**МІНІСТЕРСТВО КУЛЬТУРИ ТА**
**ІНФОРМАЦІЙНОЇ ПОЛІТИКИ УКРАЇНИ**
**ХАРКІВСЬКА ДЕРЖАВНА АКАДЕМІЯ КУЛЬТУРИ**
**Факультет соціальних комунікацій і**
**музейно-туристичної діяльності**
**Кафедра інформаційних технологій**

# КОНСПЕКТ ЛЕКЦІЙ
# З ДИСЦИПЛІНИ
# «ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ»

**для здобувачів**
**першого (бакалаврського) рівня вищої освіти**

**Галузі знань** 02 Культура і мистецтво

**Спеціальності** 029 Інформаційна, бібліотечна та архівна справа

**Освітня програма** Інформаційна та документаційна діяльність

Харків, 2022

**УДК 004.89**
**І-73**

Рекомендовано до видання науково-методичною радою ХДАК, протокол № 6 від 25.01.2023 р.

**Укладач:** ***Брусенцев В. О.,*** доцент, кандидат технічних наук, доцент кафедри інформаційних технологій

**Рецензенти:**
***Асєєв Г. Г.,*** професор, доктор технічних наук, завідувач кафедри інформаційних технологій ХДАК
***Ткачук М. А.,*** доцент, доктор технічних наук, завідувач кафедри інформаційних технологій теорії і систем автоматизованого проектування механізмів і машин Національного технічного університету «Харківський політехнічний інститут»

І-73 Інтелектуальні системи : конспект лекцій до курсу для здобувачів першого (бакалавр.) рівня вищої освіти галузі знань 02 «Культура і мистецтво», спец. 029 «Інформаційна, бібліотечна та архівна справа» освітньої програми «Інформаційна та документаційна діяльність» / Харків. держ. акад. культури ; [уклад. В. О. Брусенцев]. – Харків : ХДАК, 2022. – 182 с.

**УДК 004.89(042.4)**

## ЗМІСТ

# ВСТУП

В основі будь-якої організованої діяльності лежить інформація. Різні інформаційні ресурси необхідні в кожній сфері людської діяльності. Для людського суспільства характерне прагнення до пізнання і перетворення навколишнього світу шляхом придбання інформації, її обробки, зберігання, використання та передачі. Протягом своєї історії людство накопичувало знання, отримані зі спостережень і досвіду, намагалося зберегти їх і передати наступним поколінням, робило вірні і невірні висновки із сукупності доступних знань і було нестримно в спробах застосування отриманих знань на практиці.

Історична фаза розвитку людства, в якій головними продуктами виробництва є інформація і знання, призвела до формування інформаційного суспільства. Сучасний етап розвитку дуже тісно пов'язаний із стрімким технологічним зростанням в усіх сферах діяльності. Збільшилася роль інформації і знань в житті суспільства, збільшилася доля інформаційних комунікацій, продуктів та послуг, розширився глобальний інформаційний простір та доступ до його інформаційних ресурсів.

За останні 50 років з моменту появи електронних обчислювальних машин почали розвиватися методи та способи обробки і подання інформації. З 70-х років минулого століття велися розробки автоматизованих інформаційних систем, систем управління, систем підтримки прийняття рішень, які сприяли розвитку методів інтелектуалізації розроблюваних систем. Інтелектуалізація вивявилася дуже перспективним напрямом вдосконалення автоматизованих систем і досі набирає великої важливості. Інтелектуальна підтримка прийняття управлінських рішень набувають значної актуальності, враховуючи сучасні інформаційно-комунікаційні технології та значне поширення комп'ютерної і мобільної техніки.

Наразі іде постійне вдосконалення методів та алгоритмів опрацювання інформації, вдосконалення автоматизованих інформаційних систем і технологічних засобів, розвиток систем штучного інтелекту.

# ЛЕКЦІЯ 1. ШТУЧНИЙ ІНТЕЛЕКТ: ЗАГАЛЬНІ ПОНЯТТЯ ТА ПІДХОДИ

## 1.1. Історія розвитку штучного інтелекту як науки

Термін «*штучний інтелект*» (*artificial intelligence*) був запропонований у 1956 р. Слово *intelligence* означає «*уміння міркувати розумно*», а зовсім не «*інтелект*», для якого є термін *intellect*. Під Штучним Інтелектом (ШІ) розуміється область досліджень, в якій ставиться задача вивчення розумної поведінки (у людей, тварин і машин) і спроба знайти способи моделювання подібної поведінки в будь-якому типі штучно створеного механізму, а також технологій програмування самої такої поведінки.

Незважаючи на те, що терміну більше півстоліття єдиного визначення не існує. Різні дослідники по-різному визначають цю науку, в залежності від свого погляду на неї і працюють над створенням систем, які

– думають подібно людям;
– думають раціонально;
– діють подібно людям;
– діють раціонально.

Синтезуючи десятки визначень штучного інтелекту з різних джерел як робоче визначення можна запропонувати наступне.

*Штучний інтелект* – це один з напрямків інформатики, метою якого є розробка апаратно-програмних засобів, що дозволяють користувачеві-непрограмісту ставити і вирішувати свої задачі, що традиційно вважаються інтелектуальними, спілкуючись з ЕОМ на обмеженій підмножині природної мови.

При відтворенні розумних міркувань і дій виникають певні труднощі. По-перше, в більшості випадків, виконуючи якісь дії, людина не усвідомлює, як це робить, ніхто не знає точний спосіб, метод або алгоритм розуміння тексту, розпізнавання обличь, доказів теорем, рішення задач, віршування і т.ін. По-друге, на сучасному рівні розвитку комп'ютер занадто далекий від людського рівня компетентності і працює за іншими принципами.

Штучний інтелект завжди був міждисциплінарною наукою, будучи одночасно і наукою і мистецтвом, і технікою і психологією. Методи штучного інтелекту різноманітні. Вони активно

запозичуються з інших наук, адаптуються і змінюються під вирішувану задачу. Для створення інтелектуальної системи необхідно залучати фахівців з прикладної області, у рамках штучного інтелекту співпрацюють лінгвісти, нейрофізіологи, психологи, економісти, інформатики, програмісти і т.д. Інженерні методи і навички в області штучного інтелекту називають *технологією* або *інженерією* знань (*knowledge engineering*).

Основними додатками штучного інтелекту є:

– обробка природної мови;
– автоматизація програмування;
– керування роботами;
– інформаційний пошук;
– експертні системи;
– розпізнавання образів та обробка зображень;
– системи прийняття рішень.

Штучний інтелект є в деякому сенсі наукою майбутнього, в якій немає жорсткого поділу по областям і ясно видно зв'язок між окремими дисциплінами, які лише відображають певну грань пізнання.

Формування штучного інтелекту як науки відбулося у 1956 році. Джон Маккарті, Марвін Лі Мінський, Клод Шеннон і Натаніель Рочестер організували двомісячний семінар в Дартмуті для американських дослідників, що займаються теорією автоматів, нейронними мережами, інтелектом. Хоча дослідження в цій області вже активно велися, але саме на цьому семінарі з'явилися термін і окрема наука – *штучний інтелект*.

Одним із засновників теорії штучного інтелекту вважається відомий англійський учений Алан Тюрінг, який в 1950-му році опублікував статтю «*Обчислювальні машини й розум*» (перекладену на російську мову під назвою «*Может ли машина мыслить?*»). Саме в ній описувався «*тест Тюрінга*», який став класичним і дозволив оцінити «*інтелектуальність*» комп'ютера за його здатністю до осмисленого діалогу з людиною.

Перші десятиліття розвитку (1952–1969) штучного інтелекту були сповнені успіхів і ентузіазму. Дослідницьким полігоном для розвитку методів ШІ на першому етапі з'явилися всілякі ігри, головоломки, математичні задачі. Деякі з цих задач стали класичними в літературі з ШІ (задача про мавпу і банани, задача

про місіонерів і людожерів, Ханойські вежі, гра у 15 та інші). Вибір таких задач обумовлювався простотою і ясністю проблемного середовища (середовища, в якому розгортається рішення задачі), її відносно малою громіздкістю, можливістю досить легкого підбору і навіть штучного конструювання «*під метод*». Аллен Ньюелл, Дж. Шоу і Герберт А. Саймон створили програму для гри в шахи, на основі методу, запропонованого в 1950 році Клодом Шенноном, формалізованого Аланом Тюрінгом і промодельованого ним же вручну. До роботи була залучена група голландських психологів під керівництвом А. де Гроота, які вивчали стилі гри видатних шахістів.

У 1956 році цим колективом була створена мова програмування IPL (*Information Processing Language* – «*мова обробки інформації*») – практично першу символьну мову обробки списків і написана перша програма «*Логік-Теоретик*», призначена для автоматичного доказу теорем в обчисленні висловлювань. Цю програму можна віднести до перших досягнень в області штучного інтелекту. У 1959 році цією ж групою написана програма GPS (*General Problem Solver*) – універсальний вирішувач задач, сформульованих на мові хорновських диз'юнктів. Вона могла вирішувати ряд головоломок, обчислювати невизначені інтеграли, вирішувати деякі інші задачі. Ці результати привернули увагу фахівців в області обчислень і з'явилися програми автоматичного доведення теорем з планіметрії і розв'язання алгебраїчних задач. З 1952 року Артур Семюель написав ряд програм для гри в шашки, які грали на рівні добре підготовленого любителя, причому одна з його ігор навчилася грати краще, ніж її творець.

У 1958 році Джон Маккарті визначив нову мову високого рівня Lisp (від англ. *LISt Processing language* – «*мова обробки списків*»), яка стала домінуючою для штучного інтелекту. Перші нейромережі з'явилися в кінці 50-х років. У 1957 році Френком Розенблаттом була зроблена спроба створити систему, що моделює людське око і його взаємодію з мозком – *перцептрон*.

У Стенфордському університеті, Стенфордському дослідницькому інституті і деяких інших місцях були розроблені експериментальні роботи, що функціонують у лабораторних умовах. Проведення цих експериментів показало необхідність вирішення кардинальних питань, пов'язаних з проблемою подання знань про середовище функціонування, і одночасно недостатню

дослідженість таких проблем, як зорове сприйняття, побудова складних планів поведінки в динамічних середовищах, спілкування з роботами на природній мові.

У США з'явилися перші комерційні системи, засновані на знаннях, – *експертні системи*. Відбувається комерціалізація штучного інтелекту. Ростуть щорічні капіталовкладення та інтерес до самонавчальних систем, створюються промислові експертні системи. Розробляються методи подання знань. Перша експертна система була створена Едвардом Фейгенбаумом в 1965 році. Але до комерційного прибутку було ще далеко. Лише в 1986 році перша комерційна система R1 компанії DEC дозволила заощадити приблизно 40 мільйонів доларів на рік. У 1988 році компанією DEC було розгорнуто 40 експертних систем. У компанії Du Pont застосовувалося 100 систем, і економія становила приблизно 10 мільйонів на рік.

У середині 70-х рр. почався третій етап досліджень систем ШІ. Його характерною рисою стало зміщення центру уваги дослідників зі створення автономно функціонуючих систем, самостійно розв'язуючих в реальному середовищі поставлені перед ними задачі, до створення людино-машинних систем, інтегруючих в єдине ціле інтелект людини і здібності обчислювальної машини для досягнення спільної мети – вирішення задачі, поставленої перед інтегральною людино-машинною обчислювальною системою. Таке зміщення обумовлювалося двома причинами.

1. До цього часу з'ясувалося, що навіть прості на перший погляд задачі, які постають перед інтегральним роботом під час його функціонування в реальному часі, не можуть бути вирішені методами, розробленими для експериментальних задач в спеціально сформованих проблемних середовищах.

2. Стало ясно, що поєднання доповнюючих одна одну можливостей людини та ЕОМ дозволяє обійти гострі кути шляхом перекладання на людину тих функцій, які поки ще не доступні для ЕОМ. На перший план висувається не розробка окремих методів машинного рішення задач, а розробка методів, засобів, що забезпечують тісний контакт людини і обчислювальної системи протягом всього процесу рішення задачі з можливістю оперативного внесення людиною змін в ході цього процесу.

У 1981 році Японія приступила до розробки комп'ютера 5-го

покоління, заснованого на знаннях – 10-річного плану по розробці інтелектуальних комп'ютерів на базі Prolog. 1986 рік став роком відродження інтересу до нейронних мереж. У 1991 році Японія припиняє фінансування проекту комп'ютера 5-го покоління і починає проект створення комп'ютера 6-го покоління – *нейрокомп'ютера*.

У 1997 році комп'ютер «*Діп Блю*» переміг у грі в шахи чемпіона світу Гаррі Каспарова, довівши можливість того, що штучний інтелект може зрівнятися або перевершити людину в ряді інтелектуальних задач (нехай і в обмежених умовах).

Штучний інтелект переслідує безліч цілей. Однією з основних задач штучного інтелекту є *створення повного наукового опису інтелекту людини, тварини і машини та обчислення принципів, спільних для всіх трьох*. Моделювання розуму необхідно для вирішення задач. До інтелектуальних задач можна віднести всі задачі, алгоритм знаходження яких невідомий.

Але, наприклад, перебір всіх можливих комбінацій також є алгоритмом. Застосувати його на практиці, на жаль, на сучасному рівні розвитку техніки до більшості задач неможливо (сучасна ЕОМ не зможе згенерувати всі перестановки великої кількості різних предметів). Комбінаторний вибух, з яким зіткнулися дослідники вже в ранніх дослідженнях – приклад цього. У таких випадках, коли незначне збільшення вхідних даних задачі веде до зростання кількості повторюваних дій в степеневій залежності, говорять про неполіноміальні алгоритми, які характеризуються тим, що кількість операцій в них зростає в залежності від числа входів за законом, близьким до експоненти. Подібні алгоритми рішення мають надзвичайно велике коло задач, особливо комбінаторних проблем, пов'язаних із знаходженням поєднань, перестановок, розміщень будь-яких об'єктів.

Вирішенням подібних задач і займається штучний інтелект. Дослідники вивчають процеси мислення, розумну поведінку для того, щоб знайти шляхи вирішення подібних задач, тому що людина в своїй діяльності стикається з ними досить часто і успішно вирішує.

Хоча до цих пір багато задач не вирішено, певні досягнення в цій галузі є. Дослідники використовували різні підходи і методи, щоб отримати результат. В кінці 50-х років народилася *модель*

*лабіринтового пошуку і з'явилася теорія розпізнавання образів*, як наслідок початку використання ЕОМ для вирішення необчислювальних задач. Початок 60-х років називають епохою евристичного програмування, коли використовувалися стратегії дій на основі відомих, заздалегідь заданих евристик. Евристики дозволяють скорочувати кількість розглянутих варіантів. В середині 60-х років до вирішення задач стали активно підключати методи математичної логіки. З середини 70-х років дослідники стали приділяти увагу системам, що базуються на експертних знаннях. Такі системи застосовують до слабо формалізованих задач.

Неформалізовані задачі зазвичай володіють наступними особливостями:

– помилковістю, неоднозначністю, неповнотою і суперечливістю вхідних даних;

– помилковістю, неоднозначністю, неповнотою і суперечливістю знань про проблемну область і розв'язувану задачу;

– великою розмірністю простору рішення, тобто перебір при пошуку рішення досить великий;

– динамічно змінюючимися даними і знаннями.

В даний час в дослідженнях з ШІ можна виділити шість основних напрямків:

– подання знань;
– маніпулювання знаннями і пошук рішень;
– системи спілкування;
– системи сприйняття;
– машинне навчання;
– моделювання розумної поведінки.

## 1.2. Напрями та підходи до досліджень в області штучного інтелекту

Незабаром після визнання штучного інтелекту окремою галуззю науки відбувся поділ його на два напрямки: *нейрокібернетика* та *кібернетика чорної скрині*. Ці напрямки розвиваються практично незалежно, істотно розрізняючись як в методології, так і в технології.

*Нейрокібернетики* взяли за основу структуру і принципи функціонування єдиного створеного природою пристрою, здатного

міркувати – мозку. Клітини мозку називаються нейронами, звідси і назва напрямку. Дослідники вважали, що, змоделювавши мозок, зможуть відтворити і його роботу.

Дослідники напрямку *кібернетика чорної скрині* дотримувалися думки, що не важливо за якими принципами працює пристрій, які засоби і методи лежать в його основі, головне, імітувати функції мозку, навіть якщо крім результату це не матиме нічого спільного з природним розумом. В даний час стали помітні тенденції до об'єднання цих частин знову в єдине ціле. Стало з'являтися безліч гібридних методів і систем, наприклад, експертна система на базі нейронної мережі або нейронна мережа, що навчається за генетичним алгоритмом.

Дослідники, що моделюють лише окремі функції інтелекту, наприклад, розпізнавання образів, синтез мови, прийняття рішень, працюють у рамках напряму *слабкий* штучний інтелект. Спроби відтворити роботу інтелекту в повному обсязі відносяться до напрямку *сильний* штучний інтелект. Усі основні досягнення в області штучного інтелекту відносяться до слабкого штучного інтелекту.

Крім цього, виділяють *спадний* (семіотичний) і *висхідний* (біологічний) підходи. Спадний підхід передбачає моделювання високорівневих психічних процесів, таких як мислення, мова, емоції і т.д. Висхідний підхід досліджує інтелектуальну поведінку систем на базі більш дрібних «неінтелектуальних» елементів. Нейронні мережі та еволюційне моделювання відносяться до висхідного підходу.

Інтелектуальні системи розробляються із залученням різних засобів і методів. Існує чотири основні підходи до їх побудови: *логічний*, *структурний*, *еволюційний* та *імітаційний*.

Основою для даного логічного підходу служить Булева алгебра. Така інтелектуальна система являє собою машину доведення теорем. При цьому вихідні дані зберігаються в базі даних у вигляді аксіом, правила логічного висновку як відношення між ними. Крім того, кожна така машина має блок генерації цілі, і система виведення намагається довести дану мету як теорему. Якщо мета доведена, то трасування застосованих правил дозволяє отримати ланцюжок дій, необхідних для реалізації поставленої мети. Потужність такої системи визначається можливостями

генератора цілей і машиною доведення теорем. Для більшості логічних методів характерна велика трудомісткість, оскільки під час пошуку доказів можливий повний перебір варіантів. Тому даний підхід вимагає ефективної реалізації обчислювального процесу, і хороша робота зазвичай гарантується при порівняно невеликому розмірі бази даних.

Під структурним підходом маються на увазі спроби побудови інтелектуальної системи шляхом моделювання структури людського мозку, тобто системи, побудованої в рамках напряму нейрокібернетика.

При побудові інтелектуальної системи за допомогою еволюційного підходу основна увага приділяється побудові початкової моделі, і правилам, за якими вона може змінюватися (еволюціонувати). Причому модель може бути складена з найрізноманітніших методів, це може бути і нейронна мережа і набір логічних правил і будь-яка інша модель. На підставі перевірки моделей відбирає найкращі з них, на підставі яких по самим різним правилам генеруються нові моделі, з яких знову обираються найкращі і т.д.

Імітаційний підхід використовується в рамках напряму кібернетика чорної скрині. Інтелектуальні системи при такому підході повинні моделювати якусь інтелектуальну функцію, тобто встановити необхідну відповідність між входами і виходами системи.

Тематика штучного інтелекту охоплює величезний перелік наукових напрямків, починаючи з таких задач загального характеру, як навчання і сприйняття, і закінчуючи такими спеціальними задачами, як гра в шахи, доказ математичних теорем, створення поетичних творів і діагностика захворювань. У штучному інтелекті систематизуються і автоматизуються інтелектуальні задачі, і тому ця область стосується будь-якої сфери інтелектуальної діяльності людини.

Серед безлічі напрямків штучного інтелекту є кілька ведучих, які в даний час викликають найбільший інтерес у дослідників і практиків.

*1. Подання знань і розробка систем, заснованих на знаннях.* Це основний напрямок в області розробки систем штучного інтелекту. Основною метою побудови таких систем є виявлення,

дослідження та застосування знань висококваліфікованих експертів для вирішення складних задач, що виникають на практиці. При побудові систем, заснованих на знаннях (СЗЗ), використовуються знання, накопичені експертами у вигляді конкретних правил вирішення тих чи інших задач. Цей напрямок має на меті імітацію людського мистецтва аналізу неструктурованих і слабоструктурованих проблем. У даній області досліджень здійснюється розробка моделей подання, вилучення і структурування знань, а також вивчаються проблеми створення баз знань (БЗ), що утворюють ядро СЗЗ. Окремим випадком СЗЗ є експертні системи (ЕС).

*2. Програмне забезпечення систем штучного інтелекту.* В рамках цього напрямку розробляються спеціальні мови для вирішення інтелектуальних задач, в яких наголос робиться на перевазі логічної і символьної обробки над обчислювальними процедурами. Мови орієнтовані на символьну обробку інформації – LISP, SMALLTALK, РЕФАЛ, мови логічного програмування (PROLOG), мови представлення знань (OPS 5, KRL, FRL), інтегровані програмні середовища, що містять арсенал інструментальних засобів для створення систем ШІ (KE, ARTS, GURU, G2), а також оболонки експертних систем (BUILD, EMYCIN, EXSYS Professional, ЕКСПЕРТ), які дозволяють створювати прикладні ЕС, не вдаючись до програмування.

*3. Розробка природно-мовних інтерфейсів і машинний переклад.* Починаючи з 50-х років, однією з популярних тем досліджень в галузі штучного інтелекту є комп'ютерна лінгвістика, і, зокрема, машинний переклад. Ідея машинного перекладу виявилася зовсім не так проста, як здавалося першим дослідникам і розробникам. Системи машинного перекладу з однієї природної мови на іншу забезпечують швидкість і систематичність доступу до інформації, оперативність і однаковість перекладу великих потоків, як правило, науково-технічних текстів. Системи машинного перекладу будуються як інтелектуальні системи, оскільки в їх основі лежать БЗ в певній предметній області і складні моделі, що забезпечують додаткову трансляцію «вхідна мова оригіналу – мова сенсу – мова перекладу». Вони базуються на структурно-логічному підході, що включає послідовний аналіз і синтез природно-мовних повідомлень. Крім того, в них

здійснюється асоціативний пошук аналогічних фрагментів тексту і їх перекладів в спеціальних базах даних (БД). Даний напрямок охоплює також дослідження методів і розробку систем, що забезпечують реалізацію процесу спілкування людини з комп'ютером на природній мові (так звані системи ПМ-спілкування).

*4. Інтелектуальні роботи.* Роботи – це електротехнічні пристрої, призначені для автоматизації людської праці. Виділяють кілька поколінь роботів:

I покоління. Роботи з жорсткою схемою управління. Практично всі сучасні промислові роботи належать до першого покоління. Фактично це програмовані маніпулятори.

II покоління. Адаптивні роботи з сенсорними пристроями. Є зразки таких роботів, але в промисловості вони поки використовуються мало.

III покоління. Самоорганізовані або інтелектуальні роботи. Це – кінцева мета розвитку робототехніки. Основні невирішені проблеми при створенні інтелектуальних роботів – проблема машинного зору, адекватного зберігання і обробки тривимірної візуальної інформації.

В даний час в світі виготовляється більше 60000 роботів на рік. Фактично робототехніка сьогодні – це інженерна наука, яка не відкидає технологій штучного інтелекту, але не готова поки до їх впровадження в силу різних причин.

*5. Навчання і самонавчання.* Область штучного інтелекту, яка активно розвивається. Включає моделі, методи і алгоритми, орієнтовані на автоматичне накопичення і формування знань на основі аналізу та узагальнення даних, навчання за прикладами (або індуктивний), а також традиційні підходи з теорії розпізнавання образів. В останні роки до цього напрямку тісно примикають системи аналізу даних і пошуку закономірностей в базах даних, такі як Data-mining і Knowledge Discovery, які дуже стрімко розвиваються останнім часом.

*6. Розпізнавання образів.* Напрямок штучного інтелекту, що бере початок з самих його витоків, але в даний час виділився в самостійну науку. Її основний підхід – опис класів об'єктів через певні значення значущих ознак. Кожному об'єкту ставиться у відповідність матриця ознак, за якою відбувається його

розпізнавання. Процедура розпізнавання використовує найчастіше спеціальні математичні процедури і функції, що розділяють об'єкти на класи. Цей напрямок близький до машинного навчання і тісно пов'язаний з нейрокібернетикою.

*7. Нові архітектури комп'ютерів.* Найсучасніші процесори сьогодні засновані на традиційній послідовній архітектурі фон Неймана, використовуваній ще в комп'ютерах перших поколінь. Ця архітектура вкрай неефективна для символьної обробки. Тому зусилля багатьох наукових колективів і фірм вже десятки років націлені на розробку апаратних архітектур, призначених для обробки символьних і логічних даних. Створюються Пролог- і Ліспмашини, комп'ютери V і VI поколінь. Останні розробки присвячені комп'ютерам баз даних, паралельним і векторним комп'ютерам. І хоча вдалі промислові рішення існують, висока вартість, недостатнє програмне оснащення і апаратна несумісність з традиційними комп'ютерами істотно гальмують широке використання нових архітектур.

*8. Ігри.* Цей напрямок став більше історичним та пов'язаний з тим, що на зорі досліджень штучного інтелекту традиційно включав в себе ігрові інтелектуальні задачі – шахи, шашки, го. В основі перших програм лежить один з ранніх підходів – лабіринтова модель мислення плюс евристики. Зараз це швидше комерційний напрямок, так як в науковому плані ці ідеї вважаються тупиковими. В даний час в комп'ютерних іграх почали застосовуватися інші ідеї штучного інтелекту – нейронні мережі, інтелектуальні агенти, генетичні алгоритми і т.д., які дозволяють створювати персонажів (ботів) з різним ступенем «інтелекту». Використання методів штучного інтелекту в іграх дозволяє отримувати нові ефективні рішення, створювати шаблони проектування, підвищувати розважальність і достовірність ігор. Крім того, комп'ютерні ігри надають потужний арсенал різноманітних засобів, що використовуються для навчання.

*9. Машинна творчість.* Напрямок охоплює створення комп'ютером музики (У. Айзексон, Л. Хіллер, Р. Заріпов), віршів (Д. Лінк), живопису (Х. Фарід, Л. Моура) і навіть казок і афоризмів. Основним методом подібної «творчості» є метод пермутації (перестановок) плюс використання деяких баз знань і даних, що

містять результати досліджень по структурам текстів, рим, сценаріям і т. п.

*10. Нечіткі моделі та м'які обчислення.* Цей напрямок представлений нечіткими схемами «виведення за аналогією», поглядом на теорію нечітких мір з імовірнісних позицій, нечітким поданням аналітичними моделями для опису геометричних об'єктів, алгоритмами еволюційного моделювання з динамікою, такими, як час життя і розмір популяції, методами вирішення оптимізаційних задач з використанням технологій генетичного пошуку, гомеостатических і синергетичних принципів та елементів самоорганізації.

*11. Евристичне програмування.* В рамках напряму досліджують послідовності розумових операцій, виконання яких призводить до успішного вирішення того чи іншого завдання, моделюють розумову діяльність людини для вирішення завдань, що не мають суворого формалізованого алгоритму або пов'язаних з неповнотою вхідних даних.

*12. Штучне життя.* Напрямок досліджень, метою якого є створення штучних істот, здатних діяти не менш ефективно, ніж живі істоти. М'яке штучне життя створює обчислювальні системи і моделі, що діють на базі біологічних і еволюційних принципів. Вологе штучне життя синтезує нові штучні біологічні форми. В рамках цього напрямку використовують генетичні алгоритми, клітинні автомати, автономні агенти і т.д.

*13. Когнітивне моделювання.* Науковий напрямок, що є плідним синтезом когнітивної графіки і обчислювального моделювання, що дозволяє істотно підвищити пізнавальну ефективність сучасних ЕОМ та ПК. Методологія когнітивного моделювання призначена для аналізу і прийняття рішень в погано визначених ситуаціях, ґрунтується на моделюванні суб'єктивних уявлень експерта.

*14. Еволюційне моделювання.* При еволюційному моделюванні процес моделювання складної соціально-економічної системи зводиться до створення моделі його еволюції або до пошуку допустимих станів системи, до процедури (алгоритму) відстеження безлічі допустимих станів (траєкторій).

*15. Багатоагентні системи.* Напрямок штучного інтелекту, який розглядає рішення однієї задачі кількома інтелектуальними

підсистемами – агентами. *Агент* – апаратна або програмна сутність, здатна діяти в інтересах досягнення мети, поставленої перед всією системою. Соціальні системи дають ще одне модельне уявлення інтелекту за допомогою глобальної поведінки, яке дозволяє їм вирішувати проблеми, які б не вдалося вирішити окремим їх членам. Агенти в таких системах автономні або напівавтономні, у кожного агента є певне коло підзадач, причому він має в своєму розпорядженні обмаль знань (або зовсім не має в своєму розпорядженні знань) про те, що роблять інші агенти або як вони це роблять. Кожен агент виконує свою незалежну частину рішення проблеми і видає власний результат (щось робить) або повідомляє результат іншим агентам.

*16. Онтології.* В рамках цього напрямку досліджується можливість всеосяжної і детальної формалізації деякої області знань за допомогою концептуальної схеми – ієрархічної структури даних, що містить всі релевантні класи об'єктів, їх зв'язків і правил предметної області. Онтології використовуються і людьми, і програмними агентами, дозволяють повторно використовувати знання предметної області, відокремити їх від оперативних знань і аналізувати їх. Розробляються мови опису онтологій (RDF, DAML, OWL, KIF).

*17. Комп'ютерні віруси.* Останнє покоління комп'ютерних вірусів володіють всіма атрибутами систем штучного інтелекту. Вони здатні до розмноження, мутації, еволюції, навчання. Сучасні проблеми щодо захисту від них виявляються незначними, коли вони повністю проникнуть в сферу штучного інтелекту. Методи штучного інтелекту необхідні як для їх створення, так і для розробки ефективних засобів захисту.

*18. Інтелектуальне математичне моделювання.* В даному напрямку системи імітують творчу діяльність математика-професіонала, що займається рішенням, наприклад, крайових задач математичної фізики. Для цього використовуються бази знань, що містять теореми, математичні залежності, евристичні правила, такі системи здатні до навчання і самонавчання.

Це далеко не всі напрямки штучного інтелекту, існує безліч напрямків для вирішення безлічі конкретних задач.

# ЛЕКЦІЯ 2. КЛАСИФІКАЦІЯ ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ

## 2.1. Визначення інтелектуальної інформаційної системи

Існує велика безліч інтелектуальних інформаційних систем (ІІС). Однак загальноприйнятого єдиного визначення інтелектуальної інформаційної системи немає.

Інтелектуальною інформаційною системою називають автоматизовану інформаційну систему, засновану на знаннях, або комплекс програмних, лінгвістичних і логіко-математичних засобів для реалізації основної задачі – здійснення підтримки діяльності людини і пошуку інформації в режимі просунутого діалогу на природній мові.

Крім того, інформаційно-обчислювальними системами з інтелектуальною підтримкою для вирішення складних задач називають ті системи, в яких логічна обробка інформації превалює над обчислювальною. Таким чином, будь-яка інформаційна система, яка вирішує інтелектуальну задачу або в якій застосовують методи штучного інтелекту, відноситься до інтелектуальних.

Для інтелектуальних інформаційних систем характерні наступні ознаки:

– розвинені комунікативні здібності;

– вміння вирішувати складні погано формалізовані задачі;

– здатність до самонавчання;

– адаптивність.

Кожна з перерахованих ознак умовно відповідає своєму класу ІІС. Різні системи можуть мати одну або декілька ознак інтелектуальності з різним ступенем прояву.

Комунікативні здібності ІІС характеризують спосіб взаємодії (інтерфейсу) кінцевого користувача з системою, зокрема, можливість формулювання довільного запиту в діалозі з ІІС на мові, максимально наближеній до природньої.

Складні погано формалізовані задачі – це задачі, які вимагають побудови оригінального алгоритму рішення в залежності від конкретної ситуації, для якої можуть бути характерні невизначеність і динамічність вхідних даних і знань.

Здатність до самонавчання – це можливість автоматичного вилучення знань для вирішення задач з накопиченого досвіду конкретних ситуацій.

Адаптивність – здатність до розвитку системи відповідно до об'єктивних змін моделі проблемної області.

## 2.2. Класифікація інтелектуальних систем

У відповідності з перерахованими ознаками ІС діляться на (рис. 2.1):

– системи з комутативними здібностями (з інтелектуальним інтерфейсом);

– експертні системи (системи для вирішення складних задач);

– самонавчаємі системи (системи здатні до самонавчання);

– адаптивні системи (адаптивні інформаційні системи), причому дана класифікація одна з можливих.


Рис.2.1. Класифікація інтелектуальних інформаційних систем за типами систем

Застосування ШІ для посилення комунікативних здібностей інформаційних систем призвело до появи систем з інтелектуальним інтерфейсом, серед яких можна виділити наступні типи.

*Інтелектуальні бази даних* відрізняються від звичайних баз даних можливістю вибірки за запитом необхідної інформації, яка може явно не зберігатися, а виводитися з наявної в базі даних.

Природно-мовний інтерфейс (ПМ-інтерфейс) використовується для:

– доступу до інтелектуальних баз даних;

– контекстного пошуку документальної текстової інформації;

– голосового введення команд в системах управління;

– машинного перекладу з іноземних мов.

*Природно-мовний інтерфейс передбачає* трансляцію природно-мовних конструкцій на внутрішньомашинний рівень представлення знань. Для цього необхідно вирішувати задачі морфологічного, синтаксичного і семантичного аналізу та синтезу висловлювань на природній мові. Так, морфологічний аналіз передбачає розпізнавання і перевірку правильності написання слів за словниками, синтаксичний контроль – розкладання вхідних повідомлень на окремі компоненти (визначення структури) з перевіркою відповідності граматичним правилам внутрішнього представлення знань і виявлення відсутніх частин і, нарешті, семантичний аналіз – встановлення змістової правильності синтаксичних конструкцій. Синтез висловлювань вирішує зворотну задачу перетворення внутрішнього подання інформації в природно-мовну.

*Гіпертекстові системи* призначені для реалізації пошуку за ключовими словами в базах текстової інформації. Інтелектуальні гіпертекстові системи відрізняються можливістю більш складної семантичної організації ключових слів, яка відображає різні смислові відносини термінів. Таким чином, механізм пошуку працює, перш за все, з базою знань ключових слів, а вже потім безпосередньо з текстом. У більш широкому плані сказане поширюється і на пошук мультимедійної інформації, що включає крім текстової і цифрову інформацію.

*Системи контекстної допомоги* можна розглядати як окремий випадок інтелектуальних гіпертекстових і природно-мовних систем. На відміну від звичайних систем допомоги, які нав’язують користувачеві схему пошуку необхідної інформації, в системах контекстної допомоги користувач описує проблему (ситуацію), а система за допомогою додаткового діалогу її

конкретизує і сама виконує пошук рекомендацій, що стосуються ситуації. Такі системи відносяться до класу систем поширення знань (Knowledge Publishing) і створюються як додаток до систем документації (наприклад, технічної документації по експлуатації товарів).

*Системи когнітивної графіки* дозволяють здійснювати інтерфейс користувача з ІІС за допомогою графічних образів, які генеруються відповідно до подій. Такі системи використовуються в моніторингу і управлінні оперативними процесами. Графічні образи в наочному і інтегрованому вигляді описують безліч параметрів досліджуваної ситуації. Наприклад, стан складного керованого об'єкта відображається у вигляді людського обличчя, на якому кожна риса відповідає за будь-який параметр, а загальний вираз обличчя дає інтегровану характеристику ситуації. Системи когнітивної графіки широко використовуються також в навчальних і тренажерних системах на основі використання принципів віртуальної реальності, коли графічні образи моделюють ситуації, в яких тому, хто навчається, необхідно приймати рішення і виконувати певні дії.

*Експертні системи* призначені для вирішення задач на основі накопичуваної бази знань, що відбиває досвід роботи експертів в даній проблемній області. Експертні системи, як самостійний напрям в штучному інтелекті, сформувався в кінці 1970-х рр. Історія ЕС почалася з повідомлення японського комітету з розробки ЕОМ п'ятого покоління, в якому основна увага приділялася розвитку «інтелектуальних здібностей» комп'ютерів з тим, щоб вони могли оперувати не тільки даними, але і знаннями, як це роблять фахівці (експерти) при виробленні розумових висновків. Група з експертних систем при Комітеті British Computer Society визначила ЕС як «втілення в ЕОМ компоненти досвіду експерта, заснованого на знаннях, в такій формі, що машина може дати інтелектуальну пораду або прийняти рішення щодо оброблюваної функції». Однією з важливих властивостей ЕС є здатність пояснити хід своїх міркувань зрозумілим для користувача чином.

Область дослідження ЕС називають «інженерією знань». Цей термін був введений Е. Фейгенбаумом і в його трактуванні означає «привнесення принципів і інструментарію з області штучного інтелекту в рішення важких прикладних проблем, що вимагають

знань експертів». Іншими словами, ЕС застосовуються для вирішення неформалізованих проблем, до яких відносять задачі, що володіють однією (або декількома) з наступних характеристик:

– задачі не можуть бути представлені в числовій формі;

– вихідні дані і знання про предметну область володіють неоднознач-ністю, неточністю, суперечністю;

– мету не можна виразити за допомогою чітко визначеної цільової функції;

– не існує однозначного алгоритмічного рішення задачі;

– алгоритмічне рішення існує, але його не можна використовувати через велику розмірність простору рішень і обмежень на ресурси (час, пам'ять).

Головна відмінність ЕС і систем штучного інтелекту від систем обробки даних полягає в тому, що в них використовується символьний, а не числовий спосіб представлення даних, а в якості методів обробки інформації застосовуються процедури логічного висновку і евристичного пошуку рішень.

ЕС охоплюють найрізноманітніші предметні області (рис. 2.2), серед яких лідирують бізнес, виробництво, медицина, проектування і системи управління.

У багатьох випадках ЕС є інструментом, що підсилює інтелектуальні здібності експерта. Крім того, ЕС може виступати в ролі:

– консультанта для недосвідчених або непрофесійних користувачів;

– асистента експерта-людини в процесах аналізу варіантів рішень;

– партнера експерта в процесі вирішення задач, що вимагають залучення знань з різних предметних областей.

Для класифікації ЕС використовуються наступні ознаки:

– спосіб формування рішення;

– спосіб обліку тимчасової ознаки;

– вид використовуваних даних і знань;

– число використаних джерел знань.

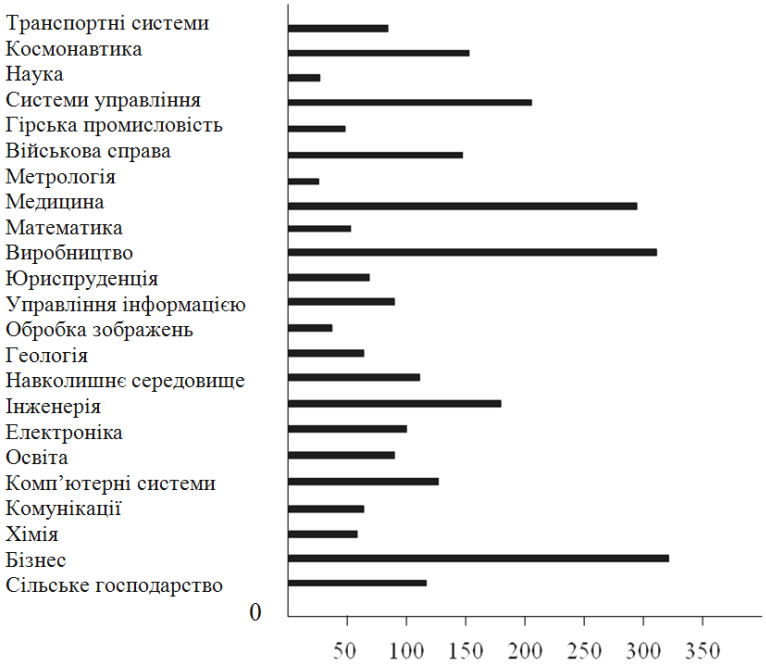


Рис. 2.2. Області застосування експертних систем

За *способом формування рішення* ЕС можна розділити на ті, що аналізують і ті, що синтезують. У системах першого типу здійснюється вибір рішення з багатьох відомих рішень на основі аналізу знань, в системах другого типу рішення синтезується з окремих фрагментів знань.

Залежно від *способу обліку тимчасової ознаки* ЕС ділять на статичні та динамічні. Статичні ЕС призначені для вирішення задач з незмінними в процесі вирішення даними і знаннями, а динамічні ЕС допускають такі зміни.

За *видами використовуваних даних і знань* розрізняють ЕС з детермінованими і невизначеними знаннями. Під невизначеністю знань і даних розуміється їх неповнота, ненадійність, нечіткість.

ЕС можуть створюватися з використанням одного або декількох *джерел знань.*

У відповідності з перерахованими ознаками можна виділити чотири основні класи ЕС (рис. 2.3): классифікуючі, довизначаючі, трансформуючі і мультиагентні.


Рис. 2.3. Основні класи експертних систем

Класифікуючі ЕС вирішують задачі розпізнавання ситуацій. Основним методом формування рішень в таких системах є дедуктивний логічний висновок.

Довизначаючі ЕС використовуються для вирішення задач з не повністю визначеними даними і знаннями. У таких ЕС виникають задачі інтерпретації нечітких знань і вибору альтернативних напрямків пошуку в просторі можливих рішень. В якості методів обробки невизначених знань можуть використовуватися байєсівський імовірнісний підхід, коефіцієнти впевненості, нечітка логіка.

Трансформуючі ЕС відносяться до синтезуючих динамічних експертних систем, в яких передбачається повторення перетвореннь знань в процесі вирішення задач. В ЕС даного класу використовуються різні способи обробки знань:

– генерація і перевірка гіпотез;

– логіка припущень і замовчувань (коли за неповними даними формуються уявлення про об’єкти певного класу, які згодом адаптуються до конкретних умов змінних ситуацій);

– використання метазнань (більш загальних закономірностей) для усунення невизначеностей у ситуаціях.

Багатоагентні системи. Для таких динамічних систем характерна інтеграція в базі знань декількох різнорідних джерел знань, які обмінюються між собою одержуваними результатами на

динамічній основі. Для багатоагентних систем характерні наступні особливості:

1. Проведення альтернативних міркувань на основі використання різних джерел знань з механізмом усунення протиріч.

2. Розподілене рішення проблем, які розбиваються на підпроблеми, що вирішуються паралельно, відповідні самостійним джерелам знань.

3. Застосування безлічі стратегій роботи механізму виведення включень в залежності від типу розв'язуваної проблеми.

4. Обробка великих масивів даних, що містяться в базі даних.

5. Використання різних математичних моделей і зовнішніх процедур, що зберігаються в базі моделей;

6. Здатність переривання вирішення задач у зв'язку з необхідністю отримання додаткових даних і знань від користувачів, моделей, паралельно вирішуваних підпроблем.

Самонавчаємі інтелектуальні системи засновані на методах автоматичної класифікації ситуацій з реальної практики, або на методах навчання на прикладах. Приклади реальних ситуацій складають так звану навчальну вибірку, яка формується протягом певного історичного періоду. Елементи навчальної вибірки описуються безліччю класифікаційних ознак.

Характерними ознаками самонавчаємих систем є:

– самонавчаємі системи "з вчителем", коли для кожного прикладу задається в явному вигляді значення ознаки його належності певному класу ситуацій (класотворюючої ознаки);

– самонавчаємі системи "без вчителя", коли за ступенем близькості значень ознак класифікації система сама виділяє класи ситуацій.

