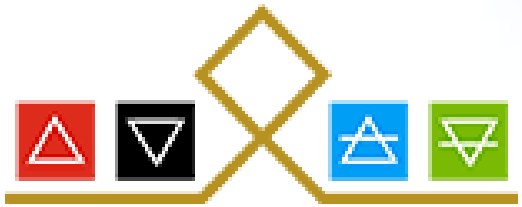
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ



Теорія розпізнавання образів та класифікації в системах штучного інтелекту

Тема №3. Подання знань у системах штучного інтелекту

Київ - 2025

Зміст

Мета лекції

Вивчення та дослідження основних понять подання знань у системах штучного інтелекту

Список літератури

1. Адаменко А. Логическое программирование и Visual Prolog / А. Адаменко, А. Кучуков. – СПб. : БХВ-Петербург, 2003. – 982 с.
2. Борисов В. В. Нечеткие модели и сети / В. В. Борисов, В. В. Круглов, А. С. Федулов. – М. : Горячая Линия – Телеком, 2007. – 284 с.
3. Боровиков В. Нейронные сети. Statistica Neural Networks. Мето- дология и технологии современного анализа данных / В. Боровиков. – М. : Горячая Линия – Телеком, 2008. – 392 с.
4. Братко И. Язык Prolog (Пролог): алгоритмы искусственного интеллекта, 3-е изд.: пер. с англ. / И. Братко. – М. : Вильямс, 2004. – 640 с.
5. Бураков М. В. Интеллектуальные системы управления: учеб. пособ. / М. В. Бураков, О. С. Попов. – СПб. : ГААП., 1997. – 108 с.
6. Гаврилова Т. А. Базы знаний интеллектуальных систем / Т. А. Гаврилова, В. Ф. Хорошевский. – СПб. : Питер, 2000. – 384 с.
7. Джарратано Дж. Экспертные системы: принципы разработки и программирование, 4-е изд. / Дж. Джарратано, Г. Райли. – М. : Вильямс, 2006. – 1152 с.
8. Джексон П. Введение в экспертные системы, 3-е изд. : пер. с англ. / П. Джексон. – М. : Вильямс, 2001. – 624 с.
9. Искусственный интеллект: справочник: в 3 кн. / под ред. Э. В. Попова. Кн. 1: Системы общения и экспертные системы. – М. : Радио и связь, 1990. – 462 с.
10. Каллан Р. Основные концепции нейронных сетей: пер. с англ. / Р. Каллан. – М. : Вильямс, 2001. 287 с.

11. *Коста Э. Visual Prolog 7.1 для начинающих: пер. с англ. / Э. Коста – М., 2008. – 210 с.*
12. Круглов В. В. Нечеткая логика и искусственные нейронные се- ти / В. В. Круглов, М. И. Дли, Р. Ю. Голунов. – М. : ФИЗМАТЛИТ, 2001. – 224 с.
13. Круглов В. В. Искусственные нейронные сети. Теория и практи- ка, 2-е изд. Стериотип. / В. В. Круглов, В. В. Борисов. – М. : Горячая линия – Телеком, 2002. – 382 с.
14. Люгер Джордж Ф. Искусственный интеллект: стратегии и ме- тоды решения сложных проблем, 4-е изд. : пер. с англ. / Джордж Ф. Люгер – М. : Вильямс, 2005. – 864 с.
15. Поспелов Г.С. Искусственный интеллект – основа новой инфо- рмационной технологии / Г. С. Поспелов – М. : Наука, 1988. – 289 с.
16. Рассел С. Искусственный интеллект: современный поход, 2-е изд. : пер. с англ. / С. Рассел, П. Норвиг. – М. : Вильямс, 2016. – 1408 с.
17. Рутковская Д. Нейронные сети, генетические алгоритмы и не- четкие системы: пер. с польск. / Д. Рутковская, М. Пилиньский, Л. Рутковский. – М. : Горячая линия – Телеком, 2006. – 452 с.
18. Рутковский Л. Методы и технологии искусственного интеллек- та / Л. Рутковский. – М. : Горячая Линия – Телеком, 2010. – 520 с.
19. Хайкин С. Нейронные сети: полный курс. 2-е изд.: пер. с англ. / С. Хайкин. – М. : Вильямс, 2008. – 1103 с.
20. Цуканова Н. И. Логическое программирование на языке Visual Prolog: учебч. пособ. для вузов / Н. И. Цуканова, Т. А. Дмитри- ева. – М. : Горячая линия-Телеком, 2008. – 144 с.
21. Частиков А. П. Разработка экспертных систем. Среда CLIPS. / А. П. Частиков, Д. Л. Белов, Т. А. Гаврилова. – СПб. : БХВ- Петербург, 2003. – 608 с.
22. Штовба С. Д. Проектирование систем средствами MATLAB / С. Д. Штовба. – М. : Горячая Линия – Телеком, 2007. – 288 с.

Знання та моделі подання знань у системах штучного інтелекту

Дані (*Data*) **–** це те, що може реєструватися в тій або іншій формі органами чуття людини або приладами. Тиск, вологість, яскра- вість, радіаційний фон тощо.

База даних – це певним чином організована сукупність даних, що дозволяє забезпечити ефективний пошук, розміщення і модифікацію даних.

Знання (*Knowledge*) – це зафіксована закономірність щодо фактів, процесів, явищ і причинно-наслідкових відносин між ними.

База знань (*Knowledge Base*) – організована певним чином сукупність знань, що дозволяє забезпечити ефективний логічний вивід рішення поставленої задачі, розміщення, модифікації і поповнення знань.

Класифікація моделей знань

Ф*ормальні* (логічні) методи, в основі яких лежить чітка математична теорія.

Неформальні – моделі такої теорії не дотримуються.

В основі ***логічного методу*** подання знань у СШІ лежить формальна система – логіка предикатів І порядку, яка мовою теорії множин описується наступною четвіркою:

$$M = <T, P, A, B>,$$

де T – множина базових елементів; P – множина синтаксичних правил, за допомогою яких із елементів множини T утворюють синтаксично правильні сукупності; A – елементи цієї множини утворюють аксіоми, визначені на множині P; $P(A)$ – процедура, що визначає приналежність до A; B – множина правил виводу, які застосовуються до елементів множини A.

