МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

Галан Василь Юрійович

«Продукційна система діагностування диспластичних процесів грудної залози / Production system for diagnosing dysplastic processes of the breast»

спеціальність: 123 - Комп'ютерна інженерія
освітньо-професійна програма - Комп'ютерна інженерія
Кваліфікаційна робота

Виконав студент групи КІм-22
В.Ю. Галан
_____

Науковий керівник:
д.т.н., проф. О.М. Березький
_____

Кваліфікаційну роботу допущено
до захисту:

"___" _____ 20___ р.

Завідувач кафедри
_____О. М. Березький

Тернопіль – 2021

ЗМІСТ

ПЕРЕЛІК ПОЗНАЧЕНЬ

ЕС	–	Експертна система
БЗ	–	База знань
СППР	–	Система підтримки прийняття рішень
ШНМ	–	Штучна нейронна мережа
БД	–	База даних
DSS	–	Decision support system
ПО	–	предметна область

ВСТУП

Актуальність теми. В останні десятиліття стрімко розвивалися медичні, біологічні, фармакологічні науки. Це продовжується і зараз, тим більше, в зв’язку з пандемією, що призводить до значного розширення та поглиблення знань про діяльність людського організму, його закономірності, появу нових методів огляду і лікування пацієнтів. Тому зростає значення розвитку медико-технічних наук [1].

Але ще й сьогодні, практична медицина все ще залишається важко формалізованою областю людської діяльності. Досить часто медичні фахівці при прийнятті рішень використовують попередній професійний досвід та власну інтуїцію, а не аналіз об'єктивних даних [2-4]. В таких випадках досить часто припускаються лікарських помилок, які мають надзвичайно високі соціально і економічно значення.

Проблемам комп'ютерної підтримки прийняття рішень задач діагностики традиційно приділяється багато уваги [5-6]. Це пов'язано з постійним збільшенням складності об'єктів діагностики та зростаючою роллю вирішення задач діагностики в сучасному суспільстві. В даний період ця задача стала особливо актуальною через швидкий розвиток таких галугей ІТ як аналіз даних, машинне навчання, штучний інтелект. За допомогою них стала можливою переробка величезних масивів інформації та перетворення їх в бази знань для інтелектуальних розв’язуючих систем. Бази знань дозволяють суттєво підвищити якість діагностичних рішень за рахунок використання сучасних математичних методів.

Для створення діагностичних експертних системи (ЕС) досить наявності бази знань та інтерпретатора. Двома основними проблемами [4-13], які виникають при розробці таких систем, є: 1) складність організації механізму отримання знань від експертів, полягає в необхідності високої кваліфікації експертів (навчання, сертифікація), їх узгодженості, формалізації експертних

знань і фактів, а також подолання ряду суб'єктивних для кожного експерта факторів (наприклад, особистої зацікавленості); 2) великі витрати часу на вирішення задач представлення, перетворення даних в моделі і вибору їх параметрів, а також безпосереднього створення вирішальних правил.

Тому використання нових інформаційних технологій для створення експертних систем медичної діагностики, що поєднують знання і досвід лікарів -експертів, є важливим завданням. Завдання побудови алгоритмів, що дозволяють спростити процедуру наповнення бази знань і прискорити процес побудови ЕС, є актуальним.

Мета і завдання дослідження. Метою роботи є розроблення та програмна реалізація алгоритмів визначення відстані між контурами зображень в нечіткі метриці Фреше для оцінки похибок сегментації.

Мета і завдання дослідження. Метою роботи є розробка продукційної моделі представлення змедичних знань і програмна реалізація модуля експертної системи на прикладі цитологічних зображень раку молочної заози.

Об’єкт дослідження – експертні системи медичної діагностики.

Предмет дослідження – моделі представлення знань.

Задачі дослідження:

– проаналізувати експертні системи медичної діагностики
– формалізувати діагностичні знання
– розробити продукційну модель представлення медичних знань
– розробити компоненти експертної системи
– провести тестування розробленої системи

Методи дослідження базуються на теорії алгоритмів, теорії експертних систем, обєктно-орієнтованому програмуванні.

Наукова новизна одержаних результатів. Розроблено продукційну модель представлення медичних знань.

Практичне значення отриманих результатів. Розроблено компоненти експертної системи для представлення знань диспластичних процесів молочної залози.

Публікації результатів досліджень. За результатами досліджень опубліковані двоє тез доповідей V науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп'ютерні системи та мережі» (2 грудня 2021 р., м. Тернопіль, Західноукраїнський національний університет) [14, 15]:

1. Мачуляк М. В., Галан В. Ю., Іпіроті В. О., Николин І. П. Системи підтримки прийняття рішень в медичній діагностиці. *Інтелектуальні комп'ютерні системи та мережі* : тези доп. V Наук.-практ. конф. молодих вчених і студентів (2 груд. 2021 р.). Тернопіль : ЗУНУ, 2021. С.
2. Галан В. Ю., Мачуляк М. В., Іпіроті В. О., Николин І. П. Моделювання знань в системах медичної діагностики. *Інтелектуальні комп'ютерні системи та мережі* : тези доп. V Наук.-практ. конф. молодих вчених і студентів (2 груд. 2021 р.). Тернопіль : ЗУНУ, 2021. С.

Дипломна робота складається із трьох розділів, висновків, списку використаної літератури та додатків [16, 17].

У першому розділі проведено аналіз експертних систем медичної діагностики.

У другому розділі розроблено продукційну модель представлення медичних знань.

У третьому розділі розроблено компоненти експертної системи .

У додатках приведено довідку про використання результатів дипломної роботи, світлокопію виданих публікацій.

1 АНАЛІЗ ЕКСПЕРТНИХ СИСТЕМ МЕДИЧНОЇ ДІАГНОСТИКИ

1.1 Основні поняття та визначення

Експертні системи – це яскравий і швидко прогресуючий напрям в області штучного інтелекту (ШІ). Причиною підвищеного інтересу, який ЕС викликають до себе впродовж всього свого існування є можливість їх застосування до рішення задач з найрізноманітніших областей людської діяльності. Мабуть, не знайдеться такої проблемної області, в якій не було б створено жодної ЕС або принаймні, такі спроби не робилися б.

ЕС – це набір програм або програмне забезпечення, яке виконує функції експерта при рішенні якої-небудь задачі в області його компетенції. ЕС, як і експерт-людина, в процесі своєї роботи оперує із знаннями [4-10]. Знання про предметну область, необхідні для роботи ЕС, певним чином формалізовані і представлені у вигляді бази знань, яка може змінюватися і доповнюватися в процесі розвитку системи.

ЕС видають поради, проводять аналіз, виконують класифікацію, дають консультації і ставлять діагноз. Вони орієнтовані на рішення задач, що зазвичай вимагають проведення експертизи людиною-фахівцем. На відміну від машинних програм, що використовує процедурний аналіз, ЕС вирішують задачі у вузькій предметній області (конкретній області експертизи) на основі дедуктивних міркувань. Такі системи часто виявляються здатними знайти рішення задач, які неструктуровані і погано визначені. Вони справляються з відсутністю структурованості шляхом залучення евристик, що може бути корисним в тих системах, коли недолік необхідних знань або часу виключає можливість проведення повного аналізу.

Головна перевага ЕС – можливість накопичувати знання, зберігати їх тривалий час, оновлювати і тим самим забезпечувати відносну незалежність конкретної організації від наявності в ній кваліфікованих фахівців.

Накопичення знань дозволяє підвищувати кваліфікацію фахівців, що працюють на підприємстві, використовуючи якнайкращі, перевірені рішення.

ЕС від інших програмних продуктів відрізняється в основному використанням не тільки даних, але й знань. Крім цього ЕС має спеціальний механізм виведення рішень і нових знань на основі тих, що наявні. Знання в експертних системах представляються в такій формі, яка може бути легко оброблена алгоритмічно. Тому в ЕС є відомим алгоритм обробки знань, а не алгоритм рішення задачі. По цій причині застосування алгоритму обробки знань може привести до отримання непередбаченого результату при рішенні конкретної задачі. Більш того, алгоритм оброблення знань заздалегідь невідомий і будується на підставі евристичних правил по ходу розв'язку задачі. Розв'язок задачі в ЕС доповнюється зрозумілими користувачеві поясненнями. Отримувані рішення є якісними і зазвичай не гіршими, а іноді і кращими тих, що досягаються фахівцями. У системах, що грунтуються на знаннях, правила, за якими вирішуються проблеми в визначеній предметній області, зберігаються в базі знань. Проблеми формуються перед системою у вигляді сукупності фактів, що окреслюють деяку ситуацію (рисунок 1.1), і система за допомогою бази знань старається вивести висновок з цих фактів.

Рисунок 1.1 – Схема роботи експертної системи

ЕС буде продуктивною у випадку якщо вона має велику за розміром і якісну базу знань (правил або евристик) [7]. Циклічні режими системи наступні: вибір (запит) даних або результатів аналізів, спостереження, пояснення результатів, засвоєння нової порції інформації, висунення за допомогою правил

тимчасових припущень і потім вибір наступних даних або результатів аналізів (рисунок 1.2). Такі цикли продовжуються до тих пір, поки не поступить інформування, достатнє для вирішального висновку.

Рисунок 1.2 – Схема роботи ЕС

У системі існують три типи знань:

– Структуровані знання – статичні (незмінні) знання про предметну область. Після виявлення знань, вони вже не змінюються.

– Структуровані динамічні знання – змінні знання про предметну область. У міру виявлення нової інформації вони постійно оновлюються.

– Робочі знання для рішення конкретної задачі або проведення консультації.

Всі перераховані типи знань зберігаються в базі знань. Для її побудування необхідно провести опитування фахівців, що є експертами в конкретній предметній області. Після опитування знання потрібно систематизувати, організувати і забезпечити знаками, щоб згодом їх можна було легко витягувати з бази знань.

Експертна система – це інтелектуальний програмний засіб, здатних у ході діалогу з людиною одержувати, накопичувати та коригувати знання із заданої предметної галузі, виводити нові знання, розв'язувати на основі цих знань практичні задачі та пояснювати хід їх розв'язку. Експертні системи акумулюють знання експертів – провідних спеціалістів у даній предметній галузі. В основі роботи експертних систем лежить дедуктивне виведення нових тверджень з існуючих. Типове застосування експертних систем – консультації для фахівців середньої кваліфікації і неспеціалістів у тій галузі, для якої вона розроблена [2, 7].

Можна створювати лише експертні системи, які належать до конкретної предметної галузі. Остання передбачає лімітований набір явищ і понять з певної сфери людської діяльності та обмежене коло задач, які вирішуються в цій галузі. Існує чимало експертних систем у таких сферах життєдіяльності, як медична і технічна діагностика, пошук корисних копалин, юриспруденція, аналіз інвестицій і комерційних ризиків і т. п.

В основі ЕС лежить база знань (БЗ). Знаннями інтелектуальної системи називається трійка $<F, R, P>$, де F - сукупність явних фактів, які зберігаються в пам'яті системи в явному вигляді, R – сукупність правил виведення, які дозволяють на основі відомих знань набувати нових знань, P – сукупність процедур, які визначають, яким чином слід застосовувати правила виведення. Базою знань інтелектуальної системи називатимемо сукупність усіх знань, що зберігаються в пам 'яті системи [2, 7].

У базі знань у деякому закодованому виді зберігаються формалізовані знання експерта. На сучасному етапі розвитку ЕС використовується кілька основних форм представлення знань:

1. "Трійка" об'єкт - атрибут - значення, наприклад: будинок - колір - зелений; пацієнт - температура - висока. Ця форма представлення знань визначає "об'єкт", що володіє деякими атрибутами (властивостями), які можуть приймати значення з відомого набору.