В процесі навчання проводиться автоматична побудова узагальнюючих правил або функцій, що описують належність ситуацій до класів, якими система згодом буде користуватися при інтерпретації незнайомих ситуацій. З узагальнюючих правил, в свою чергу, автоматично формується база знань, яка періодично коригується в міру накопичення інформації про аналізовані ситуації.

Побудовані відповідно до цих принципів самонавчаємі системи мають такі недоліки:

– відносно низьку адекватність баз знань виникаючим

реальним проблемам через неповноту та/або зашумленість навчальної вибірки;

– низький ступінь пояснювальності отриманих результатів;

– поверхневий опис проблемної області та вузьку спрямованість застосування через обмеження в розмірності простору ознак.

Індуктивні системи використовують узагальнення прикладів за принципом від окремого до загального. Процес класифікації прикладів здійснюється наступним чином:

1. Обирається ознака класифікації з безлічі заданих (або послідовно, або за будь-яким правилом, наприклад, відповідно до максимального числа отриманих підмножин прикладів).

2. За значенням обраної ознаки безліч прикладів розбивається на підмножини.

3. Виконується перевірка, чи належить кожна з утворених підмножин прикладів до одного підкласу.

4. Якщо якась підмножина прикладів належить одному підкласу, тобто у всіх прикладів підмножини збігається значення класоутворюючої ознаки, то процес класифікації закінчується (при цьому інші ознаки класифікації не розглядаються).

5. Для підмножин прикладів з неспівпадаючим значенням класоутворюючої ознаки процес класифікації продовжуєте, починаючи з пункту 1. (Кожна підмножина прикладів стає класифікованою множиною).

*Нейронні мережі* являють собою пристрої паралельних обчислень, що складаються з безлічі взаємодіючих простих процесорів – узагальнена назва групи математичних алгоритмів, що володіють здатністю навчатися на прикладах, «впізнаючи» згодом риси зразків і ситуацій, що вже зустрічалися. Кожен процесор такої мережі має справу тільки з сигналами, які він періодично отримує, і сигналами, які він періодично посилає іншим процесорам.

Нейронна мережа – це кібернетична модель нервової системи, яка являє собою сукупність великого числа порівняно простих елементів – нейронів, топологія з'єднання яких залежить від типу мережі. Щоб створити нейронну мережу для вирішення якого-небудь конкретного завдання, слід вибрати спосіб з'єднання

нейронів один з одним і підібрати значення параметрів міжнейронних з'єднань.

В експертних системах, *заснованих на прецедентах* (аналогіях), база знань містить описи не узагальнених ситуацій, а власне самі ситуації або прецеденти. Пошук рішення проблеми в експертних системах заснованих на прецедентах зводиться до пошуку по аналогії (тобто абдуктивний висновок від окремого до окремого) і включає наступні етапи:

– отримання інформації про поточну проблему;

– зіставлення отриманої інформації зі значеннями ознак прецедентів з бази знань;

– вибір прецеденту з бази знань, найбільш близького до даної проблеми;

– адаптація обраного прецеденту до поточної проблеми;

– перевірка коректності кожного отриманого рішення;

– занесення детальної інформації про отримане рішення до баз знань.

Прецеденти описуються безліччю ознак, за якими будуються індекси швидкого пошуку. Однак в системах, заснованих на прецедентах, на відміну від індуктивних систем, допускається нечіткий пошук з отриманням безлічі допустимих альтернатив, кожна з яких оцінюється деяким коефіцієнтом впевненості. Найбільш ефективні рішення адаптуються до реальних ситуацій за допомогою спеціальних алгоритмів.

Системи, засновані на прецедентах, застосовуються для поширення знань і в системах контекстної допомоги.

На відміну від інтелектуальної бази даних *інформаційне сховище* являє собою сховище витягнутої значимої інформації з оперативної бази даних, яке призначене для оперативного ситуаційного аналізу даних (реалізації OLAP-технології). Типовими задачами оперативного ситуаційного аналізу є:

– визначення профілю споживачів конкретних об'єктів зберігання;

– передбачення змін об'єктів зберігання в часі;

– аналіз залежностей ознак ситуацій (кореляційний аналіз).

Сховище даних – це предметно-орієнтоване, інтегроване, прив'язане до часу, незмінне зібрання даних, що застосовуються для підтримки процесів прийняття управлінських рішень.

Предметна орієнтація означає, що дані об'єднані в категорії і зберігаються відповідно до тих областей, які вони описують, а не з додатками, які їх використовують. У сховищі дані інтегруються з метою задоволення вимог підприємства в цілому, а не окремої функції бізнесу. Прихильність даних до часу висловлює їх «історичність», тобто атрибут часу завжди явно присутній в структурах сховища даних. Незмінюваність означає, що, потрапивши одного разу в сховище, дані вже не змінюються на відміну від оперативних систем, де дані присутні тільки в останній версії, тому постійно змінюються.

Технології вилучення знань зі сховищ даних засновані на методах статистичного аналізу і моделювання, орієнтованих на пошук моделей і відносин, прихованих в сукупності даних. Ці моделі можуть в подальшому використовуватися для оптимізації діяльності підприємства або фірми.

Для вилучення значущої інформації зі сховищ даних є спеціальні методи (OLAP-аналізу, Data Mining або Knowledge Discovery), засновані на застосуванні методів математичної статистики, нейронних мереж, індуктивних методів побудови дерев рішень та ін. Технологія OLAP (On-Line Analytical Processing – оперативний аналіз даних) надає користувачеві засоби для формування і перевірки гіпотез про властивості даних або відносини між ними на основі різноманітних запитів до бази даних. Вони застосовуються на ранніх стадіях процесу здобування знань, допомагаючи аналітику сфокусувати увагу на важливих змінних. Засоби Data Mining відрізняються від OLAP тим, що крім перевірки передбачуваних залежностей вони здатні самостійно (без участі користувача) генерувати гіпотези про закономірності, що існують в даних, і будувати моделі, що дозволяють кількісно оцінити ступінь взаємного впливу досліджуваних факторів на основі наявної інформації.

*Адаптивна інформаційна система* – це інформаційна система, яка змінює свою структуру відповідно до зміни моделі проблемної області. При цьому адаптивна інформаційна система повинна:

– в кожен момент часу адекватно підтримувати організацію бізнес-процесів;

– проводити адаптацію щоразу, як виникає потреба в реорганізації бізнес-процесів;

– реконструкція інформаційної системи повинна проводитися швидко і з мінімальними витратами.

Ядром адаптивної інформаційної системи є модель проблемної області (підприємства), що постійно розвивається, підтримувана в спеціальній базі знань – репозиторії, на основі якого здійснюється генерація або конфігурація програмного забезпечення (ядро системи управляє цими процесами). Таким чином, проектування і адаптація ІС зводиться, перш за все, до побудови моделі проблемної області та її своєчасного корегування.

В процесі розробки адаптивних інформаційних систем застосовується *оригінальне* або *типове* проектування. Оригінальне проектування передбачає розробку інформаційної системи з «чистого аркуша» на основі сформульованих вимог. Реалізація цього підходу заснована на використанні систем автоматизованого проектування, або CASE-технологій (Designer2000, SilverRun, Natural Light Storm та ін.). При типовому проектуванні здійснюється адаптація типових розробок до особливостей проблемної області. Для реалізації цього підходу застосовуються інструментальні засоби *компонентного* (*складального*) *проектування* інформаційних систем (R/3, BAAN IV, Prodis та ін.).

Головна відмінність підходів полягає в тому, що при використанні CASE-технології на основі сховища при зміні проблемної області кожен раз виконується *генерація* програмного забезпечення, а при використанні складальної технології – *конфігурація* програм і тільки в рідкісних випадках їх *переробка*.

Так як немає загальноприйнятого визначення, чітку єдину класифікацію інтелектуальних інформаційних систем дати важко. Якщо розглядати інтелектуальні інформаційні системи з точки зору розв’язуваної задачі, то можна виділити системи управління і довідкові системи, системи комп’ютерної лінгвістики, системи розпізнавання, ігрові системи і системи створення інтелектуальних інформаційних систем (див. рис. 2.4).


Рис.2.4. Класифікація інтелектуальних інформаційних систем за задачами, що розв'язуються

При цьому системи можуть вирішувати не одну, а кілька задач або в процесі вирішення однієї задачі вирішувати і ряд інших. Наприклад, під час вивчення іноземної мови система може вирішувати задачі розпізнавання мови учня, тестувати, відповідати на питання, перекладати тексти з однієї мови на іншу і підтримувати природно-мовний інтерфейс роботи.

Якщо класифікувати інтелектуальні інформаційні системи за критерієм «використовувані методи», то вони діляться на жорсткі, м'які і гібридні (див. рис. 2.5).

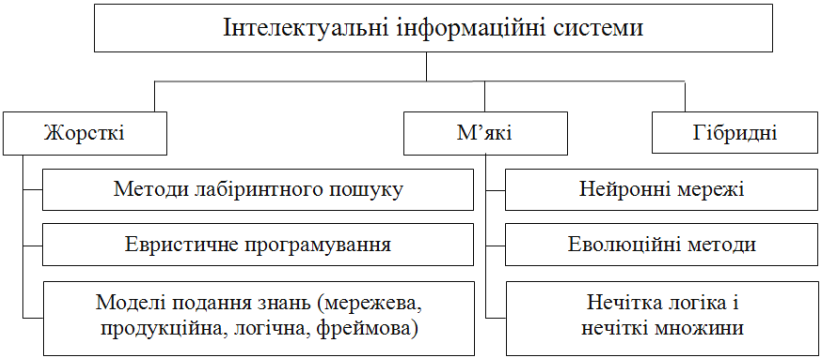


Рис. 2.5. Класифікація інтелектуальних інформаційних систем за методами

*М'які обчислення* – це складна комп'ютерна методологія, заснована на нечіткій логіці, генетичних обчисленнях, нейрокомп'ютингу та ймовірнісних обчисленнях. *Жорсткі* обчислення – традиційні комп'ютерні обчислення (не м'які). *Гібридні* системи – системи, що використовують більш ніж одну комп'ютерну технологію (в разі інтелектуальних систем – технології штучного інтелекту).

Можливі й інші класифікації, наприклад, виділяють системи загального призначення і спеціалізовані системи (див. рис. 2.6).

Крім того, ця схема відображає ще один варіант класифікації за методами: системи, що використовують методи представлення знань, системи, що самоорганізуються і, створені за допомогою евристичного програмування. Також в цій класифікації системи генерації музики віднесені до систем спілкування.


Рис. 2.6. Класифікація інтелектуальних систем за призначенням

До інтелектуальних систем *загального призначення* відносяться системи, які не тільки виконують задані процедури, але на основі метапроцедур пошуку генерують і виконують процедури вирішення нових конкретних задач.

*Спеціалізовані* інтелектуальні системи виконують рішення фіксованого набору задач, передбаченого при проектуванні системи.

Відсутність чіткої класифікації також пояснюється різноманіттям інтелектуальних задач і інтелектуальних методів, крім того, штучний інтелект – наука, що активно розвивається, в якій нові прикладні області освоюються щодня.

# ЛЕКЦІЯ 3. ТЕХНОЛОГІЇ РОЗРОБКИ ЕКСПЕРТНИХ СИСТЕМ

## 3.1. Основні поняття і визначення

Технологія створення інтелектуального програмного забезпечення суттєво відрізняється від розробки традиційних програм з використанням відомих алгоритмічних мов (табл. 3.1).

Розглянемо відпрацьовані на сьогоднішній день елементи технології створення ІІС на прикладі розробки експертних систем. Цей вибір обумовлений тим, що ЕС отримали досить широке поширення в багатьох сферах людської діяльності, а технології їх створення мають універсальний характер і не потребують апаратних реалізацій.

*Експертними системами* називають складні програмні комплекси, що акумулюють знання фахівців в конкретних предметних областях і тиражують цей емпіричний досвід для консультацій менш кваліфікованих користувачів.

Таблиця 3.1

**Відмінності систем штучного інтелекту від звичайних програмних систем**

| Характеристика | Програмування в системах штучного інтелекту | Традиційне програмування |
|---|---|---|
| Тип обробки | Символьний | Числовий |
| Метод | Евристичний пошук | Точний алгоритм |
| Завдання кроків вирішення | Неявне | Явне |
| Шукане рішення | Задовільний | Оптимальне |
| Управління та дані | Змішані | Розділені |
| Знання | Неточні | Точні |
| Модифікації | Часті | Рідкісні |

У найперших ЕС не враховувалася зміна знань, використовуваних в процесі вирішення конкретного завдання. Їх назвали *статичними ЕС*.

Типова статична ЕС містить наступні основні компоненти:

– базу знань;

– робочу пам’ять, яка також називається базою даних;

– вирішувач (інтерпретатор);

– систему пояснень;

– компоненти придбання знань;

– інтерфейс з користувачем.

*База знань ЕС* призначена для зберігання довгострокових даних, що описують розглянуту область, і правил, що описують доцільні перетворення даних цієї області.

*База даних* (*робоча пам’ять*) служить для зберігання поточних даних розв’язуваної задачі.

*Вирішувач* (інтерпретатор) формує послідовність застосування правил і здійснює їх обробку, використовуючи дані з робочої пам’яті і знання з БЗ.

*Система пояснень* показує, яким чином система отримала розв’язок задачі і які знання при цьому використовувалися. Це полегшує тестування системи і підвищує довіру користувача до отриманого результату.

*Компоненти придбання знань* необхідні для заповнення ЕС знаннями в діалозі з користувачем – експертом, а також для додавання і модифікації закладених в систему знань.

До розробки ЕС залучаються фахівці з різних предметних областей, а саме:

– експерти тієї проблемної області, до якої відносяться задачі, які вирішуються ЕС;

– інженери по знаннях, що є фахівцями з розробки ІІС;

– програмісти, які здійснюють реалізацію ЕС.

*Експерти* поставляють знання в ЕС і оцінюють правильність одержуваних результатів.

*Інженери по знаннях* допомагають експертам виявити і структурувати знання, необхідні для роботи ЕС, виконують роботу за поданням знань, обирають методи обробки знань, проводять вибір інструментальних засобів для реалізації ЕС, найбільш придатних для вирішення поставлених задач.

*Програмісти* розробляють програмне забезпечення ЕС і здійснюють його сполучення з середовищем, в якому воно буде використовуватися.

Будь-яка ЕС повинна мати, принаймні, два режими роботи. У *режимі придбання знань* експерт наповнює систему знаннями, які

згодом дозволять ЕС самостійно (без допомоги експерта) вирішувати певні задачі з конкретної проблемної області. Експерт описує проблемну область у вигляді сукупності даних і правил. Дані визначають об'єкти, їх характеристики і значення, що існують в області експертизи. Правила визначають взаємні зв'язки, що існують між даними, і способи маніпулювання даними, які характерні для даного класу задач.

У *режимі консультації* користувач ЕС повідомляє системі конкретні дані про розв'язувану задачу і прагне отримати з її допомогою результат. Користувачі-неспеціалісти звертаються до ЕС за результатом, не вміючи отримати його самостійно, користувачі-фахівці використовують ЕС для прискорення і полегшення процесу отримання результату. Слід підкреслити, що термін «користувач» є багатозначним, так як використовувати ЕС можуть і експерт, і інженер по знаннях, і програміст. Тому, коли хочуть підкреслити, що мова йде про те, для кого робилася ЕС, використовують термін «кінцевий користувач».

У режимі консультації вхідні дані про задачу надходять в робочу пам'ять. Вирішувач на основі вхідних даних з робочої пам'яті і правил з БЗ формує рішення. На відміну від традиційних програм комп'ютерної обробки даних експертна система при вирішенні задачі не тільки виконує визначену послідовність операцій, але і сама формує її.

Існує широкий клас додатків, в яких потрібно враховувати зміни, що відбуваються в навколишньому світі за час виконання програми. Для вирішення таких задач необхідно застосовувати динамічні ЕС, які поряд з компонентами статичних систем містять підсистему моделювання зовнішнього світу і підсистему зв'язку із зовнішнім оточенням. Підсистема моделювання зовнішнього світу необхідна для прогнозування, аналізу та адекватної оцінки стану зовнішнього середовища. Зміни оточення розв'язуваної задачі вимагають зміни знань, що зберігаються в ЕС, для того щоб відобразити тимчасову логіку подій, які відбуваються в реальному світі. Компонента зв'язку із зовнішнім світом актуальна для автономних інтелектуальних систем (роботів), а також для інтелектуальних систем управління. Зв'язок із зовнішнім світом здійснюється через систему датчиків і контролерів.

### 3.2. Класифікаційні ознаки експертних систем

В основі класифікації експертних систем (будемо називати їх додатками) лежать наступні параметри:

– тип додатка;

– стадія існування;

– масштаб;

– тип проблемного середовища;

– тип розв'язуваної задачі.

*Тип додатка*. Характеризується такими ознаками.

1. Можливість взаємодії додатка з іншими програмними засобами:

– *ізольований додаток* – ЕС, не здатна взаємодіяти з іншими програмними системами (наприклад, з БД, електронними таблицями, пакетами прикладних програм, контролерами, датчиками і т.п.);

– *інтегрований додаток* – ЕС та інші програмні системи, з якими вона взаємодіє в ході роботи. Більшість сучасних ЕС, використовуваних для вирішення практично значимих задач, є інтегрованими.

2. Можливість виконувати додаток на різнорідній апаратурі і переносити його на різні платформи:

– *закриті додатки* – виконуються тільки в програмному середовищі даної фірми і можуть бути перенесені на інші платформи тільки шляхом перепрограмування додатку;

– *відкриті додатки* – орієнтовані на виконання в різнорідному програмно-апаратному оточенні і можуть бути перенесені на інші платформи без перепрограмування.

3. Архітектура програми:

– *централізований додаток* – реалізується на базі центральної ЕОМ, з якою пов'язані термінали;

– *розподілений додаток* – зазвичай використовується архітектура клієнт-сервер.

*Стадія існування*. Характеризує ступінь завершеності розробки ЕС. Прийнято виділяти наступні стадії:

– *дослідницький прототип* – вирішує представницький клас задач проблемної області, але може бути нестійкий в роботі і не повністю перевірений. При наявності розвинених інструментальних

засобів для розробки дослідного прототипу потрібно приблизно 2 – 4 місяці. База знань дослідного прототипу зазвичай містить невелику кількість виконуваних тверджень;

– *діючий прототип* – надійно вирішує будь-які завдання проблемної області, але при вирішенні складних завдань може зажадати надмірно багато часу та (або) пам'яті. Доведення системи від початку розробки до стадії діючого прототипу вимагає приблизно 6 – 9 місяців, при цьому кількість виконуваних тверджень в БЗ збільшується в порівнянні з дослідним зразком;

– *промислова система* – забезпечує високу якість вирішення всіх завдань при мінімумі часу і пам'яті. Зазвичай процес перетворення діючого прототипу в промислову систему полягає в розширенні бази знань і її ретельного налагодження. Доведення ЕС від початку розробки до стадії промислової системи з застосуванням розвинених інструментальних засобів вимагає не менше 12 – 18 місяців;

– *комерційна система* – придатна не тільки для використання розробником, а й для продажу різним споживачам. Доведення системи до комерційної стадії вимагає приблизно 1,5 – 2 роки. Наведені тут терміни справедливі для ЕС середньої складності.

*Масштаб ЕС*. Характеризує складність вирішуваних завдань і пов'язаний з типом використовуваної ЕОМ. За цією ознакою розрізняють:

– *малі ЕС* – призначені для первинного навчання і дослідження можливості застосування технології ЕС для розглянутого класу задач. Системи такого типу можуть бути реалізовані на персональних комп'ютерах;

– *середні ЕС* – охоплюють весь спектр необхідних додатків і зазвичай інтегровані з БД, електронними таблицями і т.д. Системи такого масштабу найчастіше реалізуються на робочих станціях;

– *великі ЕС* – мають доступ до високопотужних БД і реалізуються на робочих станціях або на спеціалізованих комп'ютерах;

– *символьні ЕС* – створюються з дослідницькими цілями і реалізуються на спеціалізованих комп'ютерах, орієнтованих на обробку символьних даних.

*Тип проблемного середовища*. Це поняття включає опис предметної області (безліч сутностей, що описують безліч об'єктів,

їх характеристик і відносин між об'єктами) і розв'язуваних в ній задач. Інакше кажучи, проблемне середовище включає структури даних і вирішені з ними задачі, представлені у вигляді виконуваних тверджень (правил, процедур, формул і т. п.). У зв'язку з цим проблемне середовище визначається характеристиками відповідної предметної області і характеристиками типів вирішуваних в ній задач.

*Характеристики предметної області*

1. Тип предметної області:

– *статичний* – вхідні дані не змінюються за час сеансу роботи програми, значення інших (вхідних) даних змінюються тільки самою експертною системою;

– *динамічний* – вхідні дані, що надходять із зовнішніх джерел, змінюються в часі, значення інших даних змінюються ЕС або підсистемою моделювання зовнішнього оточення.

2. Спосіб опису сутностей предметної області:

– сукупність атрибутів та їх значень (фіксований склад сутностей);

– сукупність класів (об'єктів) та їх примірників (змінний склад сутностей).

3. Спосіб організації сутностей у БЗ:

– неструктурована БЗ;

– структурування сутностей у БЗ по різних ієрархіях («окреме – загальне», «частина – ціле», «рід – вид»), що забезпечує успадкування властивостей сутностей.

Структурування БЗ сприяє:

– обмеженню кола сутностей, які повинні розглядатися механізмом виведення, і скорочення кількості варіантів, що перебираються, в процесі вибору рішення;

– забезпечення *успадкування властивостей сутностей*, тобто передачі властивостей вищерозташованих в ієрархії сутностей нижчерозташованим, що значно спрощує процес придбання і використання знань.

*Характеристики задач:*

1. Тип вирішуваних задач:

– задачі аналізу або синтезу. У *задачі аналізу* задана модель сутності і потрібно визначити невідомі характеристики моделі. У *задачі синтезу* задаються умови, яким повинні задовольняти

характеристики «невідомої» моделі сутності, і потрібно побудувати модель цієї сутності. Рішення задачі синтезу зазвичай включає задачу аналізу як складової частини;

– статичні або динамічні задачі. Якщо задачі, які вирішуються ЕС, явно не враховують фактор часу та/або не змінюють в процесі свого рішення знання про навколишній світ, то кажуть, що ЕС вирішує *статичні задачі*, в іншому випадку говорять про рішення *динамічних задач*. З огляду на значимість часу в динамічних проблемних середовищах, багато фахівців називають їх додатками, що працюють в реальному часі. Зазвичай виділяють наступні системи реального часу: *псевдореального* часу, «*м'якого*» реального часу і «*жорсткого*» реального часу. Системи псевдореального часу, як випливає з назви, не є системами реального часу, проте вони, на відміну від статичних систем, отримують і обробляють дані, що надходять із зовнішніх джерел. Системи псевдореального часу вирішують задачу швидше, ніж відбуваються значні зміни інформації про навколишній світ.

2. Спільність виконуваних тверджень:

– часткові виконувані твердження, які містять посилання на конкретні сутності (об'єкти);

– загальні виконувані твердження, що відносяться до будь-яких сутностей заданого типу (незалежно від їх числа та імені). Використання загальних тверджень дозволяє значно лаконічніше представляти знання. Однак оскільки загальні твердження не містять явних посилань на конкретні сутності, для їх використання кожен раз потрібно визначати ті сутності, до яких вони повинні застосовуватися.

Не всі поєднання перерахованих вище параметрів, що характеризують проблемну середу, зустрічаються на практиці. Найбільш поширені такі типи проблемних середовищ:

– статична предметна область: уявлення сутностей у вигляді сукупності атрибутів і їх значень, незмінний склад сутностей, БЗ не структурована, вирішуються статичні задачі аналізу, використовуються тільки частково виконувані твердження; уявлення сутностей об'єктами, змінюваний склад сутностей, БЗ структурована, вирішуються статичні задачі аналізу і синтезу, використовуються загальні і часткові виконувані твердження;

– динамічна предметна область: уявлення сутностей сукупністю атрибутів і їх значень, незмінний склад сутностей, БЗ не структурована, вирішуються динамічні задачі аналізу, використовуються часткові виконувані твердження; уявлення сутностей у вигляді об'єктів, змінюваний склад сутностей, БЗ не структурована, вирішуються динамічні задачі аналізу, використовуються часткові виконувані твердження; уявлення сутностей у вигляді об'єктів, змінюваний склад сутностей, БЗ структурована, вирішуються динамічні задачі аналізу і синтезу, використовуються загальні та часткові виконувані твердження.

*Тип розв'язуваної задачі*. За цією ознакою розрізняють наступні задачі:

– *інтерпретація даних* – процес визначення сенсу даних, результати якого повинні бути узгодженими і коректними. Експертні системи, як правило, проводять багатоваріантний аналіз даних;

– *діагностика* – процес співвіднесення об'єкта з деяким класом об'єктів та/або виявлення несправностей в системі (відхилень параметрів системи від нормативних значень);

– *моніторинг* – безперервна інтерпретація даних в реальному масштабі часу і сигналізація про вихід тих або інших параметрів за допустимі межі;

– *проектування* – створення об'єкта, що раніше не існував, і підготовка специфікацій на створення об'єктів із заздалегідь визначеними властивостями. Ступінь новизни може бути різною і визначається видом знань, закладених в ЕС, і методами їх обробки. Для організації ефективного проектування і реінжинірінгу потрібно формувати не лише самі проектні рішення, але і мотиви їх прийняття. ЕС, які вирішують задачі проектування, реалізують процедури виведення рішення і пояснення отриманих результатів;

– *прогнозування* – передбачення наслідків деяких подій або явищ на основі аналізу наявних даних. Прогнозуючі ЕС логічно виводять вірогідні наслідки з заданих ситуацій. В прогнозуючих ЕС в більшості випадків використовуються динамічні моделі, в яких значення параметрів «підганяються» під задану ситуацію. Виведені з цих моделей слідства складають основу для прогнозів з ймовірними оцінками;

– *планування* – побудова планів дій об'єктів, здатних виконувати деякі функції. Робота ЕС з планування заснована на моделях поведінки реальних об'єктів, які дозволяють проводити логічний висновок наслідків планованої діяльності;

– *навчання* – використання комп'ютера для вивчення будь-якої дисципліни або предмету. Експертні системи навчання виконують такі функції, як діагностика помилок, підказування правильних рішень; акумулювання знань про гіпотетичного «учня» та його характерні помилки; діагностування слабкості в пізнаннях учнів і знаходження відповідних засобів для їх ліквідації. Системи навчання здатні планувати акт спілкування з учнем залежно від успіхів учня для передачі необхідних знань;

– *управління* – функція організованої системи, що підтримує певний режим її діяльності. Експертні системи даного типу призначені для управління поведінкою складних систем відповідно до заданих специфікацій;

– *підтримка прийняття рішень* – сукупність процедур, що забезпечує особу, яка приймає рішення, необхідною інформацією та рекомендаціями, що полегшують процес прийняття рішення. Такого роду ЕС надають допомогу фахівцям у виборі та/або генерації найбільш раціональної альтернативи з безлічі можливих при прийнятті відповідальних рішень.

Задачі інтерпретації даних, діагностики, підтримки прийняття рішень відносяться до задач аналізу; задачі проектування, планування та управління – до задач синтезу. До комбінованого типу задач належать навчання, моніторинг і прогнозування.

### 3.3. Характеристика інструментальних засобів

Трудомісткість розробки ІІС в значній мірі залежить від використовуваних інструментальних засобів. Інструментальні засоби для розробки інтелектуальних додатків можна класифікувати за такими основними параметрами:

– рівень використовуваної мови;

– парадигми програмування і механізми реалізації;

– спосіб подання знань;

– механізми виведення і моделювання;

– засоби придбання знань;

– технології розробки додатків.

*Рівень використовуваної мови*. Потужність і універсальність мови програмування визначає трудомісткість розробки ЕС.

1. Традиційні (в тому числі об'єктно-орієнтовані) мови програмування типу С, С ++ (як правило, вони використовуються не для створення ЕС, а для створення інструментальних засобів).

2. Спеціальні мови програмування (наприклад, мова LISP, орієнтована на обробку списків; мова логічного програмування PROLOG, мова рекурсивних функцій РЕФАЛ і т.ін.). Їх недоліком є слабка пристосованість до об'єднання з програмами, написаними на мовах традиційного програмування.

3. Інструментальні засоби, що містять багато, але не всі компоненти ЕС (наприклад, система OPS 5, яка підтримує продукційний підхід до подання знань; мови KRL і FRL, використовувані для розробки ЕС з фреймовим поданням знань). Таке програмне забезпечення призначене для розробників, які володіють технологіями програмування і вміють інтегрувати різнорідні компоненти в програмний комплекс.

4. Оболонки ЕС загального призначення, що містять всі програмні компоненти, але не мають знань про конкретні предметні середовища. Засоби цього типу і подальшого не вимагають від розробника додатку знання програмування. Прикладами є ЕКО, Leonardo, Nexpert Object, Kappa, EXSYS, GURU, ART, KEE та ін. Останнім часом все рідше вживається термін «оболонка», його замінюють більш широким терміном «середовище розробки». Якщо хочуть підкреслити, що засіб використовується не тільки на стадії розробки програми, але і на стадіях використання і супроводу, то вживають термін «повне середовище» (complete environment). Для підтримки всього циклу створення і супроводу програм використовуються інтегровані інструментальні системи типу Work Bench, наприклад KEATS, Shelly, VITAL. Основними компонентами системи KEATS є:

ACQUIST – засоби фрагментування текстових джерел знань, що дозволяють розбивати текст або протокол бесіди з експертом на безліч взаємопов'язаних, анотованих фрагментів і створювати поняття (концепти);

FLIK – мова представлення знань засобами фреймовой моделі;

GIS – графічний інтерфейс, використовуваний для створення гіпертекстів і концептуальних моделей, а також для проектування фреймових систем;

ERI – інтерпретатор правил, який реалізує процедури прямого і зворотного виводу;

TRI – інструмент візуалізації логічного висновку, що демонструє послідовність виконання правил;

Tables – інтерфейс маніпулювання таблицями, використовуваними для зберігання знань в БЗ;

CS – мова опису і поширення обмежень;

TMS – немонотонна система підтримки істинності.

При використанні інструментарію даного типу можуть виникнути такі труднощі:

– керуючі стратегії, закладені в механізм виведення, можуть не відповідати методам вирішення, які використовує експерт, що взаємодіє з даною системою, яка може привести до неефективних, а можливо, і неправильних рішень;

– спосіб подання знань, який використовується в інструментарії, мало підходить для опису знань конкретної предметної області.

Велика частина цих труднощів розв'язана в проблемно/предметно-орієнтованих засобах розробки ІІС.

5. Проблемно/предметно-орієнтовані оболонки і середовища (не вимагають знання програмування):

– проблемно-орієнтовані засоби – призначені для вирішення задач певного класу (задачі пошуку, управління, планування, прогнозування та ін.) і містять відповідні цьому класу альтернативні функціональні модулі;

– предметно-орієнтовані засоби – включають знання про типи предметних областей, що скорочує час розробки БЗ.

При використанні оболонок і середовищ розробник програми повністю звільняється від програмування, його основні трудовитрати пов'язані з формуванням бази знань.

*Парадигми програмування і механізми реалізації*. Способи реалізації механізму виконуваних тверджень часто називають *парадигмами програмування*. До основних парадигм відносять такі:

– процедурне програмування;

– програмування, орієнтоване на дані;

– програмування, орієнтоване на правила;

– об'єктно-орієнтоване програмування.

Парадигма *процедурного програмування* є найпоширенішою серед існуючих мов програмування. У процедурній парадигмі активна роль відводиться процедурам, а не даним; причому будь-яка процедура активізується викликом. Подібні способи завдання поведінки зручні для описів детермінованої послідовності дій одного процесу або декількох взаємопов'язаних процесів.

При використанні *програмування, орієнтованого на дані*, активна роль належить даним, а не процедурам. Тут зі структурами активних даних зв'язують деякі дії (процедури), які активізуються тоді, коли здійснюється звернення до цих даних.

У парадигмі, *орієнтованій на правила*, поведінка визначається безліччю правил виду «умова – дія». Умова задає образ даних, при виникненні якого дія правила може бути виконана. Правила в даній парадигмі відіграють таку ж роль, як і оператори в процедурній парадигмі. Однак, якщо в процедурній парадигмі поведінка задається детермінованою послідовністю операторів, що не залежить від значень оброблюваних даних, то в парадигмі, орієнтованої на правила, поведінка не задається заздалегідь визначеною послідовністю правил, а формується на основі значень даних, які в поточний момент обробляються програмою. Підхід, орієнтований на правила, зручний для опису поведінки, гнучко і різноманітно реагує на велике різноманіття станів даних.

Парадигма *об'єктного програмування* на відміну від процедурної парадигми не поділяє програму на процедури і дані. Тут програма організується навколо сутностей, званих об'єктами, які включають локальні процедури (методи) і локальні дані (змінні). Поведінка (функціонування) в цій парадигмі організується шляхом пересилання повідомлень між об'єктами. Об'єкт, отримавши повідомлення, здійснює його локальну інтерпретацію, грунтуючись на локальних процедурах і даних. Такий підхід дозволяє описувати складні системи найбільш природним чином. Він особливо зручний для інтегрованих ЕС.

*Спосіб подання знань*. Наявність багатьох способів подання знань викликано прагненням представити різні типи проблемних середовищ з найбільшою ефективністю. Зазвичай спосіб представлення знань в ЕС характеризують моделлю представлення

знань. Типовими моделями подання знань є правила (продукції), фрейми (або об'єкти), семантичні мережі, логічні формули. Інструментальні засоби, що мають у своєму складі більше однієї моделі подання знань, називають гібридними. Більшість сучасних засобів, як правило, використовує об'єктно-орієнтовану парадигму, об'єднану з парадигмою, орієнтованою на правила.

*Механізми виведення і моделювання.* У статичних ЕС єдиним активним агентом, що змінює інформацію, є механізм виведення експертної системи. У динамічних ЕС зміна даних відбувається не тільки внаслідок функціонування механізму виконуваних тверджень, але також у зв'язку зі змінами оточення задачі, які моделюються спеціальною підсистемою або надходять ззовні. Механізми виведення в різних середовищах можуть відрізнятися способами реалізації наступних процедур.

1. Структура процесу отримання рішення:

– побудова дерева виводу на основі навчальної вибірки (індуктивні методи придбання знань) і вибір маршруту на дереві виведення в режимі рішення задачі;

– компіляція мережі виведення зі специфічних правил в режимі придбання знань і пошук рішення на мережі виводу в режимі рішення задачі;

– генерація мережі виводу і пошук рішення в режимі рішення задачі, при цьому генерація мережі виводу здійснюється в ході виконання операції зіставлення, що визначає пари «правило – сукупність даних», на яких умови цього правила задовольняються;

– в режимі вирішення задач ЕС здійснює вироблення правдоподібних припущень (при відсутності достатньої інформації для вирішення); виконання міркувань щодо обґрунтування (спростування) припущень; генерацію альтернативних мереж виведення; пошук рішення в мережах виведення.

2. Пошук (вибір) рішення:

– напрямок пошуку – від даних до мети, від цілей до даних, двонаправлений пошук;

– порядок перебору вершин в мережі виводу – «пошук в ширину», при якому спочатку обробляються всі вершини, безпосередньо пов'язані з поточною оброблюваною вершиною $G$;

– «пошук в глибину», коли спочатку розкривається одна найбільш значуща вершина – $G_b$ пов'язана з поточною $G$, потім

вершина $G_x$ робиться поточною, і для неї розкривається одна найбільш значуща вершина $G_2$ і т. ін.

3. Процес генерації припущень і мережі виводу:

– режим – генерація в режимі придбання знань, генерація в режимі рішення задачі;

– повнота генеруємої мережі виводу – операція зіставлення застосовується до всіх правил і до всіх типів зазначених в правилах сутностей в кожному циклі роботи механізму виведення (забезпечується повнота генеруємої мережі); використовуються різні засоби для скорочення кількості правил та (або) сутностей, що беруть участь в операції зіставлення; наприклад, застосовується алгоритм зіставлення або використовуються знання більш загального характеру (метазнання).

Механізм виведення для динамічних проблемних середовищ додатково містить: планувальник, керуючий діяльністю ЕС відповідно до пріоритетів; засоби, що гарантують отримання кращого рішення в умовах обмеженості ресурсів; систему підтримки істинності значень змінних, що змінюються в

часі.

У динамічних інструментальних засобах можуть бути реалізовані наступні варіанти підсистеми моделювання:

– система моделювання відсутня;

– існує система моделювання загального призначення, що є частиною інструментального середовища;

– існує спеціалізована система моделювання, що є зовнішньою по відношенню до програмного забезпечення, на якому реалізується ЕС.

*Засоби придбання знань*. В інструментальних системах вони характеризуються такими ознаками:

1. Рівень мови придбання знань:

– формальна мова;

– обмежена природна мова;

– мова піктограм і зображень;

– ПМ і мова зображень.

2. Тип придбаних знань:

– дані у вигляді таблиць, що містять значення вхідних і вихідних атрибутів, за якими індуктивними методами будується дерево виводу;

– спеціалізовані правила;
– загальні і спеціалізовані правила.
3. Тип придбаних даних:
– атрибути і значення;
– об’єкти;
– класи структурованих об’єктів та їх екземпляри, які отримують значення атрибутів шляхом успадкування.

### 3.4. Технологія проектування та розробки експертних систем

Промислова технологія створення інтелектуальних систем включає наступні етапи:
– дослідження здійсненності проекту;
– розробку загальної концепції системи;
– розробку і тестування серії прототипів;
– розробку і випробування головного зразка;
– розробку і перевірку розширених версій системи;
– прив’язку системи до реального робочого середовища.

Проектування ЕС засноване на трьох головних принципах:

1. Потужність експертної системи обумовлена, перш за все потужністю БЗ і можливостями її поповнення і тільки потім – використовуваними методами (процедурами) обробки інформації.

2. Знання, що дозволяють експерту (або експертній системі) отримати якісні та ефективні рішення задач, є в основному евристичними, емпіричними, невизначеними, правдоподібними.

3. Неформальний характер вирішуваних задач і використовуваних знань робить необхідним забезпечення активного діалогу користувача з ЕС в процесі її роботи.

Перед тим як приступити до розробки ЕС, інженер по знаннях повинен розглянути питання, чи слід розробляти ЕС для цього додатка. Позитивне рішення приймається тоді, коли розробка ЕС можлива, виправдана і методи інженерії знань відповідають розв’язуваній задачі.

Щоб розробка ЕС була *можливою* для цієї програми, необхідно виконання, принаймні, таких вимог:

– існують експерти в цій галузі, які вирішують задачу значно краще, ніж фахівці-початківці;

– експерти сходяться в оцінці пропонованого рішення, тому що в противному випадку буде неможливо оцінити якість розробленої ЕС;

– експерти здатні вербалізувати (висловити природною мовою) і пояснити використовувані ними методи, інакше важко розраховувати на те, що знання експертів будуть «витягнуті» і закладені в ЕС;

– рішення задачі вимагає тільки міркувань, а не дій;

– задача не повинна бути занадто важкою (тобто її рішення повинне займати у експерта декілька годин або днів, а не тижнів або років);

– задача хоча й не повинна бути виражена в формальному вигляді, але все ж повинна віноситися до досить «зрозумілої» і структурованої області, тобто повинна існувати можливість виділення основних понять, відносин і способів отримання рішення задачі;

– рішення задачі не повинно в значній мірі спиратися на «здоровий глузд» (тобто широкий спектр загальних відомостей про світ і про спосіб його функціонування, які знає і вміє використовувати будь-яка нормальна людина), так як подібні знання поки не вдається в достатній кількості закласти в системи штучного інтелекту.

Додаток *відповідає* методам ЕС, якщо розв'язувана задача володіє сукупністю наступних характеристик:

– задача може бути природним чином вирішена за допомогою маніпулювання символами (за допомогою символічних міркувань), а не маніпулювання числами, як прийнято в математичних методах і в традиційному програмуванні;

– задача повинна мати евристичну, а не алгоритмічну природу, тобто її рішення має вимагати застосування евристичних правил. Для завдань, які можуть бути гарантовано вирішені (при дотриманні заданих обмежень) за допомогою формальних процедур, існують більш ефективні підходи, ніж технології ЕС.

При розробці ЕС, як правило, використовується концепція *швидкого прототипу*, суть якої полягає в тому, що розробники не намагаються відразу побудувати кінцевий продукт. На початковому етапі вони створюють прототип (можливо, не єдиний) ЕС, що задовольняє двом суперечливим вимогам: вміння вирішувати типові

завдання конкретної програми та незначні час і трудомісткість його розробки. При виконанні цих умов стає можливим паралельно вести процес накопичення і налагодження знань, здійснюваний експертом, і процес вибору (розробки) програмних засобів, що виконується інженером по знаннях і програмістами. Для задовільнення вказаним вимогам при створенні прототипу використовуються різноманітні інструментальні засоби, що прискорюють процес проектування.

Традиційна технологія реалізації ЕС включає шість основних етапів (див. рис. 3.1): ідентифікацію, концептуалізацію, формалізацію, виконання, тестування, дослідну експлуатацію.

На етапі *ідентифікації* визначаються задачі, що виникають, цілі розробки, експерти і типи користувачів.

На етапі *концептуалізації* проводиться змістовний аналіз проблемної області, виявляються використовувані поняття та їх взаємозв'язки, визначаються методи розв'язання задач.


Рис. 3.1. Етапи розробки експертних систем

На етапі *формалізації* обираються інструментальні засоби і способи подання всіх видів знань, формалізуються основні поняття, визначаються способи інтерпретації знань, моделюється робота системи, оцінюється адекватність системи зафіксованих понять, методів вирішення, засобів представлення та маніпулювання знаннями розглянутої предметної області.

На етапі *виконання* здійснюється заповнення бази знань. У зв'язку з тим, що основою ЕС є знання, даний етап є одним з

найважливіших і найбільш трудомістких. Процес придбання знань розділяють на вилучення знань в діалозі з експертами; організацію знань, що забезпечує ефективну роботу системи, і уявлення знань у вигляді, «зрозумілому» ЕС. Процес придбання знань здійснюється інженером по знаннях на основі аналізу діяльності експерта по вирішенню реальних завдань.

На етапі *тестування* експерт і інженер по знаннях в інтерактивному режимі з використанням діалогових і пояснювальних засобів перевіряють компетентність ЕС. Процес тестування триває до тих пір, поки експерт не вирішить, що система досягла необхідного рівня компетентності.

На етапі *дослідної експлуатації* перевіряється придатність ЕС для кінцевих користувачів. Отримані результати можуть показати необхідність суттєвої модифікації ЕС.

Процес створення ЕС не зводиться до суворої послідовності перерахованих вище етапів. В ході розробки доводиться неодноразово повертатися на більш ранні етапи і переглядати прийняті там рішення.

Інструментальні засоби розрізняються залежно від того, яку технологію розробки ЕС вони допускають. Можна виділити, принаймні, чотири підходи до розробки ЕС:

– підхід, який базується на поверхневих знаннях;

– структурний підхід;

– підхід, заснований на глибинних знаннях;

– змішаний підхід, який спирається на використанні поверхневих і глибинних знань.