Вибір МПЗ є важливим питанням при створенні СШІ. При використанні *логіки* предикатів І порядку база знань може розглядатися як сукупність логічних формул, які забезпечують частковий опис проблемного середовища. Перевага – можливість безпосередньо запрограмувати механізм виведення. Недоліки: відсутність наочності та зручності читання, громіздкість запису, що при- зводить до складності знаходження помилок.

Великі труднощі виникають при створенні моделей неповних, нечітких знань. Моделі на основі *нечіткої логіки* Л. Заде дозволяють оперувати розмитими поняттями, проте такі результати інтерпретувати важче і навіть не завжди можливо.

Семантичні сітки дозволяють описувати властивості й відношення об'єктів, подій, понять, ситуацій або дій за допомогою направленого графа, що складається з вершин і помічених ребер. Переваги: наочність, зрозумілість, зручність для програмування. Недолік – втрата наочності при розширенні моделі.

Фрейми, як і семантичні сітки, є декларативно-процедурними структурами. Можливе наслідування атрибутів більш абстрактних об'єктів. Перевага – економне розміщення БЗ у пам'яті комп'ютера. Недолік – через складність зменшується швидкість роботи механізму виводу.

Продукційні моделі використовують найчастіше. У БЗ містяться правила продукцій, а в базі даних (БД) міститься інформація, яка відображає поточний стан розв'язуваної задачі. Ініціалізацію необхідного правила здійснює інтерпретатор або блок керування. Перевага – природність виведення знань. Недолік – процес виводу менш ефективний, ніж в інших системах та важко піддається керуванню.

Логіка числення висловлювань

Семантика ***логіки числення висловів*** дуже близька до природної мови, тому результати формального виведення легко інтерпретувати.

Виводи, одержані на основі числення предикатів I порядку, вже не повною мірою збігаються з семантикою природної мови, їх складніше інтерпретувати. Проте моделі на основі числення предикатів виходять набагато більш компактними і, як наслідок, осяжними.

Числення висловлювань (його також називають пропозиціональним численням – від лат. *propositio* – пропозиція, думка, вислів) – найпростіший розділ математичної логіки, що лежить в основі решти її розділів.

Основними об’єктами розгляду є ***висловлювання*** – це речення (вислів), про яке можна сказати одне з двох: істинне воно або помилкове. Позначаються: I (істина – *true*) і H (неправда – *false*); або числа 1 і 0.

Логічні операції

Назва операції	Зв’язка	Позначення	Відповідна математична операція
Заперечення	НІ	Ø	
Кон’юнкція	І	& ∨ ×	множення
Диз’юнкція	АБО	∨ +	складання
Імплікація	ЯКЩО – ТО	→ ɔ	слідування
Тотожність (еквівалентність)	ТОДІ-І-ТІЛЬКИ-ТОДІ	↔ ≡	

Таблиця істиності

A	B	$\neg A$	$A \,\&\, B$	$A \vee B$	$A \supset B$	$A \equiv B$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Основні закони числення висловлювань

Комутативний закон (закон переміщення):

$a \,\&\, b = b \,\&\, a$; $a \acute{\text{У}} b = b \acute{\text{У}} a$.

Асоціативний закон (сполучний):

$a \,\&\, (b \,\&\, c) = (a \,\&\, b) \,\&\, c$; $a \acute{\text{У}} (b \acute{\text{У}} c) = (a \acute{\text{У}} b) \acute{\text{У}} c$.

Дистрибутивний закон (розділення):

$a \,\&\, (b \acute{\text{У}} c) = a \,\&\, b \acute{\text{У}} a \,\&\, c$; $a \acute{\text{У}} (b \,\&\, c) = (a \acute{\text{У}} b) \,\&\, (a \acute{\text{У}} c)$.

Закони де Моргана

$\neg(a \acute{\text{У}} b) = \neg a \,\&\, \neg b$; $\neg(a \,\&\, b) = \neg\, a \acute{\text{У}} \neg b$; $\neg(\neg a) = a$.

Принцип подвійності: будь-яка теорема булевої алгебри залишається істинною, якщо поміняти місцями операції кон'юнкції і диз'юнкції, константи *true* і *false* у всій теоремі.

Наприклад:

$a \vee \neg\, a = true$ $\quad a \,\&\, \neg\, a = false$

$a \vee true = true$ $\quad a \,\&\, \text{false} = false$

$\neg\,(a \vee b) = \neg a \,\&\, \neg b$ $\quad \neg\,(a \,\&\, b) = \neg a \vee \neg b$

Приведення виразів до нормальних форм

Для вирішення логічних задач важливо вміти знаходити для кожної формули числення висловлювань еквівалентну найпростішу формулу. Існує дві канонічні форми:

кон'юнктивна нормальна форма (КНФ) $D_1 \& D_2 \& \ldots \& D_m$;

диз'юнктивна нормальна форма (ДНФ) $K_1 \vee K_2 \vee \ldots \vee K_m$.

K_i і D_i – кон'юнктивний та диз'юнктивний терми, що становлять кон'юнкцію (диз'юнкцію) змінних і їх заперечень. Змінні – елементарні висловлювання, які називаються ***атомами***. Вони містять лише три операції – Ú, &, Ø, причому заперечення стоїть безпосередньо перед атомом.

Для перетворення формули в КНФ і ДНФ використовуються такі тотожності:

1. $A \supset B = Ø\, A \vee B$
2. $\neg \neg A = A$
3. $\neg A\&B) = \neg A \vee \neg B$
4. $\neg(A \vee B) = \neg A \& \neg B$

$A \vee A = A$

$A \& A = A$

7. $A \equiv B = (A \supset B) \& (B \supset A)$

Непрямі методи виведення

Прямий метод виведення дозволяє за допомогою правил виведення безпосередньо перейти від початкових посилок і аксіом до цілі.

Непрямі методи виведення дозволяють замінити один ланцюжок виведення іншим ланцюжком, у якому початкові посилки і виведення відрізняються від початкових. При цьому з істинності другого виведення випливає істинність першого виведення. Але одержати друге виведення значно легше.