2. Правила продукційних правил у вигляді: ЯКЩО пацієнт хворий грипом І стадія захворювання початкова, ТО температура висока з імовірністю = 0.95 І головний біль є з імовірністю = 0.8.

3. Фрейм. Являє собою іменовану таблицю з деякою кількістю слотів - комірок, що мають свої імена й одержують у процесі роботи машини виводу висновку деякі значення. Як значення можуть бути присутнім константи, посилання на фрейми більш високого чи більш низького рівня, а також деякі обчислювальні процедури.

4. Семантична мережа. Це орієнтований граф, вершини якого відповідають об'єктам (подіям), а дуги описують відносини між вершинами.

Продукційною називається модель знань, в якій база знань складається із сукупності продукцій, тобто правил "Якщо А, тоді В". Термін "продукція" був введений американським математиком Н. Постом. Як і логічні, продукційні моделі насамперед концентруються на дедуктивному виведенні, але є менш формалізованими і тому більш наочними, гнучкими і зручними. На сучасному етапі найбільш вживаним формалізмом для задання знань в експертних системах є саме продукційні моделі.

Типова експертна система, побудована на основі продукційних правил працює за схемою наведеною на рисунку 1.3

Рисунок 1.3 – Типова схема роботи продукційної системи

До робочої пам'яті заносяться факти, що задаються користувачем, а також його запити [2, 18]. Логічний блок зіставляє цю інформацію з правилами, що зберігаються в базі знань, і застосовує ті правила, які можуть бути зіставлені із вмістом робочої пам'яті. Існують дві основні стратегії логічного виведення в продукційних системах: пряма і зворотна [19, 20]. Пряме виведення на основі наявних правил передбачає аналіз: 1) усіх наслідків з фактів, 2) наслідків з наслідків і т. д. Процес продовжується, поки не буде встановлено істинність або хибність запиту користувача. При зворотному виведенні логічний блок починає роботу із запиту користувача, тобто з твердження, яке перевіряється. Розглядається одне з правил, на основі яких можна вивести це твердження, після чого перевіряється істинність лівої частини цього правила. Процес повторюється, доки не дійде до фактів, які вважаються істинними.

Сфери застосування систем, що грунтуються на знаннях, можна згрупувати в декілька основних класів: медична діагностика, контроль і управління, діагностика несправностей в механічних і електричних пристроях, прогнозування, планування, інтерпретація, навчання.

Діагностика в медицині. Такого роду системи використовуються для встановлення зв'язку між порушеннями поведінки організму і її можливими мотивами. Найвідомішою діагностичною системою є система MYCIN. Її призначення – діагностика і спостереження за перебігом хвороби при менінгіті і бактерійних інфекціях. Перша версія MYCIN була розроблена в Стенфордському університеті в середині 70-х років 20 століття [21]. Існує вона і по цей час: система ставить діагноз на рівні лікаря-фахівця. MYCIN має розширену базу знань, і за цією причиною вона може застосовуватися і в інших галузях медицини.

Однією з типових задач експертної системи є задача діагностики. Діагностика - це процес пошуку несправностей в обстежуваній системі (чи визначення стадії захворювання в живій системі), заснований на інтерпретації даних, можливо зашумлених. Знаходження узгоджених і коректних інтерпретацій є основною вимогою в цій задачі. Одна з необхідних умов

досягнення результату - розуміння діагностом структурної організації досліджуваної області та механізмів взаємодії між різними підсистемами.

У задачах діагностики необхідне міркування з припущеннями. У багатьох діагностичних процедурах з успіхом використовуються припущення щодо ступеня надійності датчиків, тобто ступеня надійності інформації, що вводиться. Так само, в задачі діагностики можна зіткнутися з ситуацією, яка змінюється в часі в міру того, як відбувається розвиток хвороби (або у зв'язку з застосованим лікуванням). І нарешті дані, що надходять від датчиків, часто виявляються зашумленими. Це істотний момент у задачі діагностики, де міркування проводяться на підставі результатів вимірювань.

Наприклад, завдання медичної діагностики полягає у виявленні захворювань на основі інтерпретації даних про поточний стан хворого, що утворюються в результаті аналізу скарг пацієнта, його об'єктивного огляду, результатів лабораторних обстежень та аналізів.

Серед задач діагностики найбільш складними є завдання диференціальної діагностики. Їхня складність визначається тим, що серед множини захворювань, що мають спільні ознаки, треба вибрати найбільш імовірні.

Критерій використання ЕС для рішення задач.

Існує ряд прикладних задач, які вирішуються за допомогою систем, заснованих на знаннях, успішніше, ніж будь-якими іншими засобами. При визначенні доцільності застосування таких систем потрібно керуватися наступними критеріями:

1. Дані і знання надійні і не змінюються з часом.

2. Простір можливих рішень відносно невеликий.

3. В процесі рішення задачі повинні використовуватися формальні міркування. Існують системи, засновані на знаннях, поки що не придатні для рішення задач методами проведення аналогій або абстрагування (людський мозок справляється з цим краще). У свою чергу традиційні комп'ютерні програми виявляються ефективнішими за системи, засновані на знаннях, в тих випадках, коли рішення задачі зв'язане із застосуванням процедурного аналізу.

Системи, засновані на знаннях, більш підходять для рішення задач, де потрібні формальні міркування.

4. Має бути принаймні один експерт, який здатний явно сформулювати свої знання і пояснити свої методи застосування цих знань для рішення задач.

У таблиці 1.1 приведені порівняльні властивості прикладних задач, по наявності яких можна судити про доцільність використання для їх рішення ЕС.

Таблиця 1.1 – Критерії застосовності експертних систем

Застосовні	Непридатні
Не можуть бути побудовані строгі алгоритми або процедури, але існують евристичні методи рішення	Є ефективні алгоритмічні методи
Є експерти, які здатні вирішити задачу	Відсутні експерти або їх число недостатньо
По своєму характеру завдання відносяться до області діагностики, інтерпретації або прогнозування	Завдання носять обчислювальний характер
Доступні дані “зашумленні”.	Відомі точні факти і строгі процедури.
Завдання вирішуються методом формальних міркувань.	Завдання вирішуються процедурними методами, за допомогою аналогії або інтуїтивно
Знання статичні (незмінні)	Знання динамічні (міняються з часом)

В цілому ЕС не рекомендується використовувати для рішення таких типів задач [2, 7]:

– математичних, які можуть бути розв’язані шляхом формальних перетворень і процедурного аналізу;

– задач розпізнавання, адже їх можна в основному розв’язати чисельними методами;

– задач, які не мають знань про методи рішення (тоді неможливо побудувати базу знань).

1.2 Технології штучного інтелекту в медичній діагностиці

Ранні експертні системи викликали ажіотаж і призвели до високого рівня очікувань наприкінці 1960-х. і 1970-х років. Проте, кількість необхідної інформації та необхідність строгої організації і обгрунтованих критеріїв рішення для надійного висновку стримувала застосування цих програм. Вони також не забезпечили таку підтримку, яку хотіли лікарі: помічника, а не заміни самого лікаря [2-5].

Ранні експертні системи використовували байєсівської ймовірності і евристичні міркування, які можуть бути описані як емпіричний підхід. В 1970-х роках почалось впровадження експертних системи заснованих на правилах (таблиця 1.2), таких як Mycin, яка використовувала свою базу правил для збору інформації для ідентифікації організмів, що викликають бактеріємію і менінгіт. Багато систем заснованих на правилах були розроблені протягом років, однак, через надзвичайну складність підтримки наборів правил з більш ніж декількох тисяч, системи засновані на правилах історично були присвячені вузьким областям застосування.

Застосування штучних нейронних мереж і генетичних алгоритмів (таблиця 1.2) є однією з останніх тенденцій у розвитку комп'ютерної діагностики. Хоча раніше згадувані СППР є, системами заснованими на знаннях на основі існуючих закодованих медичних знаннях, нейронні мережі і генетичні алгоритми повинні "отримати" свої знання інтерактивно від користувача. Ці типи додатків використовувалися для лікування болю в спині, діагностики раку молочної залози і класифікації гігантоклітинного артриту і гострого інфаркту міокарда.

Оскільки СППР технологія продовжує розвиватися, пристрої, що не використовують такі системи можуть застаріти.

Таблиця 1.2 – Технології створення СППР і експертних систем

Технології	Опис	Використання	Переваги	Недоліки
Системи на продукційних правилах	Знання експертів виражається в наборах правил "якщо - то"	Багато успішних систем реалізовано у вузькоспеціалізованих прикладних областях	Правила легкі для розуміння, дозволяючи системам легко доводити висновки	Важко підтримувати великі множини правил. Погана підтримка невизначеності
Статистичні імовірнісні системи, Баєсівські мережі довіри	Outcome based on statistical analysis і умовній імовірності	Microsoft використовує Баєсівські мережі для медико-інформаційних послуг при вагітності та догляді за дитиною у сервісах MSN	Представляє проблему природнім чином. Чітко керує невизначеністю	Щоб створити систему потрібне знання розподілів імовірності
Нейронні мережі	Використання нейронних вузлів. Вузли та їх взаємодії схожі на обробку в мозку людини	Використовується для розпізнавання образів у сфері епідеміології, радіології, діагностиці раку та інфаркту міокарда	Адаптивні, можуть вчитись на основі нових даних	Важко розробити і добути тренувальні набори даних

Продовження таблиці 1.2

Технології	Опис	Використання	Переваги	Недоліки
Добування даних	Аналізує великі інформаційні системи (сховища даних), щоб знайти тенденції або аномалій	Використовується для пошуку шаблонів в лікуванні і його результатах. Використовується для досліджень з епідеміології, токсикології, і діагностиці	Знаходить і класифікує відповідну інформацію і виявляє тенденції у великих наборів даних	Результат залежить від самих даних
Інтелектуальні агенти, багатоагентні системи	Програмне забезпечення складається з мережі неза-лежно діючих програмних модулів (агентів), які виконують автономні завдання	Використовується для пошуку і отримання відповідної інформації з Інтернету або інших сховищ знань	Ефективна розробка для деяких складних систем і задач	Information sites must be agent enabled. Інформація повинна бути призначена для оброблення агентом
Генетичні алгоритми	Процедури, які імітують еволюцію і природний відбір, щоб вирішити проблему	Використовується в задачах оптимізації, моделювання і розвитку багатоагентних систем, а також гібридних нейромережевих систем	Добре працює для складних багатовимірних задач оптимізації	Не гарантує оптимальне рішення

Продовження таблиці 1.2

Технології	Опис	Використання	Переваги	Недоліки
Нечітка логіка	Системи, що використовують розширення звичайної логіки для оброблення часткової істини	Використовується в мікроконтролерах. Використовується в поєднанні з нейронною мережею	Можна вирішувати проблеми, де ясного значення істинності або ймовірності немає	Занадто складні у використанні, де багатозначна логіка недоречна

Загалом, лікарі більшою мірою готові прийняти системи зосереджені на дуже конкретних областях, ніж ті, що намагається вирішити більш загальні проблеми, такі як діагностика (таблиця 1.3). Хоча навчальні заклади та університети давно розробляють технологію СППР, її прийняття на ринку охорони здоров'я швидко зростає. Сьогодні, СППР успішно використовуються в багатьох галузях промисловості медичних пристроїв, в тому числі контролю серцевої діяльності та автоматизованого ЕКГ, рентгенографії, клінічних лабораторних аналізів, моніторингу дихальних процесів, електроенцефалографії, і анестезії.