*Поверхневий підхід* застосовується для складних задач, які не можуть бути точно описані. Його суть полягає в отриманні від експертів фрагментів знань, релевантних розв’язуваній задачі. При цьому не робиться спроб систематичного або глибинного вивчення області, що зумовлює використання пошуку в просторі станів як універсального механізму виведення. Зазвичай в ЕС, що використовують даний підхід, як спосіб представлення знань вибираються правила. Умова кожного правила визначає зразок деякої ситуації, в якій правило може бути виконано. Пошук рішення полягає у виконанні тих правил, зразки яких зіставляються з поточними даними. При цьому передбачається, що в процесі пошуку рішення послідовність формованих таким чином ситуацій не

обірветься до отримання рішення, тобто не виникне невідомої ситуації, яка не відповідає жодному правилу. Даний підхід з успіхом застосовується до широкого класу додатків, але виявляється неефективним в тих випадках, коли задача може структуруватися або для її вирішення може використовуватися деяка модель.

*Структурний підхід* до побудови ЕС передбачає структуризацію знань проблемної області. Його поява обумовлена тим, що для ряду додатків застосування техніки поверхневих знань не забезпечує вирішення задачі. Структурний підхід до побудови ЕС багато в чому схожий на структурне програмування. Однак стосовно ЕС мова йде не про те, щоб структурування задачі було доведено до точного алгоритму (як в традиційному програмуванні), а передбачається, що частина задачі вирішується за допомогою евристичного пошуку. Структурний підхід в різних додатках доцільно поєднувати з поверхневим або глибинним.

При *глибинному підході* компетентність ЕС базується на моделі того проблемного середовища, в якому вона працює. Модель може бути визначена різними способами (декларативно, процедурно). Необхідність використання моделей в ряді програм викликана прагненням виправити недолік поверхневого підходу, пов'язаний з виникненням ситуацій, які не описані правилами, що зберігаються у БЗ. Експертні системи, розроблені із застосуванням глибинних знань, при виникненні невідомої ситуації здатні самостійно визначити, які дії слід виконати, за допомогою деяких загальних принципів, справедливих для даної галузі експертизи.

Глибинний підхід вимагає явного опису структури і взаємин між різними сутностями проблемної області. У цьому підході необхідно використовувати інструментальні засоби, що володіють можливостями моделювання: об'єкти з приєднаними процедурами, ієрархічне спадкування властивостей, активні знання (програмування, кероване даними), механізм передачі повідомлень об'єктам (об'єктно-орієнтоване програмування) і т. п.

*Змішаний підхід* в загальному випадку може поєднувати поверхневий, структурний і глибинний підходи. Наприклад, поверхневий підхід може застосовуватися для пошуку адекватних знань, які потім використовуються деякою глибинною моделлю.

# ЛЕКЦІЯ 4. СПОСОБИ ПРЕДСТАВЛЕННЯ ЗНАНЬ

## 4.1. Дані і знання

Необхідною частиною будь-якої інтелектуальної системи є знання. Теоретичними та практичними питаннями представлення та обробки знань в комп'ютерних системах активно займаються дослідники, що працюють в області *інженерії знань*. Це поняття у 1977 р. ввів Е. Фейгенбаум. Характерною ознакою інтелектуальних систем є наявність знань, необхідних для вирішення задачі конкретної предметної області. При цьому виникає природне запитання, що таке знання і чим воно відрізняються від звичайних даних, що обробляються ЕОМ.

*Дані* – це інформація, отримана в результаті спостережень або вимірювань окремих властивостей (атрибутів), що характеризують об'єкти, процеси і явища предметної області.

У процесах комп'ютерної обробки дані проходять наступні етапи перетворень:

– вихідна форма існування даних (результати спостережень і вимірів, таблиці, довідники, діаграми, графіки і т.д.);

– подання на спеціальних мовах опису даних, призначених для введення та обробки вихідних даних в ЕОМ;

– бази даних на машинних носіях інформації.

*Знання* – форма існування і систематизації результатів пізнавальної діяльності людини. Знання допомагає людям раціонально організовувати свою діяльність і вирішувати різні проблеми, що при цьому виникають; суб'єктивний образ об'єктивної реальності, тобто адекватне віддзеркалення зовнішнього і внутрішнього світу в свідомості людини у формі уявлень, понять, суджень, теорій.

Знання (*в широкому сенсі*) – сукупність понять, теоретичних побудов і уявлень.

Знання (*у вузькому сенсі*) – ознака певного обсягу інформації, що визначає її статус і відокремлює від всієї іншої інформації за критерієм здатності до вирішення поставленої задачі.

Знання (*з точки зору представлення знань в інтелектуальних системах*) – це зв'язки і закономірності предметної області (принципи, моделі, закони), отримані в результаті практичної

діяльності і професійного досвіду, що дозволяє фахівцям ставити і вирішувати задачі в цій галузі.

Знання від даних відрізняються рядом ***властивостей***:

– внутрішня інтерпретованість;

– структурованість;

– зв'язність;

– семантична метрика;

– активність.

*Внутрішня інтерпретованість.* Дані, що зберігаються в пам'яті або на зовнішніх носіях, позбавлені імен, таким чином, відсутня можливість їх однозначної ідентифікації системою. Дані може ідентифікувати лише програма, витягаючи їх за певним алгоритмом. При переході до знань в пам'ять вводиться додаткова інформація (атрибути: прізвище, рік народження, спеціальність, стаж). Атрибути можуть грати роль імен. За ними можна здійснювати пошук потрібної інформації.

*Структурованість.* Інформаційні одиниці повинні володіти гнучкою структурою. Інакше кажучи, повинна існувати можливість довільного встановлення між окремими інформаційними одиницями відносин типу «частина – ціле», «рід – вид» або «елемент – клас».

*Зв'язок*. Між інформаційними одиницями повинна бути передбачена можливість встановлення зв'язків різного типу. Семантика відносин може носити декларативний або процедурний характер. Наприклад, дві і більше інформаційні одиниці можуть бути пов'язані відношенням «одночасно», дві інформаційні одиниці – відношенням «причина – наслідок» або «бути поруч».

*Семантична метрика*. На безлічі інформаційних одиниць в деяких випадках корисно задавати відношення, що характеризує їх ситуаційну близькість, тобто силу асоціативного зв'язку. Його можна було б назвати відношенням релевантності для інформаційних одиниць. Воно дає можливість виділяти в інформаційній базі деякі типові ситуації (наприклад, «покупка», «регулювання руху на перехресті»). Відношення релевантності при роботі з інформаційними одиницями дозволяє знаходити знання, близькі до вже знайдених.

*Активність.* Всі обчислювальні процеси ініціюються командами, а дані використовуються цими командами лише в разі потреби. Інакше кажучи, дані пасивні, а команди активні.

Знання дозволяють адаптуватися і діяти в реальній дійсності. Існує величезна безліч різних знань, починаючи від рецепта приготування омлету до квантової фізики. Знання можна класифікувати за кількома критеріями (рис. 4.1).


Рис. 4.1. Класифікація знань

Знання *синтаксичного* типу характеризує синтаксичну структуру потоку інформації, яка не залежить від сенсу і змісту понять, які при цьому використовуються, тобто інтелектуальну систему не утворює.

*Семантичне* знання розглядається як структура, що утворює поточний контекст. Воно містить інформацію, безпосередньо

пов’язану з поточними значеннями і змістом описуваних понять і визначає стан зв’язків даних в інформаційній базі.

*Прагматичне* знання зумовлює найбільш ймовірні зв’язки, що описують дані з точки зору розв’язуваної задачі (узагальнений або "об’єктивний" контекст), наприклад, з урахуванням діючих в даній задачі специфічних критеріїв і угод.

*Декларативні* знання містять у собі уявлення про структуру понять. Ці знання наближені до даних, фактів. Наприклад: вищий навчальний заклад є сукупність факультетів, а кожен факультет, в свою чергу, є сукупність кафедр.

*Процедурні* знання, мають активну природу. Вони визначають уявлення про засоби і шляхи отримання нових знань, перевірки знань. Це алгоритми різного роду. З розвитком інформатики все більша частина знань зосереджувалася в структурах даних (таблиці, списки, абстрактні типи даних), тобто збільшувалася роль декларативних.

Істотними для розуміння природи знань є способи визначення понять. Один з широко застосовуваних способів заснований на ідеї інтенсіоналу та екстенсіоналу.

*Інтенсіонал поняття* – це визначення його через співвіднесення з поняттям більш високого рівня абстракції із зазначенням специфічних властивостей.

*Екстенсіонал поняття* – це визначення поняття через перерахування його конкретних прикладів, тобто понять більш низького рівня абстракції. Інтенсіоналом формують знання про об’єкти, в той час як екстенсіонал об’єднує дані.

Звідси *інтенсіональні* знання – це знання про предметну область, яка відображає факти, закономірності, властивості і характеристики, справедливі для будь-яких ситуацій, які можуть виникнути в цій предметній області.

*Екстенсіональні* знання – це знання про предметну область, що відображають факти, закономірності, властивості і характеристики, типові для конкретних ситуацій або класів однотипних ситуацій, які можуть виникнути в цій області.

*Функціональні* знання – це знання про виконувані функції окремих предметів і про застосування їх в реальній дійсності.

*Технологічні* знання – спеціалізовані знання, щоб забезпечити підтримку технологічних параметрів виробництва; виробничий

досвід і навички, які використовуються при вирішенні повсякденних виробничих питань. Це може бути знання послідовності операцій або знання технологічного ланцюжка, які дозволяють досягати поставлених цілей відповідно до прийнятої технології.

*Методологічні* знання – знання про методи перетворення дійсності, наукові знання про побудову ефективної діяльності. До методологічних знань відносять знання цілей, форм і напрямків розвитку теорії, методів і способів ефективного перетворення практики.

*Класифікаційні* знання застосовуються головним чином в науці, є узагальненими, системними знання. Приклад – система елементів Д.І. Менделєєва.

*Інтуїція* – це вид знання, специфіка якого обумовлена способом його придбання. Це знання, що не потребує доведення і сприймається як достовірне. За способом отримання інтуїція – це прямий розсуд об'єктивного зв'язку речей, що не спирається на доказ (інтуїція, від лат. intueri – споглядати, – є розсуд внутрішнім зором).

Під *здоровим глуздом* розуміють знання, які дозволяють приймати правильні рішення і робити правильні припущення, ґрунтуючись на логічному мисленні і накопиченому досвіді. У цьому значенні термін часто акцентує увагу на здатності людського розуму протистояти забобонам, помилкам, містифікаціям.

*Наукові* знання в будь-якому випадку повинні бути обгрунтованими на емпіричній або теоретичній доказовій основі.

*Теоретичні* знання – абстракції, аналогії, схеми, що відображають структуру і природу процесів, що протікають в предметній області. Ці знання пояснюють явища і можуть використовуватися для прогнозування поведінки об'єктів. Теоретичний рівень наукового знання припускає встановлення законів, що дають можливість ідеалізованого сприйняття, описи і пояснення емпіричних ситуацій, тобто пізнання сутності явищ. Теоретичні закони мають більш строгий, формальний характер, в порівнянні з емпіричними. Терміни опису теоретичного знання відносяться до ідеалізованих, абстрактних об'єктів. Подібні об'єкти неможливо піддати безпосередній експериментальній перевірці.

*Емпіричні* знання отримують в результаті застосування емпіричних методів пізнання – спостереження, вимірювання, експерименти. Це знання про видимі взаємозв'язки між окремими

подіями і фактами в предметній області. Воно, як правило, констатує якісні та кількісні характеристики об'єктів і явищ. Емпіричні закони часто носять імовірнісний характер і не є строгими.

*Позанаукові* знання можуть бути різними.

*Паранормальні* знання – знання несумісні з наявним гносеологічним стандартом.

Широкий клас *паранаукового* (пара від грец. – близько, при) знання включає в себе вчення або роздуми про феномени, пояснення яких не є переконливим з точки зору критеріїв науковості.

*Лженаукове* знання – свідомо експлуатуючі домисли і забобони. Як симптоми лженауки виділяють малограмотний пафос, принципову нетерпимість до спростовуючих доводів, а також претензійність. Лженаукове знання співіснує з науковими знаннями.

*Особистісні* (неявні, приховані) знання – це знання людей, отримані з практики і досвіду.

*Формалізовані* (явні) знання – знання, що містяться в документах, на компакт дисках, в персональних комп'ютерах, в Інтернеті, в базах знань, в експертних системах. Формалізовані знання об'єктивізуються знаковими засобами мови, охоплюють ті знання, про які ми знаємо, їх можна записати, повідомити іншим.

Для того щоб наділити ІІС знаннями, їх необхідно представити в певній формі. Існують два основних способи наділення знаннями програмних систем.

*Перший* – помістити знання в програму, написану на звичайній мові програмування. Така система буде являти собою єдиний програмний код, в якому знання не винесені в окрему категорію. Незважаючи на те, що основне завдання буде вирішене, в цьому випадку важко оцінити роль знань і зрозуміти, яким чином вони використовуються в процесі вирішення завдань. Нелегкою справою є модифікація і супровід подібних програм, а проблема поповнення знань може стати нерозв'язною.

*Другий* спосіб базується на концепції баз даних і полягає у винесенні знань в окрему категорію, тобто знання представляються в певному форматі і поміщаються в БЗ. База знань легко поповнюється і модифікується. Вона є автономною частиною інтелектуальної системи, хоча механізм логічного висновку, реалізований в логічному блоці, а також засоби ведення діалогу

накладають певні обмеження на структуру БЗ і операції з нею. В сучасних ІІС прийнятий цей спосіб.

Слід зауважити, що для того, щоб помістити знання в комп'ютер, їх необхідно представити певними структурами даних, відповідних обраному середовищу розробки інтелектуальної системи. Отже, при розробці ІІС спочатку здійснюються накопичення і представлення знань, причому на цьому етапі обов'язкова участь людини, а потім знання представляються певними структурами даних, зручними для зберігання і обробки в ЕОМ.

Знання в ІІС існують в наступних формах:

– вхідні знання (правила, виведені на основі практичного досвіду, математичні та емпіричні залежності, що відображають взаємні зв'язки між фактами; закономірності і тенденції, що описують зміну фактів з плином часу; функції, діаграми, графи і т.д.);

– опис вихідних знань засобами обраної моделі подання знань (безліч логічних формул або продукційних правил, семантична мережа, ієрархії фреймів і т. п.);

– уявлення знань структурами даних, які призначені для зберігання і обробки в ЕОМ;

– бази знань на машинних носіях інформації.

### 4.2. Класифікація моделей подання знань

Для зберігання даних використовуються бази даних (для них характерні великий обсяг і відносно невелика питома вартість інформації), для зберігання знань – бази знань (невеликого обсягу, але виключно дорогі інформаційні масиви).

*База знань* – основа будь-якої інтелектуальної системи, де знання описані на деякій мові представлення знань, наближеній до природної. Сьогодні знання придбали чисто декларативну форму, тобто знаннями вважаються пропозиції, записані на мовах подання знань, наближених до природної мови і зрозумілих неспеціалістам.

Сукупність знань потрібних для прийняття рішень, прийнято називати предметною областю або *знаннями про предметну область*. У будь-якій предметній області є свої поняття і зв'язки між ними, своя термінологія, свої закони, що зв'язують між собою об'єкти даних предметної області, свої процеси і події. Крім того,

кожна предметна область має свої методи вирішення завдань. Вирішуючи завдання такого виду на ЕОМ, використовують інформаційні системи, ядром яких є база знань, яка містить основні характеристики предметних областей.

Бази знань базуються на *моделях подання знань*, подібно базам даних, які засновані на моделях подання даних (ієрархічній, мережевій, реляційній, постреляційній і т.д.). При поданні знань в пам'яті інтелектуальної системи традиційні мови, засновані на чисельному поданні даних, є неефективними. Для цього використовуються спеціальні мови подання знань, засновані на символьному поданні даних. Вони діляться на типи за формальними моделям подання знань.

Найбільш часто використовується на практиці класифікація моделей подання знань, наведена на рис. 4.2, де моделі подання знань діляться на детерміновані (жорсткі) і м'які.


Рис. 4.2. Моделі подання знань

*Детерміновані* моделі включають в себе фрейми, логіко-алгебраїчні моделі, семантичні мережі і продукційні моделі. М'які моделі включають в себе нечіткі системи, нейронні мережі, еволюційні моделі, гібридні системи.

З моделюванням знань безпосередньо пов'язана проблема вибору мови подання. З метою класифікації моделей подання знань виділяється дев'ять ключових вимог до моделей знань:

1) спільність (універсальність);

2) наочність подання знань;

3) однорідність;

4) реалізація в моделі властивості активності знань;

5) відкритість;

6) можливість відображення структурних відносин об'єктів предметної області;

7) наявність механізму «проєктування» знань на систему семантичних шкал;

8) можливість оперування нечіткими знаннями;

9) використання багаторівневих уявлень (дані, моделі, метамоделі, метаметамоделі і т. д.).

Моделі подання знань не задовольняють повністю цим вимогам, чим і пояснюється їх різноманіття і активний розвиток цього напрямку.

*Логічна модель подання знань*

Логічна модель застосовується в основному в дослідницьких системах, оскільки пред'являє дуже високі вимоги до якості і повноти знань предметної області.

У логічних моделях знання представляються у вигляді сукупності правильно побудованих формул будь-якої формальної системи, яка задається четвіркою

$$S = <T, P, A, R>,$$

де T – множина базових (термінальних) елементів, з яких формуються всі вирази;

P – множина синтаксичних правил, що визначають синтаксично правильні вирази з термінальних елементів формальної системи;

A – множина аксіом формальної системи, відповідних синтаксично правильним виразам, які в рамках даної ФС апріорно вважаються істинними;

R – кінцева множина відносин $\{r_1, r_2, ... ,r_n\}$ між формулами, які називаються правилами виведення, що дозволяють отримувати з одних синтаксично правильних виразів інші.

Для будь-якого $r_i$ існує ціле позитивне число $j$, таке, що для кожної множини, що складається з $j$ формул, і для кожної формули $F$ ефективно вирішується питання про те, чи знаходяться ці $j$-формули у відношенні $r_i$ з формулою $F$. Якщо $r_i$ виконується, то $F$ називають безпосереднім наслідком $F$-формул за правилом $r_i$.

Наслідком (висновком) формули в теорії S називається така послідовність правил, що для будь-якого з них представлена формула є або аксіомою теорії S, або безпосереднім наслідком.

Найпростішою логічною моделлю є числення висловів, яке представляє собою один з початкових розділів математичної логіки, що є основою для побудови більш складних формалізмів. У практичному плані обчислення висловлювань застосовується в ряді предметних областей (зокрема, при проектуванні цифрових електронних схем). Розвиток логіки висловлювань знайшло відображення в численні предикатів першого порядку.

Знайомство з логікою предикатів почнемо з обчислення висловлювань.

*Висловлюванням* називається вислів, зміст якого можна висловити значеннями: вірно (*T*) або невірно (*F*). Наприклад, речення «*лебідь білий*» і «*лебідь чорний*» будуть висловлюваннями. З простих висловлювань можна скласти більш складні:

«*лебідь білий або лебідь чорний*»,

«*лебідь білий і лебідь чорний*»,

«*якщо лебідь не білий, то лебідь чорний*».

У свою чергу, складні висловлювання можна розділити на часткові, які пов'язані між собою за допомогою слів: і, або, не, якщо – то. Елементарними називаються висловлювання, які не можна розділити на частини. Логіка висловлювань оперує логічними зв'язками між висловлюваннями, тобто вона вирішує питання типу: «Чи можна на основі висловлювання *A* отримати висловлювання *B*?»; «Чи істинно *B* при істинності *A*?» і т.п. При цьому семантика висловлювань не має значення. Елементарні висловлювання розглядаються як змінні логічного типу, над якими дозволені такі логічні операції:

| | |
|---|---|
| $\neg$ | НЕ, заперечення (унарна операція), |
| $\wedge$ | І, кон'юнкція (логічне множення), |
| $\vee$ | АБО, диз'юнкція (логічне додавання), |
| $\rightarrow$ | ЯКЩО – ТО, імплікація. Результат імплікації збігається з результатом виразу $\neg A \vee B$, |
| $\leftrightarrow$ | еквівалентність, |

а також квантор існування $\exists$ і квантор спільності $\forall$, які необхідні для визначення області дії змінних.

Змінні, що знаходяться у сфері дії кванторів, називаються пов'язаними, інші змінні в логічних формулах називаються вільними. Для того щоб можна було говорити про істинність будь-якого затвердження без підстановки значень в змінні, всі вхідні в нього змінні повинні бути пов'язані кванторами.

Якщо в логічну формулу входить кілька кванторів, необхідно враховувати їх взаємне розташування, для усунення невизначеності введемо дужки і порядок застосування кванторів – зліва направо.

Значення результатів логічних операцій над змінними $A$ і $B$, які є елементарними висловлюваннями, наведені в табл. 4.1.

Таблиця 4.1

**Результати обчислення логічних операцій**

| $A$ | $\neg A$ | $B$ | $A \wedge B$ | $A \vee B$ | $A \to B$ | $A \leftrightarrow B$ |
|---|---|---|---|---|---|---|
| $T$ | $F$ | $T$ | $T$ | $T$ | $T$ | $T$ |
| $T$ | $F$ | $F$ | $F$ | $T$ | $F$ | $F$ |
| $F$ | $T$ | $T$ | $F$ | $T$ | $T$ | $F$ |
| $F$ | $T$ | $F$ | $F$ | $F$ | $T$ | $T$ |

Обчислення висловлювань дозволяє формалізувати лише малу частину множини міркувань, оскільки цей апарат не дозволяє враховувати внутрішню структуру висловлювання, яка існує в природних мовах.

Розглянемо приклад міркування про Сократа, що став класичним:

$P$: «Усі люди смертні»;
$Q$: «Сократ – людина»;
$R$: «Сократ – смертний».

Використовуючи для позначення висловлювань логічні змінні $P$, $Q$, $R$, можна скласти формулу: $(P \wedge Q) \to R$, яка може бути інтерпретована як «Якщо усі люди смертні і Сократ є людиною, то Сократ є смертним». Однак ця формула не є загальнозначущою, оскільки стосується тільки одного об'єкта – Сократа. Крім того, висловлювання $R$ не виводиться з $P$ і $Q$, тобто, якби ми не сформулювали $R$ заздалегідь, ми не змогли б записати наведену вище формулу.

Щоб здійснити цей примітивний логічний висновок, висловлювання $Q$ слід розділити на дві частини: «Сократ» (суб'єкт) і «людина» (властивість суб'єкта) і представити у вигляді

відношення «суб'єкт – властивість», яке можна записати за допомогою функції людина (Сократ).

Очевидно, що властивість конкретного суб'єкта з ім'ям «Сократ» бути «людиною» може бути притаманне і ряду інших суб'єктів, що дозволяє замінити константу «Сократ» на деяку змінну, наприклад *X*. Тоді отримаємо запис *людина (X)*, яка володіє внутрішньою структурою, тобто значення такого висловлювання буде залежати від його компонент. Записана функція вже не є елементарним висловлюванням, вона називається предикатом.

Наведемо пояснення поняття предиката, дане Д. А. Поспєловим: «Під предикатом будемо розуміти деякий зв'язок, який заданий на наборі з констант або змінних».

Приклад предиката: «*P* більше *Q*».

Якщо семантика *P* і *Q* не задана, то про предикат сказати особливо нічого. Мабуть, тільки те, що він є антирефлексивним, антисиметричним і транзитивним. Але при завданні семантики (тобто областей визначення змінних *P* і *Q* про предикат можна буде сказати значно більше. Наприклад, якщо *P* і *Q* – площі міст в Україні і Японії, то при завданні списків міст і підстановці значень з цих списків в змінні ми отримаємо відношення між двома сутностями і зможемо судити про його істинность, наприклад:

«Площа Києва більше площі Харкова» = T;

«Площа Чернігова більше площі Полтави» = F.

Іноді для твердження про істинність або хибність предиката можна обійтися без підстановки. Наприклад, якщо областю визначення змінної *X* є цілі позитивні числа, то предикат «*X* більше –5» буде тотожно істинний».

Основними синтаксичними одиницями логіки предикатів є константи, змінні, функції, предикати, квантори і логічні оператори. Формальний синтаксис обчислення предикатів першого порядку зручно представити в нормальній формі Бекуса-Наура, яка традиційно застосовується для запису граматик мов програмування.

<Константа> → <ідентифікатор 1>

<Змінна> → <ідентифікатор 2>

<Функція> → <ідентифікатор 3>

<Предикат> → <ідентифікатор 4>

<Терм> → <константа> | <змінна> | <функція> (<список термів>)

<Список термів> → <терм> | <терм>, (<список термів>)

<Атом> → <предикат> | <предикат> (<список термів>)

<Літера> → <атом> | ¬ <атом>

<Оператор> → ∧ | ∨ | → | ↔

<Список змінних> → <змінна> | <змінна>,<список змінних>

<Квантор> → <(∃<список змінних>)> | <(∀<список змінних>)>

<Формула> → <літера> | ¬ <формула> | <квантор> (<формула>) |

(<формула>) <оператор> (<формула>)

В даному записі будь-яке ім'я в кутових дужках являє собою тип синтаксичного об'єкта. Визначення кожного типу починається з появи його імені в лівій частині кожного запису, тобто ліворуч від знака →. У правій частині кожного запису наводяться можливі способи організації синтаксично коректних об'єктів визначаємого типу. Альтернативні варіанти розділені знаком |, який можна інтерпретувати як АБО. Номери ідентифікаторів слід трактувати в тому сенсі, що ідентифікатори, використовувані для позначення об'єктів різних типів, повинні бути помітними. Наприклад, константи позначаються іменами <ідентифікатор1>, які формуються з малих літер, причому першим символом повинен бути один з наступних: *a, b, c, d, e, к, l, m, n, x, y, z, v, w, u*. Імена змінних <ідентифікатор2> повинні починатися, наприклад, з великої літери. Ідентифікатори функцій <ідентифікатор3> складаються з малих літер, при цьому першою є *g, h, p* або *q*. Імена предикатів <ідентифікатор4> повинні складатися з великих літер.

Функції, як і предикати, задають деякий зв'язок між змінними або константами. Але цей зв'язок або відношення не характеризуються істинностним значенням. За допомогою функції можна представити складний об'єкт, наприклад, функція fbook (Author, Tytle, Publisher, Year) представляє набір інформації, що характеризує книгу. Предикат і функція відрізняються також на синтаксичному рівні, а саме: функції можуть бути аргументами предикатів (тобто термами), а предикати – ні. У логіці предикатів вищих порядків в порівнянні з першим аргументами предикатів можуть бути інші предикати. Функції з нульовим числом місць

(аргументів) є аналогами констант. Предикат без аргументів відповідає вислову.

Предикатні символи – це в основному дієслівна форма (наприклад: ПИСАТИ, ВЧИТИ, ПЕРЕДАТИ), але не тільки дієслівна форма, а форма прикметників, прислівників (наприклад: ЧЕРВОНИЙ, ЗНАЧЕННЯ, ЖОВТИЙ).

Будь-яку формулу в логіці предикатів можна представити у вигляді попередньої нормальної форми (ПНФ), в якій спочатку виписуються всі квантори, а потім – предикатні вирази, наприклад:

$$(\forall X)(\forall Y)(\exists Z)\big(P(X)\to Q(Y,Z,W)\big).$$

Формула, в якій всі змінні пов'язані, називається реченням. Кожному реченню можна поставити у відповідність певне значення – «вірно» або «невірно».

Приклад: нехай $f(X)$ – функція, що задає відношення «батько»; $P(X)$ – предикат, що задає відношення «людина». Тоді логічна формула буде інтерпретуватися як «Всі істоти, батьком яких є людина, – люди».

Операції в логіці предикатів мають неоднакові пріоритети. Найвищий пріоритет має квантор спільності, найнижчий – операція еквівалентності.

$$\forall, \exists, \neg, \wedge, \vee, \to, \leftrightarrow \longrightarrow$$

спадання пріоритету

Складні формули в логіці предикатів виходять шляхом комбінування атомарних формул за допомогою логічних операцій. Такі формули називаються правильно побудованими логічними формулами (ППФ). Інтерпретація ППФ можлива тільки з урахуванням конкретної *області інтерпретації*, яка представляє собою безліч всіх можливих значень термів, що входять в ППФ. Для представлення знань конкретної предметної області у вигляді ППФ необхідно перш за все встановити область інтерпретації (світ Хербранда), тобто вибрати константи, які визначають об'єкти в даній області, а також функції та предикати, які визначають залежності та відносини між об'єктами. Після цього можна побудувати логічні формули, що описують закономірності даної предметної області.

Записати знання за допомогою логічної моделі не вдається в тих випадках, коли утруднений вибір зазначених трьох груп елементів (констант, функцій і предикатів) або коли для опису цих знань не вистачає можливостей представлення за допомогою ППФ, наприклад, коли знання є неповними, ненадійними, нечіткими і т.д.

Позитивними рисами логічних моделей знань в цілому є:

– високий рівень формалізації, що забезпечує можливість реалізації системи формально точних визначень і висновків;

– узгодженість знань як єдиного цілого, що полегшує вирішення проблем верифікації БЗ, оцінки незалежності та повноти системи аксіом і т. д.;

– єдині засоби опису як знань про предметну область, так і способів вирішення завдань в цій предметній області, що дозволяє будь-яке завдання звести до пошуку логічного виведення деякої формули в тій чи іншій формальній системі.

Однак така однаковість тягне за собою основний недолік моделі – складність використання в процесі логічного висновку евристик, що відображають специфіку предметної області. До інших недоліків логічної моделі відносять:

– "монотонність";

– "комбінаторний вибух";

– слабкість структурованості описів.

*Продукційна модель подання знань*

Продукційна модель в силу своєї простоти отримала найбільш широке поширення.

Продукційна модель або модель, заснована на правилах, дозволяє подання знання у вигляді речень типу "Якщо (умова), то (дія)". До складу експертної системи продукційного типу входить база правил, база фактичних даних (робоча пам’ять) та інтерпретатор правил, який реалізує певний механізм логічного висновку. Будь-яке продукційне правило, яке міститься в БЗ, складається з двох частин: *антецедента* та *консеквента*.

Під "умовою" (антецедентом) розуміється деяке речення – зразок, за яким здійснюється пошук в базі знань, яке складається з елементарних речень, з’єднаних логічними зв’язками І, АБО, а під "дією" (консеквентом) – дії, що виконуються при успішному результаті пошуку (вони можуть бути проміжними, які виступають

далі як умови, і термінальними або цільовими, які завершають роботу системи).

Приклади продукційних правил:

ЯКЩО «*двигун не заводиться*» І «*стартер двигуна не працює*», ТО «*неполадки в системі електроживлення стартера*»;

ЯКЩО «*тварина має пір'я*», ТО «*тварина – птах*»

Будь-яке правило складається з однієї або декількох пар *атрибут – значення*. У робочій пам'яті виробничої системи зберігаються пари *атрибут – значення*, істинність яких встановлена в процесі вирішення конкретної задачі для деякого поточного моменту часу. Вміст робочої пам'яті змінюється в процесі виконання задачі. Це відбувається в міру спрацювання правил. Правило *спрацьовує*, якщо при зіставленні фактів, що містяться в робочій пам'яті, з антецедентом аналізованого правила має місце збіг, при цьому висновок правила, що спрацювало, заноситься в робочу пам'ять. Тому в процесі логічного висновку обсяг фактів в робочій пам'яті, як правило, збільшується (зменшуватися він може в тому випадку, якщо дія якого-небудь правила полягає у видаленні фактів з робочої пам'яті). В процесі логічного висновку кожне правило з бази правил може спрацювати тільки один раз. При описі реальних знань конкретної предметної області може виявитися недостатнім подання фактів за допомогою пар *атрибут – значення*.

Ширші можливості має спосіб опису за допомогою триплетів *об'єкт – атрибут – значення*. В цьому випадку окрема сутність предметної області розглядається як об'єкт, а дані, що зберігаються в робочій пам'яті, показують значення, які приймають атрибути цього об'єкта.

Приклади триплетів:

*собака – кличка – Граф;*

*собака – порода – різеншнауцер;*

*собака – окрас – чорний.*

Однією з переваг такого подання знань є уточнення контексту, в якому застосовуються правила. Наприклад, правило, що відноситься до об'єкту «собака», має бути застосовано для собак з будь-якими прізвиськами, всіх порід і забарвлень. З введенням триплетів правила з бази правил можуть спрацьовувати більше ніж один раз в процесі одного логічного висновку, оскільки одне

правило може застосовуватися до різних екземплярів об'єкта (але не більше одного разу до кожного примірника).

Продукційна модель в чистому вигляді не має механізму виходу з тупикових станів в процесі пошуку. Вона продовжує працювати, поки не будуть вичерпані всі допустимі продукції. Практичні реалізації продукційних систем містять механізми повернення в попередній стан для управління алгоритмом пошуку.

Існують два типи продукційних систем – з прямими і зворотними висновками. *Прямі* висновки реалізують стратегію «від фактів до висновків».

При *зворотних* висновках висуваються гіпотези ймовірних висновків, які можуть бути підтверджені або спростовані на підставі фактів, що надходять в робочу пам'ять. Існують також системи з двонаправленими висновками.

Основні переваги продукційних систем:

– простота і гнучкість виділення знань;

– відділення знань від програми пошуку;

– модульність продукційних правил (правила не можуть "викликати" інші правила);

– можливість евристичного управління пошуком;

– можливість трасування "ланцюжка міркувань";

– незалежність від вибору мови програмування;

– продукційні правила є правдоподібною моделлю рішення задачі людиною.

До недоліків систем продукцій можна віднести наступні:

– відміну від структур знань, властивих людині;

– неясність взаємних відносин правил;

– складність оцінки цілісного образу знань;

– низька ефективність обробки знань.

При розробці невеликих систем (десятки правил) проявляються в основному позитивні сторони систем продукцій, однак при збільшенні обсягу знань більш помітними стають слабкі сторони.

Є велика кількість програмних засобів, що реалізують продукційний підхід (наприклад, мови високого рівня CLIPS та OPS 5; "оболонки" або "порожні" ЕС – EXSYS Professional та Kappa, інструментальні КЕЕ, ARTS, PIES, а також промислові ЕС на його основі).

*Семантичні мережі*

Загальноприйнятого визначення семантичної мережі не існує. Зазвичай під нею мають на увазі систему знань деякої предметної області, що має певний сенс у вигляді цілісного образу мережі, вузли якої відповідають поняттям і об'єктам, а дуги – відносинам між об'єктами. При побудові семантичної мережі відсутні обмеження на число зв'язків і на складність мережі. Для того щоб формалізація виявилася можливою, семантичну мережу необхідно систематизувати.

*Семантична мережа* – це орієнтований граф, вершини якого – поняття, а дуги – відносини між ними. Термін "*семантична*" означає "смислова", а сама семантика – це наука, що встановлює відносини між символами і об'єктами, які вони позначають, тобто наука, яка визначає зміст знаків. Модель на основі семантичних мереж була запропонована американським психологом Куілліаном.

Семантичні мережі Куілліана систематизують функції відносин між поняттями за допомогою таких ознак:

– множина – підмножина (типи відносин «абстрактне – конкретне», «ціле – частина», «рід – вид»);

– індекси (властивості, прикметники в мові і т.п.);

– кон'юнктивні зв'язки (логічне І);

– диз'юнктивні зв'язки (логічне АБО);

– зв'язки по ВИКЛЮЧАЮЧОМУ АБО;

– відносини «близькості»;

– відносини «подібності – відмінності»;

– відносини «причина – наслідок» та ін.

При побудові семантичної мережі відсутні обмеження на число елементів і зв'язків. Тому систематизація відносин між об'єктами в мережі необхідна для подальшої формалізації. Приклад семантичної мережі представлений на рис. 4.3.

В якості понять зазвичай виступають абстрактні або конкретні об'єкти, а *відносини* це зв'язки типу: "це" ("АКО – A-Kind-Of, "is" або "елемент класу"), "має частиною" ("has part"), "належить", "любить".

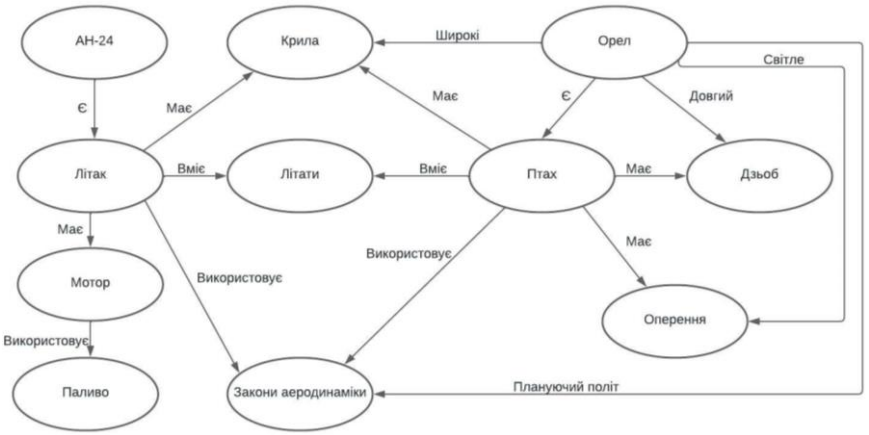


Рис. 4.3. Семантична мережа, що показує взаємини птаха та літака

Можна запропонувати кілька класифікацій семантичних мереж, пов’язаних з типами відносин між поняттями.

За кількістю типів відносин:

– однорідні (з єдиним типом відносин);

– неоднорідні (з різними типами відносин).

За типами відносин:

– бінарні (в яких відносини пов’язують два об’єкти);

– N-арні (в яких є спеціальні відносини, що зв’язують більше двох понять).

Найбільш часто в семантичних мережах використовуються наступні відносини:

– елемент класу (троянда це квітка);

– атрибутивні зв'язки / мати властивість (пам’ять має властивість – обсяг);

– значення властивості (колір має значення – жовтий);

– приклад елемента класу (троянда, наприклад – чайна);

– зв’язку типу "частина-ціле" (велосипед включає кермо);

– функціональні зв’язки (визначені зазвичай дієсловами "виробляє", "впливає" ...);

– кількісні (більше, менше, дорівнює ...);

– просторові (далеко від, близько від, за, під, над ...);

– тимчасові (раніше, пізніше, протягом ...);

– логічні зв’язки (та, або, не) та ін.

Мінімальний склад відносин в семантичній мережі такий:

– елемент класу або АКО;

– атрибутивні зв'язки / мати властивість;

– значення властивості.

Для реалізації семантичних мереж існують спеціальні мережеві мови (NET), мова реалізації систем SIMER + MIR та ін. Широко відомі експертні системи, що використовують семантичні мережі в якості мови представлення знань: PROSPECTOR, CASNET, TORUS.

Систематизація відносин конкретної семантичної мережі залежить від специфіки знань предметної області і є складним завданням. На особливу увагу заслуговують загальнозначущі відносини, присутні у багатьох предметних областях. Саме на таких відносинах заснована концепція семантичної мережі. У семантичних мережах, так само як при фреймовому поданні знань, декларативні і процедурні знання не розділені, отже, база знань не відділена від механізму виведення. Процедура логічного виведення зазвичай представляє сукупність процедур обробки мережі. Семантичні мережі отримали широке застосування в системах розпізнавання мови і експертних системах.

Недоліком цієї моделі є складність організації процедури організації виведення.

*Фрейми*

Фреймова модель подання знань є систематизованою психологічною моделлю пам'яті людини і його свідомості. Ця теорія має досить абстрактний характер, тому тільки на її основі неможливе створення конкретних мов подання знань.

*Фрейм* – це мінімально можливий опис сутності якої-небудь події, ситуації, процесу або об'єкта. В історичному плані розвиток фреймової моделі пов'язаний з теорією фреймів М. Мінського, що визначає спосіб формалізації знань, який використовується при вирішенні задач розпізнавання образів (сцен) і розуміння мови. «Відправним моментом для даної теорії служить той факт, що людина, намагаючись пізнати нову для себе ситуацію або по-новому поглянути на вже звичні речі, вибирає зі своєї пам'яті деяку структуру даних (образ), звану нами фреймом, з таким розрахунком, щоб шляхом зміни в ній окремих деталей зробити її придатною для розуміння більш широкого класу явищ або

процесів». Іншими словами, *фрейм* – це форма опису знань, що окреслює рамки розглянутого (в поточній ситуації при вирішенні даної задачі) фрагмента предметної області. Модель фрейма є досить універсальною, оскільки дозволяє відобразити все різноманіття знань про світ через:

– фрейми-структури, для позначення об'єктів і понять (позика, застава, вексель);

– фрейми-ролі (менеджер, касир, клієнт);

– фрейми-сценарії (банкрутство, збори акціонерів, святкування іменин);

– фрейми-ситуації (тривога, аварія, робочий режим пристрою) та ін.

Розрізняють фрейми-зразки або прототипи і фрейми-екземпляри, які створюються для відображення реальних фактичних ситуацій на основі даних, що надходять.

Фрейм має ім'я (назву), що служить для ідентифікації описуваного ним поняття, і містить ряд описів – *слотів*, за допомогою яких визначаються основні структурні елементи цього поняття.

Традиційно структура фрейму може бути представлена як список властивостей:

(ІМ'Я ФРЕЙМА:

(*ім'я 1-го слота: значення 1-го слота*),

(*ім'я 2-го слота: значення 2-го слота*),

.....

(*ім'я N-го слота: значення N-го слота*)).

Той же запис можна представити у вигляді таблиці (див. табл. 4.2), доповнивши її двома стовпцями.

Таблиця 4.2

**Структура фрейму**

| Ім'я фрейму | | | |
|---|---|---|---|
| Ім'я слота | Значення слота | Спосіб отримання значення | Приєднана процедура |

У таблиці додаткові стовпці (3-й і 4-й) призначені для опису способу отримання слотом його значення і можливого приєднання

до того чи іншого слоту спеціальних процедур, що допускається в теорії фреймів. Як значення слота може виступати ім'я іншого фрейма, так утворюються мережі фреймів.

За слотами слідують *шпації*, в які поміщають дані, що представляють поточні значення слотів. Слот може містити не тільки конкретне значення, але також ім'я процедури, що дозволяє обчислити це значення за заданим алгоритмом. Наприклад, слот з ім'ям *вік* може містити ім'я процедури, яка обчислює вік людини за датою народження, записаної в іншому слоті, і поточну дату. Процедури, що розташовуються в слотах, називаються пов'язаними або приєднаними процедурами. Виклик пов'язаної процедури здійснюється при зверненні до слоту, в якому вона розташована. Заповнювачами слота можуть бути також правила продукцій, які використовуються для визначення конкретного значення.

У слоті може міститися не одне, а кілька значень, тобто в якості структурних складових фреймів можуть використовуватися дані складних типів, а саме: масиви, списки, множини, фрейми і т. д. Наприклад, в слоті з ім'ям *брат* може міститися список імен, якщо об'єкт, описуваний даним фреймом, має кількох братів. Значення слота може являти собою деякий діапазон або перелік можливих значень, арифметичний вираз, фрагмент тексту і т.д.

Сукупність даних предметної області може бути представлена множиною взаємопов'язаних фреймів, що утворюють єдину фреймову систему, в якій об'єднуються декларативні і процедурні знання. Така система має, як правило, ієрархічну структуру, в якій фрейми з'єднані один з одним за допомогою родо-видових зв'язків. На верхньому рівні ієрархії знаходиться фрейм, що містить найбільш загальну інформацію, дійсну для всіх інших фреймів. Фрейми мають здатність успадковувати значення характеристик своїх батьків. Наприклад, фрейм АФРИКАНСЬКИЙ_СЛОН успадковує від фрейму СЛОН значення характеристики *колір = «сірий»*. Значення характеристики в дочірньому фреймі може відрізнятися від батьківського, наприклад, значенням даного слота для фрейма АЗІАТСЬКИЙ_СЛОН є *колір = «коричневий»*.

Над фреймами можна здійснювати деякі теоретико-множинні операції, наприклад об'єднання і перетин.