Розрізняють чотири основні стратегії:
- введення припущення;
- метод міркування шляхом розбору випадків;
- доведення від протилежного;
- метод резолюції.

Поняття предиката

Предикат (лат. *praedicatum* – сказане) – те, що висловлюється (стверджується або заперечується) в судженні про об'єкт. Предикат відображає наявність або відсутність тієї чи іншої ознаки у предмета.

Функція *P*, яка набуває одного зі значень, 0 або 1, аргументи якої набувають значення із довільної множини *M*, називається *предикатом P у предметній області M*. Кількість аргументів предиката $P(x_1, x_2, ..., x_k)$ називається його *порядком*. Множина *M*, на якій визначено предикат, називається *предметною областю*. Підмножина *QÉM*, для якої *P*(*x*) істинно, називається *екстенсіоналом*.

Розрізняють предикати: унарні (наприклад, чоловік (*X*)), бінарні (наприклад, батько (*X*, *У*)), тернарні (наприклад, громадянин (*f*, *dr*, *mr*)), *n*-арні предикати. Предикат *n*-го порядку $P(x_1, x_2, ..., x_k)$ визначає *n*-арне відношення *R* у *M*: якщо $P(c_1, c_2, ..., c_k) = 1$, то $(c_1, c_2, ..., c_k)$ знаходиться у відношенні *R*, що визначається цим предикатом. Якщо значення предиката в цій точці дорівнює 0, то ці елементи не знаходяться у відношенні *R*.

Нечіткі множини

Нечітка множина задається парами виду

$$x_1 \mid \mu_X(x_1),$$

де x_1 – елемент нечіткої множини X, а $\mu_X(x_1)$ – ступінь приналежності елемента x_1 до нечіткої множини X. Значення $\mu_X(x_1)$ змінюється в інтервалі [0…1].

Операції над нечіткими множинами

Чітких математичних методів отримання «міри приналежності» деякого елемента до множини немає, застосовуються так звані *методи прийняття рішень*.

Іншим важливим запитанням є спосіб оперування з нечіткими (лінгвістичними) оцінками (мірами приналежності).

Припустимо, є деяка ціль A, досягнення якої залежить від досягнення підцілей A_1 і A_2, причому повне досягнення підцілей можливе лише в окремому випадку.

Множини A і B рівні, якщо

$$\forall x \in M \ \mu_A(x) = \mu_B(x),$$

де M — множина елементів, які є і в A, і в B.

Множини A і B доповнюють одна одну, якщо

$$\forall x \in M \ \mu_A(x) = 1 - \mu_B(x).$$

Перетин A і B:

$$\forall x \in M \ \mu_A(x) \cap \mu_B(x) = \min(\mu_A(x), \mu_B(x)).$$

Об'єднання A і B:

$$\forall x \in M \ \mu_A(x) \cup \mu_B(x) = \max(\mu_A(x), \mu_B(x)).$$

Введені операції доповнення, перетину та об'єднання задовільняють законам комутативності, асоціативності, ідемпотентності, дистрибутивності, дії з константами де Моргана, подвійного заперечення (доповнення) і поглинання, тобто виконуються всі основні закони теорії множин.

На нечітких множинах визначені також операції алгебричного добутку та суми:

$$\forall x \in M \ \mu_{A \cdot B}(x) = \mu_A(x) \cdot \mu_B(x);$$

$$\forall x \in M \ \mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x).$$

Застосування нечітких множин

Переваги так званих *fuzzy*-систем у порівняно із іншими:

- можливість оперувати вхідними даними, заданими нечітко: наприклад, що безупинно змінюються в часі значення (динамічні задачі), значення, що неможливо задати однозначно (результати статистичних опитувань, рекламні компанії тощо);
- можливість нечіткої формалізації критеріїв оцінки і порівняння: оперування критеріями «більшість», «можливе», «переважно» тощо;
- можливість проведення якісних оцінок як вхідних даних, так і виведених результатів: ви оперуєте не лише власне значеннями даних, але їхнім ступенем вірогідності (не плутати з імовірністю) і її розподілом;
- можливість проведення швидкого моделювання складних динамічних систем і їхній порівняльний аналіз із заданим ступенем точності: оперуючи принципами поведінки системи, описаними *fuzzy*-методами, по-перше, не потрібно витрачати багато часу на з'ясування точних значень змінних і складання рівнянь, що їх описують, по-друге, можна оціни- ти різні варіанти вихідних значень.

Логічний висновок у системах із нечітк ою логікою

Функціональність нечіткої системи прийняття рішень визначається такими кроками:

- перетворення чітких вхідних змінних на нечіткі, тобто визначення ступеня відповідності входів кожній із нечітких множин;
- обчислення правил на основі використання нечітких операторів та застосування імплікації для отримання вихідних значень правил;
- агрегування нечітких виходів правил у загальне вихідне значення;
- перетворення нечіткого виходу правил на чітке значення.

Продукційні моделі подання знань

Продукційна модель або модель, що заснована на правилах, дозволяє подати знання у вигляді речень типу:

Якщо (умова), *то* (дія).

Під *умовою* розуміють деяке речення – зразок, за яким здійснюється пошук у базі знань, а під *дією* – дії, що виконуються в разі успішного результату пошуку.

У продукційних моделях процедурна інформація явно виокремлена та описується іншими засобами, ніж декларативна інформація. Замість логічного виведення, що характерне для логічних моделей, в продукційних моделях з'являється *виведення на знаннях*.

$$M = <I, Q, P, A \rightarrow B, N>,$$

де I – ім'я продукції; Q – сфера застосування продукції; P – умова застосовності ядра продукції (логічний вираз типу предиката: $P = 1$ – ядро активується, $P = 0$, то ядро не може бути використане); $A \rightarrow B$ – ядро продукції; N – постумова продукції (активується, коли ядро продукції реалізоване), це дія та умова процедури, які можуть бути виконані після P.

Усі ядра продукції можна розділити на такі групи.