Застосування при дослідженні серцевої діяльності – зокрема, аналіз ЕКГ – являють собою основні області, де відбувається підтримка прийняття діагностичних рішень. Вхідна інформація складається з серії запитань, на які повинен відповісти лікар, а також даних, взяті з ЕКГ пацієнта. Вихід є оцінкою ймовірності того, що пацієнт має АСІ. Це приклад статистичної, основаної на ймовірності СППР, яка дозволяє лікарям невідкладної медичної допомоги приймати більш обгрунтовані рішення в критичних ситуаціях, коли швидкість діагностування важлива.

Таблиця 1.3 – Області застосування СППР

Застосування	Опис
Систем оповіщення	Експертні системи під'єднані до моніторів можуть попереджати про зміни в стані хворого. Така система може сканувати результати лабораторних тестів та сповіщати користувачів, або відправляти нагадування і попередження через системи електронної пошти, якщо оцінка даних показує критичні обставини.
Діагностичне обслуговування	СППР може вносити свої пропозиції і допомогти в поставленні діагнозу базуючись на даних про пацієнта.
Системи рецензування і планування	Експертні системи можуть шукати невідповідності, помилки та упущення в поточному плані лікування або порядку медикаментів. Вони також можуть бути використані для розробки схеми лікування базуючись на стані пацієнта і прийнятих рекомендаціях по лікуванню
Розпізнавання та інтерпретація зображень	Експертні системи можуть автоматично інтерпретувати багато медичних зображень - від рентгенівських до більш складних зображень, таких як ангіографії та КТ та МРТ. Це має особливе значення в масовому обстеженні населення, в цьому випадку система може вказати потенційно ненормальні зображення для детального вивчення людиною.

Фірма GE Marquette (Milwaukee) виробляє автоматизовані системи аналізу ЕКГ , які широко використовуються. CardioSys Exercise Testing System дозволяє лікарю контролювати і аналізувати дані пацієнтів, що проходять процедуру тестування навантаження. Це пристрій, а також MAC 5000 Resting Test System, включає в себе програму аналізу Marquette 12SL ECG, інтегрована

СППР, що використовує нещодавно розроблені методи обробки і діагностичні алгоритми для інтерпретації та класифікації сигналів ЕКГ.

Ряд діагностичних ультразвукових систем був розроблений і продається ATL Ultrasound (Bothell, WA) для візуалізації та моніторингу структури серцевої тканин і її діяльності. The system uses an adaptive intelligence algorithm to examine specific tissues by actually optimizing several thousand parameters during a patient examination, thus eliminating irrelevant frequencies in returned signals. Система використовує адаптивний алгоритм для вивчення конкретних тканин оптимізуючи кілька тисяч параметрів у ході обстеження пацієнта, що виключає несуттєві частоти у відбитих сигналах. Більше 10000 систем використовуються в клініках і лікарнях по всьому світу.

Perfex, експертна система заснована на правилах розроблена в Georgia Tech, допомагає в діагностиці серцево - судинних захворювань і в даний час проходить клінічні оцінки. Система виводить ступінь і серйозність хвороби коронарної артерії після інфаркту і виробляє доповідь, що показує стан трьох основних артерій. Perfex була розроблена з використанням Blaze Software's (Mountain View, CA) Nexpert, об'єктно - орієнтованого середовища розробки для експертних систем заснованих на правилах. Амбулаторні монітори кров'яного тиску випускаються багатьма фірмами, як наприклад DynaPulse від Pulse Metric (San Diego) і Omron від Omron Healthcare Inc. (Vernon Hills, IL), використовують розпізнавання образів і алгоритми нечіткої логіки для збільшення точності вимірювань.

Системи для допомогти в ранній діагностиці бактеріального сепсису у новонароджених недоношених дітей знаходиться в стадії розробки у Medical Automation Systems (Charlottesville, VA) у співпраці з дослідниками з медичного центру Університету Вірджинії. A statistical analysis of heart rate variability in ECG data finds abnormal patterns that may help diagnose the disease 12 to 24 hours earlier than is currently possible. Статистичний аналіз варіабельності серцевого ритму в даних ЕКГ полягає у пошуку аномальних шаблонів, які можуть

допомогти діагностувати хвороби від 12 до 24 годин раніше, ніж це можливо зараз.

Рентгенографія і мікроскопія. Співставлення зображень є однією з основних областей застосування для алгоритмів штучного інтелекту. Ці алгоритми здійснюють доступ до бази даних шаблонів і намагаються зіставляти їх з даними пацієнтів, щоб визначити конкретні медичні стани.

Робоча станція Мікроскопії Micro21, розроблена Інтеліджент Медікал Імаджінг (США), виконує автоматизований диференціальний аналіз білих клітин крові, морфологічний аналіз еритроцитів, оцінку тромбоцитів, і оцінку білих клітин крові використовуючи нейромережевий алгоритм для пошуку, класифікації, і відображення як білих та червоних клітин крові.

IRIS Inc. (США) розробила діагностичні системи візуалізації, які включають автоматизовану інтелектуальну технологію мікроскопії для використання в лікарнях по всьому світу. Диференціальний аналізатор лейкоцитів White Iris, є системою автоматизованої мікроскопії, яка виявляє і класифікує лейкоцити в залежності від розміру, кольору та зображення профілю за допомогою цифрових алгоритмів обробки.

Моніторинг основних показників стану людини. Університет Медичного Центру Пенсільванії (Філадельфія) розробив "розумну" системи для відділення інтенсивної терапії, що удосконалює процес моніторингу життєвих показників пацієнтів в критичному стані. Використано поєднання штучних нейронних мереж і нечіткої логіки для перетворення вимірювань життєвих показників пацієнта в легкі для сприйняття візуальні моделі для надання допомоги лікарям та медсестрам в області моніторингу фізіологічних параметрів пацієнта. СППР підключена до пристроїв аналізу з метою представлення в реальному часі оцінки результатів вимірювань для допомоги лікарю (рисунок 1.4).

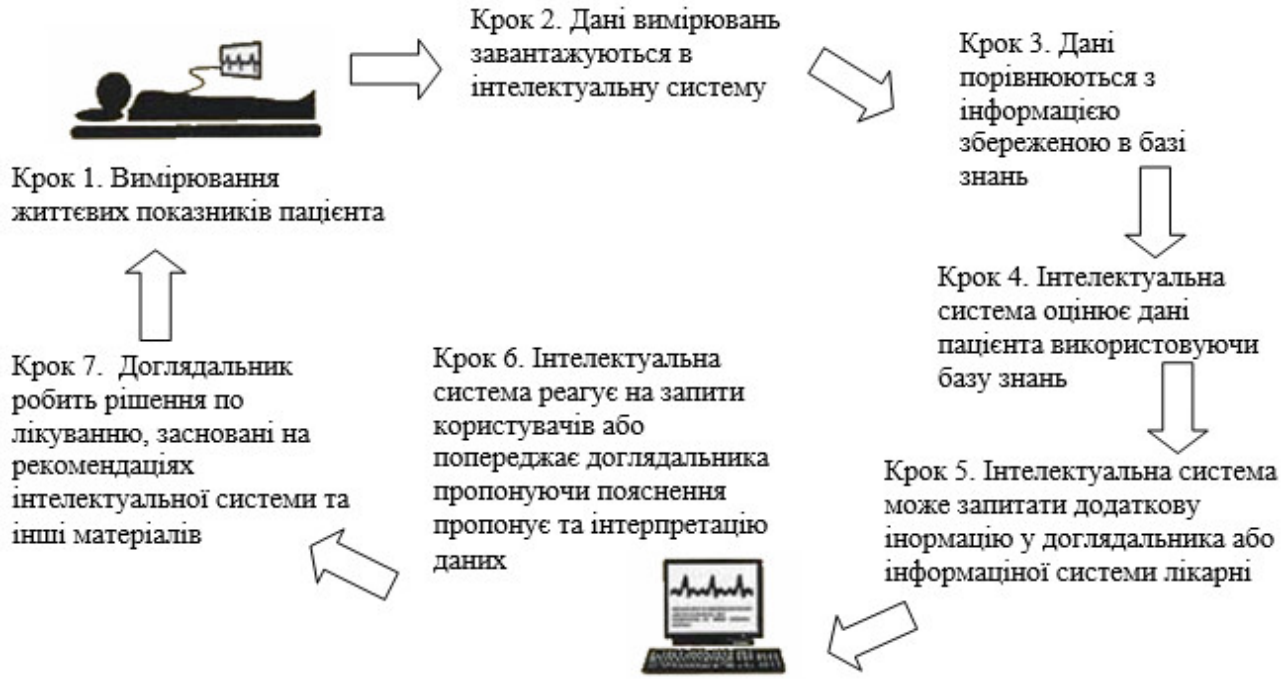


Рисунок 1.4 – Схема організації роботи СППР

Система онлайн – діагностики призначена для надання автоматизованої допомоги в діагностиці захворювань. Медична онлайн – діагностика відноситься до систем експертної класифікації. Використання штучного інтелекту на базі нечіткої логіки дозволяє отримати результати діагностики, за достовірністю близькі до клінічних [22-31]. Достовірність результатів системи на даний момент складає в середньому 58%, (достовірність клінічної діагностики за різними джерелами дорівнює 70-90%) і постійно підвищується [22].

Що експертна система медичної діагностики дає пацієнтам і лікарям? Для пацієнта це список можливих хвороб із зазначенням ймовірності, рекомендації по відвідуванню фахівців та коментарі для поточного стану. Для лікарів - можливість побачити і відразу, по можливості без додаткових питань дати висновок. Медична експертна система призначена тільки для первинного визначення характеру захворювання, коли ще немає можливості проконсультуватися з фахівцем. При першій можливості слід звернутися до лікаря.

1.3 Методи отримання знань

Перший каталог ЕС і інструментальних програмних засобів для їх розробки, був опублікований в США в 1987 році. Він містив більше 1000 систем. На даний момент зараз їх вже набагато більше. Сотні фірм розвинених зарубіжних країн (також і Україна) займаються їх розробкою і впровадженням. Втім вже на вихідних етапах розвитку систем проявилися серйозні принципові труднощі. Вони перешкодили і продовжують перешкоджати ширшому розповсюдженню, сповільнюють і переобтяжують їх розробку. Труднощі природно виходять з самих принципів розробки ЕС [8-13].

Перша трудність зв'язана з постановкою задач. Основна кількість замовників, що планують розробити ЕС, не є достатньо компетентними в питаннях застосування методів ШІ. Вони схильні суттєво перебільшувати сподівані можливості системи. Замовник хоче, щоб ЕС була здатною вирішувати широке коло задач, була самостійно мислячим експертом в досліджуваній області. Через некомпетеність перші постановки задач зі створення ЕС були на кшталт: «Розробити ЕС по обробці зображення»; «Створити медичні ЕС по лікуванню захворювань опорно-рухового апарату у дітей» [13]. Проте для такої загальної постановки задач потужність евристичних методів рішення різко зменшується. Тому рекомендується розробникам, які ще не мають досвіду створення подібних систем, обмежуватися завданням в даній області, для розв’язку яких немає простого алгоритмічного способу. Найкраще, щоб вже на початку існувала методика рішення цієї задачі “вручну” або якими-небудь розрахунковими методами. Для вдалої розробки ЕС необхідно здійснити чітку і конкретну постановку задач і розробити деталізовано (можна й словами) опис “ручного” (або розрахункового) методу її рішення. Якщо таке зробити не вдається, то побудувати ЕС неможливо.