При об'єднанні фреймів в результуючому фреймі будуть присутні всі слоти, які зустрічалися в початкових фреймах. У

слотах, які не є загальними, будуть збережені вихідні значення. Якщо в об'єднуємих фреймах були однойменні слоти, в результуючому фреймі залишиться один слот з таким ім'ям, значення його визначиться в результаті об'єднання значень однойменних слотів.

При перетині фреймів в результуючому фреймі будуть присутні тільки ті слоти, які були у всіх вхідних фреймах. Обчислити результуючі значення можна двома способами. Перший спосіб полягає в тому, що в результуючому фреймі присутні тільки ті значення, які збігалися в початкових фреймах. У другому способі результуючі значення знаходять шляхом перетину значень з вхідних фреймів.

Фреймові системи підрозділяються на статичні і динамічні, останні допускають зміну фреймів в процесі виконання задачі.

У загальному випадку структура даних фрейму може містити більш широкий набір інформації, до якого входять такі атрибути (див. табл. 4.3).

Таблиця 4.3

**Фрейм «КЕРІВНИК»**

| Ім'я слота | Значення слота | Тип значення слота |
|---|---|---|
| Ім'я | Іванчук І.І. | Рядок символів |
| Народжений | 01.01.1965 | Дата |
| Вік | *age*(*дата, народжений*) | Процедура |
| Спеціальність | Юрист | Рядок символів |
| Відділ | Відділ кадрів | Рядок символів |
| Зарплатня | 8000 | Число |
| Адреса | ДОМ_АДРЕСА | Фрейм |

*Ім'я фрейму*. Воно служить для ідентифікації фрейма в системі і має бути унікальним. Фрейм представляє собою сукупність слотів, число яких може бути довільним. Число слотів в кожному фреймі встановлюється проєктувальником системи, при цьому частина слотів визначається самою системою для виконання специфічних функцій (системні слоти), прикладами яких є: слот-покажчик батька даного фрейма (IS-A), слот-покажчик дочірніх фреймів, слот для введення імені користувача,

слот для введення дати визначення фрейму, слот для введення дати зміни фрейма.

*Ім'я слота*. Воно повинно бути унікальним в межах фрейму. Зазвичай ім'я слота є ідентифікатор, який наділений певною семантикою. У якості імені слота може виступати довільний текст. Наприклад, <Ім'я слота> = Головний герой поеми М. В. Гоголя «Мертві душі», <Значення слота> = Павло Іванович Чичиков. Імена системних слотів зазвичай зарезервовані, в різних системах вони можуть мати різні значення. Приклади імен системних слотів: IS-A, HASPART, RELATIONS і т.д. Системні слоти служать для редагування бази знань і управління виводу у фреймовій системі.

*Вказівники успадкування*. Вони показують, яку інформацію про атрибути слотів з фрейма верхнього рівня успадковують слоти з аналогічними назвами в даному фреймі. Вказівники успадкування характерні для фреймових систем ієрархічного типу, заснованих на відносинах типу «абстрактне – конкретне». У конкретних системах вказівники успадкування можуть бути організовані різними способами і мати різні позначення:

U (Unique) – значення слота не успадковується;

S (Same) – значення слота успадковується;

R (Range) – значення слота повинні знаходитися в межах інтервалу значень, зазначених в однойменному слоті батьківського фрейма;

O (Override) – при відсутності значення в поточному слоті воно успадковується з фрейма верхнього рівня, однак в разі визначення значення поточного слота воно може бути унікальним. Цей тип покажчика виконує одночасно функції покажчиків U і S.

*Вказівник типу даних*. Він показує тип значення слота. Найбільш вживані типи:

*frame* – покажчик на фрейм; *real* – дійсне число;

*integer* – ціле число; *boolean* – логічний тип;

*text* – фрагмент тексту; *list* – список; *table* – таблиця;

*expression* – вираз; *lisp* – пов'язана процедура і т.д.

*Значення слота*. Воно повинно відповідати зазначеному типу даних і умові успадкування.

*Демони*. Демоном називається процедура, яка автоматично запускається при виконанні деякої умови. Демони автоматично

запускаються при зверненні до відповідного слоту. Типи демонів пов'язані з умовою запуску процедури. Демон з умовою IF-NEEDED запускається, якщо в момент звернення до слоту його значення не було встановлено. Демон типу IF-ADDED запускається при спробі зміни значення слота. Демон IF-REMOVED запускається при спробі видалення значення слота. Можливі також інші типи демонів. Демон є різновидом пов'язаної процедури.

*Приєднана процедура*. Як значення слота може використовуватися процедура, яка називається службовою в мові Лісп або методом в мовах об'єктно-орієнтованого програмування. Приєднана процедура запускається за повідомленням, переданим з іншого фрейму. Демони і приєднані процедури є процедурними знаннями, об'єднаними разом з декларативними в єдину систему. Ці процедурні знання є засобами управління виводу у фреймових системах, причому з їх допомогою можна реалізувати будь-який механізм виведення. Подання таких знань і заповнення ними інтелектуальних систем – дуже нелегка справа, яка вимагає додаткових витрат праці та часу розробників ІІС. Тому проєктування фреймових систем виконується, як правило, фахівцями, що мають високий рівень кваліфікації в галузі штучного інтелекту.

Приклад мережі фреймів наведено на рис. 4.4. На ньому поняття УЧЕНЬ успадковує властивості фреймів ДИТИНА і ЛЮДИНА, які знаходяться на більш високих рівнях ієрархії. Якщо буде поставлено питання «Чи люблять учні солодке?», то відповідь «так», так як цією властивістю володіють всі діти, що зазначено у фреймі ДИТИНА. Успадкування властивостей може бути частковим, наприклад «вік» для учнів не успадковується з фрейма «дитина», так як явно вказано у власному фреймі.

В останні роки термін «фреймовий» часто замінюють терміном «об'єктно-орієнтований». Шаблон фрейма можна розглядати як клас, екземпляр фрейма – як об'єкт. Мови об'єктно-орієнтованого програмування (ООП) надають засоби створення класів і об'єктів, а також засоби для опису процедур обробки об'єктів (методи). Мови ООП, що не містять засобів реалізації приєднаних процедур, не дозволяють організувати гнучкий механізм логічного висновку, тому розроблені на них програми або

являють собою об’єктно-орієнтовані бази даних, або вимагають інтеграції з іншими засобами обробки знань (наприклад, з мовою PROLOG).

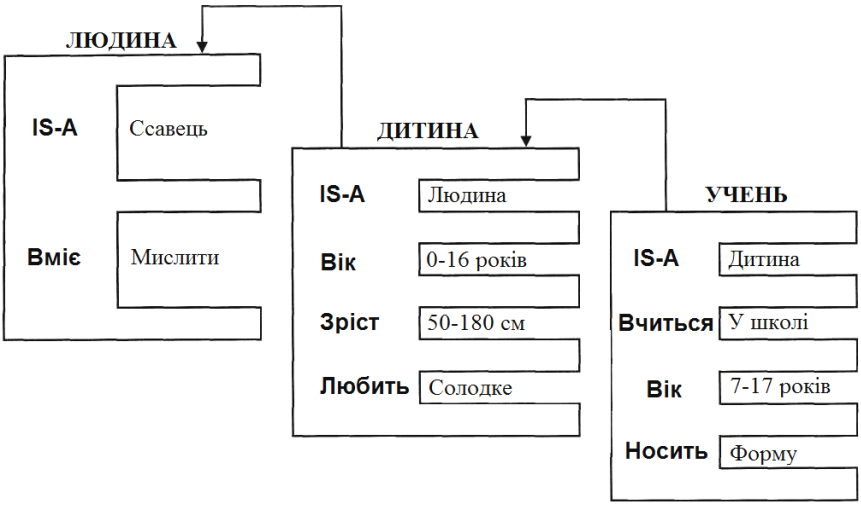


Рис. 4.4. Приклад ієрархії фреймів

Існують також спеціалізовані мови подання знань на основі фреймової моделі, прикладами яких є: FRL (Frame Representation Language), KRL (Knowledge Representation Language), фреймова «оболонка» Kappa та ін. Широко відомі такі фреймо-орієнтовані експертні системи, як ANALYST TRISTAN, ALTERID, МОДІС.

# ЛЕКЦІЯ 5. НЕЙРОННІ МЕРЕЖІ

Особливістю інтелектуальних систем є здатність вирішувати слабоструктуровані та погано формалізовані задачі. Ця здатність заснована на застосуванні різних методів моделювання міркувань для обробки символьної інформації. Традиційним підходом до побудови механізмів міркування є використання дедуктивного логічного висновку на правилах (rule-based reasoning), який застосовується в експертних системах продукційного і логічного типу. При такому підході необхідно заздалегідь сформулювати весь набір закономірностей, що описують предметну область.

Альтернативний підхід заснований на концепції навчання за прикладами (case-based reasoning). В цьому випадку при побудові інтелектуальної системи не потрібно заздалегідь знати про всі закономірності досліджуваної області, але необхідно мати достатню кількість прикладів для налаштування розроблюваної адаптивної системи, яка після навчання буде здатна отримувати необхідні результати з певним ступенем вірогідності. В якості таких адаптивних систем застосовуються штучні нейронні мережі.

## 5.1. Модель штучного нейрону

Так як основне завдання штучного інтелекту – моделювання міркувань, а природній «пристрій здатний думати» – це мозок, то очевидним завданням є створення «штучного мозку» «за образом і подобою» людського. Дослідженням цього питання займається в рамках напряму штучного інтелекту нейрокібернетика. Влаштування мозку вивчають такі науки як психологія, нейрофізіологія, нейробіологія, що володіють достатнім обсягом знань. Основною ідеєю нейрокібернетики стало відтворення «в залізі» клітини мозку – нейрону.

*Нейрони* – спеціалізовані клітини, здатні приймати, обробляти, кодувати, передавати і зберігати інформацію, організовувати реакції на подразнення, встановлювати контакти з іншими нейронами, клітинами органів. Унікальними особливостями нейрону є здатність генерувати електричні розряди і передавати інформацію за допомогою спеціалізованих закінчень – *синапсів*. Число нейронів мозку людини наближається до $10^{11}$. На одному

нейроні може бути до 10000 синапсів. Якщо тільки ці елементи вважати осередками зберігання інформації, то можна прийти до висновку, що нервова система може зберігати $10^{19}$ од. інформації, тобто здатна вмістити практично усі знання, накопичені людством. На рис. 5.1 наведена схема будови "типового" нейрону.

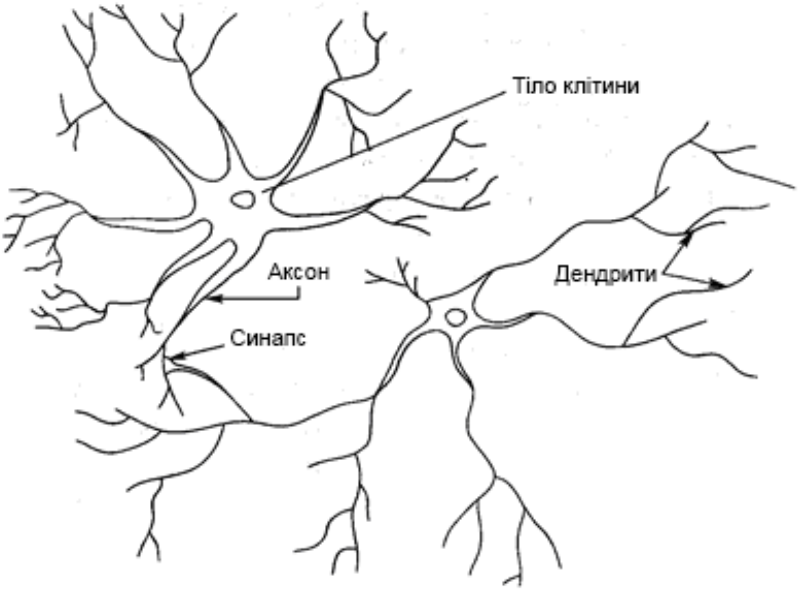


Рис. 5.1. Загальна схема будови біологічного нейрону

Природна нервова клітина (нейрон) складається з тіла – *соми*, що містить ядро, і відростків двох типів. Відростки першого типу, звані *дендритами* за їх схожість з кроною розлогого дерева, служать у якості вхідних каналів для нервових імпульсів від інших нейронів. Ці імпульси надходять в сому розміром від 3 до 100 мікрон, викликаючи її специфічне збудження, яке потім поширюється по вивідному відростку другого типу – *аксону*.

Довжина аксонів зазвичай помітно перевершує розміри дендритів, в окремих випадках досягаючи десятків сантиметрів і навіть метрів. З'єднання аксону з дендритом іншого нейрону називається *синапсом*. Нейрон може перебувати в 2 станах:

збудженому або не збудженому. Нейрон збуджується і передає сигнал через аксон, якщо число збуджуючих сигналів, що надійшли по дендритам більше, ніж число гальмуючих.

1943 рік став роком народження теорії штучних нейронних мереж. Штучна нейронна мережа (ШНМ) – це спрощена модель біологічного мозку, точніше нервової тканини. Дж. Маккалок і У. Пітт запропонували модель формального нейрону і описали основні принципи побудови нейронних мереж.

Сигнали, що надходять на вхід нейрону, нерівнозначні в тому сенсі, що інформація з одного джерела може бути більш важливою, ніж з іншого. Синапси нейрону – місця контактів нервових волокон – відповідно до цієї моделі, передають сигнали, визначаючи силу впливу сигналу з цього входу на вихідний сигнал, що надходить на аксон. Для цього кожному входу ставиться у відповідність ваговий коефіцієнт $w_i$. Дендрити отримують вхідний сигнал, представлений вектором $x_i$. Потім нейрон обробляє надійшовший сигнал, виробляючи зважене підсумовування і нелінійне перетворення, використовуючи для цього *активаційну* або з*буджуючу* функцію (найбільш поширені функції активації – лінійна, порогова і сигмоїдальна), аргументом якої буде результат підсумовування мінус граничне значення. Нейрони в мережі можуть мати однакові або різні функції збудження. Це значення визначає рівень сигналу, на який нейрон буде реагувати.

Модель штучного нейрона (рис. 5.2) являє собою дискретно-безперервний перетворювач інформації. Інформація, що надходить на вхід нейрона, підсумовується з урахуванням вагових коефіцієнтів $w_i$ сигналів $x_i$, $i = \overline{1,\ldots,n}$, де $n$ – розмірність простору вхідних сигналів. Потенціал нейрону визначається за формулою

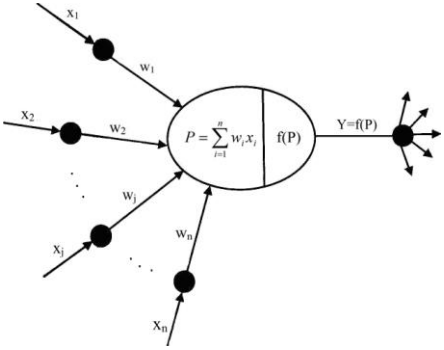


Рис. 5.2. Схема кібернетичної моделі нейрона

$$P = \sum_{i=1}^{n} w_i x_i .$$

Зважена сума вхідних сигналів (потенціал) перетворюється за допомогою передавальної функції $f(P)$ у вихідний сигнал нейрона *Y*, який передається іншим нейронам мережі, тобто $Y = f(P)$.

Нейронна мережа являє собою сукупність штучних нейронів, організованих шарами. При цьому виходи нейронів одного шару з'єднуються з входами нейронів іншого. Залежно від топології з'єднань нейронів ШНМ поділяються на однорівневі і багаторівневі, зі зворотними зв'язками і без них. Зв'язки між шарами можуть мати різну структуру. У однолінійних мережах кожен нейрон (вузол) нижнього шару пов'язаний з одним нейроном верхнього шару. Якщо кожен нейрон нижнього шару з'єднаний з декількома нейронами наступного шару, то виходить пірамідальна мережа. Воронкоподібна схема з'єднань передбачає зв'язок кожного вузла верхнього шару з усіма вузлами нижнього рівня. Існують також деревовидні і рекурентні мережі, що містять зворотні зв'язки з довільною структурою міжнейронних з'єднань. Щоб побудувати ШНМ для вирішення конкретного завдання, потрібно вибрати тип з'єднання нейронів, визначити вид передавальних функцій елементів і підібрати вагові коефіцієнти міжнейронних зв'язків. При всьому різноманітті можливих конфігурацій ШНМ на практиці набули поширення лише деякі з них. Класичні моделі нейронних мереж розглянуті нижче.

### 5.2. Моделювання нейронних мереж

Теоретичні основи нейроматематики були закладені на початку 1940-х рр.. Спроби побудувати машини, здатні до розумної поведінки, були значною мірою натхненні ідеями «батька кібернетики» Норберта Вінера, який писав у своїй знаменитій праці «Кібернетика або управління і зв'язок в тварині та машині», що всі машини, які претендують на «розумність», повинні мати здатність переслідувати певні цілі і пристосовуватися, тобто навчатися. Ідеї Вінера були застосовані Дж. Маккалохом і У. Піттсом, які розробили власну теорію діяльності головного мозку, засновану на припущенні, що функціонування комп'ютера і мозку подібне. До головних результатів їх роботи відносяться наступні:

– модель нейрону у вигляді найпростішого процесорного елементу, який обчислює значення перехідної функції від скалярного добутку вектора вхідних сигналів і вектора вагових коефіцієнтів;

– конструкція нейронної мережі для виконання логічних і арифметичних операцій;

– припущення про те, що нейронна мережа здатна навчатися, розпізнавати образи, узагальнювати отриману інформацію.

У формалізмі Дж. Маккалоха і У. Піттса нейрони мають порогову функцію переходу зі стану в стан. Кожен нейрон в мережі визначає зважену суму станів всіх інших нейронів і порівнює її з порогом, щоб визначити свій власний стан. Апаратна реалізація ШНМ на основі порогових елементів, що оперують двійковими числами, виявилася надзвичайно важкою через високу вартість електронних елементів в той час. Найдосконаліші системи тоді містили лише сотні нейронів, в той час як нервова система мурахи містить більше 20 тис.

Головною властивістю біологічного нейрону є його здатність до навчання, його універсальність, здатність вирішувати різні завдання. Описана вище модель не здатна до цього. Навченою вона стала лише у 1949 році завдяки Д.Хеббу (D. Hebb), який, спираючись на фізіологічні та психологічні дослідження, висунув гіпотезу про навченість біологічних нейронів. Його метод навчання

став відправною точкою для алгоритмів навчання нейронних мереж без вчителя.

Вже у 1957 році в світовому науковому товаристві сталася друга за значимістю в історії нейронних мереж подія: американський фізіолог Ф. Розенблатт розробив модель зорового сприйняття і розпізнавання – *перцептрон* (perceptron), а потім і побудував перший нейрокомп'ютер Марк-1. Ф. Розенблат ввів можливість модифікації міжнейронних зв'язків, що зробило ШНМ навченою. Перші перцептрони були здатні розпізнавати деякі букви латинського алфавіту. Згодом модель перцептрону була значно вдосконалена, а найбільш вдалим її застосуванням стали задачі автоматичної класифікації.

Алгоритм навчання перцептрона включає наступні кроки.

1. Системі пред'являється еталонний образ.
2. Якщо результат розпізнавання збігається з заданим, то вагові коефіцієнти зв'язків не змінюються.
3. Якщо ШНМ неправильно розпізнає результат, то ваговим коефіцієнтам дається приріст в сторону підвищення якості розпізнавання.

Нейронні мережі були прості, зрозумілі і багатообіцяючі. Складні процеси мислення, здавалося, були готові розкритися перед людиною, для їх опису були потрібні лише елементарні математичні операції: додавання, множення і лінійна функція. На перцептрон, а потім і багатошаровий перцептрон покладалися великі надії, тому бурхливий ентузіазм, з яким він був прийнятий, досить швидко змінився жорсткою критикою. У 1969 році вийшла в світ книга «Перцептрони» М. Мінського і С. Пейперта, яка ознаменувала закінчення першого етапу в історії нейронних мереж. У ній було проведено теоретичний аналіз перцептрону, який показав його обмежені можливості, оскільки не завжди існує така комбінація вагових коефіцієнтів, при якій задана множина образів буде розпізнаватися правильно. Причина цього недоліку полягає в тому, що одношаровий перцептрон реалізує лінійну поверхню, що розділяє простір еталонів, внаслідок чого відбувається невірне розпізнавання образів у випадках, коли задача не є лінійно сепарабельною (роздільною). Для вирішення таких проблем запропоновані моделі багатошарових перцептронів, здатні будувати ламану межу між розпізнаваними образами. Незважаючи на те що

перцептрон Розенблатта мав невисокі можливості навчання, розробка цієї концепції привернула увагу дослідників до проблеми ШНМ і привела до створення більш «розумних» інтелектуальних систем, яким була властива інша проблема – їх навчання.

Запропонований у 1986 році Д. Румельхардом алгоритм зворотного поширення помилки став одним з провідних чинників, що породили сучасний нейромережевий бум, тому що був ефективним способом навчання нейронної мережі досить довільної структури.

### 5.3. Класифікація штучних нейронних мереж

За топологією:

– повнозв'язкові (кожен нейрон пов'язаний з усіма іншими нейронами, в тому числі і сам з собою);

– багатошарові (нейрони розташовуються шарами і кожен нейрон наступного шару пов'язаний з усіма нейронами поточного шару).

За організацією навчання:

– з учителем (нейронну мережу навчають, подаючи на вхід значення навчальної вибірки і надаючи необхідні вихідні значення);

– без вчителя (на входи нейронної мережі подають множину об'єктів і нейронна мережа сама ділить їх на кластери або класи).

За типами структур:

– нейрони з одним типом функції активації (всі нейрони мережі мають одну функцію активації $f(x)$, наприклад, лінійну);

– нейрони з декількома типами функцій активації (нейрони мережі мають різні функції активації).

За типом зв'язків:

– прямого поширення (без зворотних зв'язків між нейронами, до таких мереж відносяться одношаровий і багатошаровий перцептрони, мережа радіальних базисних функцій);

– рекурентні (зі зворотним зв'язком, від виходів нейронів до входів, до таких мереж відносяться змагальні мережі та мережа Хопфілда).

За типом сигналу:

– бінарні (на входи подаються тільки нулі та одиниці);

– аналогові (на входи нейронів подаються значення безперервних функцій).

*Одношарові мережі*

Один нейрон здатний виконувати найпростіші процедури розпізнавання, але тільки з'єднання декількох нейронів здатне вирішити практично корисну задачу. Найпростіша мережа складається з групи нейронів, що утворюють шар.

*Навчання за дельта-правилом.*

Дельта-правило є узагальненням алгоритму навчання перцептрону. Дельта-правило працює тільки з безперервними, диференційовними функціями в режимі навчання з учителем (supervised learning). Помилка, що обчислюється в процесі навчання мережі, – це функція, що характеризує якість навчання даної мережі, тому процес навчання нейронної мережі можна уявити як процес мінімізації функції помилки. Напрямок зміни значення функції можна встановити, обчисливши похідну для функції однієї змінної або градієнт для функції багатьох змінних. При мінімізації значення функції багатьох змінних менше значення необхідно шукати у напрямку антіградієнта. У даному алгоритмі навчання початкові вагові коефіцієнти можуть бути будь-якими. Процес навчання можна вважати завершеним, якщо досягнута якась заздалегідь встановлена мінімальна помилка або алгоритм пропрацював умовлену кількість разів.

*Багатошарові мережі*

Вони володіють великими обчислювальними можливостями. Хоча створені мережі всіх конфігурацій, які тільки можна собі уявити, пошарова організація нейронів копіює шаруваті структури певних відділів мозку. Багатошарові мережі утворюються каскадами шарів. Вихід одного шару є входом для наступного шару, така організація мережі утворює багатошарову нейронну мережу з прямим розповсюдженням. У багатошарових мережах встановлюються зв'язки тільки між нейронами сусідніх шарів, як показано на рис. 5.3. Кожен елемент може бути з'єднаний модифікуємим зв'язком з будь-яким нейроном сусідніх шарів, але між елементами одного шару зв'язків нема. Шари між першим і останнім називаються проміжними або прихованими.

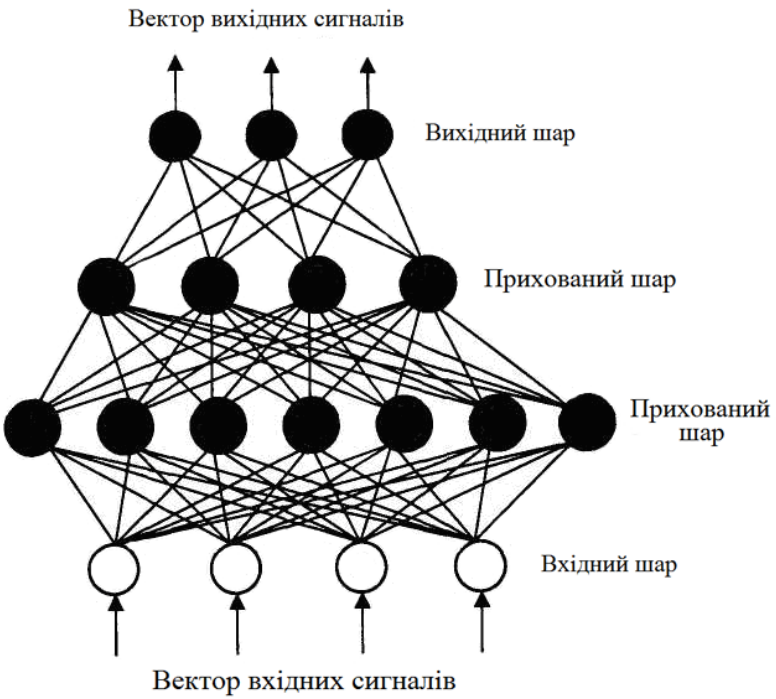


Рис. 5.3. Схема багатошарового перцептрона

Вагові коефіцієнти у такій мережі мають три індекси $i$ – номер нейрона наступного шару, для якого зв'язок вхідний, $j$ – номер входу або нейрона поточного шару, для якого зв'язок вихідний, $k$ – номер поточного шару в нейронній мережі (для входів, вектора $X$, $k = 0$).

Кожен нейрон може посилати вихідний сигнал тільки в вищерозміщений шар і приймати вхідні сигнали тільки з нижче розташованого шару. Вхідні сигнали подаються на нижній шар, а вихідний вектор сигналів визначається шляхом послідовного обчислення рівнів активності елементів кожного шару (від низу до верху) з використанням вже відомих значень активності елементів попередніх шарів. При розпізнаванні образів вхідний вектор відповідає набору ознак, а вихідний – розпізнаваним образам. Прихований шар (один або декілька) призначений для

відображення специфіки знань. У таких мережах зазвичай використовуються передавальні сигмоїдальні функції.

Структура нейронної мережі визначається типом, наприклад 25–10–5, тобто двадцять п’ять вузлів знаходиться в першому шарі, десять – в прихованому і п’ять – у вихідному. Визначення числа прихованих шарів і числа нейронів в кожному шарі для конкретної задачі є неформальною проблемою, при вирішенні якої можна використовувати евристичне правило: число нейронів в наступному шарі у два рази менше, ніж в попередньому.

Вище зазначалося, що простий перцептрон з одним шаром навчаємих зв’язків формує межі областей рішень у вигляді гіперплощин. Двошаровий перцептрон може виконувати операцію логічного «І» над півпростором, утвореним гіперплощинами першого шару вагових коефіцієнтів. Це дозволяє формувати будь-які опуклі області в просторі вхідних сигналів. За допомогою тришарового перцептрону, використовуючи логічне «АБО» для комбінування опуклих областей, можна отримати області рішень довільної форми і складності, в тому числі неопуклі і незв’язні. Те, що багатошарові перцептрони з достатньою множиною внутрішніх нейроподібних елементів і відповідною матрицею зв’язків в принципі здатні здійснювати будь-яке відображення вхід – вихід, відзначали ще М. Мінський і С. Пейперт, проте вони сумнівалися, що для таких процедур можна відкрити потужний аналог процедури навчання простого перцептрону. У даний час в результаті відродження інтересу до багатошарових мереж запропоновано декілька таких процедур. Однією з них є алгоритм зворотного поширення помилки.

*Навчання методом зворотного поширення помилки.*

Навчання алгоритмом зворотного поширення помилки передбачає два проходи по всім верствам мережі: прямого і зворотного.

При прямому проході вхідний вектор подається на вхідний прошарок нейронної мережі, після чого шириться по мережі від шару до шару. В результаті генерується набір вихідних сигналів, який і є фактичною реакцією мережі на даний вхідний образ. Під час прямого проходу всі синаптичні вагові коефіцієнти мережі фіксовані.

Під час зворотного проходу усі синаптичні вагові коефіцієнти налаштовуються відповідно до правила корекції помилок, а саме: фактичний вихід мережі віднімається з бажаного, в результаті чого формується сигнал помилки. Цей сигнал згодом поширюється мережею у напрямку, протилежному напрямку синаптичних зв'язків. Звідси і назва – алгоритм зворотного поширення помилки. Синаптичні ваги налаштовуються з метою максимального наближення вихідного сигналу мережі до бажаного.

*Рекурентні мережі*

Вони містять зворотні зв'язки, завдяки яким стає можливим отримання відмінних значень виходів при одних і тих же вхідних даних. Наявність зворотних нейронів дозволяє ШНМ накопичувати знання в процесі навчання.

Рекурентні мережі (рис. 5.4) є розвитком моделі Хопфілда на основі застосування нових алгоритмів навчання, що виключають потрапляння системи в локальні мінімуми на поверхні енергетичних станів. Важливою особливістю рекурентних мереж є їх здатність передбачати існування нових класів об'єктів.

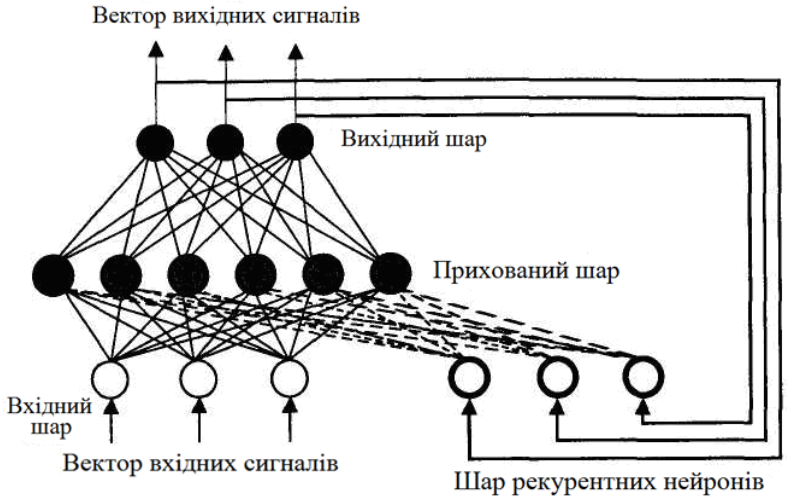


Рис. 5.4. Схема рекурентної нейронної мережі

*Модель Хопфілда*

Роботи американського біофізика Дж. Хопфілда поклали початок сучасному математичному моделюванню нейронних обчислень. Йому вдалося залучити до аналізу нейромережевих моделей потужний математичний апарат статистичної фізики. В результаті була сформульована математична модель асоціативної пам'яті на нейронній мережі з використанням правила Д. Хебба для модифікації вагових коефіцієнтів. Це правило засноване на простому припущенні: якщо два нейрона збуджуються разом, то сила зв'язку між ними зростає; якщо вони збуджуються порізно, то сила зв'язку між ними зменшується.

Мережа Хопфілда будується з урахуванням наступних умов:

– всі елементи пов'язані з усіма;

– $w_{ji} = w_{ij}$ – прямі і зворотні зв'язки симетричні,

– $w_{ii} = 0$ – діагональні елементи матриці зв'язків дорівнюють нулю, тобто виключаються зворотні зв'язки з виходу на вхід одного нейрону.

Для одношарової нейронної мережі зі зв'язками типу «все до всіх» характерна збіжність до однієї з кінцевої множини рівноважних точок, які є локальними мінімумами функції енергії, що відбиває структуру всіх зв'язків в мережі. Введена Хопфілдом функція обчислювальної енергії нейронної мережі описує поведінку мережі через прагнення до мінімуму енергії, який відповідає заданому набору образів. У зв'язку з цим мережі Хопфілда можуть виконувати функції асоціативної пам'яті, забезпечуючи збіжність до того образу, в область тяжіння якого потрапляє початковий патерн (зразок) активності нейронів мережі.

Цей підхід привабливий тим, що нейронна мережа для конкретної задачі може бути запрограмована без навчальних ітерацій. Вагові коефіцієнти зв'язків обчислюються на основі виду функції енергії, сконструйованої для розв'язуваної задачі.

Розвитком моделі Хопфілда є машина Больцмана, запропонована і досліджена Дж. Е. Хінтоном і Р. Земелом для вирішення комбінаторних оптимізаційних задач і задач штучного інтелекту. У ній, як і в інших моделях, нейрон має стани (1,0), міжнейронні зв'язки представлені ваговими коефіцієнтами, а

кожний стан мережі характеризується певним значенням функції консенсусу (аналог функції енергії). Максимум функції консенсусу відповідає оптимальному вирішенню задачі.

Мережі Хопфілда отримали застосування на практиці в основному як реалізації підсистем більш складних систем. Вони мають певні недоліки, що обмежують можливості їх застосування:

– припущення про симетрії зв'язків між елементами, без якої не можна ввести поняття енергії;

– нейронна мережа – це пристрій для запам'ятовування та обробки інформації, а не пристрій мінімізації енергії. Економія енергії грає в цих процесах допоміжну роль;

– мережі Хопфілда підтримують безліч зайвих, неефективних, іноді дублюючих одне одного зв'язків. У реальних нервових системах такі зв'язки не підтримуються, тому що їх реалізація вимагає певних витрат. У біологічних нервових системах відбувається звільнення від зайвих зв'язків за рахунок їх структуризації. При цьому замість організації зв'язків «усі до усіх» використовується багатошарова ієрархічна система зв'язків.

*Самоорганізовані мережі Т. Кохонена*

Ідея мереж з самоорганізацією на основі конкуренції між нейронами базується на застосуванні спеціальних алгоритмів самонавчання ШНМ. Мережі Кохонена зазвичай містять один (вихідний) шар обробних елементів з пороговою функцією передачі. Число нейронів у вихідному шарі відповідає кількості розпізнаваних класів. Налаштування параметрів міжнейронних з'єднань проводиться автоматично на основі міри близькості вектора вагових коефіцієнтів настроюваних зв'язків до вектору вхідних сигналів в евклідовому просторі. У конкурентній боротьбі перемагає нейрон, що має значення вагових коефіцієнтів, найближчих до нормалізованого вектору вхідних сигналів. Крім того, в самоорганізованих мережах можлива класифікація вхідних зразків (патернів). На практиці ідея Кохонена зазвичай використовується у комбінації з іншими нейромережевими парадигмами.

Нейронні мережі можуть реалізовуватися як програмно, так і апаратно. Поступово напрямок нейрокібернетика перетворився у нейрокомп'ютинг.

### 5.4. Задачі, які вирішуються нейронними мережами

*1. Класифікація образів*. Задача полягає у визначенні приналежності вхідного образа (наприклад, мовного сигналу або рукописного символу), представленого вектором ознак до одного або декількох попередньо визначених класів. До відомих додатків відносяться розпізнавання букв, розпізнавання мови, класифікація сигналу електрокардіограми, класифікація клітин крові.

*2. Кластеризація/категоризація*. При вирішенні задачі кластеризації навчаюча множина не має міток класів. Алгоритм кластеризації заснований на подобі образів і поміщає схожі образи в один кластер. Відомі випадки застосування кластеризації для видобутку знань, стиснення даних і дослідження властивостей даних.

*3. Апроксимація функцій*. Припустимо, що є навчальна вибірка (($x_1$, $y_1$), ($x_2$, $y_2$) ..., ($x_n$, $y_n$)) (пари даних вхід-вихід), яка генерується невідомою функцією $F$, спотвореної шумом. Задача апроксимації полягає в знаходженні невідомої функції $F$. Апроксимація функцій необхідна при вирішенні численних інженерних і наукових задач моделювання.

*4. Передбачення/прогноз*. Нехай задані $n$ дискретних відліків {$y$ ($t_1$), $y$ ($t_2$), ..., $y$ ($t_n$)} в послідовні моменти часу $t_1$, $t_2$, ..., $t_n$. Задача полягає в передбаченні значення $y$ ($t_{n+1}$) в наступний момент часу $t_{n+1}$.

Передбачення/прогноз мають велике значення для прийняття рішень в бізнесі, науці і техніці (передбачення цін на фондовій біржі, прогноз погоди).

*5. Оптимізація*. Численні проблеми в математиці, статистиці, техніці, науці, медицині та економіці можуть розглядатися як проблеми оптимізації. Завданням алгоритму оптимізації є знаходження такого рішення, яке задовольняє системі обмежень і максимізує або мінімізує цільову функцію.

*6. Пам'ять, що адресується за змістом*. У традиційних комп'ютерах звернення до пам'яті доступно тільки за допомогою адреси, що не залежить від змісту пам'яті. Більш того, якщо допущена помилка в обчисленні адреси, то може бути знайдена зовсім інша інформація. Асоціативна пам'ять або пам'ять, що

адресується за змістом, доступна за вказівкою заданого змісту. Вміст пам'яті може бути викликано навіть по частковому входу або пошкодженому змісту. Асоціативна пам'ять може бути використана у мультимедійних інформаційних базах даних.

*7. Управління*. Розглянемо динамічну систему, задану сукупністю $\{u(t), y(t)\}$, де $u(t)$ – вхідний керуючий вплив, а $y(t)$ – вихід системи в момент часу $t$. У системах управління з еталонною моделлю метою управління є розрахунок такого вхідного впливу $u(t)$, при якому система діє за бажаною траєкторією, заданою еталонною моделлю. Прикладом є оптимальне управління двигуном.

### 5.5. Побудова нейронної мережі

При побудові моделі ШНМ перш за все необхідно точно визначити задачі, які будуть вирішуватися за її допомогою. У даний час нейромережеві технології успішно застосовуються для прогнозування, розпізнавання і узагальнення.

Першим етапом побудови нейромережевої моделі є ретельний відбір вхідних даних, що впливають на очікуваний результат. З вхідної інформації необхідно виключити всі відомості, які не відносяться до досліджуваної проблеми. У той же час слід мати достатню кількість прикладів для навчання ШНМ. Існує емпіричне правило, яке встановлює рекомендоване співвідношення X між кількістю навчальних прикладів, що містять вхідні дані і правильні відповіді, і числом з'єднань в нейронній мережі: $X < 10$.

Для факторів, які включаються в навчальну вибірку, доцільно попередньо оцінити їх значимість, провівши кореляційний і регресійний аналіз, та проаналізувати діапазони їх можливих змін.

На другому етапі здійснюється перетворення вхідних даних з урахуванням характеру і типу проблеми, яка відображається нейромережевою моделлю, та обираються способи подання інформації. Ефективність нейромережевої моделі підвищується, якщо діапазони зміни вхідних і вихідних величин наведені до деякого стандарту, наприклад [0,1] або [–1,1].

*Третій* етап полягає у конструюванні ШНМ, тобто в проектуванні її архітектури (число шарів і число нейронів в кожному шарі). Структура ШНМ формується до початку навчання,

тому успішне вирішення цієї проблеми багато в чому визначається досвідом і мистецтвом аналітика, який проводить дослідження.

*Четвертий* етап пов'язаний з навчанням мережі, яке може проводитися на основі конструктивного або деструктивного підходу. Відповідно до першого підходу навчання ШНМ починається на мережі невеликого розміру, який поступово збільшується до досягнення необхідної точності за результатами тестування. Деструктивний підхід базується на принципі «проріджування дерева», відповідно до якого з мережі з явно надмірним об'ємом поступово видаляють «зайві» нейрони і примикаючі до них зв'язки. Цей підхід дає можливість досліджувати вплив віддалених зв'язків на точність мережі. Процес навчання нейронної мережі є уточнення значень вагових коефіцієнтів для окремих вузлів на основі поступового збільшення обсягу вхідної та вихідної інформації.

Початку навчання повинна передувати процедура вибору функції активації нейронів, що враховує характер розв'язуваної задачі. Зокрема, в тришарових перцептронах на нейронах прихованого шару застосовується в більшості випадків логістична функція, а тип передавальної функції нейронів вихідного шару визначається на основі аналізу результатів обчислювальних експериментів на мережі. Індикатором навченості ШНМ може служити гістограма значень міжнейронних зв'язків.

На *п'ятому* етапі проводиться тестування отриманої моделі ШНМ на незалежній вибірці прикладів.

Найважливішою властивістю нейронних мереж є їх здатність до навчання, що робить нейромережеві моделі незамінними при вирішенні задач, для яких алгоритмізація є неможливою, проблематичною або занадто трудомісткою. Навчання нейронної мережі полягає в зміні внутрішніх параметрів моделі таким чином, щоб на виході ШНМ генерувався вектор значень, що співпадає з результатами прикладів навчальної вибірки. Зміна параметрів нейромережевої моделі може виконуватися різними способами відповідно до різних алгоритмів навчання. Парадигма навчання визначається доступністю необхідної інформації. Виділяють три парадигми:

– навчання з вчителем (контрольоване);

– навчання без вчителя (неконтрольоване);

– змішане навчання.

При *навчанні з вчителем* всі приклади навчальної вибірки містять правильні відповіді (виходи), відповідні вхідним даним (входам). У процесі контрольованого навчання синаптичні ваги налаштовуються так, щоб мережа породжувала відповіді, найбільш близькі до правильних.

*Навчання без вчителя* використовується, коли не для всіх прикладів навчальної вибірки відомі правильні відповіді. В цьому випадку робляться спроби визначення внутрішньої структури потрапляючих в мережу даних з метою розподілити зразки за категоріями (моделі Кохонена).

При *змішаному навчанні* частина вагових коефіцієнтів визначається за допомогою навчання з вчителем, а інша частина отримується за допомогою алгоритмів самонавчання.

Навчання за прикладами характеризується трьома основними властивостями: ємністю, складністю зразків і обчислювальною складністю. Ємність відповідає кількості зразків, які може запам’ятати мережа. Складність зразків визначає здібності нейронної мережі до навчання. Зокрема, під час навчання ШНМ можуть виникати стани «перетренування», в яких мережа добре дає собі раду на прикладах навчальної вибірки, але не справляється з новими прикладами, втрачаючи здатність навчатися.

Розглянемо відомі правила навчання ШНМ.

*Правило корекції за помилкою*. Процес навчання ШНМ полягає в корекції вхідних значень вагових коефіцієнтів міжнейронних зв’язків, які зазвичай задаються випадковим чином. При введенні вхідних даних прикладу (стимулу), що запам’ятовується, з’являється реакція, яка передається від одного шару нейронів до іншого, досягаючи останнього шару, де обчислюється результат. Різниця між відомим значенням результату і реакцією мережі відповідає величині помилки, яка може використовуватися для коригування вагових коефіцієнтів міжнейронних зв’язків. Коригування полягає в невеликому (зазвичай менше 1%) збільшенні синаптичної ваги тих зв’язків, які підсилюють правильні реакції, і зменшенні тих, які сприяють помилковим. Це найпростіше правило контрольованого навчання (дельта-правило) використовується в одношарових мережах з одним

рівнем настроювання зв'язків між множиною входів і множиною виходів.