1. *Детерміновані*, де права частина виконується обов'язково (A=>B настане з ймовірністю 1), які, в свою чергу, поділяються на:
   - однозначні (якщо A, то B);
   - альтернативні (якщо A, то частіше B_1, рідше B_2).
2. *Недетерміновані* (якщо A виконується, то можливо B).

Можливі також різні оцінки реалізації ядра продукції.

1. Імовірнісна (якщо A, то з імовірністю P реалізується B).
2. Лінгвістична (мала, менша). Якщо A, то з більшою часткою впевненості B.
3. Прогнозуючого типу (якщо A, то з імовірністю P можна очікувати B).

Переваги продукційних МПЗ: розділення процедурних та декларативних знань; природність виведення знань; гнучкість родовидової ієрархії понять (зміни правил спричиняють зміни в ієрархії).

Недоліки продукційних систем: процес виведення може бути менш ефективний, ніж в інших системах; процес виведення важко піддається керуванню; лінійний ріст обсягу бази знань у міру включення нових фрагментів знань. Якщо використовуються дерева рішень, то зміни відбуваються за логарифмічним законом. Крім того, при накопиченні досить великої кількості продукцій вони починають суперечити одна одній.

Продукційна модель найчастіше застосовується в промислових експертних системах.

Механізм логічного виведення у продукційних системах

Механізм логічного виведення забезпечує формування висновків, сприймаючи факти, що вводяться, як елементи правил, відшукуючи правила, до складу яких входять введені факти, і актуалізуючи ті частини продукцій, яким відповідають введені факти. Механізм логічного виведення виконує функції пошуку в базі правил, послідовного виконання операцій над знаннями та отримання висновків.

Існує два способи проведення виведення:

- *прямий*, що реалізує стратегію від фактів до висновків;
- *зворотній*, коли висуваються гіпотези імовірнісних висновків, які можуть бути підтверджені або спростовані на підставі фактів, які надходять до робочої пам'яті.

Функцією, що реалізує роботу механізму логічного виведення, є рекурсивна процедура зіставлення зі зразком.

Рекурсія (лат. *recurso* – біжу назад, повертаюся) – спосіб розв'язання задач, що полягає в розбитті вихідної задачі на підзадачі. Якщо підзадача є зменшеним варіантом вихідної задачі, то спосіб її розбиття і вирішення ідентичний застосованому до вихідної задачі. Послідовне розбиття приводить до задачі, розв'язуваної безпосередньо. Це рішення є підставою для вирішення підзадачі верхнього рівня і т. д., поки первісна задача не буде вирішена.

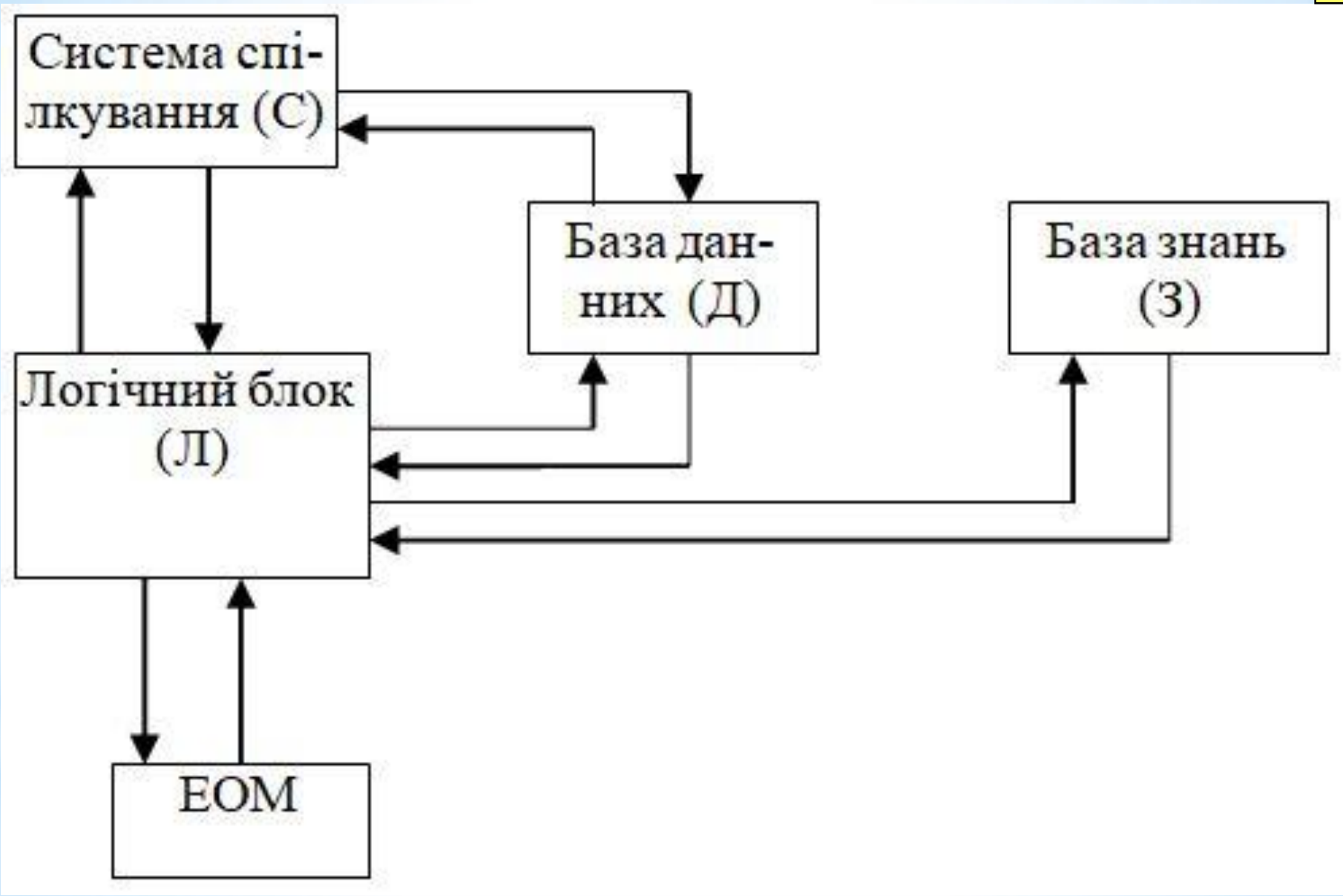
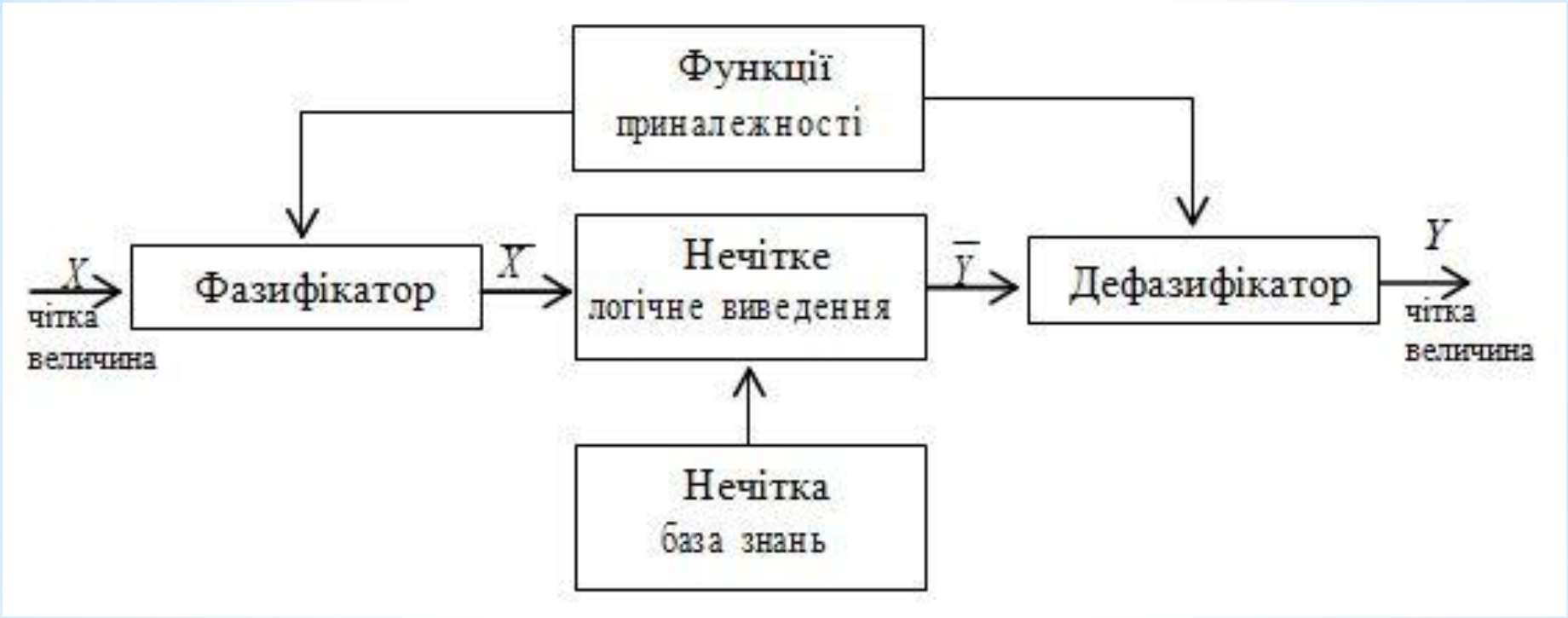