Другою перешкодою є проблема придбання (засвоєння) знань. Ця трудність виникає при “передачі” знань, від експертів-людей до ЕС. Для того, щоб “навчити” експертам комп'ютерну систему, насамперед потрібно схарактеризувати, упорядкувати і формалізувати ці знання “на папері”. На перший погляд це є легкою задачею, але більшість експертів (винятком є можливо математики), які вдало використовують в буденній діяльності свої обширні знання, відчувають великі труднощі при формулюванні і представленні в системному вигляді основної частини: «ієрархію використовуваних понять, евристики, алгоритми, зв'язки між ними» [13]. Для такої формалізації знань необхідно мати системний тип мислення. Цим типом часто володіють математики і програмісти, рідше – юристи і медики. Отже, експерти-люди мають мати знання в області математичної логіки і методів представлення знань, а також орієнтуватися в можливостях комп’ютера, в програмному забезпеченні, мовах і системах програмування.

Отже, виявляється, що для розробки ЕС потрібні особливі фахівці, що володіють вказаною сукупністю знань і “арбітри”, що будуть виконувати функції, між експертами в предметній області і комп'ютерними (експертними) системами. Ці посередники називаються інженерами знань (knowledge engineers). Процес розробки ЕС і інших інтелектуальних програм, що будуються на представленні і обробці знань називається інженерією знань (knowledge engineering). Розвинені зарубіжні країни мають спеціальність “інженер знань” в багатьох вузах. В Україні основи інженерії знань вивчаються в межах спеціалізацій по системному програмуванню. Експерт і інженер знань рідко буває однією особою. В основному інженером знань є розробник ЕС. Як показує досвід більшості розробок, для початкового придбання знань, треба щоб брали активну участь експерти, інженери знань і розробники ЕС, тобто всі три категорії фахівців. Початкове придбання знань може тривати від декількох тижнів до декількох місяців.

Зустрічаються випадки, що на етапі придбання знань виникають труднощі психологічного порядку. Експерт може не передавати всі свої знання

ЕС, вважаючи, що “машина” його замінить і це знищить його престиж як фахівця. Проте ці побоювання позбавлені підстав, адже ЕС добре працює лише в типових ситуаціях, є зручною у випадках, коли людина знаходиться в стресовому стані. В найбільш складних ситуаціях, що вимагають нестандартного мислення і оцінок Ес не може замінити експерта-людину.

Третьою серйозною перешкодою є дуже велика трудомісткість створення ЕС. Необхідно розробити засоби керування базою знань, логічного виводу, діалогову взаємодію з користувачем і т.д. Обсяг програмування дуже великий, а програми складні і нетрадиційні. Тому при розробці великих програм, спочатку створюється демонстраційний прототип системи, який містить в спрощеному вигляді основні плановані можливості. Цей прототип слугує для замовників підтвердженням того, що розробка ЕС для рішення даної задачі принципово можлива. Розробники, в свою чергу, можуть подальше поліпшувати і розвивати систему.

У останнє десятиліття ЕС існують у вигляді систем з базою знань, які тісно переплітались з існуючими діловими системами. Їх використовують в охороні здоров'я, страхуванні, банківській справі і інших областях, щоб за допомогою правил і об’єктів накопичувати досвід, підвищити якість рішень, які приймаються. Бази знань вбудовані сьогодні в найбільш сучасні великі системи. Вони знаходяться в самій серцевині програм-агентів, що здійснюють пошук в мережі Internet, і допомагають колективам користувачів справитися з потоками інформації.

Отримання знань - це процес передачі і перетворення досвіду про рішення задачі з деякого джерела знань в програму [7-9]. Процес отримання схематично зображений на рисунку 1.5.

Процес створення діагностичної експертної системи можна розділити на наступні етапи:

– Налаштування оболонки на конкретну проблемну область, тобто інженер знань спільно з експертом описує основні терміни, поняття; формує

ієрархію понять (типу " загальне - часткове "); визначає структуру основних фреймів, області значення слотів, наслідування властивостей.

– Наповнення оболонки наочними експертними знаннями. Експерт поповнює ієрархію понять конкретними фреймами; встановлює взаємозв'язки між ними; заповнює слоти фреймів.

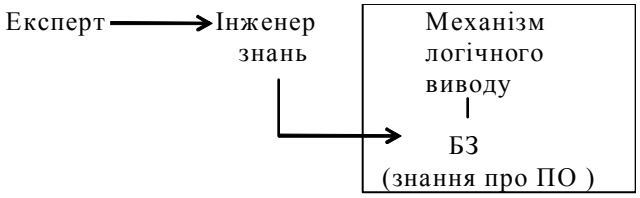


Рисунок 1.5 – Процес отримання знань

У нашому випадку експерт взаємодіє з експертною системою безпосередньо через інтелектуальну редаговану програму (рисунок 1.6). Тобто вся робота інженера знань на цих етапах вже має бути закладена в програму.

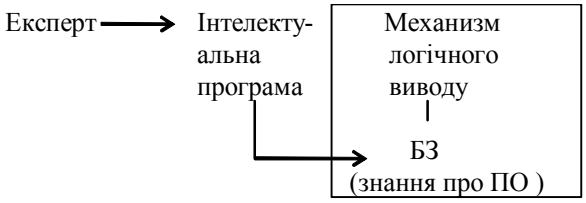


Рисунок 1.6 – Отримання знань у експертній системі

У будь-якій експертній системі можна виділити чотири основних функціональних блоки:

- базу знань;
- блок виведення висновків;
- блок "пояснення" виведення висновків;
- блок наповнення і редагування бази знань.

База знань є важливою частиною експертної системи, якість наповнення якій серйозно впливає на усю роботу системи. Практичні методи отримання знань можна розділити на два великі види: комунікативні і текстологічні.

Основний принцип ділення пов'язаний з джерелом знань. У першому випадку таким джерелом є експерт, а в другому - література, документи, підручники [7]. Саме спосіб текстології витягання знань може бути використаний для створення автоматизованої системи побудови бази знань.

Процес автоматизації прийняття рішення в експертних системах неможливий без залучення інформації, яка не може бути виражена кількісно. Це семантична (смислова) інформація. Таку інформацію можна добути з природно-мовних текстів. У разі медичних експертних систем знання витягаються із спеціальних медичних текстів, зафіксованих оповідань лікарів-експертів про різні прояви захворювань, методів їх лікування і тому подібне [7].

Упродовж останніх двох-трьох десятиліть багато дослідників, що займаються проблемами автоматичної обробки тексту і "розуміння" природної мови, отримали ряд цікавих результатів. Зокрема, можна відмітити виділення в тексті семантичних зв'язків, використання граматик для представлення тексту [7].

При стандартному підході аналіз тексту припускає наявність декількох фаз :

1) морфологічний аналіз;

2) синтаксичний аналіз;

3) семантичний аналіз;

4) прагматичний аналіз.

У літературі зазвичай зустрічається загальний опис цих фаз [3-7]. Це пояснюється трудністю побудови конкретних алгоритмів.

Системи, засновані на знаннях, мають певні переваги перед людиною-експертом:

1. У них немає упереджень.

2. Вони не роблять поспішних виводів.

3. Ці системи працюють систематизовано, розглядаючи всі деталі, часто вибираючи якнайкращу альтернативу зі всіх можливих.

4. База знань може бути дуже великою. Будучи введені в машину один раз, знання зберігаються назавжди. Людина ж має обмежену базу знань, і якщо дані довгий час не використовуються, то вони забуваються і назавжди втрачаються.

5. Системи, засновані на знаннях, стійкі до “завад”. Експерт користується побічними знаннями і легко піддається впливу зовнішніх чинників, які безпосередньо не пов'язані з вирішуваною задачею. ЕС, необтяжені знаннями з інших областей, за своєю природою менш схильні до “шумів”. З часом системи, засновані на знаннях, можуть розглядатися користувачами як різновид тиражування – нового способу запису і розповсюдження знань. Подібно до інших видів комп'ютерних програм вони не можуть замінити людини у вирішенні задач, а швидше нагадують знаряддя праці, які дають йому можливість вирішать завдання швидше і ефективніше.

6. Ці системи не замінюють фахівця, а є інструментом в його руках.

Визначимо вхідні дані для розроблюваної системи медичної діагностики :

– відповіді користувача на питання системи;

– база знань з описами симптомів;

– база знань з описом хвороб;

– таблиця відповідностей між хворобами і симптомами;

– таблиця «ваг» (ймовірностей) симптомів для хвороб;

– база знань з даними про пацієнтів.

Іншими словами, вхідні дані можна розбити на два великі блоки:

– Дані, що поступають з інтерфейсу користувача, куди входить і база даних про пацієнтів.

– Вміст бази знань, яка заповнена експертом.

Вихідними даними є діагноз, побудований на основі спостережуваних симптомів і бази знань про хвороби. Цей діагноз видається на екран як остаточна відповідь експертної системи користувачеві. Крім того, інформація про виявлену хворобу і спостережувані симптоми заносяться в персональну картку пацієнта.

В процесі роботи система генерує декілька робочих версій остаточного діагнозу, і в кінці «відсіювання» зайвих гіпотез, які мають вагу, меншу, ніж деяке значення, що заздалегідь задається системним програмістом.

Наприклад, в процесі роботи сформувалося 5 версій з імовірністю від 67% до 98%. Поріг упевненості, заданий програмістом, – 75%. Тоді система видасть всі версії, імовірність яких більше 75%. Наприклад, їх 3. Система «відсіє» решту хвороб, окрім цих три, і видасть ці три в порядку убування їх імовірності:

У вас, швидше за все, Хвороба 1. Імовірність – 94%

Імовірність Хвороби 2 – 93%

Імовірність Хвороби 3 – 87%

Отже для вирішення поставлених завдань потрібно побудувати підсистему отримання (добування) знань, розробити концептуальну модель діагностичних знань, побудувати продукційну систему представлення знань про хвороби і симптоми. Для успішної програмної реалізації експертної системи медичної діагностики потрібно розробити загальну її архітектуру, обрати інструментальні засоби та операційне середовище, розробити окремі компоненти системи та провести їх тестування.

2 ПРОДУКЦІЙНА МОДЕЛЬ МЕДИЧНИХ ЗНАНЬ

2.1 Підсистема отримання знань

Узагальнена структура розроблюваної ЕС зображена на рисунку 2.1.