*Правило Хебба*. Воно базується на наступному нейрофізіологічному спостереженні: якщо нейрони по обидві сторони синапсу активізуються одночасно і регулярно, то сила їх синаптичного зв'язку зростає. При цьому зміна вагового коефіцієнта кожного міжнейронного зв'язку залежить тільки від активності нейронів, що утворюють синапс. Це істотно спрощує реалізацію алгоритмів навчання.

*Навчання методом змагання*. На відміну від правила Хебба, де безліч вихідних нейронів може збуджуватися одночасно, в даному випадку вихідні нейрони змагаються (конкурують) між собою за активізацію. В процесі змагального навчання здійснюється модифікація вагових коефіцієнтів зв'язків нейрону, що виграв, і нейронів, розташованих в його околиці («переможець забирає усе»).

*Метод зворотного поширення помилки*. Він є узагальненням процедури навчання простого перцептрону з використанням дельта-правила на багатошарові мережі. В даному методі необхідно мати навчальну вибірку, що містить «правильні відповіді», тобто вибірка повинна включати множину пар зразків вхідних і вихідних даних, між якими потрібно встановити відповідність. Перед початком навчання міжнейронним зв'язкам присвоюються невеликі випадкові значення. Кожен крок навчальної процедури складається з двох фаз. Під час першої фази вхідні елементи мережі встановлюються у заданий стан. Вхідні сигнали поширюються мережею, породжуючи деякий вихідний вектор. Для роботи алгоритму потрібно, щоб характеристика вхід-вихід нейроподібних елементів була неспадаючою і мала обмежену похідну. Зазвичай для цього використовують сигмоїдальні функції. Отриманий вихідний вектор порівнюється з необхідним (правильним). Якщо вони збігаються, то вагові коефіцієнти зв'язків не змінюються. В іншому випадку обчислюється різниця між фактичними і необхідними вихідними значеннями, яка передається послідовно від вихідного шару до вхідного.

# ЛЕКЦІЯ 6. СПОСОБИ ОБРОБКИ ЗНАНЬ

Комп'ютерна обробка знань є однією з областей обробки інформації. Традиційна технологія обробки інформації орієнтована на обчислення, засновані на теорії Тюрінга, яка реалізована в ЕОМ з архітектурою фон Неймана. Цей підхід заснований на концепції послідовної обробки інформації в часі, тому в таких ЕОМ використовуються в основному мови процедурного типу, а в якості операційних механізмів – пристрої управління і арифметичні пристрої. Процеси обробки знань моделюють такі сфери людської діяльності, як міркування, систематизація, навчання і т.д. Якщо спробувати «перекласти подібні задачі на плечі комп'ютерів», останнім буде явно недостатньо жорсткого набору інструкцій у вигляді програми на процедурній мові програмування, навіть якщо ця програма дуже складна. Адекватне уявлення знань вимагає розробки і застосування мов декларативного типу в інтелектуальних системах, а в якості операційних механізмів – різних способів реалізації логічного висновку. Очевидно, майбутнє в цьому напрямку належить комп'ютерам з архітектурою, відмінною від фон-неймановської. Якими повинні бути комп'ютери, орієнтовані на виконання інтелектуальних задач, покажуть подальші теоретичні і практичні дослідження. В даний час експерименти проводяться в основному на машинах Тюрінга – фон Неймана і на машинах з паралельною обробкою даних.

Характерна риса ШІС, що відрізняє їх від традиційних систем обробки інформації, – використання знань. Вибір способу представлення знань в інтелектуальній системі є ключовим моментом розробки. З точки зору людини, бажано, щоб описові можливості використовуваної моделі були якомога вище. З іншого боку, складне уявлення знань вимагає спеціальних способів обробки (ускладнюється механізм виведення), що не тільки ускладнює проєктування і реалізацію експертної системи, але може привести до втрати достовірності результатів або неможливості їх інтерпретації. В кінцевому підсумку неминучий компроміс між найважливішими умовами та вимогами, висунутими до ШІС на етапі проєктування. Питання проєктування та реалізації інтелектуальних програмних засобів – окрема область

досліджень, а зараз розглянемо способи реалізації логічного висновку в системах з класичними моделями подання знань.

### 6.1. Способи доказу та виведення в логіці

Традиційною логікою називають формальну систему, запропоновану Арістотелем понад 2500 років тому. Арістотель прагнув встановити і формально записати способи, за допомогою яких можна було б встановити правильність міркувань в розумній полеміці. Множини правил, що визначають, які висновки можуть бути отримані з безлічі суджень, він назвав *силогізмом*. В даному випадку під *судженням* розуміється закінчена думка, яку можна висловити на природній мові в одній з наступних чотирьох форм:

– всі $X$ є $P$ – $(\forall X)P(X)$;

– ніякий $X$ не є $P$ – $(\forall X)\neg P(X)$;

– деякі з $X$ є $P$ – $(\exists X)P(X)$;

– деякі з $X$ не є $P$ – $(\exists X)\neg P(X)$.

Кожне правило силогізму визначає перехід від передумов до висновку, що є інтуїтивно очевидним. Силогізм початково призначений для управління розумною дискусією на природній мові. Як формальна теорія, він надмірно складний і не вичерпний. Силогізм можна назвати логікою класів, яка не містить поняття доповнення класу. Наприклад, визначивши клас $A$ (припустимо, суб'єкти, які є водіями автомобілів), ми не можемо побудувати його доповнення (тобто встановити всіх осіб, які не є водіями).

Будучи тісно пов'язаним з природною мовою, силогізм іноді призводить до абсурдних результатів, зокрема, не допускаючи логічну еквівалентність двох способів вираження однієї і тієї ж думки. Силогізм неповний в тому сенсі, що він не дозволяє здійснювати логічні висновки, в яких порушуються питання існування елемента деякого класу.

Послідовники Арістотеля, грунтуючись на силогізмі, сформулювали принципи дедуктивного виведення для висловлювань, які знаходяться на більш високому рівні абстракції в порівнянні з судженнями.

Існують два основні методи вирішення проблеми доказів в логіці: семантичний і синтаксичний.

*Семантичний метод* полягає в наступному. Можна перелічити всі атоми, що входять до формули $A_1$, $A_2$, ..., $A_n$, $B$, і скласти таблицю істинності для всіляких комбінацій значень цих атомів. Потім слід здійснити перегляд отриманої таблиці, щоб перевірити, чи в усіх її рядках, де формули $A_1$, $A_2$, ..., $A_n$ мають значення «істина», формула $B$ також має значення «істина». Цей метод можна застосовувати завжди, але він може виявитися дуже трудомістким.

При *синтаксичному методі* доказу спочатку записують посилки і, застосовуючи до них правила виведення, намагаються отримати з них інші справжні формули. З цих формул і вхідних посилок виводять наступні формули, і цей процес продовжують до тих пір, поки не буде отримано необхідного висновку (зауважимо, що це не завжди можливо). Цей процес, по суті справи, і є логічним висновком, він часто застосовується при доказі теорем в математиці.

Іноді значення конкретної логічної формули не залежить від значень вхідних в них атомів. Правильно побудовані логічні формули, значенням яких буде «істина» при будь-яких значеннях вхідних в них атомів, називаються *тавтологіями*. Тавтології, або теореми логіки, мають наступну властивість: якщо замість всіх входжень деякого атома в тавтологію підставити довільну логічну формулу, то знову буде отримана справжня формула. Ця нова формула називається окремим випадком вхідної формули, або результатом підстановки.

*Правило підстановки*. Якщо $C(A)$ – тавтологія і замість всіх входжень формули $A$ в $C$ підставити формулу $B$, то $C(B)$ – тавтологія. Для позначення тавтології використовується символ $\models$, який читається «загальнозначуще» або «завжди істинно».

$\models A \wedge (A \to B) \to B$;

$\models \neg B \wedge (A \to B) \to \neg A$;

$\models (A \to B) \to ((A \vee C) \to (B \vee C))$.

Доведемо те, що перша наведена тавтологія (*Modus Ponendo Ponens*) завжди буде мати значення «істина», для чого побудуємо скорочену таблицю істинності (табл. 6.1).

У наведеній таблиці $T$ – істина, $F$ – неправда. Значення імплікації збігається зі значенням другого аргументу в тому випадку, якщо перший аргумент має значення $T$, тому в перших двох рядках другого стовпця присутній другий аргумент – $B$, значення якого може бути довільним. Кон'юнкція $A \wedge (A \rightarrow B)$ при істинному $A$ також матиме результат, що співпадає із значенням $B$. Останній стовпець таблиці коментарів не потребує, так як наведені в ньому результати очевидні.

Таблиця 6.1

**Таблиця істинності для доказу тавтології**

| $A$ | $A \rightarrow B$ | $A \wedge (A \rightarrow B)$ | $A \wedge (A \rightarrow B) \rightarrow B$ |
|---|---|---|---|
| $T$ | $B$ | $B$ | $T$ |
| $T$ | $B$ | $B$ | $T$ |
| $F$ | $T$ | $F$ | $T$ |
| $F$ | $T$ | $F$ | $T$ |

Спробуємо встановити тавтологічність цієї ж формули синтаксичним методом. Для цього розкриємо всі дужки за розподільчим законом і переконаємося, що результат спрощується до формул з очевидними значеннями істинності

$(A \wedge (A \rightarrow B)) \rightarrow B = \neg(A \wedge (\neg A \vee B)) \vee B = (\neg A \vee (A \wedge \neg B)) \vee B =$

$= ((\neg A \vee A) \wedge (\neg A \vee \neg B)) \vee B = (T \wedge (\neg A \vee \neg B)) \vee B = \neg A \vee \neg B \vee B =$

$= \neg A \vee (\neg B \vee B) = \neg A \vee T = T.$

Крім використання тавтологій і підстановок корисним засобом для виведення є еквівалентність. Необхідно вміти правильно здійснювати заміну взаємно еквівалентних формул.

Наприклад, можна підставити $Q \vee P$ замість $P \vee Q$, так як $Q \vee P \leftrightarrow P \vee Q$.

Еквівалентність $A \leftrightarrow B$ можна замінити двосторонньою імплікацією $(A \rightarrow B) \wedge (B \rightarrow A)$, так як ці вирази мають одну і ту ж таблицю істинності. Звідси можна зробити висновок, що логічна еквівалентність – це імплікація в обох напрямках. Еквівалентність можна привести в кон'юнктивну нормальну форму, яка має

вигляд: $(\neg A \vee B) \wedge (\neg B \vee A)$. Якщо в цьому виразі розкрити дужки, отримаємо діз'юнктивну нормальну форму: $(\neg A \wedge \neg B) \vee (A \wedge B)$. Іншими словами, якщо $A$ і $B$ еквівалентні, то вони або обидва істинні, або обидва хибні.

Приклади тавтологій з еквівалентностями:

$\vDash (A \rightarrow B) \leftrightarrow (\neg B \rightarrow \neg A)$;

$\vDash (A \vee B) \leftrightarrow (B \vee A)$;

$\vDash A \wedge (A \vee B) \leftrightarrow A$;

$\vDash (X \rightarrow (Y \rightarrow Z)) \leftrightarrow (X \wedge Y \rightarrow Z)$.

У процедурах логічного доказу еквівалентності можна використову-вати два способи:

1) розписувати в вигляді двох окремих імплікацій;

2) використовувати при замінах.

При цьому важливо розуміти відмінність заміни від підстановки, яка полягає в наступному: якщо $A \leftrightarrow B$, то $A$ можна замінити на $B$ в будь-якому входженні в формулу $C$, не змінюючи її значення, причому цю заміну не обов'язково здійснювати у всіх входженнях.

На противагу тавтологіям в логіці існують протиріччя – формули, значенням яких завжди буде «неправда» незалежно від значень атомів, що до них входять. Прикладом є вираз $B \wedge \neg B = F$ при будь-якому значенні $B$.

Множина ППФ (правильно побудованих формул) для деякої предметної області називається теорією заданої області знань, а кожна окрема ППФ іменується аксіомою. Мета побудови теорії полягає в описі потрібних знань найбільш економічним способом. Якщо вдається побудувати теорію, яка адекватно описує задану область знань, то всі справжні факти з області інтерпретації будуть наслідками аксіом цієї теорії, іншими словами, їх можна буде вивести з множини ППФ. Відповідно неправдиві факти не будуть наслідками теорії, отже, їх не можна буде отримати шляхом логічного висновку на підставі аксіом даної теорії. Теорію називають *повною*, якщо всі справжні факти є наслідками цієї теорії. Це означає, що кожна справжня для даної теорії ППФ може бути доведена на підставі її аксіом.

Про теорію кажуть, що вона є *синтаксично послідовною* (*несуперечливою*), якщо з аксіом теорії неможливо вивести протиріччя.

Теорія, в якій можна довести і $P$, і $\neg P$, непослідовна.

Як приклад розглянемо побудову теорії та її перевірку на повноту і несуперечність для наступного фрагмента знань:

*«Якщо у Вас немає будинка,*
*Пожежі йому не страшні,*
*І дружина не піде до іншого,*
*Якщо у Вас немає дружини».*

Постараємося обійтися засобами логіки висловлювань і почнемо з формування області інтерпретації. Для цього введемо позначення:

*A – «У Вас є будинок»*
*B – «Будинок може згоріти»*
*C – «У Вас є дружина»*
*D – «Дружина може піти до іншого»*

На базі цих чотирьох висловлювань побудуємо логічні формули:

$$\neg A \rightarrow \neg B; \qquad \neg C \rightarrow \neg D.$$

Крім логічних формул, що виражають зв'язок фактів, будь-яка теорія містить справжні факти, на основі яких стає можливим конкретна інтерпретація ППФ. Припустимо, що в нашій теорії такими фактами є $A$, $B$, $\neg C$, $\neg D$.

Отримана теорія є повною і послідовною, оскільки кожен факт цієї теорії виводиться з інших аксіом, при цьому не виникне протиріччя. Довести це нескладно як семантичним, так і синтаксичним способом, однак для більш складних областей знань необхідно використовувати певні стратегії доказу, що дозволяють подолати хаотичність процесів логічного висновку.

Коротко розглянемо три основні стратегії доказу:

– доказ з введенням допущення;

– доказ методом «від супротивного» (приведення до протиріччя);

– доказ методом резолюції.

*Доказ з введенням допущення*. Для доказу імплікації виду $A \rightarrow B$ допускається, що ліва частина імплікації істинна, тобто $A$ приймається в якості додаткової посилки, після чого роблять спроби довести праву

частину *B*. Така стратегія доказу часто застосовується в геометрії при доказі теорем.

*Приведення до протиріччя*. Цей метод доказу, запропонований К. Робінсоном у 1960-х рр., вивів дослідження в галузі штучного інтелекту на новий рівень. Метод зумовив появу *зворотних висновків* і ефективних способів виявлення суперечностей. Суть його полягає в наступному. Наприклад, потрібно довести $A \rightarrow B$. Замість цього можна спробувати довести $\neg B \rightarrow \neg A$, використовуючи еквівалентність $A \rightarrow B$. Такий доказ можна провести двояко, а саме: чи допустити *A* і довести потім *B* (це буде прямий висновок), або зробити допущення про те, що *B* – помилкове, після чого зробити спробу спростувати посилку *A* (зворотний висновок). Приведення до протиріччя – комбінація прямого і зворотного виводу, тобто для доказу $A \rightarrow B$ можна одночасно допустити *A* і $\neg B$ (посилка істинна, а висновок – помилковий):

$$\neg(A \rightarrow B) = \neg(\neg A \vee B) = A \wedge \neg B.$$

У процесі доказу можна рухатися по шляху, який починається від *A* чи від ¬B. Якщо *B* виводиться з *A*, то, допустивши істинність *A,* ми довели б *B*. Тому, зробивши припущення ¬B, отримаємо протиріччя. Якщо висновок призведе до успіху (тобто суперечність не буде отримано), це буде свідчити про несумісність або суперечливості вхідних посилок. Ми також не отримаємо протиріччя, якщо доказувана пропозиція $A \rightarrow B$ є хибною.

*Доказ методом резолюції*. Цей метод вважається більш важким для розуміння, проте має важливу перевагу перед іншими: він легко формалізуємий. В основі методу лежить тавтологія, що отримала назву «правило резолюції»:

$$\models (X \vee A) \wedge (Y \vee \neg A) \rightarrow (X \vee Y).$$

## 6.2. Способи виведення в експертних системах продукційного типу

Будь-яка експертна система продукційного типу повинна містити три основних компонента: базу правил, робочу пам’ять і механізм виведення.

*База правил* (БП) – формалізовані за допомогою правил продукцій знання про конкретну предметну область.

*Робоча пам'ять* (РП) – область пам'яті, в якій зберігається безліч фактів, що описують поточну ситуацію, і всі пари атрибут-значення, які були встановлені до певного моменту. Вміст РП в процесі виконання завдання змінюється, зазвичай збільшуючись в обсязі в міру застосування правил. Іншими словами, РП – це динамічна частина бази знань, вміст якої залежить від оточення розв'язуваної задачі. У найпростіших ЕС збережені в РП факти не змінюються в процесі виконання завдання, однак існують системи, в яких допускається зміна і видалення фактів з РП. Це системи з немонотонним висновком, що працюють в умовах неповноти інформації.

*Механізм виведення* виконує дві основні функції: перегляд існуючих в робочій пам'яті фактів і правил з БП, а також додавання в РП нових фактів і визначення порядку перегляду і застосування правил. Порядок може бути прямим або зворотним.

*Прямий порядок* – від фактів до висновків. В експертних системах з прямими висновками з відомих фактів відшукується висновок, який витікає з цих фактів. Якщо такий висновок вдається знайти, він заноситься в робочу пам'ять. Прямі висновки часто застосовуються в системах діагностики, їх називають висновками, керованими даними.

*Зворотний порядок* виведення – від висновків до фактів. У системах зі зворотним висновком спочатку висувається деяка гіпотеза про кінцеве судження, а потім механізм виведення намагається знайти в робочій пам'яті факти, які могли б підтвердити або спростувати висунуту гіпотезу. Процес відшукання необхідних фактів може включати досить велике число кроків, при цьому можливе висування нових гіпотез (цілей). Зворотні висновки управляються цілями.

Для виконання зазначених функцій механізм виведення включає компоненту виведення і керуючу компоненту.

*Компонента виведення*. Її дія заснована на застосуванні правила логічного висновку *Modus Ponendo Ponens*. Суть застосування цього правила в продукційних системах полягає в наступному. Якщо в РП присутній істинний факт *A* і в БП існує правило виду «ЯКЩО *A*, ТО *B*», то факт *B* визнається істинним і заноситься в РП. Такий висновок легко реалізується на ЕОМ, однак при цьому часто виникають проблеми, пов'язані з розпізнаванням

значень слів, а також з тим, що факти можуть мати внутрішню структуру і між елементами цієї структури можливі різного роду зв'язки. Наприклад, нехай є факт *А* – «Автомобіль Іванчука – білий» і правило «ЯКЩО Автомобіль – білий, ТО Автомобіль легко помітити вночі». Людина легко виведе висновок «Автомобіль Іванова легко помітити вночі», але це не під силу ЕС чисто продукційного типу. Вона не зможе сформувати такий висновок, тому що *А* не збігається точно з антецедентом (основа, причина) правила. Подібна проблема вже зачіпалася, коли розглядалися відмінності логіки висловлювань і логіки предикатів. Крім того, невисока інтелектуальна потужність продукційних систем обумовлена тим, що людина виводить висновки, маючи в своєму розпорядженні всі свої знання, тобто БЗ величезного обсягу, в той час як ЕС здатні вивести порівняно невелику кількість висновків, використовуючи задану множину правил. Зі сказаного можна зробити висновок про те, що компонента виведення в ЕС повинна бути організована так, щоб бути здатною функціонувати в умовах нестачі інформації.

*Керуюча компонента*. Вона визначає порядок застосування правил, а також встановлює, чи є ще факти, які можуть бути змінені в разі продовження роботи (при немонотонному виведенні). Механізм виведення працює циклічно, при цьому в одному циклі може спрацювати тільки одне правило. Схема циклу наведена на рис. 6.1.

У циклі виконуються наступні основні операції:

– зіставлення – зразок (антецедент) правила порівнюється з наявними в РП фактами;

– дозвіл конфліктного набору – вибір одного з декількох правил в тому випадку, якщо їх можна застосувати одночасно;

– спрацьовування правила – в разі збігу зразка деякого правила з бази правил з фактами, наявними в робочій пам'яті, відбувається спрацьовування правила, при цьому воно відзначається в БП;

– дія – зміна вмісту РП шляхом додавання туди висновку спрацювавшого правила.

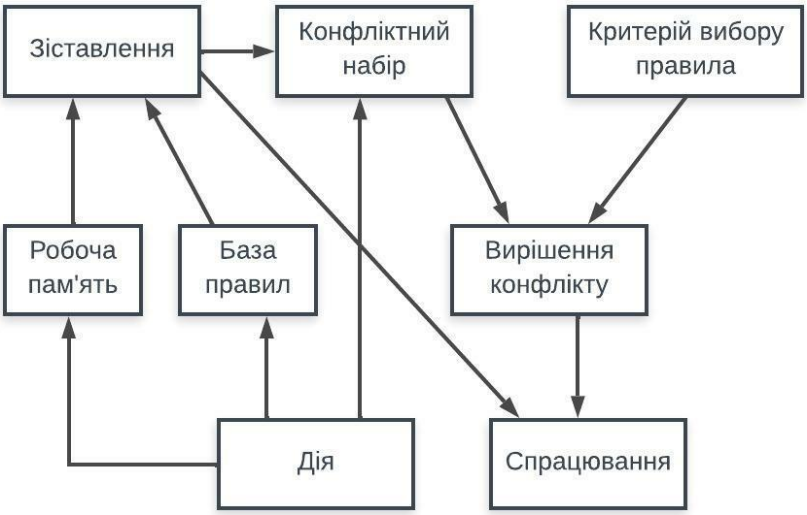


Рис. 6.1. Схема циклу роботи механізму виведення

Якщо у висновку міститься директива на виконання деякої процедури, остання виконується.

Оскільки механізм виведення працює циклічно, слід знати про способи завершення циклу. Традиційними способами є або вичерпання всіх правил з БП, або виконання деякої умови, якій задовольняє вміст робочої пам’яті (наприклад, поява в ній якогось зразка), або комбінація цих способів.

Особливістю ЕС є те, що вони не мають у своєму розпорядженні процедур, які могли б побудувати в просторі станів відразу весь шлях вирішення завдання. Траєкторія пошуку рішення повністю визначається даними, отриманими від користувача в процесі логічного висновку.

Розглянемо найпростіші приклади прямого і зворотного виведення в системах продукційного типу.

*Приклад прямого виведення*. Нехай в БП є наступні правила:

Правило 1. «ЯКЩО *Двигун не заводиться*, І *Фари не горять*, ТО *Сів акумулятор*».

Правило 2. «ЯКЩО *Покажчик бензину знаходиться на нулі*, ТО *Двигун не заводиться*».

Припустимо, що в робочу пам'ять від користувача ЕС надійшли факти: *Фари не горять*, і *Покажчик бензину знаходиться на нулі*.

Розглянемо основні кроки алгоритму прямого виведення.

1. Зіставлення фактів з РП із зразками правил з БП. Правило 1 не може спрацювати, а правило 2 спрацьовує, тому що зразок, що співпадає з його антецедентом, присутній в РП.

Дія спрацювавшого Правила 2: в РП заноситься висновок цього правила – зразок *Двигун не заводиться*.

2. Другий цикл зіставлення фактів в РП із зразками правил. Тепер спрацьовує Правило 1, так як кон'юнкція умов в його антецеденті стає справжньою.

Дія Правила 1, яка полягає у видачі користувачеві остаточного діагнозу – *Сів акумулятор*.

3. Кінець роботи (БП вичерпана).

*Приклад прямого виведення з конфліктним набором*. Тепер припустимо, що в БП крім Правила 1 і Правила 2 присутнє Правило 3:

«ЯКЩО *Покажчик бензину знаходиться на нулі*, ТО *Немає бензину*».

В РП знаходяться ті ж факти, що в попередньому прикладі.

В результаті зіставлення в першому ж циклі можливе застосування двох правил – Правила 2 і Правила 3, тобто виникає конфліктний набір і постає завдання вибору: яке з цих правил застосувати першим. Якщо виберемо Правило 2, то в РП додасться факт «*Двигун не заводиться*» і на наступному кроці знову виникне конфліктний набір, так як можна буде застосувати Правило 1 і Правило 3. Якщо буде вибрано Правило 1, то до висновку «*Сів акумулятор*» прийдемо за два кроки. При будь-якому іншому виборі порядку застосування правил до цього ж висновку приходимо за три кроки. Якщо завершення циклу роботи ЕС настає після перегляду всіх правил, то число кроків буде дорівнювати трьом, причому порядок застосування правил не матиме жодного значення.

*Приклад зворотного виведення*. Припустимо, що в БП є два правила (Правило 1 і Правило 2), а в РП – ті ж факти, що в попередніх прикладах з прямим висновком.

Алгоритм зворотного виведення містить наступні кроки.

1. Було висунуто гіпотезу остаточного діагнозу – «*Сів акумулятор*».

2. Відшукується правило, висновок якого відповідає висунутій гіпотезі, в нашому прикладі – це Правило 1.

3. Досліджується можливість застосування Правила 1, тобто вирішується питання про те, чи може воно спрацювати. Для цього в робочій пам'яті повинні бути присутніми факти, що збігаються зі зразком цього правила. У розглянутому прикладі Правило 1 не може спрацювати через відсутність в РП зразка «*Двигун не заводиться*». Цей факт стає новою метою на наступному кроці виведення. Пошук правила, висновок якого відповідає новій меті і таке правило є – Правило 2.

4. Досліджується можливість застосування Правила 2 (зіставлення). Воно спрацьовує, тому що в РП присутній факт, що співпадає з його зразком.

5. Дія Правила 2, що складається в занесенні висновку «*Двигун не заводиться*» в РП.

6. Умовна частина Правила 1 тепер підтверджена фактами, отже, воно спрацьовує, і висунута початкова гіпотеза підтверджується.

7. Кінець роботи.

При порівнянні цього прикладу з прикладом прямого виведення можна помітити перевагу зворотних виведень перед прямими.

*Приклад зворотного виведення з конфліктним набором*. Припустимо, що в БП записані Правило 1, Правило 2, Правило 3 і Правило 4:

«ЯКЩО *Засмітився бензонасос*, ТО *Двигун не заводиться*».

В РП присутні ті ж самі факти: *Фари не горять* і *Покажчик бензину знаходиться на нулі*.

В даному випадку алгоритм зворотного виведення з конфліктним набором включає наступні кроки.

1. Було висунуто гіпотезу «*Сів акумулятор*».

2. Пошук правила, висновок якого збігається з поставленою метою. Це Правило 1.

3. Досліджується можливість застосування Правила 1. Воно не може спрацювати, висувається нова підціль «*Двигун не заводиться*», що відповідає недостаючому зразку.

4. Пошук правил, висновки яких збігаються з новою підціллю. Таких правил два – Правило 2 і Правило 4. Якщо виберемо Правило 2, то подальші кроки співпадають з прикладом безконфліктного набору. Якщо виберемо Правило 4, то воно не спрацює, тому що в РП немає зразка «*Засмітився бензонасос*». Після цього буде застосовано Правило 2, що приведе до успіху, але шлях виявиться довшим на один крок.

Слід звернути увагу на те, що Правило 3, не пов'язане з поставленою метою, взагалі не піднімалося в процесі виведення. Цей факт свідчить про більш високу ефективність зворотних виведень у порівнянні з прямими, тому що при зворотних виведеннях існує тенденція виключення з розгляду правил, що не мають відношення до поставленої мети. В експертних системах процедури управління логічним виведенням закриті не тільки для користувача, але і для інженера по знаннях, однак про них необхідно мати уявлення, щоб коректно інтерпретувати результати. Для цього потрібно знати, в якому вигляді зберігаються знання і як обираються початкова точка пошуку, правила вирішення конфліктів, структура, за допомогою якої зберігаються знання. Наприклад, у відомому сімействі ЕС OPS застосовується стратегія прямих виведень, ефективність яких істотно підвищується завдяки використанню алгоритму узгодження RETE при генерації конфліктного набору. Суть цього алгоритму зводиться до наступного: кожен раз при додаванні в РП нового зразка перевіряється правило, в якому він використовується, і якщо зразок задовольняє антецеденту деякого правила, то він запам'ятовується саме в цій якості. У конфліктний набір правило включається тільки в тому випадку, якщо додавання зразка задовольняє всім умовам. Для вирішення конфліктів в системах сімейства OPS, а також в інших системах з прямими виведеннями широкого поширення набув метод вирішення конфліктів LEX, в якому перевага віддається правилам з посиланням на самий останній згенерований

зразок. Якщо таких правил декілька, то серед них обирається правило з найбільшим числом умов в антецеденті.

У великих ЕС продукційного типу вся множина знань зазвичай зберігається у вигляді дерева, яке називається І-АБО–графом. Фрагменти такої структури наведено на рис. 6.2 і 6.3. Класична форма продукцій передбачає наявність в антецеденті тільки зв'язки І. На практиці класична форма може бути розширена, наприклад, введенням зв'язки АБО в умовну частину або включенням в антецедент обчислень на підставі вмісту робочої пам'яті і т.п.

Якщо існує безліч правил, з яких виводиться один і той же висновок, то, виконавши операцію диз'юнкції над усіма висновками, отриманими за допомогою цих правил, можна показати відношення між результатом окремого виведення і даними, на підставі яких робиться висновок.

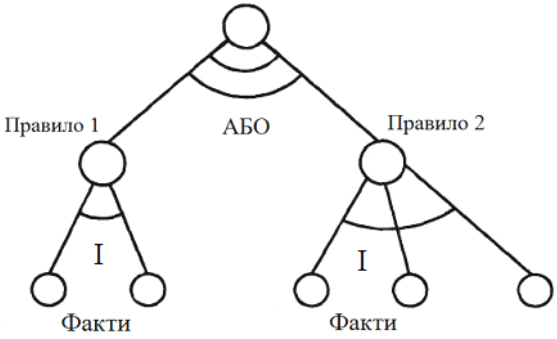


Рис. 6.2. Найпростіший фрагмент структури І-АБО–графа

За допомогою І-АБО–графа зворотне виведення в ЕС продукційного типу можна представити як проблему пошуку певного шляху на графі. Вибір однієї із зв'язок АБО відповідає вирішенню конфліктного набору, при цьому не байдужий порядок оцінки умов в антецеденті, з'єднаних зв'язкою І. Задачі та стратегії пошуку на І-АБО–графах широко висвітлені в літературі. Однак слід зупинитися на способах підвищення ефективності пошуку, тому що в системах, що мають практичну цінність, нараховуються сотні

правил, і слід знати, за допомогою яких стратегій управління виводу можна мінімізувати час вирішення задач.

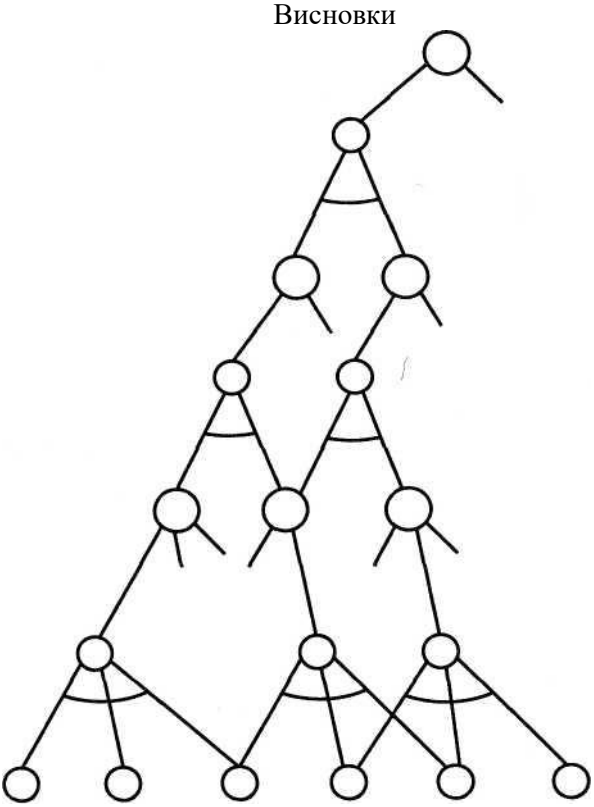


Рис. 6.3. Фрагмент структури І-АБО–графа продукційної експертної системи

*Стратегія пошуку в глибину*. При виборі чергової підцілі в процесі зворотного виведення перевага завжди, коли можливо, віддається тій, яка відповідає наступному, більш детальному рівню опису задачі. Наприклад, система діагностики, зробивши на підставі відомих симптомів припущення про причини несправності, буде запитувати уточнюючі ознаки і симптоми до тих пір, поки повністю не підтвердить (або спростує) висунуту

гіпотезу. Приклад організації пошуку в глибину показаний на рис. 6.4, де цифрами позначені номери кроків пошуку.

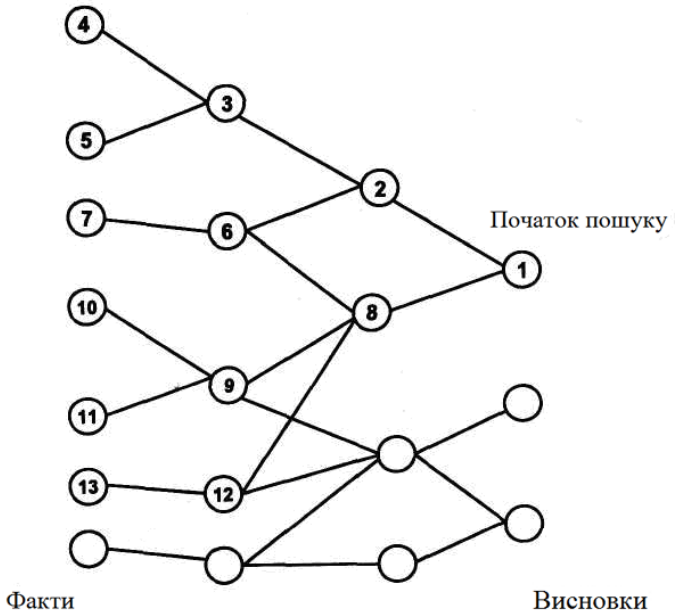


Рис. 6.4. Пошук в глибину при зворотному виведенні

*Стратегія пошуку в ширину*. При пошуку в ширину спочатку аналізуються всі симптоми (факти), що знаходяться на одному рівні простору станів задачі, навіть якщо вони належать до різних цілей (підцілей), і тільки після цього відбувається перехід до пошуку симптомів наступного рівня.

На рис. 6.5 показані кроки пошуку в ширину, позначені номерами, зазначеними в вершинах. На малюнку представлена стратегія зворотного виведення на тому ж І-АБО–графі, який наведено і на рис. 6.4. Алгоритм пошуку в глибину більш ефективний щодо часу пошуку і обробки знань, проте він характеризується більш високим ризиком втрати перспективних рішень в порівнянні з пошуком в ширину.

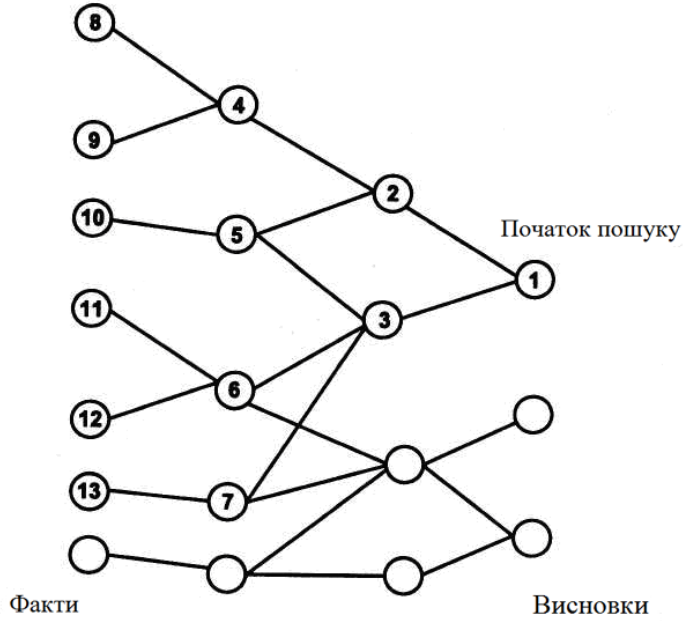


Рис. 6.5. Пошук в ширину при зворотному виведенні

*Розбиття на підзадачі*. Декомпозиція дає позитивний ефект тільки для добре структурованих областей знань, тому що застосування цієї стратегії засноване на правильному розумінні суті задачі і можливості її подання у вигляді системи ієрархічно пов'язаних цілей-підцілей, причому розбиття на підзадачі необхідно виконати оптимальним способом.

*α-β–алгоритм*. За допомогою цього алгоритму вихідна задача зводиться до зменшення простору станів шляхом видалення з нього гілок, неперспективних для пошуку успішного вирішення, тобто проглядаються тільки ті вершини, в які можна потрапити в результаті наступного кроку, після чого неперспективні напрямки виключаються. Наприклад, в БЗ продукційної системи, заповненої знаннями про тваринний світ, не слід шукати тварин, що не відносяться до ссавців, в напрямку, що бере початок від вершини, яка визначає ссавців. Дана стратегія є певним компромісом між пошуком в ширину і пошуком в глибину. Для її успішної реалізації слід мати додаткові

евристичні знання, які використовуються при виборі перспективних напрямків. Вражаючий приклад застосування варіанту цієї стратегії продемонстрований розробниками системи Deep Blue, що зуміла обіграти кращого шахіста планети.

### 6.3. Обробка знань в інтелектуальних системах з фреймовим поданням

В інтелектуальних системах з фреймовим поданням знань використовуються три способи управління логічним виведенням: демони, приєднані процедури і механізм успадкування. Останній можна назвати єдиним основним механізмом виведення, яким оснащені фреймові (об'єктно-орієнтовані) системи.

Управлінські функції механізму наслідування полягають в автоматичному пошуку і визначенні значень слотів фреймів нижчих рівнів за значеннями слотів фреймів верхніх рівнів, а також в запуску приєднаних процедур і демонів. Приєднані процедури і демони дозволяють реалізувати будь-який механізм виведення в системах з фреймовим поданням знань. Однак ця реалізація має конкретний характер і вимагає значних витрат праці проєктувальників і програмістів.

Розглянемо простий приклад. У табл. 6.3 показана структура фрейму «Наукова конференція».

Таблиця 6.3

**Фрейм «Наукова конференція»**

| Ім'я слота | Значення слота | If-needed | If-added | If-removed |
|---|---|---|---|---|
| Дата<br>Місце проведення<br>Тема доповіді<br><br>Доповідач | 1.02. 10:10<br>Аудиторія 209<br>Прогнозування тенденцій в економіці<br>Іванчук І.І. | *ХТО?* | *ЗАМОВЛЕННЯ* | |

Демон *ЗАМОВЛЕННЯ* – це процедура, яка автоматично запускається при спробі підстановки значення в слот з ім'ям *Місце проведення*.

Її головне призначення полягає в перевірці можливості замовлення аудиторії на потрібний час. Така процедура на мові LISP може виглядати приблизно так:

*LISP proc ЗАМОВЛЕННЯ (Назва конференції, Місце проведення, дата)*

*if можливо(Назва конференції, Місце проведення, дата)*

*then замовити(Назва конференції, Місце проведення, дата)*

*else повідомлення(«Замовлення неможливе», Назва конференції)*

*end.*

Демон *ХТО*? автоматично запускається при зверненні до слоту *Доповідач*, якщо значення цього слоту не визначене. Основний зміст даної процедури – генерація запиту до користувача типу «*Хто виступає*?», отримання відповіді та його запис у якості значення слота.

Реалізація виведення за допомогою приєднаних процедур вимагає наявності механізму обміну інформацією між фреймами. В якості такого механізму зазвичай використовується механізм повідомлень.

Можливість організації висновків будь-якого типу є суттєвою перевагою фреймових систем у порівнянні з продукційними і логічними. Не менш важливою перевагою є більша схожість цієї моделі подання знань зі структурою знань в пам’яті людини. Разом з тим практична реалізація фреймових систем пов’язана зі значною трудомісткістю, як на етапі проектування, так і при реалізації. Тому вартість промислових експертних систем фреймового типу на порядок перевершує вартість продукційних систем.

## ЛЕКЦІЯ 7. НЕЧІТКІ ЗНАННЯ ТА НЕЧІТКА ЛОГІКА

Найбільш вражаючою властивістю людського інтелекту є здатність приймати правильні рішення щодо неповної та нечіткої інформації. Побудова моделей наближених міркувань людини та їх використання у комп'ютерних системах майбутніх поколінь представляє сьогодні одну з найважливіших проблем науки.

На початку 1920-х років польський математик Лукашевич працював над принципами багатозначної математичної логіки, в якій значеннями предикатів могли бути не лише «істина» чи «неправда». У 1937 р. ще один американський вчений Макс Блек вперше застосував багатозначну логіку Лукашевича до списків як множин об'єктів і назвав такі множини невизначеними.

Перш ніж нечіткий підхід до моделювання складних систем отримав визнання у всьому світі, минуло не одне десятиліття з моменту зародження теорії нечітких множин.

Значне просування в цьому напрямку зроблено професором Каліфорнійського університету (Берклі) Лотфі А. Заде (Lotfi A. Zadeh). Його робота «Fuzzy Sets», що з'явилася в 1965 році в журналі Information and Control, №8, заклала основи моделювання інтелектуальної діяльності людини і стала початковим поштовхом до розвитку нової математичної теорії.

Що ж запропонував Заде? По-перше, він розширив класичне канторовське поняття *множини*, допустивши, що характеристична функція (функція приналежності елемента множині) може приймати будь-які значення в інтервалі (0; 1), а не тільки значення 0 або 1. Такі множини були названі ним *нечіткими* (fuzzy). Л. Заде визначив також ряд операцій над нечіткими множинами і запропонував узагальнення відомих методів логічного виведення *modus ponens* та *modus tollens*.

Ввівши потім поняття *лінгвістичної змінної* і допустивши, що в якості її значень (термів) виступають нечіткі множини, Л. Заде створив апарат для опису процесів інтелектуальної діяльності, включаючи нечіткість і невизначеність виразів.

Подальші роботи професора Л. Заде та його послідовників заклали міцний фундамент нової теорії і створили передумови для впровадження методів нечіткого управління в інженерну практику.

Вже до 1990 року з цієї проблематики опубліковано понад 10000 робіт, а число дослідників досягло 10000, причому в США, Європі та СРСР по 200 – 300 дослідників, близько 1000 – в Японії, 2000 – 3000 – в Індії і близько 5000 дослідників в Китаї.

Нечітка логіка як науковий напрям розвивалася складно і непросто, не уникла вона і звинувачень в лженаучності. Навіть в 1989 році, коли приклади успішного застосування нечіткої логіки в обороні, промисловості та бізнесі обчислювалися десятками, Національне наукове товариство США обговорювало питання про виключення матеріалів по нечітким множинам з інститутських підручників.

Перший період розвитку нечітких систем (кінець 60-х – початок 70 рр.) характеризується розвитком теоретичного апарату нечітких множин. У 1970 р. Беллман спільно з Заде розробив теорію прийняття рішень в нечітких умовах.

У другому періоді (70–80-і роки) з'являються перші практичні результати в області нечіткого управління складними технічними системами (парогенератор з нечітким управлінням). І. Мамдані в 1975 р. спроєктував перший функціонуючий на основі алгебри Заде контролер, керуючий паровою турбіною. Одночасно стало приділятися уваги питанням побудови експертних систем, побудованих на нечіткій логіці, розробці нечітких контролерів. Нечіткі експертні системи для підтримки прийняття рішень знаходять широке застосування в медицині та економіці.