Схема продукційної системи



Етапи нечіткого логічного виведення

Керування пошуком рішень у продукційних системах

У продукційних системах передбачені додаткові можливості з додавання евристичного керування до алгоритму пошуку. Вони включають:
- вибір стратегії (на основі даних або від цілі);
- вибір структури правил;
- вибір стратегій для вирішення конфліктів.

Керування за допомогою вибору стратегії пошуку

Пошук ***на основі даних*** (*пряме виведення*) починається з опису задачі (наприклад, у вигляді набору логічних аксіом), потім із наявних даних виводяться нові знання. Це здійснюється шляхом застосування правил виведення (наприклад, допустимих ходів у грі або інших операцій), що генерують нові стани в поточному описі предметного середовища, і додавання результатів до опису даної задачі. Цей процес продовжується, поки не буде досягнуто цільового стану.

Усі продукційні правила мають форму УМОВА→ДІЯ. Якщо УМОВА відповідає деяким елементам робочої пам’яті, виконується ДІЯ. Якщо продукційні правила сформульовані як логічні слідування і ДІЯ додає твердження в робочу пам’ять, то активізація правила відповідає застосуванню правила *modus ponens*. При цьому на графі створюється новий стан.

Керування пошуком за допомогою структури правил

Структура правил у продукційній системі, включаючи відмінності між умовою і дією, а також порядок перевірки умов, визначає метод дослідження простору.

Числення предикатів, як мова подання в продукційних системах, дозволяє представити одне істинне твердження декількома альтернативними формами. Еквівалентність цих виразів може бути продемонстрована за допомогою таблиці істинності. Хоча ці формулювання логічно еквівалентні, вони не ведуть до однакових результатів, якщо інтерпретуються як продукції (продукційні правила), оскільки реалізація виробничої системи забезпечує певний порядок перевірки відповідності та активізації правил.

Тому вибирається найбільш зручна форма подання правил для конкретної задачі. Таким чином, продукційна система накладає на декларативну мову опису правил процедурну семантику.

Оскільки продукційна система перевіряє правила в певному порядку, програміст може керувати пошуком через структуру і порядок проходження правил у продукційному наборі. Кваліфіковані фахівці кодують найбільш значущі (ключові) евристики, керуючись своїми професійними знаннями. У черговості передумов міститься важлива процедурна інформація, необхідна для успішного вирішення проблеми. Дуже важливо, щоб це формулювання зберігалося при написанні програми.

Керування пошуком через вирішення конфліктів

Продукційні системи (як і всі системи, засновані на знаннях) дозволяють представляти евристики безпосередньо в правилах, що описують знання. Крім того, вони пропонують інший метод евристичного керування – через *вирішення конфліктів*. Найпростіша стратегія зводиться до того, щоб вибирати перше відповідне правило з робочої пам'яті. Однак для вирішення конфліктів потенційно може бути застосована будь-яка стратегія.

