректність*: клас вивідних висловлень (теорем) – підклас істинних: $|- \subseteq |=$.

4) *Повнота*: клас істинних висловлень – підклас вивідних: $|= \subseteq |-$

5) *Розв'язність істинності*:

$\Phi \in Prop \Rightarrow$ можна визначити, чи Φ істинна, тобто відповісти на питання "$\Phi \in |=$"

5) *Розв'язність*: *вивідності*:

$\Phi \in Prop \Rightarrow$ можна визначити, чи Φ вивідна, тобто відповісти на питання "$\Phi \in |-$"

Типова ситуація щодо співвідношень цих понять: $|- \subseteq |= \subset Prop$.

Це означає несуперечливість істинності й вивідності та коректність вивідності.

Для логік верхнього рівня абстракції часто додатково маємо повноту: $|- = |=$.

Для достатньо складних логік повнота та розв'язність відсутні

Індивідна надабстрактна логіка. Рівень класової надабстрактної логіки формально не дозволяє змістовно відрізняти класи істинних висловлень і теорем, тому робимо подальші конкретизації.

Перша конкретизація – заперечуємо тлумачення істинності та вивідності як завершених класів.

Вважаємо, що класи тавтологій і теорем задані не як цілісності, а через свої індивідні елементи – через певні механізми розпізнавання істинних висловлень і породження теорем (це дозволяє розрізнити роль істинних висловлень і теорем у логіці).

На абстрактному рівні розгляду розпізнавання істинності елемента відбувається за допомогою певної характеристичної функції, яка набуває значення T (*true*), тобто функції $\models : Prop \rightarrow \{T\}$.

Породження задається багатозначною функцією $\vdash : \{T\} Prop$.

Це підводить до традиційних властивостей логічних понять.

Тепер можна їх властивості та проблеми записувати не в термінах завершених класів, а в термінах одиничних (індивідних) елементів.

Зокрема, замість $\models(\Phi)\downarrow = T$ пишемо $\models \Phi$.

Пишемо $\vdash \Phi$ замість $\vdash (T)\downarrow = \Phi$.

Це дозволяє переформулювати основні проблеми логіки:

1) *Несуперечливість*

– *істинності*: існує $\Phi \in Prop$, що не $\models \Phi$;

– *вивідності*: існує $\Phi \in Prop$, що не $\vdash \Phi$.

2) *Коректність*: якщо $\vdash \Phi$, то $\models \Phi$.

3) *Повнота*: якщо $\models \Phi$, то $\vdash \Phi$.

4) *Розв'язність*

– *істинності*: чи можна визначити, що $\models \Phi$;

– *вивідності*: чи можна визначити, що $\vdash \Phi$

Маємо друге наближення до визначення логіки.

Індивідна надабстрактна логіка – це трійка

$$L(IOA) = (Prop, \models, \vdash),$$

де *Prop* – певна множина, $\models : Prop \to \{T\}$, $\vdash : \{T\} Prop$

Абстрактна логіка моделей світів. Подальша конкретизація – заперечення синкретичного розгляду висловлень і перехід до їх аналітичного тлумачення.

Це пов'язано з виділенням у висловленнях двох складових – форми та змісту (синтаксису та семантики).

Форма задається класом формул *Form*, а зміст – інтерпретаціями формул у класі моделей світів.

На цьому рівні визначень логіки головним є поняття *моделі світу*.

Згідно із КНП, модель світу певного рівня (у семантичному аспекті) має інтенсіональну та екстенсіональну складові.

Інтенсіональна складова – *інтенсіональна модель* ***IM*** – описує (онтологічні та гносеологічні) властивості.

Екстенсіональна складова задає *клас* ***EM*** *екстенсіональних* (одиничних) *моделей світів*, які мають ці інтенсіональні властивості.

Інтенсіональна модель займає провідну позицію, вона

– специфікує логіку згідно з рівнем абстракції та

– індукує *мову логіки L* відповідного рівня (синтаксичний аспект), яка задається *класом формул* (*дескрипцій*) *Frm*.

Кожна $\Phi \in Frm$ інтерпретується (набуває значення) в екстенсіональній моделі $M \in \boldsymbol{EM}$ за допомогою *відображення інтерпретації* $I_M : Form \rightarrow M$.

Істинність Ф (пишемо |= Ф) індукується інтенсіональною моделлю й задається на підставі значень I_M(Ф).

Вивідність Ф (пишемо |– Ф) задається певним формалізмом, визначеним на множині формул (наприклад, формальною системою (*Frm*, *Ax*, *R*)).