Нарешті, в третьому періоді, який триває з кінця 80-х років і триває в даний час, з'являються пакети програм для побудови нечітких експертних систем, а області застосування нечіткої логіки помітно розширюються. Вона застосовується в автомобільній, аерокосмічній і транспортній промисловості, в області виробів побутової техніки, у сфері фінансів, аналізу та прийняття управлінських рішень і багатьох інших. Крім того, чималу роль у розвитку нечіткої логіки зіграло доказ знаменитої теореми FAT (Fuzzy Approximation Theorem) Б. Коско, в якій стверджувалося, що будь-яку математичну систему можна апроксимувати системою на основі нечіткої логіки.

Одним з найбільш вражаючих результатів стало створення керуючого мікропроцесора на основі нечіткої логіки, здатного автоматично вирішувати відому «задачу про собаку, що наздоганяє

кота». У 1990 р. Комітет з контролю експорту США вніс нечітку логіку до списку критично важливих оборонних технологій, що не підлягають експорту потенційному противнику.

У бізнесі та фінансах нечітка логіка отримала визнання після того, як в 1988 році експертна система на основі нечітких правил для прогнозування фінансових індикаторів єдина передбачила біржовий крах. І кількість успішних фаззі-застосувань в даний час обчислюється тисячами.

В Японії цей напрямок переживає справжній бум. Тут функціонує спеціально створена організація – Laboratory for International Fuzzy Engineering Research. Програмою цієї організації є створення найближчих людині обчислювальних пристроїв.

Інформаційні системи, що базуються на нечітких множинах і нечіткій логіці, називають нечіткими системами.

В останні 10-15 років почалося використання нових методів і моделей в промисловості. І хоча перші застосування нечітких систем управління відбулися в Європі, найбільш інтенсивно впроваджуються такі системи в Японії. Спектр додатків їх широкий: від управління процесом відправлення і зупинки поїзда метрополітену, управління вантажними ліфтами і доменною піччю до пральних машин, пилососів і СВЧ-печей. При цьому нечіткі системи дозволяють підвищити якість продукції при зменшенні ресурсів та енерговитрат і забезпечують більш високу стійкість до впливу факторів, що заважають, порівняно з традиційними системами автоматичного управління.

Іншими словами, нові підходи дозволяють розширити сферу додатка систем автоматизації за межі застосування класичної теорії. В цьому плані цікава точка зору Л. Заде: «Я вважаю, що зайве прагнення до точності стало впливати на теорію управління і теорію систем, так як воно призводить до того, що дослідження в цій області зосереджуються на тих і тільки тих проблемах, які піддаються точному рішенню. В результаті багато класів важливих проблем, в яких дані, цілі та обмеження є занадто складними або погано визначеними для того, щоб допустити точний математичний аналіз, залишалися і залишаються осторонь з тієї причини, що вони не піддаються математичному трактуванню. Для того щоб сказати що-небудь істотне для проблем подібного роду, ми повинні

відмовитися від наших вимог точності і допустити результати, які є декілька розмитими або невизначеними».

Зсув центру досліджень нечітких систем в бік практичних додатків призвело до постановки цілого ряду проблем таких, як нові архітектури комп'ютерів для нечітких обчислень, елементна база нечітких комп'ютерів і контролерів, інструментальні засоби розробки, інженерні методи розрахунку і розробки нечітких систем управління та багато іншого.

*Переваги* нечітких систем:

- функціонування в умовах невизначеності;
- оперування якісними та кількісними даними;
- використання експертних знань в управлінні;
- побудова моделей наближених міркувань людини;
- стійкість при дії на систему всіляких збурень.

*Недоліками* нечітких систем є:

- відсутність стандартної методики конструювання нечітких систем;
- неможливість математичного аналізу нечітких систем існуючими методами;
- застосування нечіткого підходу в порівнянні з імовірнісним не призводить до підвищення точності обчислень.

Широкою областю ефективного застосування інтелектуальних систем як засобу побудови інформаційних систем нового покоління є область нечітких знань. Це пов'язано з тим, що у всіх предметних областях істотне місце займають некоректні, нечітко сформульовані задачі і реальний людський спосіб міркування (спирається на природну мову) не може бути описаний в рамках традиційних математичних формалізмів, які передбачають однозначність інтерпретації. Іншими словами, знання найчастіше нечіткі. Для того щоб ІС вийшли за рамки простих символьних виведень і наблизилися до мислення людини, необхідні методи подання нечітких знань і механізми виведень, що працюють в їх середовищі. Виникла необхідність створення теорії, що дозволяє формально описувати нестрогі, нечіткі поняття і моделювати міркування, що містять такі поняття.

Усі недовизначенності в знаннях можна класифікувати наступним чином:

- недетермінованість висновків;

- багатозначність;
- ненадійність;
- неповнота;
- нечіткість або неточність.

*Недетермінованість висновків*

Це характерна риса більшості систем штучного інтелекту. Недетермінованість означає, що заздалегідь шлях вирішення конкретної задачі у просторі її станів визначити неможливо. Тому в більшості випадків методом проб і помилок обирається деякий ланцюжок логічних висновків, які узгоджуються з наявними знаннями, а в разі якщо вона не призводить до успіху, організовується перебір з поверненням для пошуку іншого ланцюжка і т.д. Такий підхід передбачає визначення деякого початкового шляху. Для вирішення подібних задач запропоновано безліч евристичних алгоритмів.

Розглянемо один з них – класичний алгоритм *A**, розроблений на етапах становлення штучного інтелекту. В алгоритмі *A** використовуються оціночні функції, побудовані на основі апріорних оцінок вартості шляху до цільового стану. Такі оцінки, по суті справи, теж є евристичними знаннями. Для пошуку в просторі станів використовуються дерево пошуку та методи горизонтального (в ширину) і вертикального (в глибину) пошуку на цьому дереві. Основні кроки і поняття алгоритму розглянемо на прикладі гри у «8», що є усіченою версією гри у «15». Метою гри є перехід з деякого початкового стану в кінцевий, як показано на рис. 7.1.

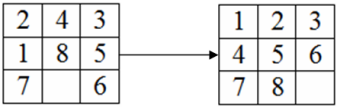


Рис. 7.1. Перехід з початкового стану в кінцевий у грі «8»

У цій грі в якості основного об’єкта зручніше розглядати не ті шашки, що пересуваються, а саме розглядати переміщення порожнього квадрата. При цьому можна визначити чотири основних оператора, виконуваних над порожнім квадратом:

- переміщення порожнього квадрата вліво;

– переміщення порожнього квадрата вгору;
– переміщення порожнього квадрата вниз;
– переміщення порожнього квадрата вправо.

Оціночна функція $f(n)$ буде формуватися як вартість *оптимального* шляху до мети з початкового стану через *n* вершин дерева пошуку. Дерево пошуку для даного прикладу показано на рис. 7.2. Значення оціночної функції в *n*-й вершині можна уявити як суму двох складових $f(n)=g(n)+h(n)$, де $g(n)$ – вартість оптимального шляху від першої вершини до *n*-ої, а $h(n)$ – вартість оптимального шляху від *n*-ої вершини до мети. Для простоти будемо вважати, що вартість переміщення однієї шашки (або порожнього квадрата) дорівнює 1. Оптимальним буде шлях, який має мінімальну вартість. Точне значення $f(n)$ неможливо знати в процесі пошуку, тому введемо апріорну оцінку значення функції: $\hat{f}(n)=g(n)+\hat{h}(n)$, де $g(n)$ – глибина пройденого шляху на дереві пошуку від 1-ої до *n*-ої вершини; $\hat{h}(n)$ – апріорне значення $h(n)$.

Основна проблема полягає у визначенні другої компоненти $h(n)$, так як цей шлях ще не пройдений. У якості апріорної оцінки $\hat{h}(n)$ можна, наприклад, взяти число шашок, що знаходяться не на своїх місцях на *n*-му кроці пошуку. Сформувавши таким чином оціночну функцію, визначимо стратегію вибору вершин (застосування операторів), в яких значення функції мінімальні. Результат пошуку показаний на рис. 7.2, де цифри у кружках показують послідовність переходів з початкового стану в кінцевий.

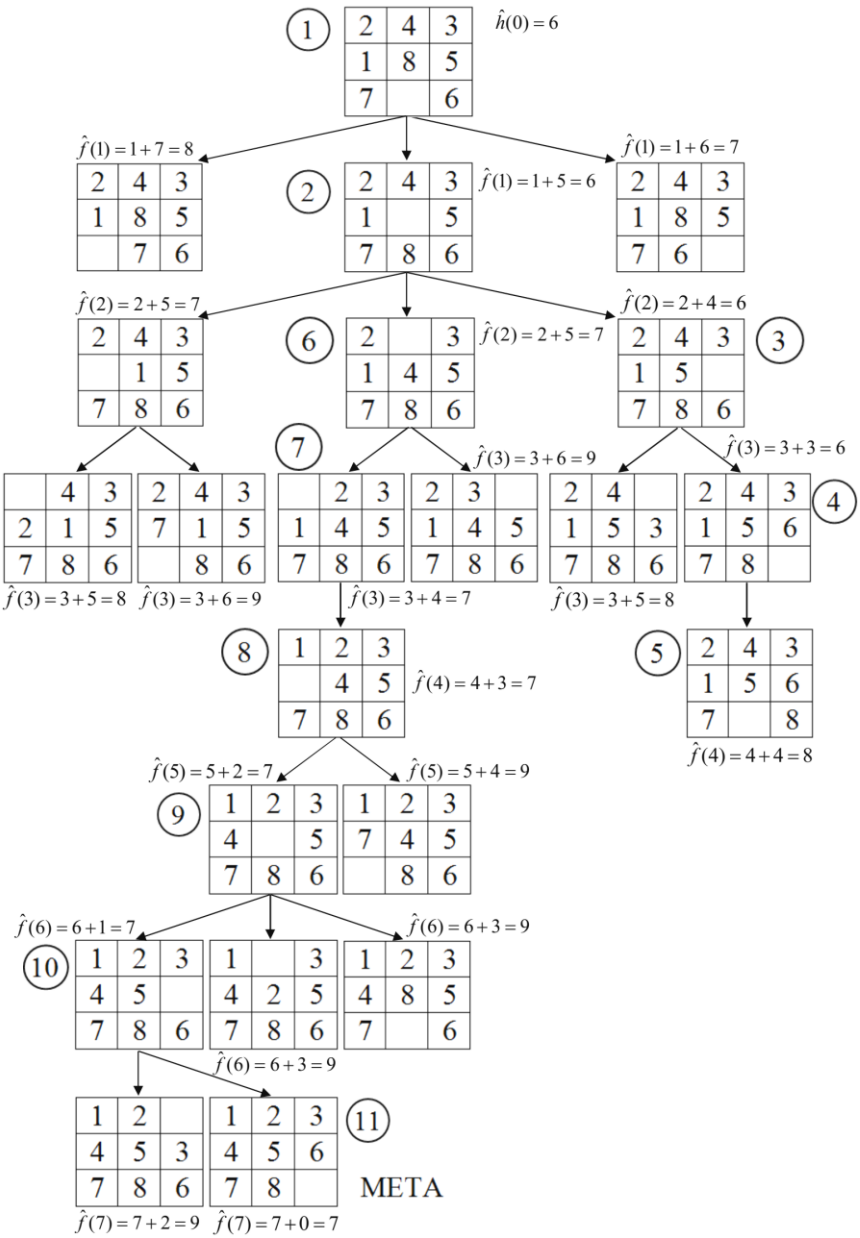


Рис. 7.2. Дерево пошуку для гри у «8»

Проаналізуємо основні кроки алгоритму.

1. Розглядаємо всі можливі оператори над порожнім квадратом в початковому стані і обираємо варіант з найменшим значенням $\hat{h}(n)$.

2. Застосовуємо обраний оператор, в результаті отримуємо новий стан.

3. Створюємо вершини наступного рівня ієрархії, аналізуючи застосування всіх можливих операторів для переходу в новий стан.

4. Обираємо стан з найменшим значенням $\hat{h}(n)$.

5. Повторюємо перераховані дії до тих пір, поки не досягнемо мети.

В даному випадку важливо, що $\hat{h}(n) \le h(n)$, так як якщо апріорна оцінка вартості оптимального шляху не перевищує істинної вартості, то знаходження оптимального шляху гарантовано. Цю умову можна інтерпретувати наступним чином: мета пошуку не буде досягнута, поки число переміщень менше числа шашок, що знаходяться не на своїх місцях.

Якщо $\hat{h}(n)$ вибрати по-іншому, наприклад $\hat{h}(n) = 0$, то будемо здійснювати горизонтальний пошук на дереві станів задачі, при якому розкриваються всі вершини нижчележачого рівня.

Таким чином, *недетермінованість висновків* – риса, органічно притаманна інтелектуальним системам, тобто неусувана компонента нечіткості знань. Її слід враховувати при виробленні ефективних способів подання і зберігання знань, а також при побудові алгоритмів пошуку і обробки знань, які дозволяють отримати рішення задачі за найменше число кроків. Для побудови таких алгоритмів зазвичай застосовуються евристичні метазнання, тобто знання про знання.

*Багатозначність*

Багатозначність інтерпретації – звичайне явище в задачах розпізнавання. При розумінні природної мови серйозними проблемами стають багатозначність значення слів, їх підпорядкованості, порядку слів в реченні і т. д. Проблеми розуміння значення виникають в будь-якій системі, яка взаємодіє з користувачем на природній мові. Розпізнавання графічних образів

також пов'язано з вирішенням проблеми багатозначної інтерпретації. При комп'ютерній обробці знань багатозначність необхідно усувати шляхом вибору правильної інтерпретації, для чого розроблені спеціальні методи.

Розглянемо один з таких методів – *метод релаксації*, призначений для систематичного усунення багатозначності при інтерпретації зображень.

Усунення багатозначності досягається за допомогою циклічних операцій фільтрації.

Розглянемо приклад: Нехай є чорно-біле зображення (рис 7.3), яке повинен розпізнати комп'ютер.

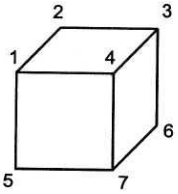


Рис. 7.3. Зображення, що розпізнається

Одним з етапів розпізнавання об'єкта є інтерпретація сенсу ліній. Для ідентифікації граней введемо такі мітки:

(+) – опукла грань;

(–) – увігнута грань;

(→) – праворуч від стрілки знаходиться видима поверхня.

Перший цикл методу релаксації здійснює «локальний погляд» на кожну вершину, при цьому будемо мати інтерпретації вершин 1–7. Якщо при інтерпретації заданого контуру розглядати відразу дві сусідні вершини, то вони повинні мати загальну грань з однією і тією ж міткою, при цьому число варіантів інтерпретації зменшується. Подібну операцію називають *фільтрацією*. Її можна застосовувати неодноразово з метою усунення багатозначності, до тих пір, поки подальше застосування фільтрації не призводить до скорочення числа вершин-кандидатів для інтерпретації.

*Ненадійність знань та висновків*

Ненадійність знань означає, що для оцінки їх достовірності не можна застосувати двобальну шкалу (1 – абсолютно достовірні; 0 – недостовірні знання). Для більш тонкої оцінки достовірності знань

застосовується імовірнісний підхід, заснований на теоремі Байєса, та інші методи. Наприклад, в експертній системі *MYSIN*, призначеній для діагностики і вибору методу лікування інфекційних захворювань, розроблений метод виведення з використанням коефіцієнтів впевненості. Широке застосування на практиці отримали нечіткі висновки, що будуються на базі нечіткої логіки, яка веде своє походження від теорії нечітких множин.

В системах з ненадійними знаннями на І–АБО–графі з'являється ще один тип зв'язку КОМБ – комбінований зв'язок. Фрагмент структури такого графа показаний на рис. 7.4. Якщо ненадійні знання об'єднані зв'язкою ТА, то ступінь надійності висновку традиційно вибирається як мінімальне значення, при об'єднанні зв'язкою АБО – як максимальне значення ступеня надійності.

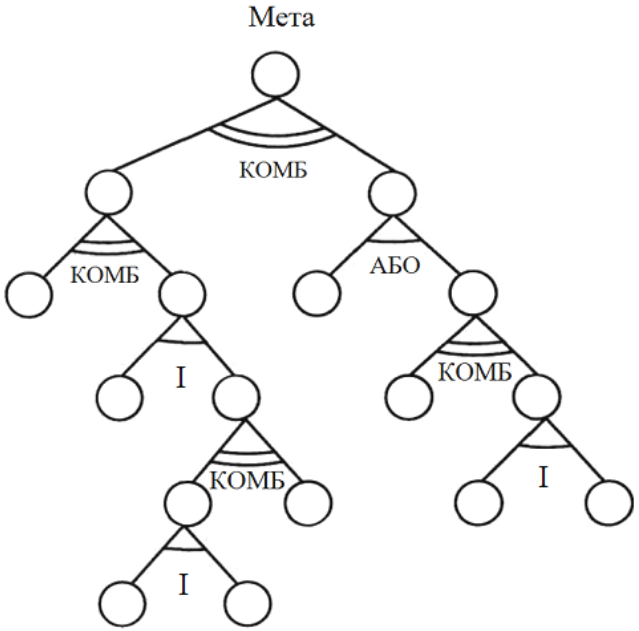


Рис. 7.4. Фрагмент структури І–АБО–графа для системи з ненадійними знаннями

Зв'язок КОМБ означає, що висновок, заснований на фактах, що об'єднуються цим видом зв'язку, буде отримано з оцінкою достовірності, що обчислюється тим чи іншим способом.

Одним з видів оцінки достовірності знань є коефіцієнти впевненості, які використовуються в експертній системі *MYCIN*, які можуть приймати значення, що належать відрізку [–1,1], при цьому 1 відповідає істинному, а (–1) – неправдивому твердженню.

Незважаючи на відсутність суворого теоретичного обґрунтування, коефіцієнти впевненості знаходять широке застосування в експертних системах продукційного типу завдяки простоті сприйняття та інтерпретації отриманих результатів, які непогано узгоджуються з реальністю.

Імовірнісний метод оцінки надійності знань отримав в інженерії знань широкий розвиток. Розглянемо один з таких підходів, *суб'єктивний байєсівський метод*. В даному підході зв'язки між елементами знань не поділяються на типи; замість цього кожному елементарному фрагменту знань (факту, представленому парою *атрибут – значення* або ствердженням) ставиться у відповідність мінімальне або максимальне значення байєсівської ймовірності. Після чого ступені надійності виведених висновків розраховуються як апостеріорні (умовні) ймовірності за формулами, отриманими на базі формули Байєса.

Суб'єктивний байєсівський підхід набув широкого поширення завдяки своїй простоті. До того ж він досить обгрунтований теоретично. Однак цей підхід, як і попередній, має невирішені проблеми: наприклад, сума ймовірностей подій, що спростовують одна одну, може виявитися більше 1. Складною задачею для експертів є призначення апріорних ймовірностей умовних подій. Теоретичні дослідження в даному напрямку активно тривають, і у розпорядженні проектувальників експертних систем вже є такі потужні засоби, як імовірнісна логіка, нечітка логіка, теорія Демпстера-Шафера і т.п.

*Неповнота знань і немонотонна логіка*

Абсолютно повних знань не буває, оскільки процес пізнання нескінченний. У зв'язку з цим стан бази знань має змінюватися з плином часу. На відміну від простого додавання інформації, як в базах даних, при додаванні нових знань виникає небезпека отримання суперечливих висновків, тобто висновки, отримані з

використанням нових знань, можуть спростовувати ті, що були отримані раніше. Ще гірше, якщо нові знання будуть перебувати в протиріччі зі «старими», тоді механізм виведення може стати непрацездатним. Багато експертних систем першого покоління були засновані на моделі закритого світу, обумовленої застосуванням апарату формальної логіки для обробки знань. *Модель закритого світу* передбачає жорсткий відбір знань, що включаються в базу, а саме: БЗ заповнюється виключно вірними поняттями, а все, що ненадійно або невизначено, свідомо вважається хибним. Іншими словами, все, що відомо базі знань, є істиною, а решта – неправдою. Така модель має обмежені можливості подання знань і таїть в собі небезпеку отримання протиріч при додаванні нової інформації. Проте, ця модель досить поширена; наприклад, на ній базується мова PROLOG. Недоліки моделі закритого світу пов'язані з тим, що формальна логіка виходить з передумови, згідно з якою набір визначених у системі аксіом (знань) є *повним* (теорія є повною, якщо кожен її факт можна довести, виходячи з аксіом цієї теорії). Для повного набору знань справедливість раніше отриманих висновків не порушується з додаванням нових фактів. Ця властивість логічних висновків називається *монотонністю*. На жаль, реальні знання, які закладаються в експертні системи, вкрай рідко бувають повними.

Розглянемо простий приклад. Припустимо, в БЗ містяться такі твердження:

*«Птахи літають». «Пінгвін не літає». «Лоло – птах».*

На основі цих знань можна отримати заключення *«Лоло літає»* і зробити висновок про те, що *«Пінгвін не є птахом»*.

Якщо в БЗ додати факт *«Лоло – пінгвін»*, то отримаємо суперечні попереднім висновки: *«Лоло не літає»* і *«Лоло не є птахом»*.

У якості засобів формальної обробки неповних знань, для яких необхідні *немонотонні виведення*, розробляються методи немонотонної логіки: немонотонна логіка Макдермотта і Доула, в якій вводяться умовні логічні операції, логіка умовчання Рейтера, немонотонна логіка Маккарті і т.п. Багато з цих теорій ще не повністю відпрацьовані, але запропоновані в них елементи вже знайшли застосування у практичних розробках (перевірка і облік несуперечності елементів знань, встановлення значень за

замовчуванням у фреймових системах і т.п.). Так, приклад з пінгвіном не викликає протиріч при використанні фреймового подання знань (рис. 7.5).

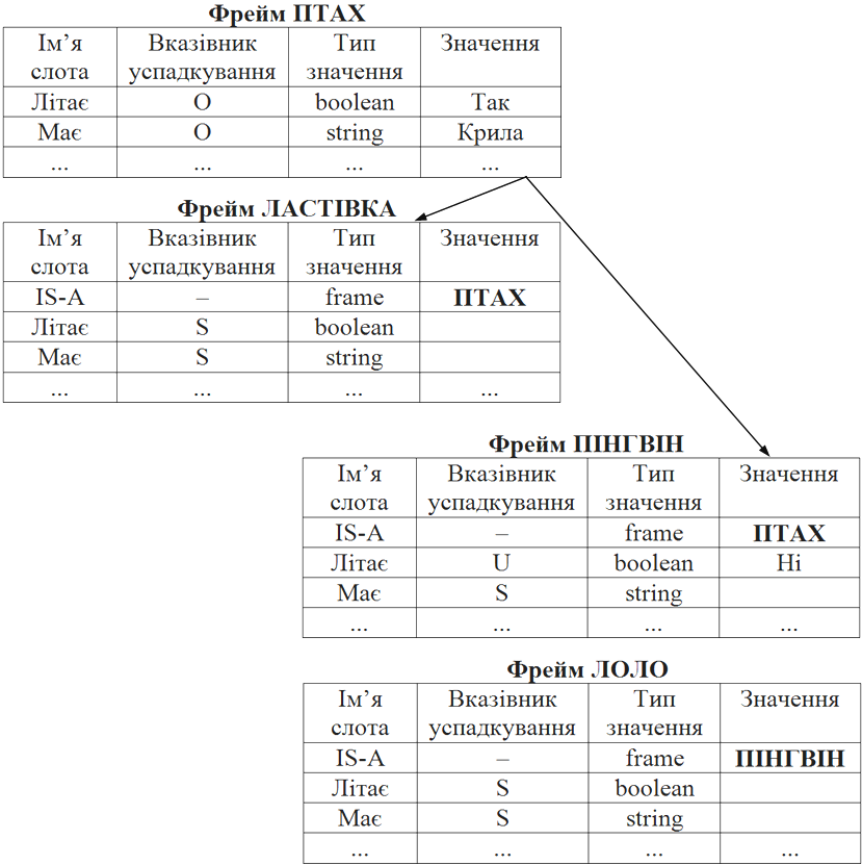


Рис. 7.5. Установка значень за замовчуванням в системах фреймів

Для організації логічних висновків в інтелектуальних системах з неповними знаннями замість традиційної *дедукції* застосовується *абдукція*. *Абдукцією* називається процес формування пояснюючої гіпотези на основі заданої теорії і наявних спостережень (фактів). Розглянемо найпростіший приклад абдуктивного виведення. Припустимо, теорія містить правило:

«ЯКЩО студент відмінно знає математику, ТО він може стати гарним інженером» і факт: «Студент Іванчук відмінно знає математику». Крім того, є спостереження: «Студент Іванчук став гарним спеціалістом-економістом», яке не виводиться із заданої теорії. Для того щоб його вивести, необхідно сформувати абдуктивну (пояснюючу) гіпотезу, яка не суперечитиме вищенаведеній теорії. Такою гіпотезою може бути, наприклад, наступна: «Гарний математик може стати гарним спеціалістом-економістом».

Абдуктивні виведення використовуються в задачах діагностики для виявлення причин спостережуваної неправильної поведінки систем, в задачах, пов'язаних з розумінням природної мови, для вирішення проблем накопичення та засвоєння знань і т.д.

Для роботи з неповними знаннями призначена також *система підтримки значень істинності*, в якій усі знання діляться на достовірні і недостовірні, при цьому передбачається систематичне упорядкування БЗ з метою усунення недостовірних знань. Достовірні на даний момент знання відносять до класу IN, а сумнівні і недостовірні – до класу OUT. Якщо при додаванні нових знань виникає протиріччя, то виконується перевірка класів знань, при цьому можливі міграції з класу в клас.

Дослідження в області немонотонних виведень – це спроби розширити межі формальної логіки, в які не вписуються реальні знання, необхідні інтелектуальним системам.

*Неточність знань*

Відомо, що кількісні дані (знання) можуть бути неточними, при цьому існують кількісні оцінки такої неточності (довірчий інтервал, рівень значущості, ступінь адекватності і т.д.). Лінгвістичні знання також можуть бути неточними. Для обліку неточності лінгвістичних знань використовується теорія нечітких множин, запропонована Л. Заде у 1965 р. Цьому вченому належать слова: «Фактично нечіткість може бути ключем до розуміння здатності людини справлятися з задачами, які занадто складні для вирішення на ЕОМ». Розвиток досліджень в області нечіткої математики призвело до появи нечіткої логіки і нечітких виведень, які виконуються з використанням знань, представлених нечіткими множинами, нечіткими відносинами, нечіткими відповідностями і т. д.

# ЛЕКЦІЯ 8. МЕТОДИ ВИЛУЧЕННЯ І ПРИДБАННЯ ЗНАНЬ

Проблема придбання знань включає дві основні задачі: отримання інформації та її систематизацію. Процес отримання знань від експертів є ключовим при розробці інтелектуальних інформаційних систем, які постійно потребують нові знання, тому на стадії розробки необхідно розглядати проблеми їх перманентного навчання. З розвитком засобів інформатизації з'явилися нові задачі, пов'язані з управлінням знаннями в ІІС.

Придбанням знань називається виявлення знань з джерел і перетворення їх в потрібну форму, а також перенесення в базу знань ІС. Джерелами знань можуть бути книги, архівні документи, вміст інших баз знань і т. п., тобто деякі об'єктивізовані знання, переведені в форму, яка робить їх доступними для споживача. Іншим типом знань є експертні знання, які є у фахівців, але не зафіксовані в зовнішніх по відношенню до нього сховищах. Експертні знання є суб'єктивними. Ще одним видом суб'єктивних знань є емпіричні знання. Такі знання можуть видобуватися ІС шляхом спостереження за навколишнім середовищем (якщо у ІС є засоби спостереження).

Введення в базу знань об'єктивізованих знань не представляє особливої проблеми, виявлення і введення суб'єктивних та особливо експертних знань досить важкі. Щоб розробити методологію придбання суб'єктивних знань, одержуваних від експерта, треба чітко розрізняти дві форми репрезентації знань. Одна форма пов'язана з тим, як і в яких моделях зберігаються ці знання у людини-експерта. При цьому експерт не завжди усвідомлює повністю, як репрезентовані у нього знання. Інша форма пов'язана з тим, як інженер по знанням, який проєктує ІС, збирається їх описувати і представляти. Від ступеня узгодженості цих двох форм репрезентації між собою залежить ефективність роботи інженера по знанням.

*Вилученням* знань називають процес отримання знань від експертів. Вилучення знань – складна і трудомістка процедура, в результаті якої інженеру по знанням (когнітологу, аналітику) необхідно створити власну модель предметної області на основі інформації, отриманої від експертів. Спроби отримати знання, необхідні для розробки ІІС, безпосередньо від експертів і

обійтися без когнітологів зазвичай не призводять до успіху, так як в цьому випадку пред'являються дуже високі вимоги до експерта, який, будучи фахівцем в предметній області, буде змушений набути кваліфікацію інженера по знанням. Крім того, існує ще декілька причин, що викликають необхідність участі аналітиків в процесах вилучення знань, а саме:

− кращим способом для вербалізації знань експерта є діалог;

− досвідчений аналітик, використовуючи сучасну методологію системного аналізу, може допомогти експерту в структуризації знань предметної області;

− інженер по знанням допомагає експерту усвідомити «приховані» знання, пропонуючи йому встановити причинно-наслідкові зв'язки (а також зв'язки іншої природи) на множині виділених понять.

Успіх на етапі вилучення знань багато в чому залежить від кваліфікації аналітика, який повинен мати освіту, що включає знання з різних областей, в тому числі з когнітивної психології, системного аналізу, математичної логіки, штучного інтелекту і т.д.

У когнітивній психології вивчаються форми репрезентації знань (когнітивні структури знань) характерні для людини. Прикладами можуть служити: представлення класу понять через його елементи (наприклад, поняття "*птах*" репрезентується рядом *чайка, горобець, шпак, ...*), представлення понять класу за допомогою базового прототипу, що відображає найбільш типові властивості об'єктів класу (наприклад, поняття "*птах*" репрезентується прототипом *дещо з крилами, дзьобом, літає, ...*), представлення за допомогою ознак (для поняття "*птах*", наприклад, *наявність крил, дзьоба, двох лап, пір'я*).

Крім понять репрезентуються і відносини між ними. Як правило, відношення між поняттями визначаються процедурним способом, а відносини між складовими понять (визначаючих структуру поняття) – декларативним способом. Наявність двох видів описів змушує в моделях подання знань одночасно мати обидва компонента, наприклад семантичну мережу і продукційну систему, як це представлено в когнітивній моделі.

Як правило, кожен когнітолог сам винаходить мову для опису отриманих від експерта знань шляхом поповнення сформованої мови конкретної науки спеціальними термінами і знаками.

Стандарту таких мов поки не існує. Проте бажано, щоб такі мови були зрозумілими і містили якомога менше неточностей. Розробка мов інженерії знань ведеться в різних напрямках, зокрема відомі мови-класифікації, логіко-конструктивні мови, структурно-логічні та ін. Перспективний підхід до створення подібних мов відкриває *семіотика* – наука про знакові системи. Класична семіотика є чисто гуманітарною наукою, основні інтереси якої зосереджені в області культури людської поведінки, мистецтва і мови. Область досліджень *прикладної семіотики* пов'язана із застосуванням знакових систем для представлення та обробки знань в практичних додатках ШІ.

При придбанні знань важливу роль відіграє так зване поле знань в якому містяться основні поняття, що використовуються при описі предметної області, і властивості всіх відношень, що використовуються для встановлення зв'язків між поняттями. Поле знань пов'язано з концептуальною моделлю проблемної області, в якій ще не враховані обмеження, які неминуче виникають при формальному поданні знань в базі знань. Перехід від опису деякої області в поле знань до опису в базі знань аналогічний переходу від концептуальної моделі бази даних до її логічної схеми, коли вже зафіксована система управління базою даних. Важливо відзначити, що перехід безпосередньо до формального подання в базі знань без етапу концептуального опису в поле знань призводить до численних помилок, що уповільнює процес формування БЗ ІС.

Процеси вилучення знань розглядають в трьох основних аспектах: психологічному, лінгвістичному і гносеологічному (див. рис. 8.1).

*Психологічний аспект.* Це найважливіший з усіх аспектів, так як вилучення знань відбувається в процесі *спілкування* когнітологів з експертами, де психологія грає домінуючу роль.

Процес вилучення знань для інтелектуальних систем необхідно організувати не як односпрямований процес передачі повідомлень від експерта аналітику, а як спільний пошук істини.

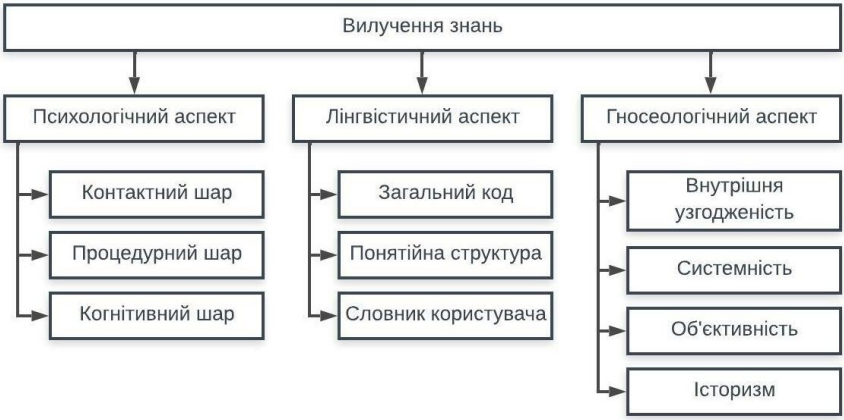


Рис. 8.1. Аспекти вилучення знань

В процесі розмовного спілкування багато інформації втрачається, тому важливою проблемою є збільшення інформативності спілкування аналітика та експерта за рахунок використання методик, вироблених в психології (див. рис. 8.2).

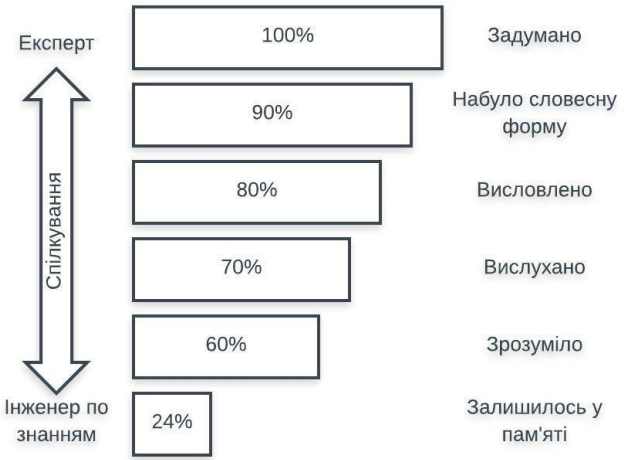


Рис. 8.2. Втрати інформації при розмовному спілкуванні

Модель спілкування включає учасників спілкування, засоби спілкування і предмет спілкування (знання). Відповідно до цих компонентів виділяються три шари психологічних проблем: контактний, процедурний, когнітивний.

Ступінь інформативності спілкування аналітика та експерта на рівні *контактного шару* залежить в основному від статі, віку, темпераменту особистості і мотивації учасників спілкування. Встановлено, що кращі результати дають гетерогенні пари (чоловік/жінка) і співвідношення вікових груп:

$$5 < (B_E - B_A) < 20$$

де $B_E$ – вік експерта;

$B_A$ – вік аналітика.

Бажано, щоб учасники процесу спілкування володіли такими здібностями, як доброзичливість, хороша пам'ять, увага, спостережливість, уява, вразливість, зібраність, наполегливість і винахідливість.

В рамках контактного шару найбільш переважними з чотирьох класичних типів темпераменту є сангвініки і холерики.

Параметри *процедурного шару* забезпечують ефективність вилучення знань. До цих параметрів відносяться: ситуація спілкування (місце, час, тривалість); обладнання (допоміжні засоби, освітленість, меблі); професійні прийоми (темп, стиль, методи та ін.). Для підвищення ефективності процесу вилучення знань інженер по знанням повинен підібрати значущі для експерта стимули, оскільки останній передає аналітику один з найцінніших ресурсів – знання.

*Когнітивний шар* зв'язаний з вивченням семантичного простору пам'яті експерта та з відтворенням його понятійної структури і моделі міркувань. Когнітивний шар характеризується когнітивним стилем і семантичною репрезентативністю.

Під *когнітивним стилем* людини розуміється специфічна сукупність критеріїв, використовуваних ним в процесі пізнання світу для вирішення різних задач. Когнітивний стиль – це система засобів та індивідуальних прийомів, до яких вдається людина для організації своєї діяльності, що забезпечує досягнення бажаних результатів. Для підвищення ефективності вилучення знань доцільно підбирати експертів і аналітиків, що володіють цілком

певними характеристиками когнітивного стилю. Найбільш важливими з них є наступні: поленезалежність (незалежність від шумового поля), імпульсивність – рефлексивність, жорсткість – гнучкість, когнітивна еквівалентність.

Бажано, щоб і аналітики, і експерти мали такі когнітивні характеристики:

- висока *поленезалежність,* яка має на увазі здатність виділяти головні аспекти даної проблеми і відкидати все зайве, що не відноситься до поставленої задачі. Цю якість бажано мати і аналітику, і експерту, однак слід враховувати той факт, що полезалежні люди більш контактні і схильні до спілкування;
- *рефлексивність*, що характеризує схильність до розсудливості і самоаналізу (в той час як імпульсивність характеризується швидким, часто недостатньо обґрунтованим прийняттям рішень);
- *когнітивна еквівалентність*, що визначає здатність людини до розрізнення понять і розбиття їх на класи та підкласи;
- експерти – стійкість уявлень, тобто *жорсткість* структури сприйняття, а аналітики – *гнучкість*, тобто уміння легко пристосовуватися до нової обстановки.

Для ефективної побудови ІІС інженер по знанням повинен володіти спеціальними неформальними методами і математичним апаратом, що дозволяє йому відтворювати отримані від експерта знання за допомогою різних моделей, наприклад, таких, як продукційна або фреймова. Не нав’язуючи експерту будь-якої моделі, аналітик повинен підібрати засоби подання знань, що мають максимально високу семантичну репрезентативність.

*Лінгвістичний аспект.* Актуальність дослідження цього аспекту визначається тим, що мова є основним засобом спілкування в процесі вилучення знань. В області лінгвістичних проблем найбільш важливими є поняття: загальний код, понятійна структура, словник користувача.

*Спільним кодом* називають спеціальну проміжну мову спілкування між експертом та інженером по знанням. Ця мова включає сукупність загальнонаукових і спеціальних понять з професійної літератури, елементів базової мови, неологізмів і т.п. Спільний код дозволяє подолати мовний бар’єр в процесі спілкування когнітологів з експертами. Вироблення спільного коду

для партнерів здійснюється відповідно до інформаційних потоків, представлених на рис. 8.3.

Надалі загальний (спільний) код перетворюється в *понятійну структуру*, або семантичну мережу, яка пов'язує поняття, що зберігаються в пам'яті людини. Виявлення відношень між елементами знань, представлених поняттями, є однією з найскладніших проблем в процесах вилучення знань.

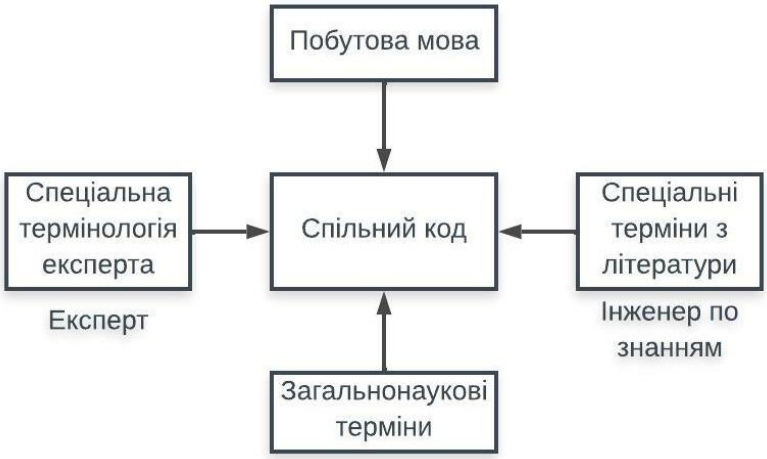


Рис 8.3. Структура формування загального коду

Добре відомо, що природні знання людини це пов'язані структури, а не розрізнені фрагменти. Однак до теперішнього часу при розробці БЗ враховується вельми обмежений набір зв'язків між поняттями, в той час як в дійсності існує велика різноманітність таких відношень. Поспєлов Д. О. виділяє понад 200 базових видів відношень між поняттями. Таке різноманіття унеможливлює однозначне визначення набору ознак, що описують конкретне поняття, і, як наслідок, однозначну класифікацію понять. Очевидно, складність даної проблеми є головною причиною того, що на сьогоднішній день відсутні надійні методики формування понятійних структур.

Проте побудова тієї чи іншої ієрархії понять входить в задачі концептуального аналізу структури знань будь-якої предметної

області. Останнім часом в ІІ став широко вживатися термін *онтологія*, що має багатозначну інтерпретацію.

1. Онтологія як філософська дисципліна являє собою систему категорій, які є наслідком певного погляду на світ.

2. Онтологія як неформальна система концептуалізації знань передбачає побудову опису множини виділених об'єктів, понять, зв'язків і відношень в заданій області знань. При цьому можуть використовуватися формальні або неформальні засоби. У найпростішому випадку онтологія може описувати тільки ієрархію понять, пов'язаних відношеннями «елемент-клас». Крім цього вона може містити набір аксіом і правил виведення, що дозволяють висловити інші відношення між поняттями та обмежити область інтерпретації понять.

3. Онтологія як представлення концептуальної системи у вигляді логічної теорії означає використання певного синтаксису для подання знань.

Розробка *словника користувача* необхідна в зв'язку з тим, що кінцевий користувач не зобов'язаний володіти професійною мовою предметної області, який використовувався при побудові БЗ. Інтерфейс користувача створюється, як правило, шляхом доопрацювання словника спільного коду.

*Гносеологічний аспект.* Він об'єднує методологічні проблеми отримання нового наукового знання, так як процес пізнання часто супроводжується появою нових понять і теорій. В процесі розробки БЗ експерти досить часто вперше формулюють деякі закономірності на основі накопиченого емпіричного досвіду. Послідовність «*факт→узагальнений факт→емпіричний закон→теоретичний закон»* називається гносеологічним ланцюжком. Теорія – це не тільки система узагальнення накопичених знань, але й спосіб отримання нового знання.

Основними критеріями якості нових знань є внутрішня узгодженість, системність, об'єктивність, історизм.

У процесі вилучення знань аналітиків насамперед цікавлять емпіричні знання експертів, що представляють собою результати спостережень, які можуть виявитися неузгодженими. *Внутрішня узгодженість* емпіричних знань характеризується поняттями модальності, суперечливості, неповноти. Під модальністю знання розуміється можливість його існування в різних категоріях.

Суперечливість є природною властивістю емпіричних знань і не завжди може і повинна бути усунена. Навпаки, суперечливість може служити відправною точкою в міркуваннях експертів. Неповнота знань пов'язана з неможливістю вичерпного опису будь-якої предметної області.

*Системність* знань заснована на визначенні місця нових знань в багаторівневій ієрархічній організації. При цьому необхідно знайти відповіді на питання: які поняття деталізують або узагальнюють нові знання і в яких відношеннях вони є з відомими фактами і закономірностями?