Рефракція означає, що після активізації правила воно не може бути використане знову, поки не зміняться елементи робочої пам'яті, відповідні його умовам. Це перешкоджає зацикленням.

Новизна. Стратегія новизни віддає перевагу правилам, умови яких відповідають зразкам, доданим до робочої пам'яті останніми. Це дозволяє зосередити пошук на одній лінії міркування.

Специфічність. Згідно з цією стратегією доцільніше використовувати більш конкретне, а не більш загальне правило. Одне правило більш специфічне (конкретне) ніж інше, якщо воно містить більше умов, а значить, відповідає меншій кількості зразків у робочій пам'яті.

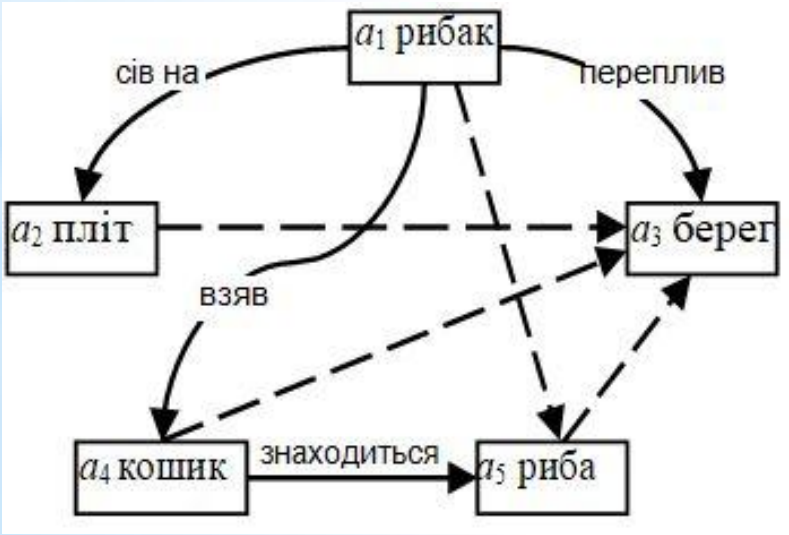
Семантичні сітки як модель подання знань

Основна ідея підходу до подання знань у вигляді мережевих моделей полягає у тому, щоб розглядати проблемне середовище як сукупність об'єктів та зв'язків між ними. Тобто, ***семантична сітка*** – це орієнтований граф, вершини якого – *поняття*, а дуги – *від ношення* між ними.

Така конструкція може бути описана мовою теорії множин:

$$H = \{I, C_1, C_2, \ldots, C_n, \Gamma\},$$

де I – множина інформаційних одиниць; $C_1, C_2, \ldots, C_n$ – множина типів зв'язків між інформаційними одиницями; Γ – множина відображення між інформаційними одиницями.



Приклад семантичної сітки

Залежно від типів зв’язків, між інформаційними одиницями, які використовуються в моделі, розрізняють такі типи мереж.

Класифікуючі мережі – застосовують відношення структуризації. Дозволяють у базах знань вводити різні ієрархічні відношення між інформаційними одиницями.

Функціональні мережі – функціональні відношення між інформаційною одиницею. Прикладом є обчислення, тому можуть виступати ще як обчислювальні. Дозволяють описувати процедури «обчислень» одних інформаційних одиниць через інші.

Сценарії – часто використовуються казуальні (причинно - наслідкові) відношення між інформаційними одиницями. Крім того, можуть зустрічатися відношення типів: «засіб – результат», «знаряддя – дія».

Якщо в мережевій моделі застосовують відношення всіх типів, то таку мережу називають ***семантичною.***

Типи об'єктів

Основні типи об'єктів:

Поняття є константами або параметрами предметної області, описуваної семантичною сіткою, і зазвичай вказують предмети й абстракції.

Події представляють собою дії, які можуть відбутися.

Методи подання подій:

- *глибинно-відмінникові семантичні відносини*, які вказують характеристики та діючих осіб даної події;

- *зміни, які може спричинити подія*. Результатом події є також деяка ситуація, яку можна визначити як зразок у деякій про- цедурі, що описує таким чином послідовність дій, що при- водить до цієї ситуації.

Властивості використовуються для уточнення або модифікації понять, подій та інших властивостей. У разі понять властивості можуть бути особливостями, рисами або характеристиками. У разі подій властивості описують деякі загальні, універсальні, постійні характеристики, наприклад, місце, час, тривалість тощо.

Властивість є бінарним відношенням, яке відображає область свого визначення, тобто вершини, до яких властивість застосовується, в область значень, тобто значення, якого властивість може набувати.

Можливе розширення поняття властивості від бінарного до тернарного і багатомісного відношення. При цьому додаткові характеристики властивості пов'язуються з вершиною властивості дугами, поміченими «щодо».

Вершини, що входять до семантичної сітки, незалежно від їх типу розділені на два класи:

загальні – поняття, події та властивості загального характеру, що описують у сукупності закони, які діють в предметній області;

фактуальні – окремі випадки загальних об’єктів, які описують конкретні прояви зазначених вище законів, або просто деякі факти.

Іноді для підвищення ефективності виведення вводяться *процедурні* вершини.

Важливим є подання в семантичних сітках загального вигляду елементів пізнавальної активності, наприклад, у вигляді логічних міркувань і прикладних програм. Ці елементи можуть бути реалізовані за наявності в семантичній сітці віртуальних відношень, які дозволяють у системі подання знань реалізувати інформаційно - логічний режим функціонування систем подання знань.

Логічне виведення на семантичних сітках

Виведення на семантичних сітках відрізняється повнотою та має концептуальну інтерпретацію. Послідовне застосування правил виведення може привести до утворення так званих «ланцюжків виведення», які в окремих випадках можуть досягати значної довжини.

Особливий тип генерації виведення, що використовується в семантичних сітках, – це так званий *метод «активності, що розповсюджується, і техніки перетинань»*. Цей метод відіграє важливу роль в обробці контекстів. Процес здійснюється побудовою ланцюжків виведення на основі введених висловлювань в усіх напрямках доти, поки не виявиться перетинання в сітці.