Отже, *абстрактну логіку моделей світів* можна уточнити як об'єкт вигляду (***IM***, ***EM***, *Frm*, ***I***, |=, |–), де

IM – інтенсіональна модель,

EM – клас екстенсіональних моделей,

Frm – клас формул мови,

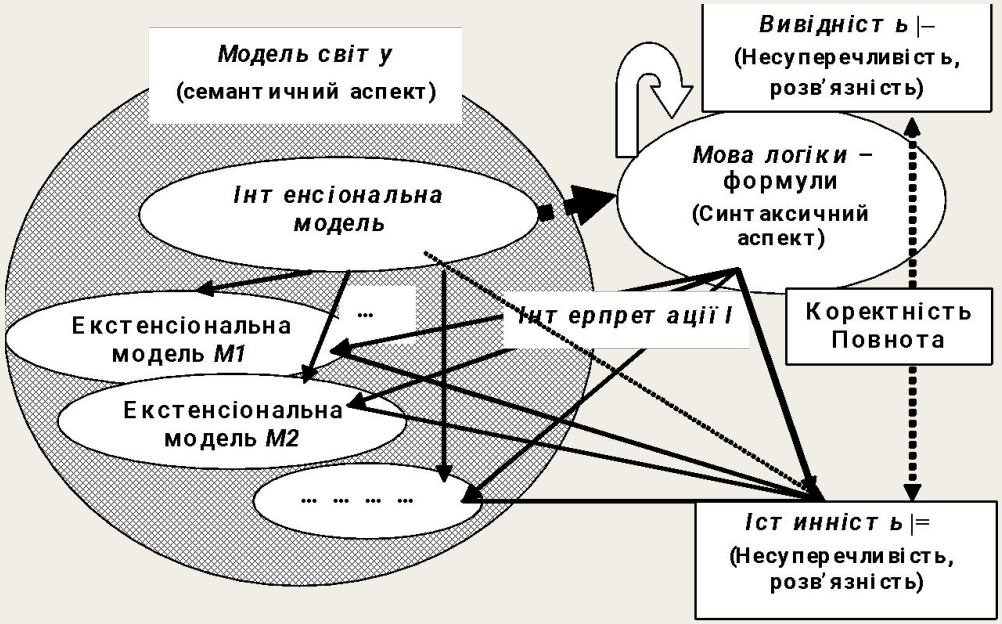
I – клас відображень інтерпретації формул в екстенсіональних моделях,

|= – клас істинних формул,

|– – клас вивідних формул (теорем).

Абстрактна логіка моделей світів не розкриває структуру інтенсіональної та екстенсіональних моделей, тому не дає змоги продемонструвати зв’язки мови логіки з моделями світів. Це – на наступних рівнях конкретизації.

Перехід до конкретніших рівнів дозволить підкреслити *провідну роль інтенсіональних аспектів*: саме інтенсіональні аспекти визначають тип логіки: її мову, інтерпретації, істинність і, певним чином, вивідність



Поняття абстрактної логіки моделей світів та їх зв'язки

Композиційно-номінативні логіки предикатів. На цьому рівні модель світу задається:

– певним класом D станів світу (їх назвемо даними);

– класом предикатів Pr, заданих на станах світу;

– операторами (композиціями) $\boldsymbol{C}$ породження нових предикатів.

Такий розгляд дає змогу виділити як інтенсіональну складову цих понять (даного, предиката, композиції), так і екстенсіональну.

Зазначимо: з інтенсіонального погляду предикат є особливою функцією, результати якої (істиннісні значення) завжди конкретні ("білі скриньки").

Екстенсіональна модель задається як

предикатна композиційна система (D, Pr, $\boldsymbol{C}$).

Така система задає дві алгебри:

– алгебру даних (D, Pr)

– алгебру предикатів (Pr, $\boldsymbol{C}$).

Класи даних (світ) і предикатів на даних (на станах світу) задають *предметну* складову логіки певного типу, а композиції – її *універсальну* складову.

Тому центральним поняттям у моделях світів є поняття *композиції*.

Саме композиції визначають універсальні методи побудови предикатів, виступаючи ядром логіки певного типу.

Визначення композицій є інтенсіональним, що дозволяє далі інтерпретувати їх уніформно в різних екстенсіональних моделях.

Отже, на даному рівні розгляду можна казати про *композиційні* моделі світів.

Основними проблемами, що виникають при їх вивченні, є

– проблема *інтенсіональної експлікації* класу композицій на основі інтенсіональних визначень класів світів і предикатів

– проблема *вербалізації* композицій, тобто *побудови мови логіки* певного типу, що має формальну семантику композицій, базовану на їх експлікаціях.

Останню проблему розв'язуємо визначенням базових композицій алгебри предикатів. Це дозволяє трактувати терми цієї алгебри як *формули* мови логіки.

Означування даних і предикатів може відбуватися за допомогою відношень номінації (іменування).

Беручи до уваги фундаментальну роль відношень іменування у визначенні основних логічних понять, на цьому рівні будемо говорити про *композиційно-номінативні* логіки (КНЛ)

Поняття *композиційної алгебри предикатів* лежить в основі КНЛ.

Дослідження семантичних аспектів КНЛ зводиться до вивчення властивостей таких алгебр.

КНЛ можна назвати аксіоматичними системами алгебр предикатів.