*Об'єктивність* знань визначити практично неможливо. Процеси накопичення, опису, представлення, обробки, інтерпретації та оцінювання якості знань виконуються конкретними людьми, тому їх результати мають суб'єктивний характер. Об'єктивність деяких закономірностей часто пов'язують з широтою області їх застосовності. Межі цієї області можна встановити експериментальним або теоретичним шляхом, але далеко не завжди. У якості непрямих свідчень об'єктивності іноді допускають збіг представлень різних експертів і підтвердження висунутих гіпотез відомими фактами.

*Історизм* знань пов'язаний з розвитком і зміною представлень про предметну область з плином часу.

Процес пізнання можна представити наступними етапами:

- опис та узагальнення фактів;
- виявлення зв'язків між фактами, формулювання правил і закономірностей;
- побудова моделі знань предметної області;
- пояснення і прогнозування явищ на основі моделі.

На початкових етапах інженер по знанням, досліджуючи структуру висновків експерта, може використовувати різні теорії і підходи для побудови формальної моделі знань предметної області. Найбільш відомими і часто застосованими прийомами є математична логіка, теорія асоціацій і гештальт-психологія.

*Математична логіка* формує критерії, які гарантують точність, значимість і несуперечливість загальних понять, міркувань і висновків. Застосовуючи логічний підхід, когнітолог виконує наступні операції: визначення понять, виявлення подібності та відмінності, аналіз, абстрагування, узагальнення,

класифікацію, утворення суджень, складання силогізмів і т.д. Проблема застосування логічного підходу до подання знань полягає в тому, що людина не завжди мислить категоріями суворої класичної логіки, а природна система знань не є повною, допускає протиріччя і багатозначні оцінки істинності.

В *теорії асоціацій* мислення представляється у вигляді ланцюжка ідей, пов'язаних загальними поняттями. Тут застосовуються такі прийоми:

- асоціації, придбані на основі зв'язків різної природи;
- залучення минулого досвіду;
- метод проб і помилок з випадковим успіхом;
- звичні («автоматичні») реакції та ін.

*Гештальт-психологія* орієнтує аналітика на виділення цілісного образу або структури знань (*гештальта*) як основи для розуміння процесів і явищ навколишнього світу. Поняття гештальта багато в чому узгоджується з поняттям фрейму. Застосування даної теорії орієнтує експерта на формування моделі знань відповідно до критеріїв простоти, зв'язності і гармонії.

Ідеалізована модель знань предметної області будується на основі встановлених логічних зв'язків між поняттями. Модель формалізується за допомогою категориального апарату, формально-знакових засобів математики і логіки. Для адекватного відображення в моделі реальної картини світу інженеру по знанням необхідно володіти такими прийомами, як ідеалізація, абстрагування, огрубіння. Критерієм якості побудованої моделі є здатність ІІС робити прогнози і пояснювати безліч явищ із заданою предметною областю. Інженер по знанням повинен прагнути, щоб результуюча модель знань була достатньо повною, зв'язковою і несуперечливою.

Для структурування знань використовуються структурний та об'єктний підходи. *Структурний* підхід заснований на ідеї алгоритмічної декомпозиції, де кожен модуль системи виконує один з важливих етапів загального процесу. В рамках структурного підходу розроблено велику кількість виразних засобів: діаграми потоків даних (DFD), структуровані словники (тезауруси), мови специфікацій систем, таблиці рішень, стрілочні діаграми, діаграми переходів, дерева цілей, засоби управління проектом (PERT-діаграми) та ін.

*Об'єктний (об'єктно-орієнтований)* підхід пов'язаний з об'єктною декомпозицією, при якій кожен об'єкт розглядається як екземпляр певного класу. До базових понять цього підходу відносяться наступні:

1. *Абстрагування*, яке М. Шоу визначив як спрощений опис системи, де виділяються найбільш суттєві для розгляду властивості і деталі, а незначні аспекти опускаються. Абстрактне уявлення реальності відображено моделлю сутності (об'єкт) і моделлю поведінки (метод). Об'єкти відповідають поняттям предметної області. Методи являють собою операції, які можна виконувати над об'єктами.

2. *Клас* – множина об'єктів, пов'язаних спільністю структури і властивостей.

3. *Ієрархія* – упорядкована система абстракцій (класів).

4. *Спадкування* – таке співвідношення між класами, коли один клас використовує структурну або функціональну частину іншого класу (або декількох інших).

5. *Типізація* – обмеження, що накладається на клас, яке перешкоджає взаємозамінності об'єктів, що належать різним класам.

6. *Модульність* – властивість системи, пов'язана з можливістю її декомпозиції на ряд взаємопов'язаних частин (модулів).

7. *Інкапсуляція* – обмеження доступу до внутрішньої структури і механізмів функціонування об'єкта.

8. *Поліморфізм* – можливість наділення об'єкта різними властивостями і стратегіями поведінки. Іншими словами, одне ім'я може відповідати різним класам об'єктів, що входять в один суперклас. Отже, об'єкт, позначений цим ім'ям, може по-різному реагувати на деяку множину дій.

На цей час продовжує розвиватися об'єктно-структурний підхід до структурування знань на основі узагальнення існуючих підходів. Основна ідея об'єктно-структурного підходу пов'язана з проведенням послідовного об'єктно-структурного аналізу інформації про розглянуту предметну область, для подання якої використовується стратифікована модель (див. табл. 8.1). В процесі об'єктно-структурного аналізу відбуваються виділення і

структуризація понять із застосуванням різноманітних методів аналізу знань.

Таблиця 8.1.

**Стратифікація знань предметної області**

| Рівень страти | Категорія знань | Вид аналізу знань |
|---|---|---|
| 1 | НАВІЩО | Стратегічний аналіз: призначення і функції системи |
| 2 | ХТО | Організаційний аналіз: колектив розробників системи |
| 3 | ЩО | Концептуальний аналіз: основні принципи, понятійна структура |
| 4 | ЯК | Функціональний аналіз: гіпотези і моделі прийняття рішень |
| 5 | ДЕ | Просторовий аналіз: оточення, обладнання, комунікації |
| 6 | КОЛИ | Часовий аналіз: часові параметри і обмеження |
| 7 | ЧОМУ | Причинно-наслідковий (каузальний) аналіз: формування підсистеми пояснень. |
| 8 | СКІЛЬКИ | Економічний аналіз: ресурси, витрати, прибуток, окупність |

На стадії структурування знань предметної області необхідно вирішити наступні задачі: складання словника використовуваних термінів; виявлення понять та їх атрибутів; виявлення зв'язків і визначення відношень між поняттями; деталізація і узагальнення понять; побудова узагальненої структури знань предметної області.

Першим кроком структурування знань є визначення вхідних і вихідних даних, які в подальшому будуть деталізуватися і уточнюватися. На підставі цих даних, а також за результатами аналізу протоколів сеансів вилучення знань складається набір ключових слів (термінів), в процесі обробки якого виявляються об'єкти, поняття і їх атрибути. Під *поняттям* мається на увазі узагальнення предметів деякого класу за специфічними ознаками. Формування понять – серйозна проблема. Для виявлення понять використовуються традиційні методи розпізнавання образів і

класифікації, а також нетрадиційні методи, що базуються на методології інженерії знань. Найбільш поширеними методиками виявлення об'єктів і понять є:

- методика формування переліку понять;
- інтерв'ювання фахівців;
- складання списку елементарних дій;
- складання змісту підручника.

Практичне використання цих методик показало, що найбільш результативними з них є методики інтерв'ювання та складання змісту підручника.

Існує думка, що теорія понять є тільки в одній науці – математиці, де можливе їх строге визначення. У гуманітарних науках визначення понять найчастіше відсутні, отже, там замість понять доводиться мати справу з ідеями. Для строго певних понять існують мови, здатні висловити зв'язки між ними, але для опису зв'язків між ідеями таких мов поки немає.

Відношення між поняттями можуть мати різну природу: «ціле-частина», «причина-наслідок», домінування, часові і просторові відношення, ситуативні, асоціативні, функціональні та ін. Виявлення зв'язків і відношень між поняттями є складною задачею, для вирішення якої використовуються різноманітні засоби. Багато інженерів по знанням самі винаходять різні методи і прийоми в процесі роботи з експертом. До таких методів можна віднести «сортування карток» і побудова замкнутих кривих. Один із сучасних підходів до подання взаємопов'язаних структур знань заснований на використанні *сценаріїв*, які будуються за аналогією з організацією людської пам'яті, де усі знання об'єднані зв'язками різних типів. Сценарії складаються з фрагментів (сцен), пов'язаних просторовими або часовими відношеннями. Елементи знань з фрагментів можуть бути пов'язані відношеннями різної природи: функціональними, асоціативними, ситуативними, причинно-наслідковими та ін.

Процедури узагальнення та деталізації понять погано формалізуємі та потребують від експертів і аналітиків високої кваліфікації. Ієрархічне подання знань вимагає встановлення відношень між поняттями всередині кожного рівня ієрархії і між ними. Заключним етапом структурування знань є аналіз можливих ланцюжків міркування і вироблення правил прийняття рішень, які

дозволяють об’єднати сформовані поняття і відношення в динамічну модель знань предметної області.

Послідовність структурування знань залежить від особливостей конкретної області і від напрацьованих для даної категорії знань предметно-орієнтованих методів структурування.

Семіотичний підхід до моделювання людських знань вважається на цей час одним з найперспективніших. На рис. 8.4 показана схема, відома в семіотиці як трикутник Фреге. Об’єкти реального світу, звані *денотатами*, відображаються у свідомості людини (ментальному світі), і в результаті цього відображення виникають уявлення про денотат.


Рис. 8.4. Трикутник Фреге

Уявлення – це інтегрований образ денотату (званий в психології гештальтом), отриманий на основі відчуттів та інших джерел інформації.

Для розрізнення денотатів людина використовує імена, пов’язані з уявленнями про сутності, і формує відповідні поняття, використовуючи процедури виявлення подібності та відмінності уявлень.

Між ментальними об’єктами (уявлення, ім’я, поняття), що утворюють трикутник Фреге, існують зв’язки, що відповідають певним механізмам мислення. Зв’язок 1 дозволяє по імені сутності активізувати в пам’яті всі відомості про її властивості (поняття). Цей же зв’язок в

зворотному напрямку дозволяє за описом сутності визначити її ім'я («*Не гавкає, не кусає, а в будинок не пускає. Що це?*»). Зв'язок 2 дозволяє за уявленням про сутність знайти інформацію про її властивості або сформувати уявлення про сутність за сукупністю властивостей. Зв'язок 3 з'єднує уявлення про денотат з його ім'ям, тобто ім'я може активізувати уявлення і навпаки.

Інформаційна структура, відповідна трикутнику Фреге, в якому вершини ототожнюються з ім'ям, поняттям і уявленням, називається *знаком* або *семою*.

Між знаками існують різного роду відношення. Особливе значення в придбанні знань мають відношення спадкування, до яких відносяться наступні типи: «елемент – клас», «частина – ціле», «вид – рід». Важливою властивістю відношень «елемент – клас» $(x \in X)$ є те, що між елементами класу може не бути ніякої схожості, крім їх приналежності до одного класу. Визначення класу в подібних випадках можливе шляхом перелічення імен елементів, що до нього входять. У відношеннях «частина – ціле» $(y \subset X)$ успадковується частина властивостей сутності, якій приписана роль «цілого» ($X$). При цьому частини одного цілого можуть бути абсолютно не схожі одне на одного. Відношення «вид – рід» базуються на спадкуванні всіх властивостей роду $X$ видом $z$ (наприклад, будь-який вид ссавців успадковує ознаку вигодовування дитинчат молоком), тому між видами одного роду обов'язково існує схожість.

Відношення спадкування утворюють ієрархічні структури в системах знаків. На рис. 8.5 показано приклад подібної ієрархії, де перший рівень утворений відношенням типу «елемент – клас», другий – відношенням «вид – рід», а третій – відношенням «частина – ціле». У відношенні «елемент – клас» ієрархічний зв'язок встановлюється між іменами знаків, а у відношеннях «частина – ціле» і «вид – рід» – між поняттями або уявленнями.

Між знаками, що знаходяться на одному ієрархічному рівні, теж можуть існувати зв'язки. Типовим прикладом таких зв'язків є відношення типу «причина – наслідок». Наявність відношень всередині ієрархічних рівнів призводить до утворення мереж, в яких вершини відповідають знакам, а дуги – відношенням між знаками. По суті справи, це і є семантичні мережі, про які говорилося раніше. Крім цієї аналогії слід зазначити схожість

понять знака та фрейму, що породило концепцію *знака-фрейму*, який являє собою трикутник з вершинами:

- *ім'я* (ім'я фрейма = ім'я знака);
- *протофрейм* (поняття, представлене набором слотів «пустого» фрейма-прототипу);
- *екзофрейм* (уявлення, реалізоване конкретним фреймом-екземпляром).

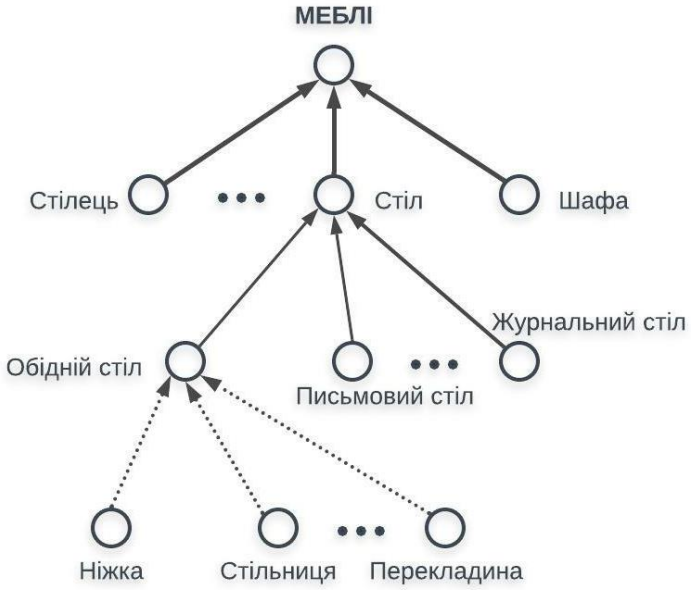


Рис. 8.5. Приклад ієрархії знаків з відношеннями спадкування різних типів

Для дослідження операцій над знаками-фреймами необхідно ввести метарівень, на якому в якості денотатів виступають знаки-фрейми, а ментальними об'єктами є метазнаки, які характеризуються синтаксисом (іменем або іншим способом кодування), семантикою (поняттям, конкретним змістом для суб'єкта) і прагматикою (процедурами, пов'язаними зі знаками). Взаємозв'язок цих аспектів також можна представити трикутником Фреге, в якому існують зв'язки між синтаксисом та семантикою,

що дозволяють отримувати по імені всю інформацію про сутність, і, навпаки, за описом метазнака визначити його синтаксис. Зв'язки між семантикою і прагматикою дозволяють формувати певні дії на основі аналіза ситуації, яка характеризується даним знаком. Зв'язок між цими аспектами в протилежному напрямку можна інтерпретувати як пояснення причин подій (процедур), що відбулися. Зв'язок синтаксису з прагматикою передбачає перехід до дій при згадці імені без аналізу семантики (натиснути на гальмо при червоному сигналі світлофора), а також дозволяє відновити синтаксис (ім'я) ситуації по здійсненим діям.

Синтаксис, семантика і прагматика знака (метазнака) не прив'язані жорстко і однозначно до тієї сутності, яку вони характеризують. Розглянемо приклад, що підтверджує договірний характер використання знаків. Припустимо, в поїздці за грибами учасники домовилися запалити багаття в тому випадку, якщо буде виявлено місце з великою кількістю грибів. Результат цієї домовленості можна розглядати як створення знака, позначенням (синтаксисом) якого є дим від багаття, семантика відповідає факту виявлення грибів, а прагматика (дії) очевидна: якщо хтось не знайшов грибів, потрібно йти до багаття. Однак учасники поїздки можуть прийняти інші угоди. Наприклад, при виявленні грибів не розпалювати багаття, а голосно крикнути. В цьому випадку зміниться тільки синтаксис. Якщо вони домовляться про те, що дим багаття є сигналом настання обіду, то зміниться семантика, а прагматика залишиться колишньою – йти до багаття. Нарешті, якщо прийнято угоду про те, що дим від багаття означає сигнал повернення додому, то зміниться не тільки семантика, але і прагматика, так як в цьому випадку потрібно йти не в бік багаття, а до автобусної зупинки.

Структура семіотичної моделі знань деякої предметної області, або *поля знань*, на якому створюється БЗ інтелектуальної системи, показана на рис. 8.6.

Синтаксис семіотичної моделі поля знань можна описати такими ознаками:

$$P = \{I, O, M\}$$

де $I$ – структура вихідних даних, що підлягають обробці та інтерпретації в ІІС;

*O* – структура вихідних даних;

*M* – модель предметної області, що описує перетворення *I* в *O*, яка може бути представлена двома складовими $M = \{S_k, S_f\}$.

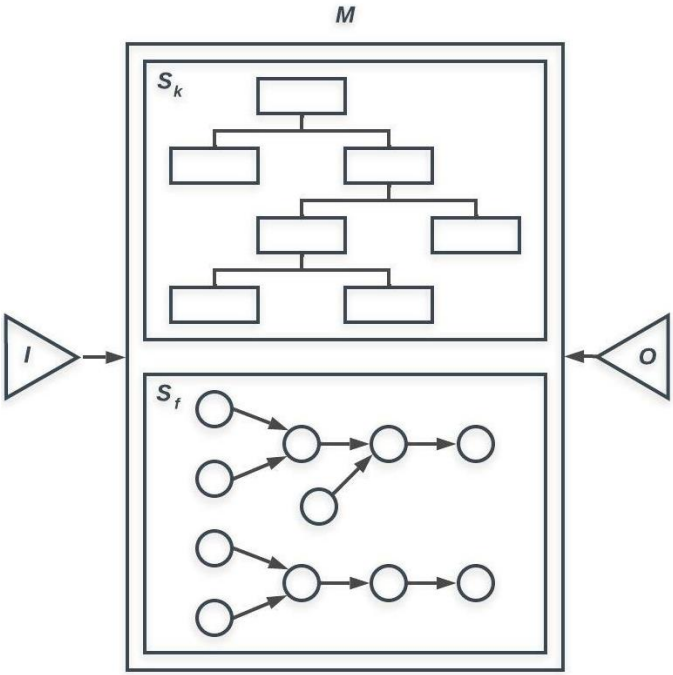


Рис. 8.6. Структура поля знань

Компонента $S_k$ відображає понятійну структуру предметної області у вигляді статичної концептуальної структури, а $S_f$ – динамічну функціональну структуру, що моделює можливі схеми міркувань експерта на основі функціональних зв'язків або відношень між поняттями, що утворюють $S_k$. Семантика моделі визначається конкретним змістом знань проблемної області. Сенс використовуваних в моделі понять і відношень визначається і поступово уточнюється в процесі отримання знань, в якому можна виділити наступні етапи.

1. Побудова первинної моделі, яка відображає уявлення експерта про предметну область. Ця модель є семантичною репрезентацією реальності і особистого досвіду експерта.

2. Експерт вербалізує свій досвід, пояснюючи способи міркувань в процесі передачі знань аналітику. Семантику знань предметної області тепер можна подати за допомогою тексту.

3. Інженер по знанням інтерпретує вербалізований досвід експерта і формує власне уявлення про знання даної предметної області.

4. Формується модель поля знань.

Під прагматичною складовою семіотичної моделі фахівці в області штучного інтелекту розуміють практичні аспекти розробки поля знань, пов'язані зі створенням і застосуванням технологій проведення структурного аналізу, узгодженням окремих фрагментів знань, вирішенням протиріч і т.п. Прагматика в даному випадку визначає вибір технологій, які використовує інженер по знанням для перетворення хаотичного досвіду експерта в побудовану модель поля знань проблемної області.

Різноманіття задач, ситуацій і джерел знань зумовило появу великої кількості методів вилучення, придбання і формування знань. Одна з можливих класифікацій методів вилучення знань наведена на рис. 8.7, на першому рівні якої виділено два великих класи. Перший клас утворюють комунікативні методи, які орієнтовані на безпосередній контакт інженера по знанням з експертом (джерелом знань), другий клас – текстологічні методи, засновані на придбанні знань з документів і спеціальної літератури.

*Комунікативні методи.* Вони поділяються на пасивні і активні. У пасивних методах провідну роль відіграє експерт, в активних – інженер по знанням. При вирішенні конкретних задач, як правило, використовуються як пасивні, так і активні методи. Активні методи діляться на індивідуальні та групові. У групових методах знання отримують від великої кількості експертів, в індивідуальних – від єдиного експерта. Індивідуальні методи отримали більш широке застосування на практиці в порівнянні з груповими.

*Пасивні комунікативні методи* включають спостереження, аналіз протоколів «думок вголос», процедури вилучення знань з лекцій.

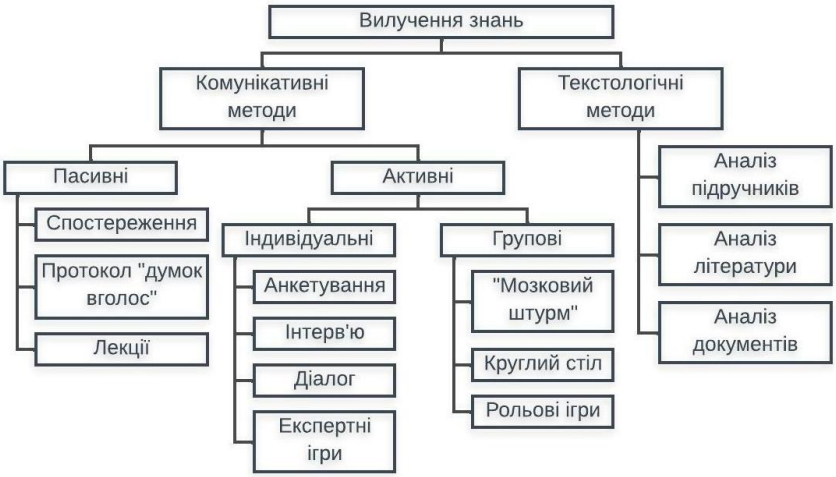


Рис. 8.7. Класифікація методів вилучення знань

*Метод спостереження* є одним з найбільш застосовуваних на початкових етапах розробки експертних систем. Його суть полягає у фіксуванні всіх дій експерта, його реплік і пояснень. При цьому аналітик не втручається в роботу експерта, а тільки спостерігає за процесом вирішення реальних задач або за вирішенням проблем, що імітують реальні задачі. Спостереження за процесом вирішення реальних задач дозволяють інженеру по знанням глибше зрозуміти предметну область. Однак експерт в цьому випадку відчуває велику психологічну напругу, розуміючи, що здійснює не тільки свою професійну діяльність, але і демонструє її інженеру по знанням. Спостереження за імітацією процесу знімає цю напругу, але призводить до зниження повноти і якості видобутих даних. Спостереження за імітацією незамінні в тих випадках, коли спостереження за реальним процесом неможливі через специфіку досліджуваної предметної області.

*Метод аналізу протоколів «думок вголос»* відрізняється від методу спостереження тим, що експерт не тільки коментує свої дії, але і пояснює ланцюжок своїх міркувань, що призводять до вирішення. Основною проблемою, що виникає при використанні цього методу, є принципова складність для будь-якої людини

словесного опису власних думок і дій. Підвищити повноту і якість видобутих знань можна за рахунок багаторазового уточнюючого протоколювання міркувань експерта.

*Метод вилучення знань з лекцій* передбачає, що експерт передає свій досвід інженеру по знанням в формі лекцій. При цьому інженер по знанням може заздалегідь сформулювати теми лекцій. Якщо цього не вдається зробити, то когнітолог конспектує лекції і ставить питання. Якість інформації, наданої експертом в ході лекції, визначається чіткістю сформульованої теми, а також здібностями лектора в структуруванні і викладі своїх знань і міркувань.

Одна з можливих класифікацій людей за психологічними характеристиками ділить всіх на три типи:

- мислитель (пізнавальний тип);
- співрозмовник (емоційно-комунікативний тип);
- практик (практичний тип).

*Мислителі* орієнтовані на інтелектуальну роботу, навчання, теоретичні узагальнення і мають властивості поленезалежності і рефлексивності. *Співрозмовники* – товариські, відкриті люди, готові до співпраці. *Практики* воліють дії розмовам, добре реалізують задуми інших.

Предметні області відрізняються рівнем документованості та структурованості. Для характеристики предметної області за рівнем документованості виділяють три класи: добре документовані, середньодокументовані і слабодокументовані області. За ступенем структурованості знань предметні області можуть бути:

- добре структурованими (з чіткою аксіоматизацією, широким застосуванням математичного апарату, усталеною термінологією);
- середньоструктурованними (з визначеною термінологією, розвинутою теорією, явними взаємозв'язками між явищами);
- слабоструктурованими (з розмитими визначеннями, багатим емпіричним матеріалом, прихованими взаємозв'язками).

*Активні індивідуальні методи* включають методи анкетування, інтерв'ювання, вільного діалогу і гри з експертом.

Перевагою *методів анкетування* є те, що анкета або опитувальник складаються інженером по знанням заздалегідь і використовуються для опитування експертів. Складання анкети слід

проводити з урахуванням рекомендацій, вироблених в соціології та психології. Основними вимогами до анкет є наступні:

1. Анкета не повинна бути монотонною і одноманітною, щоб не викликати нудьгу або втому. Для цього необхідно урізноманітнити тематику і форму завдання питань, включити питання-жарти і застосувати стиль гри.

2. Анкета повинна бути пристосована до мови експерта.

3. Слід враховувати, що питання впливають одне на одного, з цього важливо розташувати їх у правильній послідовності.

4. В анкеті повинно міститися оптимальне число надлишкових питань, частина яких призначена для контролю правильності відповідей, а інша частина – для зняття напруги.

*Метод інтерв'ювання* відрізняється від методу анкетування тим, що дозволяє аналітику опускати ряд питань у залежності від ситуації, вставляти нові питання в анкету, змінювати теми і урізноманітнити ситуацію спілкування. Важливу роль в методі інтерв'ювання відіграють питання, класифікація яких показана на рис. 8.8.

*Відкрите* питання називає тему або предмет, залишаючи експерту повну свободу щодо форми і змісту відповіді. *Закрите* питання пропонує експерту вибрати відповідь із запропонованого набору.

*Особисте* питання безпосередньо стосується особистого досвіду експерта. *Безособове* питання спрямоване на виявлення найбільш поширених закономірностей предметної області.

*Пряме* питання безпосередньо вказує на конкретний предмет або тему (використовується при «закритості» експерта). *Непряме* питання поволі зачіпає проблему, що розглядається.


Рис. 8.8. Класифікація питань при інтерв'юванні

*Вербальне* питання – традиційне усне питання. *Питання з використанням наочного матеріалу* дозволяє урізноманітнити інтерв'ю і зняти втому експерта (використовуються фотографії, рисунки, картки).

Основне питання спрямоване на виявлення знань. *Зондуюче* питання направляє міркування експерта в потрібну сторону. *Контрольне* питання перевіряє достовірність і об'єктивність інформації, отриманої в інтерв'ю раніше.

*Нейтральне* питання підкреслює неупередженість інженера по знанням до предмету дослідження. *Навідне* питання орієнтує експерта взяти до уваги позицію інженера по знанням.

Додатково в інтерв'ю рекомендується включати наступні питання: *контактні* (знімають психологічний бар'єр між аналітиком і експертом), *буферні* (розмежовують окремі теми інтерв'ю), *оживляючі пам'ять* експертів (реконструюють окремі випадки з практики), *«провокуючі»* (сприяють отриманню непідготовлених відповідей).

При використанні методу інтерв'ювання слід мати на увазі, що його ефективність багато в чому визначається мовою питань (зрозумілістю, лаконічністю, термінологією); порядком питань (логічна послідовність); доречністю питань (етичністю і ввічливістю).

Перш ніж готувати питання, аналітик повинен опанувати ключовий набір знань досліджуваної предметної області, оскільки будь-яке питання має сенс тільки в контексті.

*Метод вільного діалогу* дозволяє вилучати знання в формі бесіди з експертом, тому тут не передбачається використання жорсткого опитувальника або плану. У той же час підготовка до вільного діалогу повинна проводитися за спеціальною методикою, в яку входить загальна, спеціальна, конкретна і психологічна підготовка. Загальна підготовка спрямована на підвищення наукової ерудиції, оволодіння загальною культурою, знайомство з системною методологією. Спеціальна підготовка зводиться до оволодіння теорією і навичками інтерв'ювання. Конкретна підготовка передбачає вивчення предметної області, підготовку ситуації спілкування, знайомство з експертом, тестування експерта. Психологічна підготовка включає знайомство з теорією спілкування і з когнітивною психологією.

*Ігри з експертом* істотно відрізняються від наведених вище індивідуальних активних методів вилучення знань і розглядаються в класі групових активних методів, де особливе місце належить рольовим і експертним методам.

*Активні групові методи* включають «мозковий штурм», дискусії за круглим столом і рольові ігри. Групові методи дозволяють творчо інтегрувати знання безлічі експертів.

Метод *«мозкового штурму»* – один з найбільш відомих і широко застосовуваних методів генерування нових ідей шляхом творчого співробітництва групи фахівців. Будучи в деякому сенсі єдиним мозком, група намагається штурмом подолати труднощі, що заважають вирішити проблему, що розглядається. У процесі такого штурму учасники висувають і розвивають власні ідеї, стимулюючи появу нових і комбінуючи їх. Для забезпечення максимального ефекту «мозковий штурм» повинен підкорятися певним правилам і грунтуватися на поділі в часі процесу висунення ідей і процесу їх обговорення та оцінки. На першій стадії штурму

забороняється засуджувати висунуті ідеї і пропозиції (вважається, що критичні зауваження відводять до частковості, переривають творчий процес, заважають висуненню ідей). Роль аналітика полягає в тому, щоб активізувати творче мислення учасників засідання і забезпечити висунення якомога більшої кількості ідей.

Після висунення ідей виконуються ретельне їх обговорення, оцінка і відбір кращих. На стадії обговорення учасники «мозкового штурму» повинні сконцентруватися на позитивних сторонах ідей, знайти в них раціональні зерна і запропонувати напрямки їх розвитку. Висунуті в процесі обговорення додаткові ідеї можуть базуватися на ідеях інших учасників або, навпаки, служити для них фундаментом, каталізатором. Значний ефект дає комбінування ідей при одночасному виявленні переваг і недоліків синтезованих при цьому варіантів.

Метод «мозкового штурму» ефективний при вирішенні не дуже складних задач загального організаційного характеру, коли проблема добре знайома всім учасникам засідання і з даного питання є достатня інформація. Існує ряд модифікацій цього методу.

*Індивідуальний «мозковий штурм»* проводиться за тими ж правилами, що і колективний, але виконується одним експертом, який одночасно генерує ідеї, дає їм об’єктивну оцінку і критикує їх.

*Масовий «мозковий штурм»* проводиться в масовій аудиторії (до декількох десятків людей). Відбір ідей проводиться на проміжних етапах. Експерти групуються по 6-8 чоловік, при цьому важливо, щоб безпосереднє відношення до задачі мав лише керівник групи, а решта були лише знайомі з нею (інакше амбіції можуть зіграти негативну роль). Штурм проводиться в два етапи. На першому етапі оперативні групи здійснюють прямий колективний «мозковий штурм». При цьому бажано, щоб кожна група працювала над задачею, найбільш близькою за тематикою до профілю фахівців. На другому етапі керівники кожної групи протягом декількох хвилин оцінюють висунуті ідеї, відбирають з них найбільш цікаві і повідомляють їх на «пленарному засіданні».

*Подвійний «мозковий штурм»* органічно з’єднує в собі процеси генерування ідей і їх доброзичливої позитивної критики.

*Зворотний «мозковий штурм»* відрізняється від прямого тим, що в ньому більше уваги приділяється критиці висловлених ідей.

*Метод дискусії за круглим столом* передбачає рівноправне обговорення експертами поставленої проблеми. Відмінною особливістю методу дискусії є колективний розгляд предметної області з різних точок зору і дослідження суперечних гіпотез.

*Експертні ігри* призначені для вилучення знань і базуються на ділових, діагностичних і комп'ютерних іграх.

За кількістю учасників ігри поділяють на індивідуальні (ігри з експертом) і групові (рольові ігри в групі). За застосуванням спеціального обладнання – ігри з тренажерами та ігри без реквізиту. Особливий клас є комп'ютерні ігри.

В *іграх з експертом* інженер по знанням бере на себе чию-небудь роль в моделюємій ситуації. *Рольові ігри* в групі передбачають участь в грі декількох фахівців. Учасники гри наділяються певними ролями, а власне гра проводиться за складеним когнітологом сценарієм. З метою підвищення ефективності рольових ігор до них необхідно вводити елементи змагальності.

Ігри *із застосуванням тренажерів* дозволяють зафіксувати важко вловимі знання, які виникають в реальних ситуаціях і можуть бути втрачені при виході з них.

*Комп'ютерні* експертні ігри в даний час використовуються в основному з метою навчання. Вони корисні для «розминки» експертів перед сеансом вилучення знань.

Текстологічні методи включають методи вилучення знань, засновані на вивченні текстів підручників, спеціальної літератури і документів. Текстологія – це наука, метою якої є практичне прочитання текстів, вивчення та інтерпретація літературних джерел, а також розгляд семіотичних, психолінгвістичних та інших аспектів вилучення знань з текстів.

Особливу складність представляє витяг знань зі спеціальної літератури і методик, оскільки в них дуже високий ступінь концентрації спеціальних знань.

# ЛЕКЦІЯ 9. СИСТЕМИ ПРИКЛАДНОЇ ЛІНГВІСТИКИ І МАШИННОГО ЗОРУ

Прикладна лінгвістика (прикладне мовознавство) – поряд з теоретичною лінгвістикою є частиною науки, що займається мовою. Спеціалізується на вирішенні практичних завдань, пов'язаних з вивченням мови, а також на практичному використанні лінгвістичної теорії в інших областях.

Як самостійна дисципліна зі своїм апаратом і термінологією, прикладна лінгвістика найбільш поширена в англомовних країнах і азіатсько-тихоокеанському регіоні, при цьому за змістом вона майже цілком збігається з лінгводидактикою, з деяким залученням корпусної лінгвістики. Такі сфери, як перекладознавство, комп'ютерна лінгвістика, лінгвістична експертиза, в англомовних країнах зазвичай розглядаються як дисципліни, окремі від прикладної лінгвістики.

Істотним фактором для європейсько-української традиції є широке проникнення методів і термінології теоретичної лінгвістики, тоді як в англомовному світі теоретична і прикладна лінгвістика мають багато в чому різний термінологічний апарат.

Комп'ютерна лінгвістика – напрямок в прикладній лінгвістиці, орієнтований на використання комп'ютерних інструментів – програм, комп'ютерних технологій організації та обробки даних – для моделювання функціонування мови в тих чи інших умовах, ситуаціях, проблемних сферах і т.д.

Комп'ютерна лінгвістика – науковий напрям в області математичного і комп'ютерного моделювання інтелектуальних процесів у людини і тварин при створенні систем штучного інтелекту, яке ставить собі за мету використання математичних моделей для опису природних мов.

Комп'ютерна лінгвістика частково перетинається з обробкою природних мов. Однак останній акцент робиться не на абстрактні моделі, а на прикладні методи опису та обробки мови для комп'ютерних систем.

## 9.1. Системи розуміння природної мови

Обробка текстів на природній мові (Natural Language Processing, NLP) – загальний напрямок штучного інтелекту і математичної лінгвістики. Він вивчає проблеми комп'ютерного аналізу і синтезу текстів на природних мовах. Стосовно до штучного інтелекту аналіз означає розуміння мови, а синтез – генерацію грамотного тексту.

Процес спілкування з машиною довгий час залишався долею фахівців і був недоступний для розуміння «простим смертним», які були споживачами комп'ютерних послуг. Комп'ютерний інтерфейс на перших етапах розвитку обчислювальної техніки в якості обов'язкового елемента неодмінно включав людину-фахівця.

Мало хто знає, як людина спілкувався з першими обчислювальними машинами. Відбувалося це так: оператор, використовуючи дроти з роз'ємами на кінцях, з'єднував між собою тригери (з яких, власне, і складалася машина) таким чином, щоб при запуску виконувалася потрібна послідовність команд. Зовні це дуже нагадувало маніпуляції телефонних АТС початку століття, а по суті – було дуже кваліфікованою роботою. Можна сказати, програмування тоді здійснювалося навіть не в машинних командах, а на апаратному рівні. Потім задача спростилася: послідовність потрібних команд стали записувати безпосередньо в пам'ять машини. Для введення інформації стали застосовуватися більш продуктивні пристрої. Спочатку це були групи тумблерів, перемикаючи які, оператор (або програміст – тоді ці поняття означали одне і те ж) міг набрати потрібну команду і ввести її в пам'ять машини. Потім з'явилися перфокарти. Слідом – перфострічки. Швидкість спілкування з машиною зросла, число помилок, що виникають при введенні, різко зменшилася. Але сутність цього спілкування, його характер – не змінилися.

Можливість вперше поспілкуватися безпосередньо з'явилася на так званих малих машинах. Наприклад, діалоговий інтерфейс «Наїрі» дозволяв на клавіатурі набирати команду, адресовану безпосередньо машині і отримати осмислену відповідь. На той момент діалоговий режим командного рядка здавався верхом досконалості. Будь-який користувач комп'ютерних послуг, не вдаючись у технічні труднощі і вивчивши команди операційної

системи, міг спілкуватися з комп'ютером без посередників. Далі з розвитком операційних систем з'явився графічний інтерфейс і відпала потреба в знанні спеціальних команд. На наступному етапі розвитку з'явився звуковий або мовний інтерфейс.

*Розуміння в діалозі*

На роль інтерфейсу, який влаштував би всіх, зараз претендує інтерфейс мовний, до чого людство завжди прагнуло в спілкуванні з комп'ютером. За порівняно короткий період був вироблений вичерпний теоретичний базис, і практичні досягнення обумовлювалися тільки продуктивністю комп'ютерної техніки. Побудова мовного інтерфейсу розпадається на три складові.

Перша задача полягає в тому, щоб комп'ютер міг «зрозуміти» те, що йому говорить людина, тобто він повинен вміти отримувати з промови людини корисну інформацію. Поки що, на нинішньому етапі, ця задача зводиться до того, щоб витягти з промови смислову її частину, текст (розуміння таких складових, як скажімо, інтонація, поки взагалі не розглядається). Тобто ця задача зводиться до заміни клавіатури на мікрофон. Перешкодою для остаточного вирішення цієї задачі є те, яким чином можна розчленувати нашу мову на смислові частини, щоб витягти з неї складові, в яких міститься смисл. У тому звуковому потоці, який ми видаємо при розмові, не можна розрізнити ні окремих букв, ні складів: навіть, здавалося б, однакові букви і склади в різних словах на спектрограмах виглядають по-різному. Проте, багато фірм вже мають свої методики (ретельно приховувані), що дозволяють вирішити цю задачу. У всякому випадку, після попереднього тренування сучасні системи розпізнавання мови працюють досить стерпно і роблять помилок не більше, ніж робили оптичні системи розпізнавання друкованих символів в перші роки функціонування.

Друга задача полягає в тому, щоб комп'ютер сприйняв смисл сказаного. Поки мовне повідомлення складається з деякого стандартного набору зрозумілих комп'ютеру команд (наприклад, дублюючих пункти меню), нічого складного в її реалізації немає. Однак навряд чи такий підхід буде зручнішим, ніж введення цих же команд з клавіатури або за допомогою миші. Мабуть, навіть зручніше просто клацнути мишкою по іконці програми, ніж чітко вимовляти і при цьому заважати оточуючим. В ідеалі комп'ютер повинен чітко «осмислювати» природну мову людини і розуміти,

що деякі слова означають в одній ситуації різні поняття, а в іншій – одне і те ж. На думку більшості фахівців, дана задача не може бути вирішена без допомоги систем штучного інтелекту. Великі надії є на квантові комп'ютери. Поки доля мовного інтерфейсу – всього лише дублювання голосом команд, які можуть бути введені з клавіатури або за допомогою миші. Звичайно, мовне введення текстів в комп'ютер набагато зручніше, ніж звичайний набір текстового файлу. При цьому зовсім не потрібно, щоб комп'ютер осмислював почуту промову, а задача перекладу промови в текст більш-менш вирішена. Багато випущених програм «мовного інтерфейсу» орієнтовані на введення промови.

Третя задача полягає в тому, щоб комп'ютер міг перетворити інформацію, з якою він оперує, в мовне повідомлення, зрозуміле людині. По суті, синтез мови – це чисто математична задача, яка в даний час вирішена на досить хорошому рівні. І найближчим часом, швидше за все, буде вдосконалюватися тільки її технічна реалізація. Вже є різного роду програми для читання вголос текстових файлів, озвучуванням діалогових вікон, пунктів меню і з генерацією розбірливих текстових повідомлень вони справляються без проблем.

Становлення сучасної комп'ютерної індустрії проходило при розвитку графічного інтерфейсу, альтернативи якому в колі задач, що вирішуються сьогодні комп'ютерами, не існує. Масові додатки: САПР, офісні та видавничі пакети, СУБД складають основний обсяг інтелектуальної складової комп'ютерів, залишаючи в їх нинішньому вигляді дуже мало місця для застосування альтернативних моделей користувальницького інтерфейсу, в тому числі і мовного.

Для подачі команд, пов'язаних з позиціонуванням в просторі, людина завжди користувалася і буде користуватися жестами, тобто системою «руки-очі». На цьому принципі побудований сучасний графічний інтерфейс. Перспектива заміни клавіатури і миші блоком розпізнавання мови на даний момент поки відпадає. При цьому виграш від покладання на нього частини функцій управління настільки малий, що не зміг надати достатніх підстав навіть для пробного впровадження в масових комп'ютерах протягом вже більше тридцяти років. Саме таким терміном оцінюється існування комерційно застосовних систем розпізнавання мови.

Сьогодні серед провідних виробників систем розпізнавання мови не прийнято віддавати належне досягненням дослідників минулих років. Причина зрозуміла: це не тільки в значній мірі знизить видимі показники досягнутого ними прогресу, але і посприяє виникненню цілком обгрунтованих сумнівів в перспективності здійснюваних підходів взагалі.

У 1969 році в своєму знаменитому листі редактору журналу Акустичного суспільства Америки Дж. Пієс, співробітник фірми Bell Laboratories, вказав на відсутність явного прогресу в той час і можливості такого прогресу технології розпізнавання мови в найближчому майбутньому у зв'язку з нездатністю комп'ютерів аналізувати синтаксичну, семантичну і прагматичну інформацію, що міститься в висловлюванні. Наявний бар'єр може бути подоланий тільки з розвитком систем штучного інтелекту – напрямком, що натрапив в 70-х на бар'єр складності. Важко сподіватися на подальше поліпшення характеристик пристроїв мовного вводу, враховуючи, що вже в 70-х роках їх здатність розпізнавати звуки мови перевершувала людську. Даний факт був підтверджений серією експериментів в порівнянні впевненості розпізнавання людиною і комп'ютером слів іноземної мови і безглуздих ланцюжків звуків. При відсутності можливості підключення прагматичних (смислових), семантичних та інших аналізаторів людина явно програє.