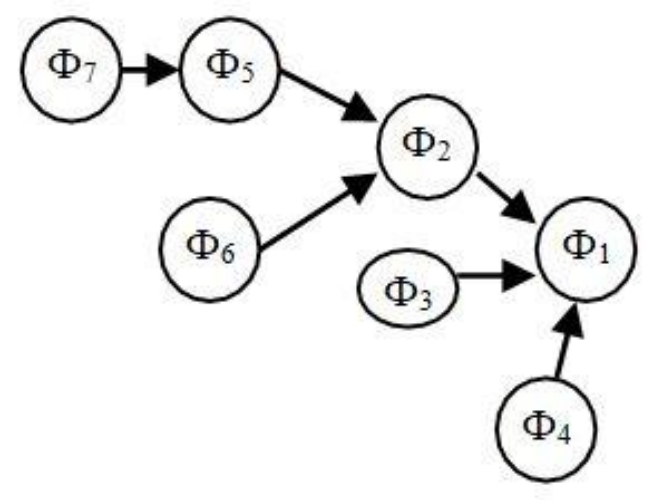
Ці методи подібні до тих, що використовуються в системах подання знань на базі логіки предикатів: розширення семантичних сіток за рахунок введення в них знань про застосування; тематична структуризація; предметно-орієнтована ієрархія, розробка глобальних схем подання, в яких використовувалися б семантичні сітки, що містять локальні знання.

Сценарії

У системі подання знань можуть бути стереотипи знання, що описують відомі стандартні ситуації. Окремим видом семантичних сіток є сценарії. ***Сценарії*** – це формалізований опис стандартної послідовності взаємопов'язаних фактів, що визначають типову ситуацію предметної області.

Сценарій включає такі компоненти:

- *початкові умови*, які мають бути істинними при виклику сценарію;
- *результати* або факти, які є істинними, коли сценарій завершується;
- *припущення*, які підтримують контекст сценарію;
- *ролі* є діями, які виконують окремі учасники;
- *сцени*, які є часовими аспектами сценарію.



Приклад сценарію

Фрейми: основні поняття, структура фрейму, фреймові системи

Фрейм (англ. *frame* – каркас або рамка) запропонований М. Мінським у 70-ті рр. XX ст. як структура знань для сприйняття просторових сцен. Ця модель, як і семантична сітка, має глибоке психологічне обґрунтування. Фрейм – одиниця подання знань, деталі якої можуть змінюватися відповідно до поточної ситуації.

Фрейми часто використовують як структуру для подання стереотипних ситуацій. Структура фрейму така, що він складається з характеристик описуваних ситуацій та їх значень, які називаються відповідно ***слотом*** та ***заповнювачем слоту***. Ця структура, доки вона не заповнена якимись значеннями, називається ***протофреймом***.

Ім’я фрейму:
Ім’я слоту 1 (Значення слоту 1);
Ім’я слоту 2 (Значення слоту 2);
… … …
Ім’я слоту *k* (Значення слоту k).

Значення слоту: число, математичний вираз, текст природною мовою, програма для ЕОМ, правила виведення, посилання на інші слоти даного фрейму або інших фреймів.

При заповненні фрейму йому та слотам присвоюються конкретні імена та відбувається заповнення слотів. З протофрейму отримують ***фрейм-екземпляр*** (экзофрейм).

Фрейми утворюють ієрархію. Ієрархія у фреймових моделях породжує єдину багаторівневу структуру, що описує або об'єкт, якщо слоти описують лише властивості об'єкта, або ситуацію чи процес, якщо окремі слоти є іменами процедур, приєднаних до фрейму і спричинених при його актуалізації.

Формально фрейм – це тип даних виду:

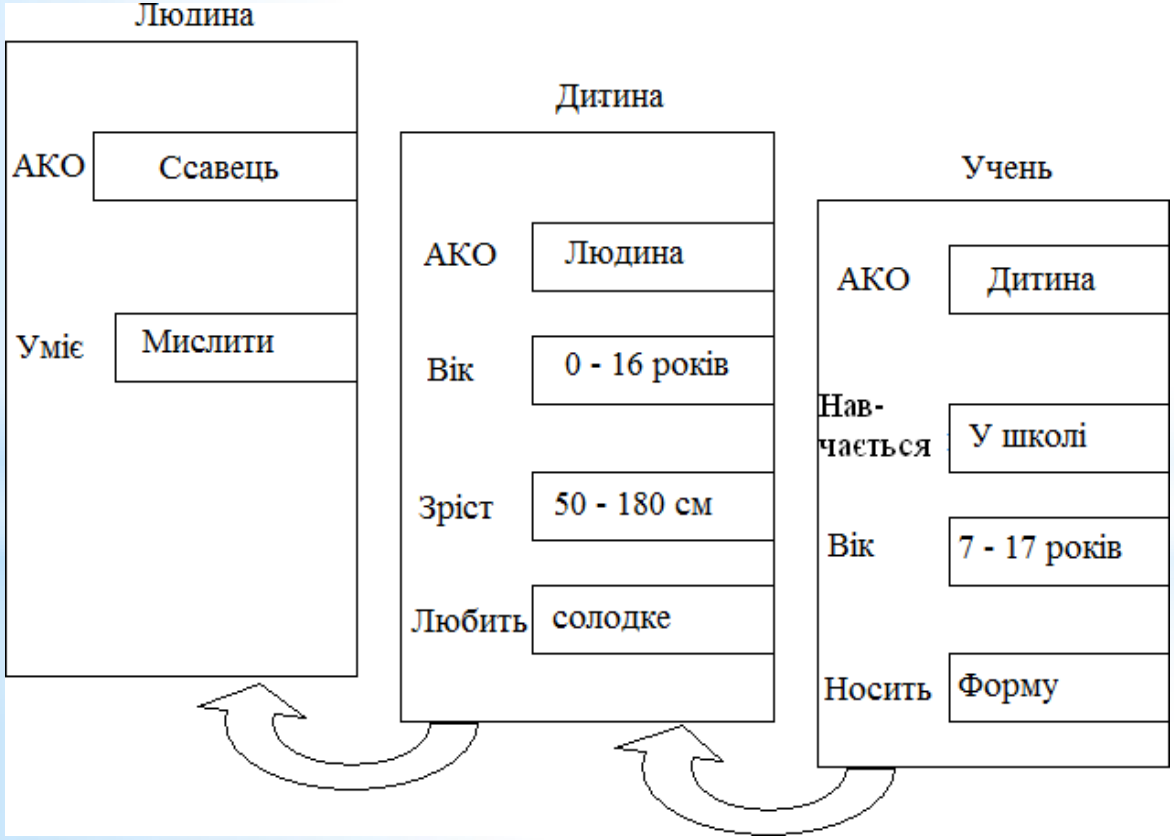
$$F=<N, S_1, S_2, S_3>,$$

де N – ім'я об'єкта; S_1 – множина слотів, що містять факти, які визначають декларативну семантику фрейму; S_2 – множина слотів, що забезпечують зв'язки з іншими фреймами (каузальні, семантичні тощо); S_3 – множина слотів, яка забезпечує перетворення, що визначають процедурну семантику фрейму.

Фрейми поділяються на:

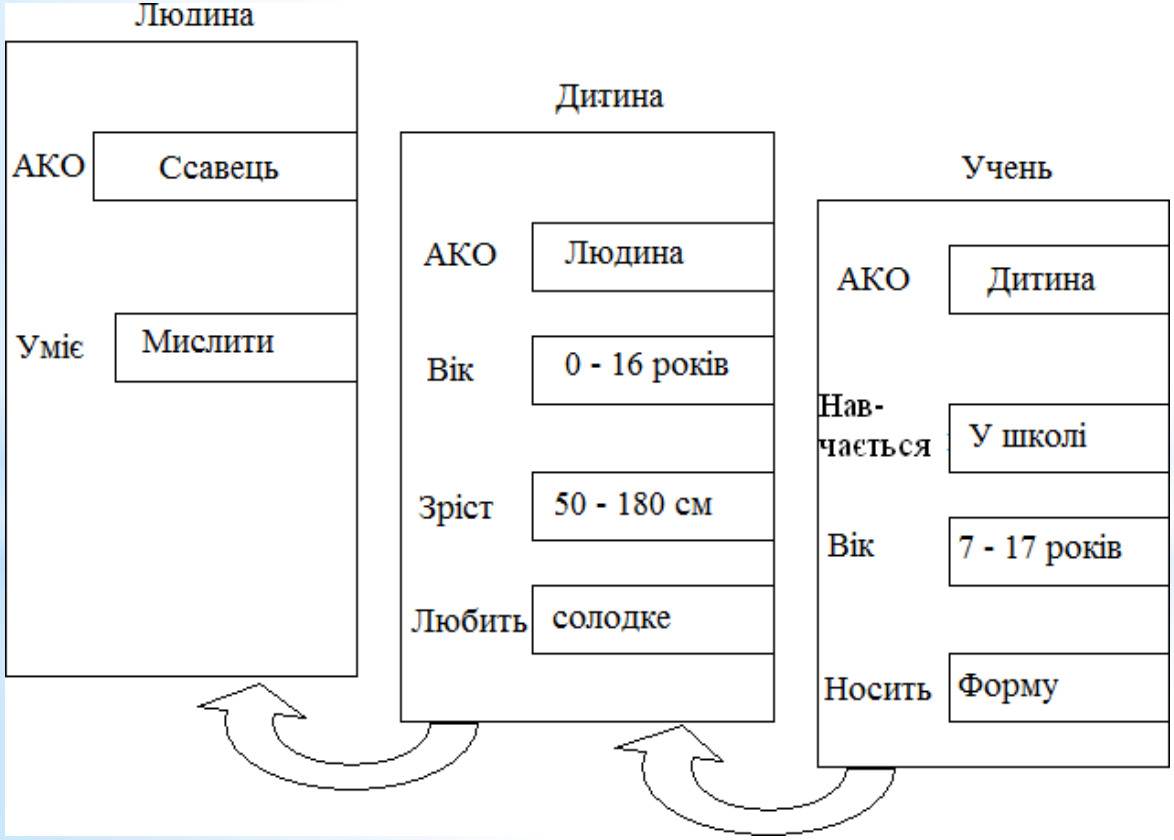
- *фрейм-екземпляр* – конкретна реалізація фрейму, яка описує поточний стан у предметній області;
- *фрейм-зразок* – шаблон для опису об'єктів або допустимих ситуацій предметної області;
- *фрейм-клас* – фрейм верхнього рівня для подання сукупності фреймів-зразків.

У мережевому і фреймовому поданні знань найчастіше використовуються відносні зв'язки типу IS-A і АКО. Зв'язок типу IS-A означає, що окремий об'єкт «є екземпляром» певного класу. Зв'язок типу АКО (A-KIND-OF) визначає відношення між родовими класами. Загальний клас, на який указує стрілка АКО, називається суперкласом. У випадку якщо суперклас має зв'язок АКО, що указує на інший вузол, то він, разом з тим, є класом суперкласу.



Мережа фреймів

У мережевому і фреймовому поданні знань найчастіше використовуються відносні зв'язки типу IS-A і АКО. Зв'язок типу IS-A означає, що окремий об'єкт «є екземпляром» певного класу. Зв'язок типу АКО (A-KIND-OF) визначає відношення між родовими класами. Загальний клас, на який указує стрілка АКО, назива- ється суперкласом. У випадку якщо суперклас має зв'язок АКО, що указує на інший вузол, то він, разом з тим, є класом суперкласу.



Мережа фреймів

Переваги фреймових систем

1. Дозволяють маніпулювати як декларативними, так і процедурними знаннями, тобто значення будь-якого слоту може бути обчислене за допомогою відповідних процедур або визначене евристичним методом.
2. Економне розміщення бази знань у пам’яті комп’ютера.

Недоліки фреймових систем:

1. Відносно висока складність цих систем, що призводить до зменшення швидкості роботи механізму виведення.
2. Доцільна область застосування там, де родовидові зв’язки змінюються нечасто та предметна область має небагато виключень.