Визначальними для КНЛ є проблеми виділення й дослідження класів алгебр предикатів, які задаються в єдності їх інтенсіональних і екстенсіональних аспектів.

Для конкретного опису інтенсіональних аспектів основних понять логіки, визначення композицій та мови, треба розглянути

інтенсіонали даних і предикатів.

Стосовно зв’язку з основними поняттями логіки і програмування, дані трактуємо як об’єкти, до яких застосовуються функції (предикати). Тому найперше розглянемо розвиток поняття даного з інтенсіонального погляду

Розвиток поняття даного в логіці й програмуванні ведемо за напрямами *ціле – частина* та *абстрактне – конкретне*.

Перший напрям розвивається за тріадою

ціле (whole) – *частина* (parts) – *ієрархія* (hierarchy).

Це означає:

– спочатку дане розглядаємо як цілісність, у структуру якої не проникаємо

– на другому рівні дане тлумачимо як структуроване, що складається з частин

– на третьому, синтетичному рівні, частини розглядаємо як цілі, що можуть мати свої частини, тому на цьому рівні дані тлумачимо як ієрархічні.

Звідси три рівні розвитку поняття даного:

– D.W (дане як ціле),

– D.P (дане як структуроване, із частинами),

– D.H (дане як ієрархічне).

Другий напрям розвивається за тріадою

абстрактне – конкретне – синтетичне

Звідси кожен із трьох рівнів далі розпадається на три підрівні:

– A (абстрактний),

– C (конкретний),

– S (синтетичний).

Підсумовуючи, отримуємо 9 *рівнів розгляду* (типів) *даних*:

D.W.A	D.W.C	D.W.S;
D.P.A	D.P.C;	D.P.S;
D.H.A	D.H.C	D.H.S.

Кожен рівень визначає певний інтенсіонал.

Об’єкти, які мають певний інтенсіонал, задають екстенсіонал.

Для абстрактних рівнів А екстенсіонал багатий,
для конкретних рівнів С – бідний.

Приклади даних рівня D.W:

1) Дані як "чорна скринька" (напр., ПІН-код у конверті) – D.W.A.

2) Дані як "біла скринька" (напр., символи певного алфавіту) – D.W.C.

3) Дані як "чорна" або "біла скринька" (зовнішній "байдужий" синтез, можемо відрізнити "чорну скриньку" від "білої") – D.W.S.

Головне питання подальшого розгортання цієї схеми:
як відбувається перехід з рівня W на рівень P.

Найважливішим тут є *зв’язок між частинами*

На рівні D.P частини даних вважаються незв’язаними (слабко зв’ язаними). Такі не зв’язані між собою (слабко зв’язані) частини назвемо *елементами* даних.

Дані рівня D.P назвемо *сукупностями* (aggregate).

Приклади даних рівня D.P:

1. Пасажири в автобусі для перехожих – "чорні скриньки", які слабко зв’язані між собою, тому утворюють певну сукупність – рівень D.P.A.

Дані такого рівня назвемо *скупченнями* (assemblage).

2. Ще один приклад – купа шоколадних яєць "Кіндер-сюрприз", адже їх вміст є невідомим.

Дані рівня D.P.A можна перевіряти на порожність (має таке дане елементи чи ні?), об’єднувати їх ("звалити" дві купи в одну), проте перетин даних рівня D.P.A визначити неможливо

3. Мешканців певної квартири можна розглядати як сукупність "білих скриньок" – рівень D.P.C.

Дані такого рівня назвемо *множинами* (set).

Сформулюємо інтенсіональні властивості множин.

Множини складаються з елементів, для яких діють постулати:

– *елементи не зв'язані між собою* (аксіома екстенсіональності);

– *елементи відрізняються один від одного*; це означає, що рівність або нерівність елементів розв'язна (розв'язність тлумачимо у широкому сенсі, не вимагаючи наявності певного механізму розв'язку);

– *належність елемента множині розв'язна* (маючи елемент і множину, можна відповісти на питання, належить елемент множині чи ні).

На рівні D.P.S елементи є синтезом абстрактного та конкретного. Елемент має два боки: "білий" – конкретний; "чорний" – абстрактний.

Суть цього синтезу випливає з *якості даного* як об’єкта, що є формою подання інформації (дозволяє записувати, зберігати й надавати інформацію). При запиті інформація ще невідома ("чорна скринька"), тому запит може відбуватися лише за допомогою "білої скриньки" – імені (символу, знака).

Тому зв'язок частин у даному цього рівня є *зв’язком іменування*.

Таким чином, на рівні D.P.S говоримо про *номінативні* елементи.

Сукупності номінативних елементів називають *номінатами*

Наведене розгортання множини і номінату дозволяє чітко їх розрізнити

Поняття номінату – синтез абстрактного та конкретного.

Імена в номінаті розглядаються як конкретні об’єкти, що мають властивості елементів множини, а значення – як абстрактні.

Номінати будуються на основі відношення *ім’я→значення*.

Вони є найважливішим типом даних для логіки й програмування.