Для ілюстрації декілька спірних тверджень розглянемо перспективу і основні проблеми застосування систем мовного введення текстів, особливо тих, що активно просуваються останнім часом. Для порівняння: спонтанна промова вимовляється із середньою швидкістю 2,5 слова в секунду, професійний машинопис – 2 слова в секунду, непрофесійна – 0,4. Таким чином, на перший погляд, мовне введення має значну перевагу за продуктивністю. Однак оцінка середньої швидкості диктовки в реальних умовах знижується до 0,4 слова в секунду в зв'язку з необхідністю чіткого проголошення слів при мовному введенні і досить високим відсотком помилок розпізнавання, які потребують коригування. Мовний інтерфейс природний для людини і забезпечує додаткову зручність при наборі текстів. Однак навіть професійного диктора може не порадувати перспектива протягом декількох годин диктувати малозрозумілому і німому комп'ютеру. Крім того,

наявний досвід експлуатації подібних систем свідчить про високу ймовірність захворювання голосових зв'язок операторів, що пов'язано з неминучою при диктовці комп'ютера монотонністю мови.

Часто до переваг мовного введення тексту відносять відсутність необхідності в попередньому навчанні. Однак одне з найслабших місць сучасних систем розпізнавання мови – чутливість до чіткості вимови – призводить до втрати цієї очевидної переваги. Друкувати на клавіатурі оператор вчиться в середньому 1-2 місяці. Постановка правильної вимови може зайняти декілька років. Крім того, додаткова напруга – наслідок свідомих і підсвідомих зусиль по досягненню більш високої розпізнаваності – зовсім не сприяє збереженню нормального режиму роботи мовного апарату оператора і значно збільшує ризик появи специфічних захворювань. Існує ще одне неприємне обмеження застосовності: оператор, взаємодіє з комп'ютером через мовний інтерфейс, змушений працювати в звукоізольованому окремому приміщенні або користуватися звукоізолюючим шоломом. Інакше він буде заважати роботі своїх сусідів по офісу, які, в свою чергу, створюючи додатковий шумовий фон будуть значно ускладнювати роботу мовного розпізнавача. Таким чином, мовний інтерфейс вступає в явне протиріччя із сучасною організаційною структурою підприємств, орієнтованих на колективну працю. Ситуація дещо пом'якшується з розвитком віддалених форм трудової діяльності, проте ще досить довго сама природна для людини продуктивна і потенційно масова форма призначеного для користувача інтерфейсу приречена на вузьке коло застосування.

Обмеження застосовності систем розпізнавання мови в рамках найбільш популярних традиційних додатків змушують зробити висновок про необхідність пошуку потенційно перспективних для впровадження мовного інтерфейсу додатків за межами традиційної офісної сфери, що підтверджується комерційними успіхами вузькоспеціалізованих мовних систем.

Найуспішніший проект комерційного застосування розпізнавання мови – телефонна мережа фірми AT&T. Клієнт може запросити одну з п'яти категорій послуг, використовуючи будь-які слова. Він каже до тих пір, поки в його висловлюванні зустрінеться одне з п'яти ключових слів. Ця система в даний час обслуговує

близько мільярда дзвінків на рік. Даний висновок знаходиться в протиріччі з усталеними широко поширеними стереотипами і очікуваннями. Незважаючи на те, що одним з найбільш перспективних напрямків для впровадження систем розпізнавання мови може стати сфера комп'ютерних ігор, вузькоспеціалізованих реабілітаційних програм для інвалідів, телефонних та інформаційних систем, провідні розробники мовного розпізнавання нарощують зусилля по досягненню універсалізації та збільшення обсягів словника навіть враховуючи скорочення процедури попереднього налаштування на диктора. А тим часом саме ці програми представляють дуже низькі вимоги до обсягу розпізнаваного словника поряд з жорсткими обмеженнями, що накладаються на попереднє налаштування. Більш того, розпізнавання спонтанної злитої мови практично знаходиться на одному й тому ж місці з 70-х років в силу нездатності комп'ютера ефективно аналізувати неакустичні характеристики мови.

Говорячи про мовний інтерфейс, часто роблять упор на розпізнавання мови, забуваючи про іншу його сторону – мовний синтез. Головну роль в цьому перекосі зіграв бурхливий розвиток останнім часом систем, орієнтованих на події, що в значній мірі мають відношення до комп'ютера як активної сторони діалогу. Раніше підсистеми розпізнавання і синтезу мови розглядалися як частини єдиного комплексу мовного інтерфейсу. Однак інтерес до синтезу зник досить швидко. По-перше, розробники не зустріли навіть десятої частки складнощів, з якими вони зіткнулися при створенні систем розпізнавання. По-друге, на відміну від розпізнавання синтез мови не демонструє значних переваг перед іншими засобами виведення інформації з комп'ютера. Практично вся його цінність полягає в доповненні мовного введення. Для людини природним і звичним є саме діалог, а не монолог. Як наслідок недооцінки необхідності мовної відповіді з'являється підвищена стомлюваність операторів, монотонність мови та обмеженість застосовності мовного інтерфейсу.

Правильно організований і модульований синтез може в значній мірі знизити ризик появи у оператора захворювань, пов'язаних з монотонністю мови і додатковою напругою. Майбутнє мовного інтерфейсу в не меншому ступені залежить від уміння сучасних розробників не тільки створити технологічну основу

мовного введення, але і гармонійно злити технологічні знахідки в єдину логічно завершену систему взаємодії «людина-комп'ютер».

*Методи озвучування мови*

Найширше поділ стратегій, що застосовуються при озвучуванні мови, – це поділ на підходи, які спрямовані на побудову діючої моделі мовотворної системи людини, і підходи, де ставиться задача змоделювати акустичний сигнал. Перший підхід відомий під назвою артикуляторного синтезу. Другий підхід більш простий і краще вивчений. Усередині нього виділяється два основних напрямки – формантний синтез за правилами і компілятивний синтез.

Формантні синтезатори використовують збуджуючий сигнал, який проходить через цифровий фільтр, побудований на декількох резонансах, схожих на резонанси голосового тракту. Поділ збуджуючого сигналу і передавальної функції голосового тракту становить основу класичної акустичної теорії мовотворення.

Компілятивний синтез здійснюється шляхом склеювання потрібних одиниць компіляції з наявного інвентарю. На цьому принципі побудовано безліч систем, що використовують різні типи одиниць і різні методи складання інвентарю. У таких системах необхідно застосовувати обробку сигналу для приведення частоти основного тону, енергії і тривалості одиниць до тих, якими повинна характеризуватися синтезована мова. Крім того, потрібно, щоб алгоритм обробки сигналу згладжував розриви в формантній (і спектральній в цілому) структурі на границях сегментів. В системах компілятивного синтезу застосовуються два різних типи алгоритмів обробки сигналу: LP (скор. англ. Linear Preduction) і PSOLA (скор. англ. Pitch Synchronous Overlap and Add). LP-синтез заснований в значній мірі на акустичній теорії мовотворення, на відміну від PSOLA-синтезу, який діє шляхом простого розбиття звукової хвилі, що становить одиницю компіляції, на тимчасові вікна і їх перетворення. Алгоритми PSOLA дозволяють добиватися хорошого збереження природності звучання при модифікації вихідної звукової хвилі.

*Автоматичний комп'ютерний синтез мови по тексту*

Мовне виведення інформації з комп'ютера – проблема не менш важлива, ніж мовне введення. Фактично, завдяки синтезу мови по тексту відкривається ще один канал передачі даних від

комп’ютера до людини, аналогічний звичайному монітору. Звичайно, важкувато було б передати малюнок голосом. Але ось почути електронну пошту або результат пошуку в базі даних в ряді випадків було б досить зручно, особливо якщо в цей час погляд зайнятий чим-небудь іншим.

З точки зору користувача, найбільш розумне рішення проблеми синтезу мови – це включення мовних функцій (багатомовних, з можливостями перекладу) до складу операційної системи. Комп’ютери повинні озвучувати навігацію по меню, читати (дублювати голосом) екранні повідомлення, каталоги файлів і т. ін. Важливе зауваження: користувач повинен мати достатні можливості по налаштуванню голосу комп’ютера, зокрема, при бажанні, зуміти вимкнути голос зовсім.

Вищезазначені функції були б не зайвими для осіб, які мають проблеми із зором. Для всіх інших вони створюють новий вимір зручності користування комп’ютером і значно знижують навантаження на нервову систему і на зір.

1. *Методи синтезу мови*. Текст складається з слів, розділених пробілами і знаками пунктуації. Виголошення слів залежить від їх розташування в реченні, а інтонація фрази – від знаків пунктуації. Більш того, досить часто і від типу застосовуваної граматичної конструкції: в ряді випадків при проголошенні тексту чується явна пауза, хоча будь-які знаки пунктуації відсутні. Нарешті, проголошення залежить і від смислу слова, наприклад, вибір одного з варіантів «за'мок» або «замо'к» для одного і того ж слова «замок». Вже стартовий аналіз проблеми показує її складність. І справді, на цю тему написані десятки монографій, і величезна кількість публікацій здійснюється щомісяця.

2. *Узагальнена функціональна структура синтезатора*. Структура ідеалізованої системи автоматичного синтезу мови складається з декількох блоків:

- визначення мови тексту;
- нормалізація тексту;
- лінгвістичний аналіз: синтаксичний, морфемний аналіз і т.д.;
- формування просоідичних характеристик;
- фонемний транскриптор;
- формування керуючої інформації;

− отримання звукового сигналу.

Вона не описує жодну з існуючих реально систем, але містить компоненти, які можна виявити у багатьох системах. Автори конкретних систем, незалежно від того, чи є ці системи вже комерційним продуктом або ще знаходяться у стадії дослідницької розробки, приділяють різну увагу окремим блокам і реалізують їх дуже по-різному, відповідно до практичних вимог.

3. *Модуль лінгвістичної обробки*. Перш за все, текст, який підлягає прочитанню, надходить в модуль лінгвістичної обробки. У ньому проводиться визначення мови (в багатомовній системі синтезу), а також фільтруються символи, що не підлягають вимовленню. У деяких випадках використовуються спелчекери (модулі виправлення орфографічних і пунктуаційних помилок). Потім відбувається нормалізація тексту, тобто здійснюється поділ введеного тексту на слова та інші послідовності символів. До символів відносяться, зокрема, знаки пунктуації та символи початку абзацу. Всі знаки пунктуації дуже інформативні. Для озвучування цифр розробляються спеціальні підблоки. Перетворення цифр в послідовності слів є відносно легкою задачею (якщо читати цифри як цифри, а не як числа, які повинні бути правильно оформлені граматично ), але цифри, які мають різне значення і функцію, вимовляються по-різному. Для багатьох мов можна говорити, наприклад, про існування окремої вимовної підсистеми телефонних номерів. Пильну увагу потрібно приділити правильній ідентифікації та озвученню цифр, що позначають числа місяця, роки, час, телефонні номери, грошові суми і т. д. (список для різних мов може бути різним).

4. *Лінгвістичний аналіз*. Після процедури нормалізації кожному слову текста (кожній словоформі) необхідно приписати відомості про його вимову, тобто перетворити в ланцюжок фонем або, інакше кажучи, створити його фонемну транскрипцію. У багатьох мовах, в тому числі і в українській, існують досить регулярні правила читання – правила відповідності між буквами і фонемами (звуками), які, однак, можуть вимагати попередньої розстановки словесних наголосів. В англійській мові правила читання дуже нерегулярні, і задача даного блоку для англійського синтезу тим самим ускладнюється. У будь-якому випадку при визначенні вимови імен власних, запозичень, нових слів, скорочень

та абревіатур виникають серйозні проблеми. Просто зберігати транскрипцію для всіх слів мови не представляється можливим через великий обсяг словника і контекстних змін вимови одного і того ж слова у фразі.

Крім того, слід коректно розглядати випадки графічної омонімії: одна і та ж послідовність буквених символів в різних контекстах часом представляє два різних слова/словоформи і читається по-різному. Часто вдається вирішити проблему неоднозначності такого роду шляхом граматичного аналізу, проте іноді допомагає тільки використання більш широкої семантичної інформації.

Для мов з досить регулярними правилами читання одним з продуктивних підходів до перекладу слів в фонеми є система контекстних правил, що переводять кожну букву/буквосполучення в ту чи іншу фонему, тобто автоматичний фонемний транскриптор. Однак чим більше в мові винятків з правил читання, тим гірше працює цей метод. Стандартний спосіб поліпшення вимови системи полягає в занесенні декількох тисяч найбільш уживаних винятків в словник. Альтернативне підходу «слово-буква-фонема» рішення передбачає морфемний аналіз слова і переклад в фонеми морфів (тобто значущих частин слова: приставок, коренів, суфіксів і закінчень). Однак у зв'язку з різними граничними явищами на стиках морфів розкладання на ці елементи являє собою значні труднощі. У той же час для мов з багатою морфологією, наприклад, для українського, словник морфів був би компактніше. Морфемний аналіз зручний ще й тому, що з його допомогою можна визначати приналежність слів до частин мови, що дуже важливо для граматичного аналізу тексту і завдання його просодичних характеристик.

Особливу проблему для даного етапу обробки тексту утворюють власні імена.

5. *Формування просодичних характеристик*. До просодичних характеристик висловлювання відносяться його тональні, акцентні і ритмічні характеристики. Їх фізичними аналогами є частота основного тону, енергія і тривалість. У промові просодичні характеристики висловлювання визначаються не тільки складовими його словами, але також тим, яке значення воно несе і для якого слухача призначене, емоційним і фізичним станом мовця і багатьма

іншими факторами. Багато з цих факторів зберігають свою значимість і при читанні вголос, оскільки людина зазвичай інтерпретує і сприймає текст в процесі читання. Таким чином, від системи синтезу слід очікувати приблизно того ж, що вона зможе розуміти наявний у неї на вході текст, використовуючи методи штучного інтелекту. Більшість сучасних систем автоматичного синтезу намагаються коректно синтезувати мову з емоційно нейтральною інтонацією. Тим часом, навіть ця задача на сьогоднішній день є дуже складною.

Формування просодичних характеристик, необхідних для озвучування тексту, здійснюється трьома основними блоками: блоком розстановки синтагматичних границь (паузи), блоком приписування ритмічних і акцентних характеристик (тривалості та енергія), блоком приписування тональних характеристик (частота основного тону). При розстановці синтагматичних границь визначаються частини висловлювання (синтагми), всередині яких енергетичні і тональні характеристики поводяться одноманітно і які людина може вимовити на одному диханні. Якщо система не робить пауз на границях таких одиниць, то виникає негативний ефект: слухачу здається, що мовець (в даному випадку – система) задихається. Крім цього, розстановка синтагматичних границь істотна і для фонемної транскрипції тексту. Найпростіше рішення полягає в тому, щоб ставити границі там, де їх диктує пунктуація. Для найбільш простих випадків, коли пунктуаційні знаки відсутні, можна застосувати метод, заснований на використанні службових слів і дієслівних форм.

Задача приписування тональних характеристик зазвичай ставиться досить вузько. В системах синтезу мови реченню, як правило, приписується нейтральна інтонація. Не було спроб моделювати ефекти вищого рівня, такі, як емоційне забарвлення мови, оскільки цю інформацію витягти з тексту важко, а часто і просто неможливо.

*Синтезатор української мови*

В основі мовного синтезу лежить ідея суміщення методів конкатенації і синтезу за правилами. Метод конкатенації при адекватному наборі базових елементів компіляції забезпечує якісне відтворення спектральних характеристик мовного сигналу, а набір правил – можливість формування природного інтонаційно-

просодичного оформлення висловлювань. Існують і інші методи синтезу, може бути, в перспективі більш гнучкі, але дають поки менш природне озвучування тексту. Це, перш за все, параметричний (формантний) синтез мови за правилами або на основі компіляції, що розвивається для ряду мов зарубіжними дослідниками. Однак для реалізації цього методу необхідні статистично представницькі акустико-фонетичні бази даних і відповідна комп'ютерна технологія, які поки доступні не всім.

1. *Мова формального запису правил синтезу*. Для створення зручного і швидкого режиму зміни і верифікації правил, включених в різні блоки синтезуючої системи, була розроблена формалізована і в той же час змістовно прозора і зрозуміла мова записи правил, яка легко компілюється в вихідні тексти програм. В даний час блок автоматичного транскриптора налічує близько 1000 рядків, записаних на формалізованій мові подання правил.

2. *Інтонаційне забезпечення*. Функція розроблених правил полягає в тому, щоб визначити тимчасові і тональні характеристики базових елементів компіляції, які при обробці синтагми обираються з бібліотеки в потрібній послідовності спеціальним процесором (блоком кодування). Необхідні для цього попередні операції над синтезованим текстом: виділення синтагм, вибір типу інтонації, визначення ступеня виділення (ударності-безударності) голосних і символьного звукового наповнення складових комплексів здійснюються блоком автоматичного транскриптора.

У тимчасовий процесор входять також правила, що задають тривалість паузи після закінчення синтагми (кінцевої/некінцевої), які необхідні для синтезу зв'язкового тексту. Передбачена також модифікація загального темпу проголошення синтагми і тексту в цілому, причому в двох варіантах: в стандартному – при рівномірній зміні всіх одиниць компіляції – і в спеціальному, що дає можливість змінювати тривалість тільки голосних або тільки приголосних.

Тональний процесор містить правила формування для одинадцяти інтонаційних моделей: нейтральна оповідна інтонація (точка); точкова інтонація; типова для фокусованих відповідей на питання; інтонація речень з контрастивним виділенням окремих слів; інтонація спеціального та загального питання; інтонація особливих протиставних або порівняльних питань; інтонація

звернень, деяких типів вигуків і команд; два види незавершеності; перелічальна інтонація; інтонація вставочних конструкцій.

3. *Аллофонна база даних*. Необхідний мовний матеріал був записаний в наступному режимі оцифровки: частота дискретизації 22 кГц з розрядністю 16 біт. В якості базових елементів компіляції обрані аллофони, оптимальний набір яких і являє собою акустико-фонетичну базу синтезу. Інвентар базових одиниць компіляції включає в себе майже 1200 елементів, який займає близько 7 Мбайт пам'яті. У більшості випадків елементи компіляції є сегментами мовної хвилі фонемної розмірності. Для отримання необхідної вихідної бази одиниць компіляції був складений спеціальний словник, який містить слова і словосполучення з аллофонами у всіх врахованих контекстах. У ньому міститься 1130 слововживань.

4. *Лінгвістичний аналіз*. На основі даних, отриманих від інших модулів синтезу мови і від аллофонної бази, програма формування акустичного сигналу дозволяє здійснювати модифікацію тривалості приголосних і голосних. Вона дає можливість модифікувати тривалість окремих періодів на вокальних звуках, використовуючи дві або три точки тонування на аллофонному сегменті, здійснює модифікацію енергетичних характеристик сегмента і з'єднує модифіковані аллофони в єдину злиту мову.

На етапі синтезу акустичного сигналу програма дозволяє отримувати різноманітні акустичні ефекти, такі як: реверберація, відлуння, зміна частотного забарвлення.

5. *Інструментарій синтезу мови*. Інструментарій синтезу мови по тексту дозволяє читати вголос змішані тексти. Інструментарій є набором динамічних бібліотек (DLL), в який входять модулі українського та англійського синтезу, словник наголосів української мови, модуль правил проголошення англійських слів. На вхід інструментарію подається слово або речення, яке підлягає вимові.

*Система розпізнавання мови*

Система розпізнавання мови складається з двох частин: акустичної та лінгвістичної. Ці частини можуть бути виділені в блоки або в підпрограми. Лінгвістична частина може включати в себе фонетичну, фонологічну, морфологічну, синтаксичну і семантичну модель мови.

Акустична модель відповідає за подання мовного сигналу. Лінгвістична модель інтерпретує інформацію, що отримується від акустичної моделі, і відповідає за подання результату розпізнавання користувачеві.

1. *Акустична модель*. Існує два підходи до побудови акустичної моделі: винахідницький і біонічний. Перший базується на результатах пошуку механізму функціонування акустичної моделі. При другому підході розробник намагається зрозуміти і змоделювати роботу природних систем. Обидва підходи мають свої переваги і недоліки. При розробці технічних систем вибір підходу має першорядне значення.

2. *Лінгвістична модель*. Лінгвістичний блок підрозділяється на наступні яруси (шари, рівні); фонетичний, фонологічний, морфологічний, лексичний, синтаксичний, семантичний. Всього їх шість. Всі яруси по суті це апріорна інформація про структуру природної мови, а будь-яка апріорна інформація про потрібний предмет збільшує шанси прийняття вірного рішення. Природна мова несе вельми сильно структуровану інформацію, з чого випливає, що для кожної природної мови може знадобитися своя унікальна лінгвістична модель. Відповідно до даної моделі на першому – фонетичному рівні проводиться перетворення вхідного (для лінгвістичного блоку) подання мови в послідовність фонем, як найменших одиниць мови. Вважається, що в реальному мовному сигналі можна виявити лише аллофони – варіанти фонем, що залежать від звукового оточення. На наступному – фонологічному рівні накладаються обмеження на комбінаторику фонем (аллофонів). Обмеження – це правило навиворіт: не всі поєднання фонем (аллофонів) зустрічаються, а ті, що зустрічаються, мають різну ймовірність появи, що залежить ще і від оточення. Для опису цієї ситуації використовується математичний апарат ланцюгів Маркова. Далі, на морфологічному рівні оперують зі складоподібними одиницями мови більш високого рівня, ніж фонема. Іноді вони називаються морфемами. Вони накладають обмеження вже на структуру слова, підкоряючись закономірностям моделюємої природної мови. Лексичний ярус охоплює слова і словоформи тої чи іншої природної мови, тобто словник мови, так само вносячи важливу апріорну інформацію про те, які слова можливі для даної природної мови. Семантика встановлює

співвідношення між об’єктами дійсності і словами, які їх позначають. Вона є вищим рівнем мови. За допомогою семантичних відношень інтелект людини виробляє як би стиснення мовного повідомлення в систему образів, понять, що представляють суть мовного сполучення. Звідси випливає висновок, що система повинна бути "розумною". Чим краще у неї буде побудована модель семантичних зв’язків, еквівалента "системи уявних образів", тим більша ймовірність правильно розпізнати мову.

На рис. 9.1 представлена класифікація систем розпізнавання мови.


Рис. 9.1. Класифікація систем розпізнавання мови

## 9.2. Системи машинного зору

Компанією «*Біт*» була розроблена спеціальна технологія розпізнавання символів, яка отримала назву «*Фонтанне перетворення*», а на її основі – комерційний продукт, який отримав високу оцінку. Це система оптичного розпізнавання ABBYY FineReader. Останні версії працюють з текстом на будь-якій мові, формами, таблицями, а також розпізнають друкований і рукописний тексти.

*Основні принципи або цілісність сприйняття*

В основі фонтанного перетворення лежить принцип цілісності. Відповідно до нього будь-який сприйнятий об’єкт розглядається як ціле, що складається з частин, пов’язаних між собою певними відношеннями. Так, наприклад, друкована сторінка складається з статей, стаття – з заголовка і колонок, колонка – з абзаців, абзаци – з рядків, рядки – зі слів, слова – з букв. При цьому всі перераховані елементи тексту пов’язані між собою певними просторами і мовними відношеннями.

Для виділення цілого потрібно визначити його частини. Частини ж, в свою чергу, можна розглядати тільки в складі цілого. Тому цілісний процес сприйняття може відбуватися тільки в рамках гіпотези про сприйнятий об’єкт – цілий. Після того як висунуте припущення про сприйнятий об’єкт, виділяються та інтерпретуються його частини. Потім робиться спроба "зібрати" з них ціле, щоб перевірити правильність вихідної гіпотези. Сприйнятий об’єкт може інтерпретуватися в рамках більшого цілого.

Так, читаючи пропозицію, людина впізнає букви, сприймає слова, пов’язує їх в синтаксичні конструкції і розуміє сенс.

У технічних системах будь-яке рішення при розпізнаванні тексту приймається неоднозначно, а шляхом послідовного висунення і перевірки гіпотез і залучення як знань про саме досліджуваний об’єкт, так і загальний контекст. Цілісний опис класу об’єктів сприйняття відповідає двом умовам: по-перше, всі об’єкти даного класу задовольняють цьому опису, а по-друге, жоден об’єкт іншого класу не задовольняє йому. Наприклад, клас зображень букви "К" повинен бути описаний так, щоб будь-яке зображення букви "К" в нього потрапляло, а зображення всіх інших букв – ні. Такий опис має властивість відображуваності, тобто забезпечує відтворення описуваних об’єктів: еталон букви для системи OCR дозволяє візуально відтворити букву, еталон слова для розпізнавання мови дозволяє вимовити слово, а опис структури речення в синтаксичному аналізаторі дозволяє синтезувати правильне речення. З практичної точки зору відображуваність грає величезну роль, оскільки дозволяє ефективно контролювати якість описів.

Існує два види цілісного опису: шаблонне і структурне.

У першому випадку опис являє собою зображення в растровому або векторному поданні, і заданий клас перетворень (наприклад, повтор, масштабування та ін.).

У другому випадку опис представляється у вигляді графа, вузлами якого є сполучені елементи вхідного об’єкта, а дугами – просторові відношення між ними. У свою чергу елементи можуть виявитися складними (тобто мати свій опис).

Звичайно, шаблонний опис простіше в реалізації, ніж структурний. Однак воно не може використовуватися для опису об’єктів з високим ступенем мінливості. Шаблонний опис, наприклад, може прийматися для розпізнавання тільки друкованих символів, а структурний – ще й для рукописних.

Цілісність сприйняття пропонує два важливих архітектурних рішення. По-перше, усі джерела знання повинні працювати за можливістю одночасно. Не можна, наприклад, спочатку розпізнати сторінку, а потім піддати її словниковій та контекстній обробці, оскільки в цьому випадку неможливо буде здійснити зворотний зв’язок від контекстної обробки до розпізнавання. По-друге, досліджуваний об’єкт повинен представлятися і оброблятися за можливістю цілком.

Перший крок сприйняття – це формування гіпотези про сприйнятий об’єкт. Гіпотеза може формуватися як на основі апріорної моделі об’єкта, контексту і результатів перевірки попередніх гіпотез (процес "зверху – вниз"), так і на основі попереднього аналізу об’єкта ("знизу – вгору"). Другий крок – уточнення сприйняття (перевірка гіпотези), при якому проводиться додатковий аналіз об’єкта в рамках висунутої гіпотези і в повну силу залучається контекст.

Для зручності сприйняття необхідно провести попередню обробку об’єкта, не втративши при цьому суттєвої інформації про нього. Зазвичай попередня обробка зводиться до перетворення вхідного об’єкта в подання, зручне для подальшої роботи (наприклад, векторизація зображення), або отримання всіляких варіантів сегментації вхідного об’єкта, з якого шляхом висунення і перевірки гіпотез вибирається правильний об’єкт. Процес висунення і перевірки гіпотез повинен бути явно відображений в архітектурі програми. Кожна гіпотеза повинна бути об’єктом, який

можна було б оцінити або порівняти з іншими. Тому зазвичай гіпотези висуваються послідовно, а потім об’єднуються в список і сортуються на основі попередньої оцінки. Для остаточного ж вибору гіпотези активно використовується контекст та інші додаткові джерела знань.

*Розпізнавання символів*

Сьогодні відомо три підходи до розпізнавання символів – шаблонний, структурний і ознаковий. Але принципу цілісності відповідають лише перші два.

Шаблонний опис простіше в реалізації, однак, на відміну від структурного, він не дозволяє описувати складні об’єкти з великою різноманітністю форм. Саме тому шаблонний опис застосовується для розпізнавання лише друкованих символів, в той час як структурний – для рукописних, що мають, зазвичай, набагато більше варіантів накреслення.

1. Шаблонні системи. Такі системи перетворюють зображення окремого символу в растрове, порівнюють його з усіма шаблонами, наявними в базі і вибирають шаблон з найменшою кількістю точок, відмінних від вхідного зображення. Шаблонні системи досить стійкі до дефектів зображення і мають високу швидкість обробки вхідних даних, але надійно розпізнають тільки ті шрифти, шаблони яких їм "відомі". І якщо шрифт розпізнавання хоч трохи відрізняється від еталонного, шаблонні системи можуть робити помилки навіть при обробці дуже якісних зображень.

2. Структурні системи. У таких системах об’єкт описується як граф, вузлами якого є елементи вхідного об’єкта, а дугами – просторові відношення між ними. Системи, що реалізують подібний підхід, зазвичай працюють з векторними зображеннями. Структурними елементами є лінії, з яких складається символ. Так, для букви "р" – це вертикальний відрізок і дуга.

До недоліків структурних систем слід віднести їх високу чутливість до дефектів зображення, що порушує сполучені елементи. Також векторизація може додати додаткові дефекти. Крім того, для цих систем, на відміну від шаблонних і ознакових, до сих пір не створені ефективні автоматизовані процедури навчання. Тому для FineReader структурні описи довелося створити вручну.

3. Ознакові системи. В них усереднене зображення кожного символу представляється як об’єкт в n-мірному просторі ознак. Тут

вибирається алфавіт ознак, значення яких обчислюються при розпізнаванні вхідного зображення. Отриманий n-мірний вектор порівнюється з еталонними, і зображення відноситься до найбільш підходящого з них. Ознакові системи не відповідають принципу цілісності. Необхідна, але недостатня умова цілісності опису класу об'єктів (в нашому випадку це клас зображень, що представляють один символ) складається в тому, що опису повинні задовольняти всі об'єкти даного класу і жоден з об'єктів інших класів. Але оскільки при обчисленні ознак втрачається істотна частина інформації, важко гарантувати, що до даного класу вдасться віднести тільки «рідні» об'єкти.

*Структурно-плямний еталон*

Фонтанне перетворення поєднує в собі переваги шаблонної і структурної систем і, на нашу думку, дозволяє уникнути недоліків, властивих кожній з них окремо. В основі цієї технології лежить використання структурно-плямного еталону. Він дозволяє представити зображення у вигляді набору плям, пов'язаних між собою n-арними відношеннями, які задають структуру символу. Ці відношення (тобто розташування плям відносно інших) утворюють структурні елементи, з яких складається символ. Так, наприклад, відрізок – це один тип n-арних відношень між плямами, еліпс – другий, дуга – третій. Інші відношення задають просторове розташування елементів, що утворюють символ.

У еталоні задаються:

– ім'я;

– обов'язкові, забороняючі і необов'язкові структурні елементи;

– відношення між структурними елементами;

– відношення, що зв'язують структурні елементи з описуючим прямокутником символу;

– атрибути, використовувані для виділення структурних елементів;

– атрибути, використовувані для перевірки відношень між елементами;

– атрибути, використовувані для оцінки якості елементів і відношень;

– позиція, з якої починається виділення елемента (відношення локалізації елементів).

Структурні елементи, що виділяються для класу зображень, можуть бути вихідними і сполученими. Вихідні структурні елементи – це плями, складені – відрізок, дуга, кільце, точка. У якості складових структурних елементів, в принципі, можуть бути взяті будь-які об'єкти, описані в еталоні. Крім того, вони можуть бути описані як через вихідні, так і через інші сполучені структурні елементи.

Наприклад, для розпізнавання корейських ієрогліфів (складовий лист) сполученими елементами для опису складу є опису окремих букв (але не окремі елементи букв). У підсумку, використання складових структурних елементів дозволяє будувати ієрархічні описи класів розпізнаваних об'єктів.

У якості відношень використовуються зв'язки між структурними елементами, які визначаються або метричними характеристиками цих елементів (наприклад, «довжина більше»), або їх взаємним розташуванням на зображенні (наприклад, «правіше», «стикається»).

При завданні структурних елементів і відношень використовуються конкретизуючі параметри, що дозволяють довизначити структурний елемент або відношення при використанні цього елемента в еталоні конкретного класу. Для структурних елементів конкретизуючими можуть бути, наприклад, параметри, що задають діапазон допустимої орієнтації відрізка, а для відношень – параметри, що задають граничну допустиму відстань між характерними точками структурних елементів у відношенні «стикається».

Конкретизуючі параметри використовуються також для обчислення «якості» конкретного структурного елемента зображення і «якості» виконання даного відношення.

Побудова і тестування структурно-плямних еталонів для класів розпізнаваних об'єктів – процес складний і трудомісткий. База зображень, яка використовується для налагодження описів, повинна містити приклади хороших і поганих (гранично допустимих) зображень для кожної графеми, а зображення бази поділяються на навчальну і контрольну множини.

Розробник опису спочатку задає набір структурних елементів (розбиття на плями) і відношення між ними. Система навчання по базі зображень автоматично обчислює параметри елементів і

відношень. Отриманий еталон перевіряється і коригується по контрольній вибірці зображень даної графеми. За контрольною ж вибіркою перевіряється результат розпізнавання, тобто оцінюється якість підтвердження гіпотез.

Розпізнавання з використанням структурно-плямного еталону відбувається наступним чином. Еталон накладається на зображення, і відношення між виділеними на зображенні плямами порівнюються з відношеннями плям в еталоні. Якщо виділені на зображенні плями і відношення між ними задовольняють еталону деякого символу, то даний символ додається до списку гіпотез про результат розпізнавання вхідного зображення.

*Уроки машинного читання від Cognitive Technologies*

Працює система за принципом «однієї кнопки». Це означає, що при натисканні кнопки «Скануй і Розпізнавай» запускається весь процес обробки документа: сканування, фрагментація сторінки на текстові і графічні блоки, розпізнавання тексту, перевірка орфографії та формування вихідного файлу. Але що за всім цим стоїть? Інтелектуальний алгоритм дозволяє автоматично підібрати оптимальний рівень яскравості сканера (адаптивне сканування) у залежності від фону документа, зберегти ілюстрації (або, у залежності від розв'язуваної задачі, видалити непотрібні графічні елементи для максимального скорочення подальшого редагування).

У CuneiForm використовується декілька методів подібного зіставлення. По-перше, образ кожного символу розкладається на окремі елементи – події. Наприклад, подією є фрагмент від однієї лінії перетину до іншої. Сукупність подій є компактним описом символу.

Інші методи засновані на співвідношенні «мас» окремих елементів символів і описі їх характерних ознак (заокруглення, прямі, кути і т. д.). По кожному з цих описів існують бази даних, в яких знаходяться відповідні еталони. Елемент зображення, що поступає на обробку, порівнюється з еталоном. А потім на підставі цього порівняння вирішальна функція виносить вердикт про відповідність зображення конкретному символу. Крім того, існують алгоритми, які дозволяють працювати з текстами низької якості. Так, для розрізання «склеєних» символів існує метод оцінки оптимального розбиття. І навпаки, для з'єднання «розсипаних» елементів розроблений механізм їх з'єднання.

У CuneiForm'96 були вперше застосовані алгоритми самонавчання (або адаптивного розпізнавання). Принцип їх роботи полягає в наступному. У кожному тексті присутні чітко і нечітко надруковані символи. Якщо після того як система розпізнала текст (як це робить звичайна система, наприклад попередня версія OCR CuneiForm 2.95), з'ясовується, що точність виявилась нижче порогової, проводиться дорозпізнавання тексту на основі шрифта, який генерується системою по добре надрукованим символам. Тут розробники з'єднали гідності двох типів систем розпізнавання: омнії мультишрифтові. Перші дозволяють розпізнавати будь-які шрифти без додаткового навчання, другі ж більш стійкі при розпізнаванні низькоякісних текстів. Результати застосування Cunei-Form'96 показали, що використання самонавчальних алгоритмів дозволяє підняти точність розпізнавання низькоякісних текстів в чотири-п'ять разів! Але головне, мабуть, в тому, що самонавчальні системи мають набагато більший потенціал підвищення точності розпізнавання.

Важливу роль відіграють методи словникового і синтаксичного розпізнавання і, по суті, служать потужним засобом підтримки геометричного розпізнавання. Але для їх ефективного використання необхідно було вирішити дві важливі задачі. По-перше, реалізувати швидкий доступ до великого (порядку 100000 слів) словника. В результаті вдалося побудувати систему зберігання слів, де на зберігання кожного слова йшло не більше одного байта, а доступ здійснювався за мінімальний час. З іншого боку, треба було побудувати систему корекції результатів розпізнавання, орієнтовану на альтернативність подій (подібно системі перевірки орфографії). Сама по собі альтернативність результатів розпізнавання очевидна і обумовлена зберіганням колекцій букв разом з «оцінками відповідності». А словниковий контроль дозволяв змінювати ці оцінки, використовуючи словникову базу. У результаті застосування словника дозволило реалізувати схему дорозпізнавання символів.

Поряд з задачами підвищення точності розпізнавання на передній план вийшло питання розширення сфер застосування OCR-технологій, з'єднання технологій розпізнавання з архівними системами. Іншими словами, ми перейшли від монопрограмми, яка

виконує функції введення тексту, до автоматизованих комплексів, що вирішують задачі клієнта в області документообігу.

*Розпізнавання рукописних текстів*

Очевидно, проблема розпізнавання рукописного тексту значно складніше, ніж у випадку з текстом друкованим. Якщо в останньому випадку ми маємо справу з обмеженим числом варіацій зображень шрифтів (шаблонів), то в разі рукописного тексту число шаблонів незмірно більше. Додаткові складнощі вносять також інші співвідношення лінійних розмірів елементів зображень.

І все ж сьогодні ми можемо визнати, що основні етапи розробки технології розпізнавання рукописних (окремі символи, написані від руки) символів вже пройдені. В арсеналі Cognitive Technologies є технології розпізнавання всіх основних типів текстів: стилізованих цифр, друкованих символів і «рукодрукованих» символів. Але технології введення «рукодрукованих» символів потрібно ще пройти стадію адаптації, після чого можна буде заявити, що інструментарій для потокового введення документів в архіви дійсно реалізований повністю.

Динамічний розвиток нових комп'ютерних технологій (мережеві технології, технології «клієнт-сервер» та ін.) знайшли своє відображення і в стані сектора електронного документообігу. Якщо раніше в просуванні технологій безклавіатурного введення робився упор на переваги їх персонального використання, то сьогодні на перший план виходять переваги колективного та раціонального використання технологій введення і обробки документів. Мати одну, відокремлену систему розпізнавання сьогодні вже явно недостатньо. З розпізнаними текстовими файлами (як би добре вони розпізнані не були) потрібно щось робити: зберігати в базі даних, здійснювати їх пошук, передавати локальною мережею і т. ін. Словом, потрібна взаємодія з архівною або іншою системою роботи з документами. Таким чином, система розпізнавання перетворюється в утиліту для архівних та інших систем роботи з документами.

**Рекомендовані джерела інформації**

**Основна література**

1. Литвин В. В. Інтелектуальні системи : підручник / В. В. Литвин, В. В. Пасічник, Ю. В. Яцишин. – Львів : “Новий Світ – 2000”, 2020. – 406 с.

2. Невмержицький О. В. Аналіз сучасних моделей, орієнтованих на знання, та методів прийняття рішень / О. В. Невмержицький // Інформаційні технології проектування. – №13. – 2013. – С. 119–125.

3. Нестеренко О. В. Інтелектуальні системи і технології. Ввідний курс : навч. посіб. / О. В. Нестеренко, О. В. Ковтунець, О. О. Фаловський. – К. : Національна академія управління, 2017. – 90 с.

4. Нестеренко О. В. Інтелектуальні системи підтримки прийняття рішень / О. В. Нестеренко, О. І. Савенкова, О. О. Фаловський. [За ред. Бідюка П. І.] : навч. посіб. – К. : Національна академія управління, 2016. – 188 с.

5. Шаров С. В. Інтелектуальні інформаційні системи : навч. посіб. / С.В. Шаров, Д.В. Лубко, В.В. Осадчий. – Мелітополь : Вид-во МДПУ ім. Б. Хмельницького, 2015. – 144 с.

6. Ямпольський Л. С. Нейротехнології та нейросистеми : монографія / Л. С. Ямпольський. – К. : Дорадо-Друк, 2015. – 508 с.

7. Ярощук Л. Д. Інтелектуальні системи управління: Експертні системи – основи проектування та застосування в системах автоматизації : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп’ютерно-інтегровані технології» / Л. Д. Ярощук. – Київ : КПІ ім. Ігоря Сікорського, 2019. – 136 с.

**Допоміжна**

8. Береза А. М. Основи створення інформаційних систем : навч. посіб. / А. М. Береза. – К. : КНЕУ, 2001. – 205 с.

9. Глибовець М. М., Олецький О. В. Системи штучного інтелекту. – К. : КМ Академія, 2002. – 366 с.

10.Глинський Я. М. Штучний інтелект. Інтелектуальні роботи / Я. М. Глинський, В. А. Ряжська. – Львів : Деол, 2002. – 168 с.

11.Глібовець М. М. Штучний інтелект : підруч. для студ. вищ. навч. закладів / М. М. Глібовець, О. В. Олецький. – Київ : Видавничий дім «КМ Академія», 2002. – 336 с.

12. ДСТУ 2481–94. Системи оброблення інформації. Інтелектуальні інформаційні технології. Терміни та визначення. – Офіц. вид. – Київ : Держстандарт України, 1994. – 38 с. – (Державний стандарт України).

13. Зайченко Ю. П. Основи проектування інтелектуальних систем : навч. посіб. / Ю. П. Зайченко. – К. : Видавничий дім «Слово», 2004. – 352 с.

14. Кисіль Н. М. Класифікація інформаційних систем / Н. М. Кисіль, З. П. Гаталяк, Н. І. Горбаль // Лісове господарство, лісова, паперова і деревообробна промисловість : Міжвідомчий науково-технічний збірник. – Львів : УкрДЛТУ. – 2004. – Вип. 29. – С. 242–249.

15. Кононюк А. Ю. Нейроні мережі і генетичні алгоритми / А. Ю. Кононюк. – К. : Корнійчук, 2008. – 446 с.

16. Любарський С. В. Методологія вибору моделі подання знань в інтелектуальних навчальних системах / С. В. Любарський, П. В. Шаціло // Збірник наукових праць ВІТІ НТУУ «КПІ». – № 2. – 2010. – С. 65–71.

17. Петров Е. Г. Методи і засоби прийняття рішень у соціальноекономічних системах / Е. Г. Петров, М. В. Новожилова, І. В. Гребеннік. – К. : Техніка, 2004. – 256 с.

18. Руденко О. Г., Бодянський Є. В. Штучні нейронні мережі : навч. посіб. – Харків : ТОВ "Компанія СМІТ", 2006. – 404 с.

19. Субботін С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень : навч. посіб. / С. О. Субботін. – Запоріжжя : ЗНТУ, 2008. – 341 с.

20. Титенко С. В. Проблема подання знань на основі природної мови у освітніх системах штучного інтелекту [Електронний ресурс] / С. В. Титенко // Лабораторія СЕТ. Київ, 2006. – Режим доступу : http://www.setlab.net/?view= Philosophy_Knowledge.

21. Штучна нейронна мережа [Електронний ресурс]. – Режим доступу : https://uk.wikipedia.org/wiki/Штучна_нейронна_мережа.

22. Gupta Jatinder N.D. Intelligent Decision-making Support Systems: Foundations, Applications and Challenges / Jatinder N.D. Gupta, Guisseppi A. Forgionne, Manuel Mora. – Springer Science & Business Media, 2007. – 527 p.

*Навчальне видання*

**ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ**

Конспект лекцій з дисципліни
для здобувачів першого (бакалаврського) рівня вищої освіти
галузі знань 02 «Культура і мистецтво»
Спеціальності 029 «Інформаційна, бібліотечна та архівна справа»
освітньої програми «Інформаційна та документаційна діяльність»

**Укладач:**
Брусенцев Віталій Олександрович, кандидат технічних наук, доцент

***Друкується в авторській редакції***

План 2022
Підписано до друку 25.01.2023. Формат 60x84/16.
Гарнітура «Times». Папір для мн. ап. Друк ризограф.
Ум. друк. арк. 2,10. Обл.-вид. арк.2,20. тираж 100. Зам. №
ХДАК, 61057, Харків-3, Бурсацький узвіз, 4.