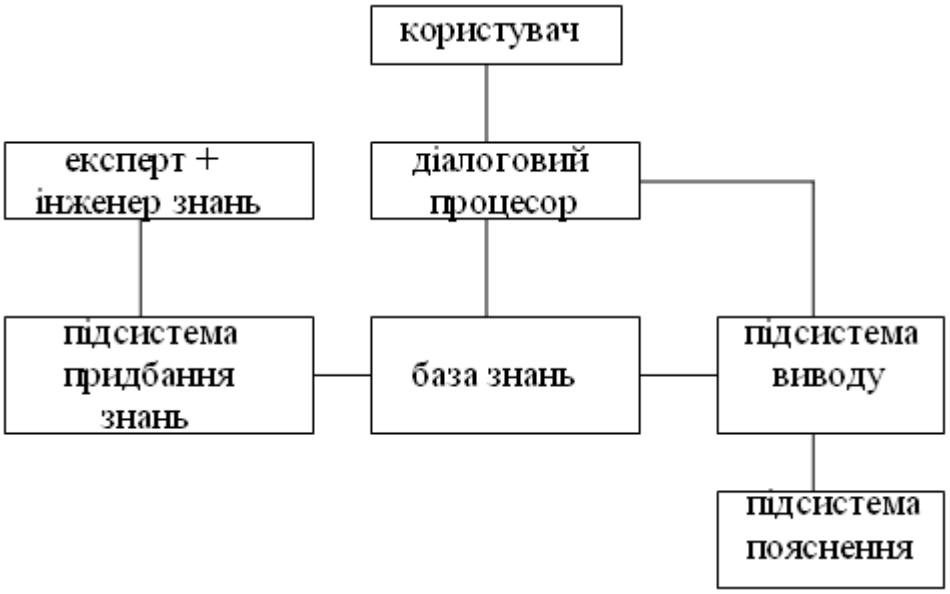


Рисунок 2.1 – Структура експертної системи

Експертні системи мають дві категорії користувачів і два окремі “входи”, відповідних різним цілям взаємодії користувачів з ЕС [7-9]:

1) звичайний користувач (експерт), якому потрібна консультація ЕС - діалоговий сеанс роботи з нею, в процесі якої вона вирішує деяку експертну задачу. Діалог з ЕС здійснюється через діалоговий процесор – спеціальну компоненту ЕС. Існують дві основні форми діалогу з ЕС - діалог на обмеженій підмножині природної мови (з використанням словника – меню (при якому на кожному кроці діалогу система пропонує вибір професійного лексикону експертів) і діалог на основі з декількох можливих дій);

2) експертна група інженерії знань, що складається з експертів в предметній області і інженерів знань. У функції цієї групи входить заповнення бази знань, здійснюване за допомогою спеціалізованої діалогової компоненти

ЕС - підсистеми придбання знань, яка дозволяє частково автоматизувати цей процес.

Підсистема набування знань призначена для додавання в базу знань нових правил і модифікації існуючих. У її завдання входить приведення правила до вигляду, який дозволить підсистемі виводу застосовувати це правило в процесі роботи. Складніші системи мають ще й засоби для перевірки правил, що вводяться або змінюються, коли зустрічаються з суперечністю з наявними правилами.

Добування знань можна розділити на аналіз структури документу, наприклад, Інтернет сторінки, та аналіз власне вмісту. Інтернет джерело містить сукупність параграфів певних типів. Типами можуть бути: визначення, теореми, пояснення, приклади, доведення тощо. Для забезпечення можливості формування структури джерела остання повинна бути побудована за певними правилами. Існують розроблені наступні правила побудови вищезгаданих параграфів. Для відзначення початку і кінця параграфів вирішено використовувати такі конструкції. Початок параграфа відзначається наступним чином:

<A NAME="мітка_початку_параграфа"> < / A >,

де мітка_початку_параграфа представляє з себе рядок, складений з ключового слова, що ідентифікує факт початку і тип параграфа, і рядки, що представляють з себе коротку назву параграфа. Наприклад, <A NAME="startdefХвороба"> </ A>. У даному прикладі ключовим словом є startdef, яке означає початок параграфа типу визначення, в якому визначається поняття " Хвороба ". Всі прогалини в назві параграфа повинні бути замінені на символи підкреслення. Це пов'язано з тим, що деякі засоби генерації і перегляду HTML сторінок не допускають прогалин у параметрі NAME тега <A>. Кінець параграфа відзначається аналогічною конструкцією, з тією тільки різницею, що ключове слова замінюється на інше, ідентифікує кінець параграфа. Вибір подібних

конструкцій заснований на наступних міркуваннях. По-перше, вставка даних конструкцій ніяк не відбивається на зовнішньому вигляді HTML-документа. По-друге, дані конструкції одночасно є мітками параграфів з точки зору HTML, тобто не вводячи ніяких додаткових міток, ми можемо побудувати посилання на будь-який описаний подібним чином параграф. По-третє, використання саме таких конструкцій полегшує побудову гіпертекстового документа, так як багато засобів розробки гіпертекстів, наприклад, Microsoft Word, дозволяють робити в тексті закладки, які перетворюються як раз на подібні теги.

Таким чином, пункт, наприклад, типу визначення, має наступний вигляд:

<A NAME = "startdefХвороба"> </A> текст визначення<A NAME="enddefХвороба"> </ A>.

Якщо в тексті визначення зустрічаються посилання на інші параграфи, вони повинні бути оформлені в наступному вигляді:

<A HREF="startdefСимптом">.

Подібне оформлення параграфів дозволяє побудувати структуру понять з урахуванням усіх наявних зв'язків між ними. Структура електронного джерела формується наступним чином. Весь процес розбитий на два етапи.

Перший етап - перегляд сторінки і складання списку всіх хвороб, побудованих за описаною вище схемою. При цьому для кожного параграфа складається список всіх посилань, виявлених усередині нього, у вигляді імені сторінки плюс безпосередньо імені посилання. Другий етап - аналіз списку симптомів, з метою побудови зв'язків між ними. Аналізуються внутрішні посилання і на їх основі будуються зв'язку між поняттями..

Алгоритми аналізу природно-мовних текстів. Вхідними даними для аналізу служать тексти, отримані в результаті пошуку в мережі Інтернет. Ці тексти сформовані системою пошуку і мають наступну структуру:

- назва хвороби;

- текст з описом хвороби.

Початковими даними для аналізу можуть також бути і тексти з описом хвороб, надані самим користувачем. На файл користувача накладаються

наступні обмеження:

- дотримання структури : назва хвороби - текст з описом хвороби;

- в одному файлі опис однієї хвороби.

Аналіз природно-мовного медичного тексту припускає розбір на симптоми, характерні для вибраної хвороби, і методи можливого лікування.

Структура природно-мовного медичного тексту. Усі природно-мовні медичні тексти, в яких міститься опис хвороб, мають загальну логічну структуру, приведену на рисунку 2.2.

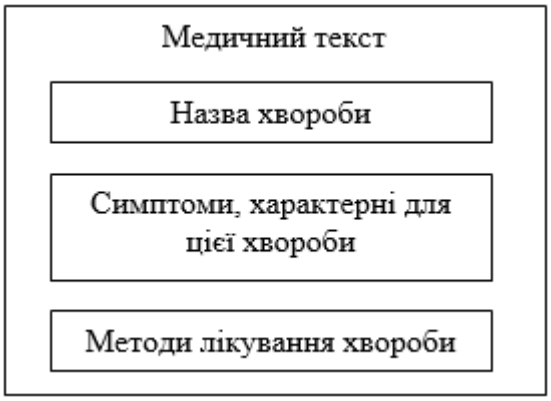


Рисунок 2.2 – Загальна структура тексту, що містить опис хвороби

Алгоритм синтаксичного і морфологічного аналізу тексту. Текст, що має структуру, приведену вище, можна обробити по наступному алгоритму.

Алгоритм аналізу природно-мовного медичного тексту.

1. Отримати вхідний текст.

2. Виділити з тексту заголовок - назву хвороби.

3. Вибрати речення з тексту.

4. Якщо в реченні описуються симптоми хвороби, то знайти в реченні опис симптомів і занести їх у базу знань.

5. Якщо в реченні описується можливе лікування хвороби, то знайти в реченні опис можливого лікування і занести його у базу знань.

Блок-схема алгоритму аналізу природно-мовного медичного тексту приведена на рисунку 2.3.

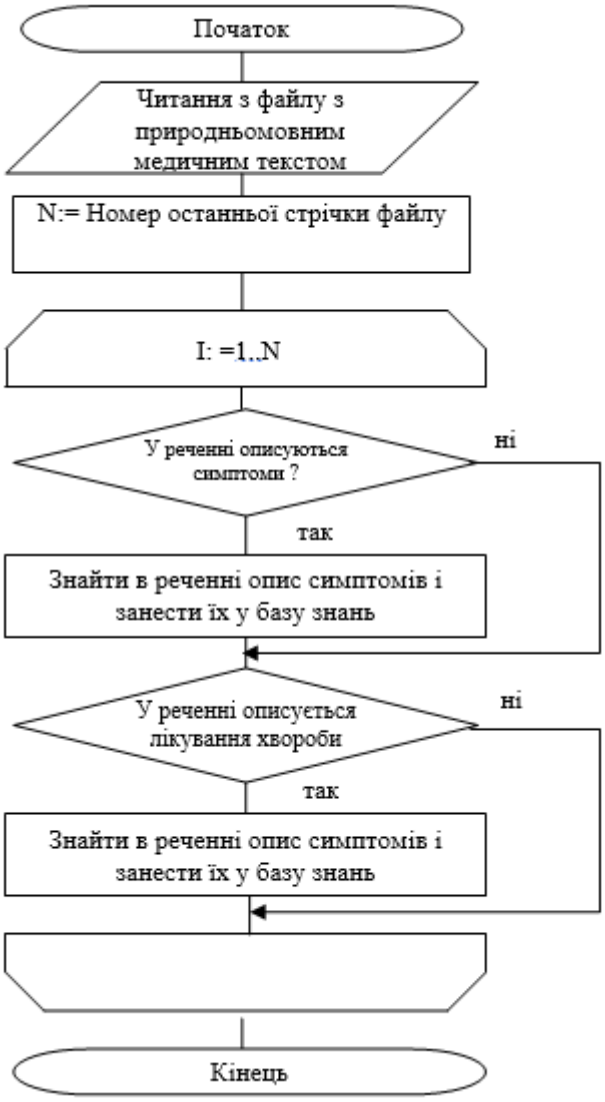


Рисунок 2.3 – Блок-схема алгоритму аналізу медичного тексту

Алгоритми семантичного аналізу тексту можна розбити на дві категорії:

1) виділення симптомів хвороби;

2) визначення шляхів можливого лікування.

Виділення симптомів хворобі здійснюється методом пошуку в реченнях характерних конструкцій для симптомів.

Такими характерними конструкціями можуть виступати наступні слова:

- симптоми;
- характерно;
- діагностика;
- спостерігається;

– помітно;

– з'являються;

– інші.

Алгоритм виділення симптомів представлений наступний. Виділення опису лікування хвороби здійснюється методом пошуку в твердженнях тексту характерних конструкцій для можливого лікування. Такими характерними конструкціями можуть виступати наступні слова:

– лікування;

– застосовуються;

– призначається;

– необхідно;

– інші.

Алгоритм виділення симптомів приведений нижче:

```
ОТРИМАТИ текст з описом хвороби
ДЛЯ усіх речень в тексті РОБИТИ
        ДЛЯ усіх характерних слів із словника для симптомів РОБИТИ
                ЯКЩО в речень є характерне слово ТО
                        Виділити з речення перераховані елементи
                КІНЕЦЬ_ЯКЩО
        КІНЕЦЬ_ДЛЯ
КІНЕЦЬ_ДЛЯ
```

Алгоритм виділення тексту можливого лікування:

```
ОДЕРЖАТИ текст із описом хвороби
ДЛЯ всіх речень у тексті РОБИТИ
        ДЛЯ всіх характерних слів зі словника для можливого лікування РОБИТИ
                ЯКЩО в речення є характерне слово ТО
```

Виділити із речення перелічені елементи

КІНЕЦЬ_ЯКЩО

КІНЕЦЬ_ДЛЯ

КІНЕЦЬ_ДЛЯ

Результатом аналізу природно-мовного медичного тексту є наступні отримані знання:

1) назва хвороби;

2) список симптомів;

3) список можливого лікування.

Отримані знання зберігаються у базі знань. Логічна структура бази знань представлена в таблиці 2.1.

Таблиця 2.1 – Логічна структура БЗ

Назва хвороби 1	Симптом 1	Лікування 1
	…	…
	Симптом M	Лікування L
Назва хвороби N	Симптом 1	Лікування 1
	…	…
	Симптом K	Лікування P

Дана струкутра є базовою і може бути легко модифікована при вузькій спеціалізації ЕС.