Номінати мають двоїсту природу: з одного боку, це сукупності, а з іншого – функції, адже зв’язок *ім’я→значення* має функціональний відтінок. Тому номінати доцільно подавати за допомогою функцій, адже функції дають змогу використовувати абстрактні значення, що не дозволяється у множинах.

Однозначні номінати називають *іменними множинами*.

ІМ тлумачимо як множини пар, першою компонентою яких є ім’я, а другою – значення цього імені

При цьому одне ім’я не може іменувати два різних значення (принцип однозначності іменування).

Рівень D.H є синтетичним, тут маємо синтез *частина – ціле*.

Частини можуть розглядатися як цілі, що мають свої частини.

Це веде до *ієрархічних* даних.

Ієрархічність можна формалізувати за допомогою рекурентних, індуктивних і рекурсивних визначень.

Звідси випливає важливість цих понять у логіці та програмології.

Дані рівня D.H класифікуються так:

– ієрархічні скупчення (D.H.A)
– ієрархічні множини (D.H.C)
– ієрархічні номінати – номінативні дані (D.H.S).

Номінативні дані дуже важливі для програмування й логіки.

Рівні D.H.A та D.H.C є виродженими

Розвиток поняття функції подібний до розвитку поняття даного.

Отримуємо 9 рівнів розгляду поняття функції:

F.W.A	F.W.C	F.W.S
F.P.A	F.P.C	F.P.S
F.H.A	F.H.C	F.H.S

Головна особливість полягає в тому, що, на відміну від даних, частини функцій зв'язані між собою.

Для функцій *F.P* – рівень (*функціональних*) *комплексів*; його частини – (*функціональні*) *компоненти*, вони мають певні нетривіальні зв'язки.

F.H – рівень *ієрархічних функціональних комплексів*.

Він пов'язаний, зокрема, з різними теоріями покрокової розробки програм (експлікативністю процесу програмування), теоріями повторного використання програм тощо

Розглянемо характеристики функцій рівня F.P.

Рівень F.P.A (абстрактний комплекс) – зв’язки та компоненти є абстрактними.

Наприклад, у кінофільмах герої намагаються знешкодити часові бомби. Є багато дротів і складових, але невідомо, що всередині та який дріт можна перерізати, щоб відключити механізм.

Рівень F.P.C (конкретний комплекс) – усе відомо, тому функції цього рівня подають самі себе.

Рівень F.P.S (синтетичний комплекс) є найважливішим, що пояснюється його широким використанням

Рівень F.P.S характеризується фіксованими зв’язками між абстрактними компонентами.

Наприклад, коли ми збираємо свою конфігурацію комп’ютера, то компоненти (пам’ять, вінчестер тощо) можна сприймати як "чорні скриньки" зі входом і виходом, але зв’язки яких відомі

Наявність компонент як абстрактних дозволяє вважати, що замість деякої компоненти може бути якась інша.

Така абстракція від компонент на рівні синтетичного комплексу – це *композиція*.

Синтетичний комплекс при цій абстракції подається двома складовими:

– компоненти ("чорні скриньки")

– їх зв’язки ("білі скриньки").

Зв'язок відбувається через імена компонент (функцій).

Отже, виникають функціональні номінати.

Таким чином, композицію можна уточнити як

оператор, визначений на іменованих функціях.

Важливість композицій і композиційності в логіці та програмології була усвідомлена В. Н. Редьком.

На основі принципу композиційності він розвинув спеціальний напрям у програмології – композиційне програмування

Головною складовою інтенсіоналу поняття є властивості відповідного рівня абстракції.

Тому позначення рівнів використовуємо і для позначення інтенсіоналів понять.

Традиційним для логіки є розгляд предикатів, заданих на рівнях

D.W.A (пропозиційні логіки)

D.P.S (першопорядкові логіки).

Досліджено спеціальні логіки рівня D.W.C (сингулярні)

Розпочато вивчення логік над ієрархічними номінативними даними (рівень D.H.S).

Логіки рівнів D.P.S та D.H.S дуже важливі для програмування.

Логіки рівнів D.W.S, D.P.A, D.P.C, D.H.A, D.H.C до певної міри вироджені й вимагають окремого дослідження

Спектр композиційно-номінативних логік

Побудову логічних систем ведемо на основі КНП.

Логіки, збудовані на його основі, названо композиційно-номінативними.

Будуємо КНЛ за семантико-синтаксичною схемою.

1. Спочатку задаємо інтенсіональні (змістовні) моделі логік. Такі моделі найперше визначаються рівнями розгляду даних, тому для їх задання фіксуємо рівень абстракції розгляду. Інтенсіональні моделі індукують клас формул (мову логіки) відповідного рівня (синтаксичний аспект).

2. Будуємо відповідні до розглянутого рівня екстенсіональні моделі, які задають семантичні аспекти логік. Екстенсіональна семантична модель задається як предикатна композиційна система (D, Pr, $\boldsymbol{C}$). Вона визначає алгебру (алгебраїчну систему) даних (D, Pr) та алгебру предикатів (Pr, $\boldsymbol{C}$).

Терми алгебри предикатів трактуються як формули мови логіки.

Композиції визначають універсальні методи побудови предикатів, вони є основою, ядром логіки певного типу

Таким чином, КНЛ будуються за семантико-синтаксичною схемою.

Визначення КНЛ реалізують

єдність інтенсіонального та екстенсіонального аспектів.

Дослідження семантичних аспектів КНЛ зводиться до вивчення властивостей алгебр предикатів, які є основним поняттям КНЛ.

Застосування КНП дає змогу побудувати низку логічних моделей різноманітних ПО, що перебувають на різних рівнях абстрактності й загальності.

Побудову КНЛ починаємо з гранично абстрактних рівнів, поступово їх конкретизуючи.

Такі рівні відрізняються трактуванням рівня розгляду даних.

Будемо тут розглядати лише фінітарні КНЛ

Фінітарність логіки означає, що її композиції скінченно-арні

Рівні D.W визначають порівняно прості класи логік

Пропозиційний рівень. На рівні D.W.A дані трактуються гранично абстрактно.

Предикати, задані на даних цього рівня, мають вигляд $A \to \{T, F\}$,

де A – сукупність абстрактних даних ("чорних скриньок").

Такі предикати трактуємо гранично абстрактно

(можна використовувати лише властивість аплікативності предикатів).

На пропозиційному рівні кожний предикат P на довільному даному може набувати єдиного фіксованого значення або бути невизначеним, адже на гранично абстрактному рівні одне дане від іншого неможливо відрізнити. Тому на рівні D.W.A композиції фактично працюють лише з істиннісними значеннями, що й пояснює трактування логічних зв'язок у класичній логіці як булевих функцій.

Базовими композиціями логік пропозиційного рівня є клінієві $\vee$ та $\neg$.

Пропозиційна КНЛ дуже близька до класичної пропозиційної.

Інфінітарні пропозиційні логіки розглянуто М.Нікітченком

Сингулярний рівень D.W.C трактуємо як конкретизацію пропозиційного.

На цьому рівні дані трактуються гранично конкретно, як "білі скриньки".

На рівні D.W.C фіксується єдиний клас даних, тому цей рівень називають сингулярним.

Предикати, задані на даних рівня D.W.C, мають вигляд $D \rightarrow \{T, F\}$,

де D – множина конкретних даних.

Такі предикати трактуються гранично абстрактно.

Композиціями сингулярного рівня є конкретні аплікативні композиції.

Для сингулярних логік побудована спеціальна інфінітарна алгебра сингулярних композицій Кліні

Рівню D.W.S відповідають пропозиційні логіки змішаного (абстрактно-сингулярного) рівня.

На наступному рівні абстракції D.P. дані розглядаються як сукупності.

Виродженим рівням D.P.A та D.P.C відповідають спеціальні логіки, які вимагають окремого дослідження

Номінативний рівень. Рівень D.P.S є синтезом двох перших рівнів – абстрактного D.W.A та конкретного D.W.C. Дані розглядаються як "сірі скриньки", побудовані з "білих" і "чорних". Такі дані називаються номінатами, тому відповідні логіки відносимо до *номінативного* рівня.

Номінативний рівень є дуже багатим і розпадається на низку підрівнів.

Найважливіший підрівень однозначних номінатів – *іменних множин*

До цього підрівня можна віднести класичні першопорядкові логіки.

КНЛ рівня ІМ називають *логіками квазіарних предикатів*.

На рівні ІМ далі можна виділити

*номінативно-безквантор*ні рівні:

- реномінативний
- реномінативно-еквaційний
- безквантoрний
- безквантoрно-еквaційний;

першопорядкові рівні:

- кванторний
- кванторно-еквaційний
- функціональний
- функціонально-еквaційний

V-іменна множина над A – це однозначна функція вигляду $d : V \to A$.

V та A трактуємо як множини предметних імен і предметних значень.

Імена трактуємо гранично конкретно, предметні значення – абстрактно.

Множину всіх V-ІМ над A позначимо ${}^{V}A$.

Повна (максимальна) V-ІМ над A – тотальна однозначна функція $V \to A$.

Множину всіх повних V-ІМ над A позначимо A^{V}.

Функції, задані на ІМ, називають *квазіарними*

Квазіарна функція $f : {}^{V}A \to R$ *монотонна*, якщо

$d \subseteq d' \Rightarrow f(d') \subseteq f(d)$.

Для логік виділимо квазіарні функції двох типів.

V-квазіарна функція на A – функція вигляду $f : {}^{V}A \to A$

V-квазіарний предикат на A – функція вигляду $P : {}^{V}A \to \{T, F\}$

Клас V-квазіарних функцій на A позначимо Fn^{A}

Клас V-квазіарних предикатів на A позначимо Pr^{A}

Однозначна функція (зокрема, предикат) g *еквітонна*, якщо

з умови $g(d)\downarrow$ та $d \subseteq d'$ випливає $g(d')\downarrow = g(d)$.