2.2 Формалізація діагностичних знань

Перш ніж перейти до побудови концептуальної моделі певної частини БЗ інтелектуальної медичної системи дамо загальну характеристику "знанням про предметну область " [5].

Під суб'єктом предметної області (ПО) будемо розуміти будь-яку систему *S*, здатну реалізувати певний проект з ПО. Приклади систем: ООН, держава, наукове співтовариство, технопарк, організація, інтелектуальна комп'ютерна система і т.д. Життєвий цикл існування системи може обмежуватися одним єдиним проектом. Очевидно також, чим складніше проект, тим складніше повинна бути система, яка його реалізує. Будь-яка складна система складається з елементів і підсистем. Отже, складний проект (глобальний проект) повинен бути перетворений в ієрархію проектів більш низького рівня, що реалізуються елементами системи або підсистемами. Під " проектом " будемо розуміти як глобальний проект, так і всю сукупність підлеглих проектів.

Поняття "проект" може включати в себе найпростішу шкільну задачку, написання / читання статті або книги, похід у магазин за покупками, створення комп'ютерної програми, організацію лікування хворого, розробку складної математичної моделі, вивчення матеріалів інших проектів, проект польоту людини на Марс і т . д. Новий проект може відкриватися в момент укладання господарського, страхового, науково - технічного або іншого договору. Обов'язковим для будь-якого проекту є наявність усвідомлених цілей, етапу планування, а, значить, рішення задач вибору, необхідних ресурсів і найближчого терміну виконання. Цілі проекту задають, у свою чергу, критерії "успішності "реалізації проекту. Критерії включають в себе також і обмеження, що накладаються на процес реалізації проекту. Для конкретності будемо розглядати тільки ті проекти, які реалізовані будь-якою системою *S* та її елементами (за весь термін їх існування) чи включені в план робіт та / або досліджень системи *S* (як відомо, політ людини на Марс заплановано NASA).

Судити про наявність знання і про глибину знання системою ПО будемо, в загальному випадку, за здатністю системи реалізувати той чи інший проект даної ПО. Більше того, будемо вважати, що дві системи мають еквівалентним знанням про ПО, якщо вони здатні потенційно реалізувати одну і ту ж множину проектів. Зв'язування знання з проектом дозволяє забезпечити необхідну цілісність і системність знання. Проте нас цікавить, головним чином, можливість створення еквівалентного знанню системи *S* "машинного знання " (в наведеному вище сенсі), як основи роботи інтелектуальної системи, тому необхідно виділити ті компоненти знання, які можуть служити прообразом "машинного знання ".

Під базовою інформаційною задачею σ_{PS} в рамках проекту *P*, реалізованого системою *S*, будемо розуміти задачу розробки максимально повного інформаційно - документального забезпечення проекту. Без втрати загальності будемо вважати, що базова інформаційне завдання вирішується також системою *S*.

Необхідно відзначити, що задача розробки інформаційно - документального забезпечення в свою чергу також є проектом. У ряді випадків може існувати стандарт (підприємства, галузевої і т.д.) інформаційно - документального забезпечення проекту, що значно полегшує рішення базової задачі.

Під документуванням деякого образу будемо розуміти фіксацію проекцій образу на одну або кілька шкал в цифровому вигляді, які містять значимі для проекту характеристики образу. Зокрема, це можуть бути електронні публікації та описи проектів, комп'ютерні програми, цифрові аудіо та відео матеріали тощо. Документування передбачає, по - перше, осмисленість образу, процесу (тобто виділення смислових та логічних елементів та етапів), по-друге, наявність природної або формальної мови опису образу, процесу і, по-третє, наявність коштів для створення цифрової проекції образу. Наприклад, одна система (лікарня) може мати томограф, а інша ні, отже, перша система володіє великими можливостями документування ходу лікувально - діагностичного

процесу. Як правило, розширення можливостей документування означає більш глибоке розуміння процесів, що протікають у рамках проекту, і, отже, веде до більш якісного виконання проекту.

Документальне забезпечення проекту має відображати всі етапи життєвого циклу проекту, наприклад: технічне завдання, ескізний проект, технічний проект, робочий проект, будь-які інструкції, креслення, схеми, описи математичних моделей, обчислювальних алгоритмів і т.д. Відображення (документальне) інформаційних та технологічних процесів не обов'язково має бути тотожним первинним процесам (найчастіше це принципово неможливо). Головне, щоб були відображені суттєві для проекту механізми.

Інформаційне забезпечення включає в себе всі незадокументовані інформаційні образи процесів, що протікають у рамках проекту, які, однак, можуть бути потенційно задокументовані. Необхідно також відзначити, що підготовка і реалізація деяких проектів ПО може взагалі не припускати будь-якого документального супроводу, проте завжди має місце інформаційний супровід (ціль, планування тощо).

Вербальними знаннями системи *S* про ПО будемо називати сукупність відомостей, що дозволяють максимально повно задокументувати базові інформаційні завдання всіх реалізованих і реалізуються системою проектів ПО. Вербальні знання повинні бути доступні всім елементам системи. Завдяки вербальному знанню різні системи чи елементи систем можуть обмінюватися знаннями між собою.

Невербальним знанням системи *S* будемо називати інформаційне відображення процесів, що сприяють реалізації деяких проектів ПО, але не піддаються документування системою *S*. Прикладом таких процесів можуть служити процеси, що протікають на підсвідомому рівні людини. Невербальні знання можуть утворитися і у випадку, якщо який - небудь елемент системи виконує свою частину проекту з грифом "таємно". Слід зазначити, що в результаті придбання нових знань невербальні знання системи *S* можуть перейти в розряд вербальних. Наявність невербального знання призводить до

утворення "білих плям" при документуванні базових інформаційних задач проектів.

Таким чином, знання системи *S* про ПО є результатом об'єднання вербального і невербального знання. Реалізація нового проекту призводить до появи нового знання у системи S. Знання складної системи залежить від сукупності знань її елементів.

Будемо вважати, що вербальне знання будь-якої системи *S* може бути представлено у вигляді "машинного знання ", відповідної інтелектуальної системи. Однак, в загальному випадку, машинного знання недостатньо для реалізації проектів, аналогічних тим, які реалізує система S.

Інтегральним знанням про ПО будемо називати гіпотетичне знання, яке утворилося б в результаті об'єднання всіх суб'єктів ПО в єдину систему. Підкреслимо, що інтегральне знання могло б проявити себе лише у разі реалізації будь -яких проектів ПО.

У загальному випадку, ніяка система не володіє інтегральним знанням про ПО, зокрема, з огляду на те, що окремо взята система володіє обмеженими ресурсами для реалізації проектів (проекти, що вимагають великих ресурсів, не включаються в план робіт системи, а значить, не утворюється інформаційно - документальне забезпечення, яке складає основу вербальних знань системи про ПО). Відзначимо також, що існує, як правило, безліч механізмів (способів, алгоритмів) реалізації проекту і кожен механізм передбачає свої ресурси. При одних ресурсах система може реалізувати проект, тобто володіє відповідним знанням технології рішення (інформаційно -документальним забезпеченням), а при інших немає. Більш того, один і той же механізм реалізації проекту у різних систем вимагає в загальному випадку різних ресурсів.

В якості ресурсів можуть бути задіяні можливості інших систем. У граничному випадку, за наявності, наприклад, достатніх фінансових коштів деякий суб'єкт ПО може використовувати в якості ресурсів можливості всіх інших суб'єктів ПО і, таким чином, система - " суб'єкт ПО + доступний ресурс " буде володіти інтегральним знанням. Приблизно така ситуація має місце, коли

фахівець використовує потужний інтегральний пакет прикладних програм, наприклад, MATLAB. Дійсно, в цьому пакеті і аналогічних пакетах сконцентровані знання сотень і тисяч суб'єктів ПО, що дозволяє говорити про деяке наближення до інтегрального знання людино - машинної системи "спеціаліст + ЕОМ + пакет MATLAB". Будемо вважати, що саме інтелектуальні інформаційні системи дозволять окремому суб'єкту ПО в максимальному ступені наблизитися до інтегрального знання про ПО.

Проведений вище аналіз показує, що в загальному вигляді базова інформаційна задача σ_{PS} (на будь-якому рівні деталізації або ієрархії) може бути представлена у вигляді кортежу $\sigma_{PS} =< A_1, A_2, \alpha, K, R_\alpha >_{PS}$, де A_1 – опис вихідного стану об'єктів задачі; A_2 – опис результуючого стану об'єктів задач; α - опис оператора (механізму, способу), що здійснює відображення: $\alpha : A_1 \to A_2$, або відношення виду $A_1 \alpha A_2$ (α може виражати, зокрема, суб'єктивні та об'єктивні причинно - наслідкові, родо - видові, ситуативні, функціональні, асоціативні та інші зв'язки між об'єктами задачі); K – опис цілей і критеріїв якості реалізації проекту, включаючи будь-які обмеження; R_α – опис ресурсів, потрібних або виділених для реалізації проекту та документування. Рішення базової інформаційної задачі полягає в тому, щоб визначити всі елементи кортежу.

Приклад опису спрощеного проекту з медицини: < "Пацієнт хворий", " Пацієнт здоровий", "Лікування ", " Тільки терапевтичним шляхом ", " У районній лікарні " >.

Зазвичай в задачі невідомий один або кілька елементів, що дозволяє розділити завдання, наприклад, на діагностичні (A_1?), прогностичні (A_2?), вибору стратегії і тактики управління (A_2, α, K, R_α?), пошуку закономірностей (зокрема, α?) і т.д. Гранично складний випадок задачі має місце, коли невідомий ні один елемент кортежу: σ_{PS}=<?, ?, ?, ?, ?>, наприклад: "на вулиці знайшли лежачу людину в коматозному стані " і проект полягає в тому, щоб повернути людині максимально можливе здоров'я.

Концептуальна модель діагностичних знань. Визначимо основні поняття:

S – довільна фіксована система - суб'єкт ПО.

ω - довільний об'єкт ПО (реальний або абстрактний), який може бути задокументований системою S.

Ω_ω - клас об'єктів, однотипних ω (з точки зору документування), або деяка множина об'єктів, наприклад, агрегат. Будь-який ω може входити в множину різних класів - множин: $\Omega^1_\omega, \Omega^2_\omega, ..., \Omega^n_\omega$. Будь-який клас Ω_ω також є об'єктом ПО. Запис $\Omega.\omega$ буде означати, що розглядається об'єкт ω з класу Ω. Оскільки можуть бути утворені класи класів, то ланцюжок вкладень може бути як завгодно довгою, наприклад: $\Omega_1.\Omega_2...\Omega_n.\omega$.

Можливість входження об'єкта в різні класи передбачає наявність у нього різних імен, алліасів, кодів і т.д., які повинні бути пов'язані між собою. Сукупність усіх кодів об'єкта може бути об'єднана в окремий клас.

Σ - вербальні знання системи S про предметну область (База Знань - БЗ). Приймемо, що будь-яке $\omega \in \Sigma$. Сама Σ також є об'єктом. Згідно з визначенням вербальні знання дозволяють максимально повно задокументувати рішення базової інформаційної задачі будь-якого проекту ПО.