Реномінативний рівень. Найабстрактнішими серед логік номінативного рівня є реномінативні. Починаючи з цього рівня, можна перейменовувати компоненти даних. Це дає змогу ввести нову композицію реномінації.

Введення реномінації (перейменування) є важливим кроком у бік номінативності, яка в класичній логіці дуже завуальована.

Базові композиції (фінітарних) реномінативних логік: $\neg$, $\vee$, $R_{\overline{x}}^{\overline{v}}$

Інфінітарні реномінативні логіки розглянуто М.Нікітченком.

Реномінативно-еквaційний рівень. Тут можна ототожнювати й розрізняти значення предметних імен за допомогою спеціальних 0-арних композицій – параметризованих за іменами предикатів рівності.

Можна розглядати дві різновидності таких предикатів – строгої (точної) рівності $\equiv_{xy}$ та слабкої рівності $=_{xy}$.

На відміну від $=_{xy}$, предикати $\equiv_{xy}$ немонотонні (нееквітонні)

Базові композиції РНЛ з слабкою рівністю: $\neg$, $\vee$, $R_{\overline{x}}^{\overline{v}}$, $=_{xy}$

Базові композиції РНЛ з строгою рівністю: $\neg$, $\vee$, $R_{\overline{x}}^{\overline{v}}$, $\equiv_{xy}$

Безквантворний рівень. Тут маємо розширені можливості формування нових аргументів для функцій і предикатів

Це дозволяє ввести композицію суперпозиції $S^{\bar{x}}$

Виділення квазіарних функцій на A та квазіарних предикатів на A індукує виділення суперпозицій двох типів: $(Fn^A)^{n+1} \to Fn^A$ та $Pr^A \times (Fn^A)^n \to Pr^A$.

Для роботи з окремими компонентами даних виділяємо спеціальні 0-арні композиції – функції деномінації (розіменування) $'x$, де $x \in V$. При їх введенні реномінації можна промоделювати за допомогою суперпозицій.

Базові композиції логік безквантворного рівня: $\neg$, $\vee$, $S^{\bar{x}}$, $'x$.

Безквантворно-еквaційний рівень. Тут додатково можна ототожнювати й розрізняти предметні значення, що дає змогу ввести спеціальну композицію рівності вигляду $Fn^A \times Fn^A \to Pr^A$.

Можна розглядати дві різновидності такої композиції:

– слабкої рівності $=$

– строгої (точної) рівності $\equiv$

Базові композиції безквантворних логік з слабкою рівністю: $\neg$, $\vee$, $S^{\bar{x}}$, $'x$, $=$

Базові композиції безквантворних логік з строгою рівністю: $\neg$, $\vee$, $S^{\bar{x}}$, $'x$, $\equiv$

Кванторний рівень. Тут можна застосовувати квазіарні предикати до всіх предметних значень

Це дозволяє ввести композиції квантифікації $\exists \boldsymbol{x}$ та $\forall \boldsymbol{x}$

Базові композиції логік кванторного рівня: $\neg$, $\vee$, $R^{\bar{v}}_{\bar{x}}$, $\exists \boldsymbol{x}$.

***Кванторно-еквaційний рівень*.** Додатково до можливостей кванторного рівня, тут можна ототожнювати й розрізняти значення предметних імен за допомогою спеціальних 0-арних композицій – предикатів рівності.

Розглядаємо предикати строгої рівності $\equiv_{\boldsymbol{xy}}$ та слабкої рівності $=_{\boldsymbol{xy}}$.

На відміну від $=_{\boldsymbol{xy}}$, предикати $\equiv_{\boldsymbol{xy}}$ немонотонні (нееквітонні)

Базові композиції логік кванторного рівня з слабкою рівністю:

$\neg$, $\vee$, $R^{\bar{v}}_{\bar{x}}$, $\exists \boldsymbol{x}$, $=_{\boldsymbol{xy}}$.

Базові композиції логік кванторного рівня з строгою рівністю:

$\neg$, $\vee$, $R^{\bar{v}}_{\bar{x}}$, $\exists \boldsymbol{x}$, $\equiv_{\boldsymbol{xy}}$

Функціональний рівень. На додаток до можливостей кванторного рівня, маємо розширені можливості формування нових аргументів для функцій і предикатів, що дозволяє ввести композицію суперпозиції. При виділенні спеціальних 0-арних композицій – функцій деномінації $'\boldsymbol{x}$, композиції реномінації можна промоделювати за допомогою суперпозицій

Базові композиції логік функціонального рівня: $\neg$, $\vee$, $S^{\bar{x}}$, $\exists x$, $'\boldsymbol{x}$.

Функціонально-еквaційний рівень. На додаток до можливостей функціонального рівня, тут можна ототожнювати й розрізняти предметні значення, що дає змогу ввести спеціальну композицію рівності.

Розглядаємо дві різновидності такої композиції:

– строгої рівності $\equiv$

– слабкої рівності $=$.