$\sigma =< A_1, A_2, \alpha, K, R_\alpha >$ - документальний опис довільної інформаційної задачі (проекту), вирішеною або розв'язуваної системою S (позначення ті ж). Очевидно, σ є об'єктом БЗ. За допомогою σ можуть бути описані будь-які відображення, проблемні ситуації та зв'язку відомі системі. Будемо використовувати нотацію $\sigma \Rightarrow \omega$, якщо на якомусь рівні опису задачі σ використовується об'єкт ω.

$\sum_\sigma = \{\omega \mid \omega \in \Sigma \wedge \sigma \Rightarrow \omega\}$ – множина об'єктів БЗ, використовуваних для опису задачі σ.

t – час (значення часу t можуть належати до різних порядкових шкал: числових, лінгвістичних і т.д.). Об'єкти ПО можуть з'являтися і зникати, змінювати свої властивості з часом, тому мають сенс записи: ω_t, Σ_t.

Якщо будь-які дві системи S_1 і S_2 володіють власними БЗ - Σ_1 і Σ_2 відповідно, то виникають задачі порівняння ($\Sigma_1 \cap \Sigma_2, \Sigma_1 \setminus \Sigma_2$ і т.д.) і / або

інтеграції (θ - об'єднання) знань різних систем: $\Sigma_1 \theta \Sigma_2$, где θ - різні операції (способи) об'єднання при реалізації тих чи інших проектів (наприклад, операції злиття знань, взаємозбагачення, варіант консиліуму і т.д.). Так, якщо S1,2 – чисто комп'ютерні системи, то теоретично можливе злиття знань. Якщо S1,2-людино - машинні системи, то можливі або консиліум, або взаємозбагачення знаннями. Нас цікавитимуть, в першу чергу, системи "Лікар - IMC ", " Пацієнт - IMC " и " IMC ".

Будь-яка інформація про об'єкти ПО, наприклад, їх властивості та характеристики, параметри управління і т.д., може бути виявлена тільки за допомогою спеціальних тестів. Виконання будь-якого тесту означає реалізацію окремого проекту. Нехай τ_ω - тест, призначений для дослідження ω.

$T_\omega = \{\tau_\omega\}$ - множина всіх тестів, призначених для дослідження ω.

T_Ω - множина тестів, призначених для дослідження класу об'єктів Ω.

У загальному випадку справедливо:

$$T_\omega = \bigcup_{i=1}^{n} T_{\Omega_\omega^i}$$

Приймемо, щ $\forall \omega \in \Sigma$, $\tau_\omega, T_\omega, T_\Omega$ також є об'єктами ПО і, отже, належать Σ.

Будь-який тест τ може мати один або кілька механізмів реалізації, інформаційні моделі (опис) яких позначимо через $\mu_{\tau\nu}$, де $\nu \in N_\tau$. Всю множину механізмів реалізації тесту τ позначимо через M_τ.

При виконанні будь-якого тесту $\tau \in T_\omega$ за допомогою механізму реалізації $\mu \in M_\tau$ повинні бути дотримані певні метрологічні вимоги (умови), які позначимо через $c_{\tau\mu}$.. Очевидно, в конкретній ситуації можуть бути задіяні тільки такі механізми реалізації тесту, які реально здійсненні і для яких здійсненні вимоги метрології.

Якщо необхідно підкреслити, що тест τ реалізується через механізм μ з дотриманням метрологічних вимог *c*, то можуть бути використані наступні нотації: $\tau / \mu, c$; τ / μ ; t/c.

Вибір системою *S* того чи іншого механізму реалізації тесту залежить від ряду критеріїв K_τ, які приймають різні значення в залежності від конкретних обставин β в моделі дійсності. Значення даних критеріїв для конкретного механізму μ і обставин β позначимо через $T_{\tau\mu}(\beta)$. Очевидно, у системи *S* повинен існувати механізм $\mu_{K\tau}$ вибору найкращого (допустимого, оптимального) способу реалізації тесту в конкретних умовах β.

Приймемо, що $\forall \omega \in \Sigma$, $\forall \tau \in T_\omega$, $\mu_{\tau\nu}(\nu \in N_\tau)$, M_τ, $c_{\tau\mu}$, K_τ, $\mu_{K\tau}$ також є об'єктами предметної області і, отже, належать Σ.

З кожним тестом $\tau \in T_\omega$ пов'язана множина питань $q \in Q_\tau$, наприклад: q_1 = "Чи може бути виконаний тест τ (за допомогою механізму μ при виконанні метрологічних умов *c*)?"; q_2 = "Чи проводився тест τ?"; q_3 = "Чи є результати тесту τ?"; q_4 = "Які результати тесту τ?"; q_5 = "Які механізми реалізації тесту τ?"; q_6 = "Які критерії вибору того чи іншого механізму реалізації тесту τ?"; q_7 = "Які чутливість (*Se*) та специфічність (*Sp*) тесту τ?" і т.д.

Відповіді на питання $q \in Q_\tau$ можуть вибиратися або формуватися з різних множин відповідей $D_{q\nu}$ (доменів). Для фіксації того, що в якості множини відповідей на запитання *q* використовується домен *D*, будемо використовувати наступну нотацію: *q*/*D*. У загальному випадку *D* може не бути множиною готових відповідей, наприклад, *D* може бути лексичним деревом [1,2,4] (відповідь формується із стандартних лексем) або множиною всіх фраз природної (професійної) мови. Отже, в загальному випадку можна говорити лише про виводимості відповіді *d* на базі конструкцій множини *D*, що будемо відображати за допомогою нотації: $D \vdash d$. Домени можуть бути неповні, тобто не всі елементи доменів можуть бути відомі. Використовуючи різні домени, можна управляти загальністю (масштабом) відповіді. Приклади доменів:

Температура тіла $\in D_1$ = [35-41]°C

Температура тіла $\in D_2$ = { Нормальна, підвищена, субфібрильна,…}

Біль $\in$ { Сильний, слабкий,…}

Тривалість $\in$ {години}, {дні}, {місяці}, {роки}, {кілька годин, днів, місяців, років;..., недавно; точно не відомо;…}, і т.д.

Нехай $\forall\omega\in\Sigma$, $\tau\in T_\omega$, $q\in Q_\tau$. Універсальну схему дослідження об'єкта визначимо наступним чином:

$t/\mu,c$ q/D ?d, где D ├ d

Якщо використовуються механізм реалізації тесту і множина відповідей, прийняті за замовчуванням, то запис може бути спрощено: τ $q?d$. За замовчуванням може застосовуватися і питання: " Який результат тесту? ". У цьому випадку можна використовувати нотацію: $t?d$ или $q/D?d$

Розглянемо приклади.

Темп. тіла /D_1? 37,5°C

Темп. тіла /D_2? Підвищена

Введемо наступні скорочення: МКБ-10 = "Міжнародний класифікатор хвороб 10-го перегляду", КлінDs = "Клінічний діагноз". Справ_Sp = {Низька, Невизначена, Висока}. Використовуємо дані позначення для запису наступних (спрощених) прикладів:

Ω = Пацієнти, $\Omega.\omega$ = Пацієнти.Петренко, t = Пацієнти.Петренко. Діагноз, $q = \tau$

Пацієнти.Список? Петренко, Сидоренко, Тарасенко,…

Пацієнти.Петренко.Номер_ІБ? 1234

τ Дата встановлення? 12.01.2004

Пацієнти.Петренко. Діагноз /МКБ-10 ? G23

Пацієнти.Петренко. Діагноз /КлінDs? Гостра, лівостороння пневмонія

Пацієнти.Петренко. Діагноз / Лікар Іванова Se? 76%

Пацієнти.Петренко. Діагноз / Консиліум, Лікарня Se? 83%

Пацієнти.Петренко. Діагноз / ІМС Se? 96%

Пацієнти.Петренко. Діагноз / Лікар Іванова Sp? 82%

Пацієнти.Петренко. Діагноз / Лікар Іванова Sp/Справ_Sp? Висока

В останніх п'яти прикладах "Лікар Іванова", " Консиліум " і " ІМС " грають роль різних механізмів реалізації тесту, а " Лікарня " грає роль необхідних умов для реалізації механізму " Консиліум ".

Розглянемо інші приклади.

Скарги? Є

Скарги.Задишка? Є

Скарги.Задишка.Особенності прояви? Постійна

Скарги.Головний біль? Є

За допомогою відступів від лівого краю можна вказувати рівень вкладеності тесту в ієрархії тестів. Це дозволяє скоротити число повторень тестів у протоколі обстеження. Приклади:

Консультація терапевта? Є

Скарги? Є

Задишка? Є

Причини появи? У спокої, після фіз.нагрузкі

Особливості прояви? Періодично

Тривалість? Кілька років

Головний біль? Є

Локалізація? Скронева область, лобова область

Характер прояви? Розпираючий

Відрижки? Не турбують

Анамнез? Є

Жовтяниця? Ні

Хвороби батьків. Рак? Ні

Консультація окуліста? Є

Скарги? Запорошеності, світлобоязнь, набряк повік

Анамнез.давнина? 2 місяці

Об’єктивно? Є

Повіки? Гіперемія, невуси

Склера? Жовта

Корисно ввести спеціальний клас тестів - Тест - агрегат. Тест - агрегат являє собою сукупність елементарних тестів (тестів, не мають підрівнів) і записується таким чином:

ТЕСТ(Парам1, Парам2,…ПарамК)
АН_КРОВІ (Лейкоцити, Еритроцити,…)
Результат тесту - агрегату записується таким чином:
ТЕСТ? Парам1=знач1, Парам2 = знач2,…, ПарамК = значК
АНАЛІЗ СЕЧІ? білок = відс., глюкоза = відс.
ЕКГ? Гіпертрофія лів.шлун.= відс., Вогнищеві зміни = відс.
Або спрощено (якщо визначалися всі параметри):
ТЕСТ? знач1, знач2,…, значК

Кожен Парам1,…, ПарамК являє собою елементарний тест з класу ТЕСТ, тому справедливі записи: ТЕСТ.Парам1? знач1 и т.д.

Для цілей діагностики надзвичайно корисним є клас тестів - Тест - АДД (АДД - Алгоритм диференціальної Діагностики). Даний клас тестів може бути реалізований на основі лексичного дерева [1,2] і спеціально тут не розглядається (це предмет окремої статті).

Конкретизація тесту у вигляді Тесту - агрегату, Тесту - АДД та інших структур є по суті частковим або повним описом одного з можливих механізмів реалізації складного тесту.

2.3 Продукційна модель представлення медичних знань

База знань - найбільш важлива компонента експертної системи, на якій засновані її "інтелектуальні здібності" [13]. На відміну від всіх останніх компонент ЕС, база знань – "змінна" частина системи, яка може поповнюватися і модифікуватися інженерами знань і досвіду використання ЕС, між консультаціями (а в деяких системах і в процесі консультації). Існує декілька способів представлення знань в ЕС, проте загальним для всіх них є те, що знання представлені в символьній формі (елементарними компонентами представлення знань є тексти, списки і інші символьні структури). Тим самим, в ЕС реалізується принцип символьної природи міркувань, який полягає в тому, що процес міркування представляється як послідовність символьних перетворень.

Найбільш поширений спосіб представлення знань – у вигляді продукційної системи з конкретних фактів і правил, по яких з наявних фактів можуть бути виведені нові. Факти представлені, наприклад, у вигляді трійок:

(Атрибут, Об'єкт, Значення).