Базові композиції логік функціонального рівня з слабкою рівністю:

$\neg$, $\vee$, $S^{\bar{x}}$, $\exists x$, $'\boldsymbol{x}$, $=$.

Базові композиції логік функціонального рівня з строгою рівністю:

$\neg$, $\vee$, $S^{\bar{x}}$, $\exists x$, $'\boldsymbol{x}$, $\equiv$.

На наступному рівні абстракції D.H дані розглядаються як ієрархічні.

Виродженим рівням D.H.A та D.H.C тут відповідають спеціальні логіки, які вимагають окремого дослідження

***Ієрархічно-номінативний рівень*.** На рівні D.H.S дані можна розглядати як ієрархічні номінати. Вони будуються індуктивно з множини предметних імен і множини предметних значень.

Відповідні логіки названо логіками ієрархічних номінативних даних.

Ієрархічно-номінативний рівень дуже багатий, зроблено лише перші кроки його дослідження. Можна виділити такі підрівні рівня D.H.S:

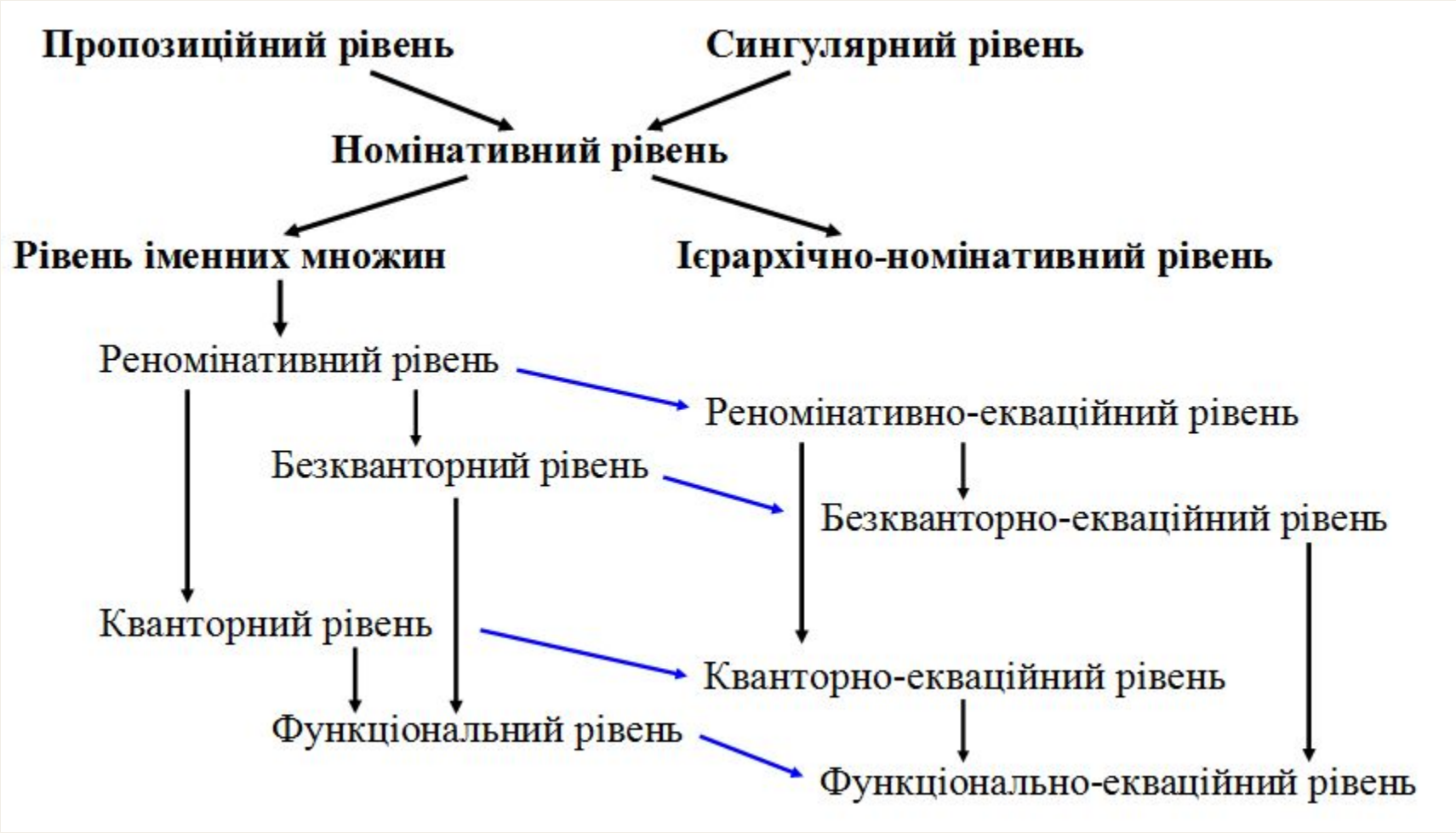
– логіки над даними зі складними іменами

– логіки над скінченними ієрархічними номінативними даними (логіки номінативних даних), вони будуються у стилі теорії допустимих множин

– логіки над структурованими даними.

Логіки останніх двох підрівнів можна трактувати як спеціальні першопорядкові КНЛ

Спектр КНЛ за рівнем абстракції розгляду



Різновидності КНЛ за обмеженнями на клас предикатів

Для логік квазіарних предикатів не виконуються деякі важливі закони класичної логіки. Тому для збереження основних властивостей класичної логіки доцільно розглянути обмеження на клас квазіарних предикатів.

Приклад. V-квазіарний предикат на Z, заданий формулою $x<y$, визначений на всіх даних, що містять значення змінних x та y, наприклад на $[x\square 5, y\square 8]$, $[x\square 5, y\square 8, z\square 3]$. Якщо цей предикат уже набув значення на якомусь даному, то на всіх більших даних, що є розширеннями першого, його значення не змінюється.

Ця властивість – еквітонність. Еквітонність можна трактувати як незмінність уже встановленого знання.

Властивість еквітонності дуже важлива для програмування.

Фактично для всіх мов програмування

– вирази та умови мови породжують еквітонні функції та предикати,

– програми та оператори породжують монотонні функції.

Еквітонність притаманна формульним предикатам класичної логіки

Інша властивість предикатів класичної логіки. – визначеність предиката на кожному повному (максимальному) даному, яке містить значення всіх предметних імен. Така властивість називається *повнототальністю*.

Властивості еквітонності й повнототальності фактично випливають із семантики Тарського.

Логіки повнототальних еквітонних предикатів. Класична логіка є логікою скінченно-арних тотальних предикатів. Логіки повнототальних ЕП є найближчими до класичної. Вони зберігають основні закони класичної логіки при істотному розширенні класу семантичних моделей.

Логіка повнототальних ЕП і класична логіка не є ідентичними.

Визначальна властивість логік квазіарних предикатів, зокрема логік повнототальних ЕП, що істотно відрізняє їх від класичної логіки, – можливість для формули бути залежною від наперед необмеженої множини предметних імен.

Для логік повнототальних ЕП відповідного рівня побудовані аксіоматичні системи гільбертівського типу – неокласичні числення. Для таких числень доведено теореми несуперечливості й повноти

Логіки еквітонних предикатів. Вони підпорядковуються основним законам класичної логіки. Проте для логік ЕП не діють деякі правила виведення (*modus ponens*, правило перетину), порушуються певні властивості класичної логіки

Клас моделей логіки ЕП є розширенням класу моделей логіки ПЕП.

Логіки ПЕП та логіки ЕП названі *неокласичними*.

Відмова від *modus ponens* спонукає проведення дослідження синтаксичних властивостей логік ЕП на базі не гільбертівських, а ґенценівських систем – секвенційних числень. Такі числення побудовано для логік ЕП відповідного рівня, для них доведено теореми коректності й повноти.

Логіки еквісумісних предикатів. Логіки, орієнтовані на такі особливості ПО, як неповнота наявної інформації, базуються на класах предикатів, визначених на даних з неповною інформацією. Такими є *еквісумісні* предикати.

Еквісумісність означає: за можливості розширення різних даних (сумісність даних) до одного більшого значення предиката на таких даних мають збігатися.

Клас ЕСП є розширенням класу ЕП.

Властивості логік ЕСП аналогічні відповідним властивостям логік ЕП.

Для логік ЕСП побудовано числення секвенційного типу, для них доведено теореми коректності й повноти

Логіки локально-еквітонних предикатів. Для ЛЕП вимагаємо збереження значення при розширенні даних лише на скінченну кількість іменованих компонент. Клас ЛЕП є розширенням класу ЕП.

Семантичні властивості ЛЕП в цілому аналогічні властивостям ЕП

Логіки локально-еквісумісних предикатів. Локально-еквісумісність предиката означає: за можливості *скінченного* розширення різних даних до одного більшого значення предиката на таких даних мають збігатися.

Властивості логік ЛЕСП аналогічні властивостям логік ЛЕП.

Секвенційні числення логік ЕП, ЛЕП, ЕСП, ЛЕСП ідентичні.

При переході від класичної до логіки ПЕП, потім до логіки ЕП, затим до до логік ЛЕП і логік ЕСП, далі до логік ЛЕСП, отримуємо все ширші класи семантичних моделей. Такі логіки мають більші виразні можливості порівняно з класичною, водночас ці логіки зберігають основні її закони.

Логіки квазіарних предикатів. Для квазіарних предикатів у загальному випадку деякі важливі закони класичної логіки вже невірні.

Властивості КНЛ часткових однозначних, тотальних неоднозначних, часткових неоднозначних квазіарних предикатів розглянемо далі.

Для КНЛ реномінативного й кванторного рівнів побудовано числення секвенційного типу, для них доведено теореми коректності й повноти

Спектр КНЛ за обмеженнями на клас квазіарних предикатів