Такий факт означає, що заданий об'єкт має заданий атрибут (властивості) із заданим значенням. Наприклад, трійка

(Температура, Пацієнт1 37.5)

представляє факт "температура хворого, що іменується Пацієнт1, рівна 37.5". У простіших випадках факт виражається неконкретним значенням атрибуту, а яким або простим твердженням, яке може бути істинним або помилковим,

Продукційна модель один з найчастіше використовуваних в експертних системах способів представлення знань. Основна ідея полягає в асоціюванні з відповідними діями набору умов у вигляді правил типу "якщо-то", званих також продукціями:

ЯКЩО умова ТО дії

"Якщо-То" – правила зазвичай виявляються природним виразним засобом представлення знань. Крім того, вони володіють наступними привабливими властивостями:

– модульність: кожне правило описує невеликий, відносно незалежний фрагмент знань;

– можливість інкрементного нарощування: додавання нових правил

– у базу знань відбувається відносно незалежно від інших правил;

– зручність модифікації (як наслідку модульності): старі правила можна змінювати і замінювати на нових відносно незалежно від інших правил;

– застосування правил сприяє прозорості системи, тобто здібності до пояснення ухвалених рішень і отриманих результатів.

Проте продукційні системи не вільні від недоліків:

– процес виводу менш ефективний, чим в інших системах, оскільки велика частина часу при виводі витрачається на непродуктивну перевірку застосовності правив;

– цей процес важко піддається управлінню;

– складно представити родовидову ієрархію понять.

Розглянемо питання побудови бази знань на прикладі наступних захворювань гортані: ларингіт гострий, ларингіт хронічний катаральний, ларингіт хронічний гіпертрофічний, ларингіт хронічний атрофічний. Кожна хвороба має свій специфічний набір ознак (в медицині – симптомів). Аналіз предметної області зведено в таблицю 2.2, що допоможе зручніше сприймати. У таблиці стовпці – це назви хвороб, а рядки – назви симптомів. На перетині відповідних стовпців і рядків стоїть знак "+", у випадку якщо симптом дійсно належить досліджуваній хворобі. Одні і ті ж симптоми можуть бути ознаками однієї і тієї ж хвороби. По цій причині деякі з симптомів повторюються.

Таблиця 2.2 – Хвороби і їх симптоми

Хвороба	Ларингіт гострий	Ларингіт хронічний	Ларингіт хронічний	Ларингіт хронічний

Симптом		катаральний	гіпертрофічний	атрофічний
1. загальне нездужання	+	+	+	+
2. сухість, першіння	+	+		+
3. кашель спочатку сухий, потім з мокротою	+			
4. голос хриплий або беззвучний	+	+		+
5. іноді біль при ковтанні	+			
6. головний біль	+			
7. підвищення температури тіла	+			
8. швидка стомлюваність голосу		+		
9. періодичний кашель з мокротою		+		
10. охрипліть			+	
11. відчуття незручності			+	
12. біль в горлі			+	
13. кашель при загостренні			+	
14. сухий кашель				+
15. слизова оболонка покрита густим слизом				+
16. відкашлювання з прожилками крові				+

Для визначення хвороби можна побудувати правило вигляду

ЯКЩО симптом$_1$ І симптом$_n$ ТО хвороба

Врахування імовірності діагностованої хвороби. Визначимо вагу як імовірність тієї або іншої хвороби у відсотках. Зберігати ваги симптомів для хвороб будемо у таблиці. Недовизначена специфікація – набір даних, на основі якого неможливо ухвалити остаточне рішення. Чинник упевненості – чинник упевненості демонструє ступінь упевненості системи в достовірності зроблених нею логічних висновків. Поріг упевненості – число, що заздалегідь визначене користувачем, означає максимальну вагу хвороби в даному випадку, нижче за яке гіпотези просто не розглядаються.

На вхід системи спочатку поступає неповна інформація, унаслідок чого система не може однозначно поставити діагноз. Для вирішення даної проблеми використовується наступний алгоритм:

1) Збір попередньої інформації.

Первинний крок. При вході в систему користувач бачить перед собою список всіх симптомів, наявних в базі. Напроти вибіркових пунктів списку симптомів користувач виставляє “помітки” (ті симптоми, які він у себе спостерігає). Натиснувши кнопку “Далі” система переходить до наступного пункту.

2) Складання первинного списку хвороб, до яких підходить даний набір симптомів.

Після введення користувача деяких первинних симптомів, система аналізує, до яких хвороб належать дані симптоми. Алгоритм визначення первинного списку хвороб наступний:

1) Відкриваємо таблицю відповідностей

2) Для всіх k=1 до максимального числа симптомів:

3) Беремо k -тий симптом з первинного списку;

4) Дивимося в таблицю відповідностей: простим циклом робимо повний перебір всіх елементів (хвороб), що стоять в стовпці даного симптому;

5) Робимо перевірку:

5.1) Якщо дана хвороба вже є в списку, то переходиться до кроку.6;

5.2) Якщо елемент таблиці =0, то хворобу в список не включаємо;

6) Збільшуємо k на 1;

7) Якщо всі симптоми (k) перебрані, то первинний список хвороб сформований;

8) Закриваємо таблицю відповідностей.

Кожен симптом може належати відразу декільком хворобам, тому в первинному списку хвороб хвороб буде не стільки ж, скільки симптомів. Слід також врахувати, що симптоми є значущі і незначущі, тобто вага симптому по відношенню до якої-небудь хвороби або велика, або мала. Даний процес регулюється системою, тому ніяк не залежить від користувача.

3) Уточнення інформації.

Маючи початковий список хвороб, система проводить їх диференціацію. Далі система починає проводити “міркування”.

Найпоширеніші методи логічного виводу - це прямий ланцюжок міркувань (прямий вивід) і зворотний ланцюжок міркувань (зворотний вивід). В основному, при вирішенні завдань діагностики використовується зворотний вивід. Можна сказати, що зворотний вивід ефективніший, коли користувач повинен вибирати з набору можливих наслідків як у разі медичної або технічної діагностики. У системі, що розробляється, реалізується механізм змішаного виводу, який дозволяє і прямий вивід від фактів до висновків, і зворотний - щоб підтвердити або спростувати гіпотезу.

В процесі уточнення інформації система, ставлячи користувачеві питання, проводить «відсіювання» зайвих гіпотез, що мають малу вагу. Для прорахунку ваги гіпотез система відкриває дані з файлу на диску, а саме таблицю вагів. Таблиця ваг розміром [К-ть хвороб] на [К-ть симптомів] має в перетині кліток число, рівне вазі даного симптому для даної хвороби; поріг упевненості заздалегідь задається в налаштуваннях.

4) Рекомендації і збір додаткової інформації.

Якщо користувач не зміг відповісти на деякі питання на етапі первинного опиту, то система дає рекомендації, як можна зібрати ці дані, (здати аналізи, провести ЕКГ) і на основі цього збирає додаткові дані. Рекомендації система

дає тільки на гіпотези, що мають велику вагу (щоб підтвердити їх ваговитість, і щоб пацієнтові не варто було здавати зайвих аналізів).

5) Ухвалення остаточного рішення.

В процесі попередніх кроків виявляється декілька версій остаточного результату, які система розподіляє по порядку зростання імовірності тієї або іншої хвороби.

Імовірність хвороб також рахуються по таблиці ваг.

Алгоритм підрахунку ваги:

– вибирається хвороба із списку хвороб, сформованого на попередніх етапах;

– система переглядає, які симптоми із списку симптомів мають відношення до даної хвороби;

– відбувається підсумовування ваг всіх симптомів, що мають відношення до даної хвороби (знову ж таки по таблиці ваг);

– запам'ятовування кінцевої ваги хвороби.

Після підрахунку ваг усіх хвороб вибирається хвороба, що має максимальну вагу, і відбувається нормування ваг хвороб (щоб вони були в межах від 1 до 100). Далі система вибирає ті хвороби, імовірність яких знаходиться в деяких рамках, заздалегідь визначених системним програмістом (так званий “поріг упевненості”). Значення порогу упевненості можна задати в налаштуваннях програми.

Формула для підрахунку ваги хвороби: $vaga\_b(j) = \sum[tab\_ves(ves(i,j))]*k(i)$

Тобто, для кінцевого підрахунку імовірності не застосовується множення ваги на коефіцієнти, а йде підсумовування всіх елементів таблиці ваг (tab_ves(ves(i,j)) що мають відношення до хвороби. Далі відбувається нормування всієї кінцевої імовірності з метою “укладання” їх в проміжок від 1..100. (Щоб остаточна відповідь вимірювалася у відсотках). k (i) – коефіцієнт присутності симптому (рівний або «0», або «1»).

Приклад:

	Голова	Провали	Часті	У вусі	Щелепу	Внутрішньочерепний

	болить	пам'яті	припадки	стріляє	зводить	тиск
Склероз	10	106	64	55	20	43

Наприклад, в процесі роботи були вибрані симптоми 2, 4 і 6. Система підсумовуватиме вагу:

Vaga_b("склероз")=10×0+106×1+64×0+55×1+20×0+43×1=204.

Далі:

У всіх хвороб таким чином буде обчислена вага.

Вибереться максимальна вага хвороби із списку хвороб.

```
J=–1;                    // від’ємне значення спочатку.
for i=1 to 15 do
( if j>max(vaga_b(i))    // якщо j більше максимального значення
j=max(vaga_b(i));        // то j=max.
```

Допустимо, максимальна вага = 300.

Ваги всіх хвороб розділяться на 300, щоб бути в межах від 0 до 1.

Таким чином, вага хвороби “склероз” = 204/300=0.68 (тобто 68%). Це і є кінцевий висновок.

6) Видача кінцевого результату.

Система видає ті хвороби, які були вибрані в попередньому пункті в порядку процентного убування, на екран.

Приклад:

У вас, швидше за все, Склероз. Імовірність – 94%

Імовірність фізичного пошкодження мозку – 93%

Імовірність простої втрати пам'яті – 87%

В даному випадку, поріг упевненості, встановлений програмістом, складає 92%, тому всі хвороби, що мають ваги, не перевищують планки 92%, просто відсіваються.

3 ЕКСПЕРТНА СИСТЕМА ДІАГНОСТУВАННЯ

3.1 Розробка архітектури експертної системи

Система, структура складається з 6 компонентів: бази знань, машини логічного виводу висновку, блоку інтерфейсу користувача, блоку пояснення висновку, блок отримання знань, і робочої області у вигляді дошки оголошень. Дошка оголошень (робоча область) – частина бази даних експертної системи для взаємодії компонентів виводу [7].

Діалог з експертною системою. Для взаємодії з системою новачку (непідготовленому користувачу) потрібно ввести засоби спілкування на природній мові. Основна кількість систем, що грунтуються на знаннях, володіють достатньо примітивним інтерфейсом на природній мові. Для користувача допускаються вхідні повідомлення, які містять набори понять з бази знань.

Під час консультації діалог веде система, а сама консультація у ЕС виглядає як і консультація у людини-експерта: задається перелік питань і на підставі їх аналізу видається експертний висновок. Втім на відміну від бесіди з спеціалістом, діалог з ЕС має свої психологічні особливості: основна кількість користувачів (через відсутність досвіду роботи на комп'ютерах, лаконічність діалогу з ЕС, відсутність пояснень в ході консультації і ін.) менше довіряє “висновку” ЕС, ніж висновку “живого” експерта.

Щоб переконатися в правильності рішень ЕС, користувач може звернутися до її підсистеми пояснення.

Процес логічного виводу в ЕС найчастіше представляється у вигляді схеми, яка називається деревом виводу. ЕС будує дерево виводу в процесі консультації і зберігає його в пам'яті в певній внутрішній формі. Правило буде успішно застосоване, якщо йому відповідає додавання вузла з його ім'ям,

нащадками цього імені будуть вузли, які відповідають деяким з вже виведених фактів, а предком буде новий вузол, який відповідає факту, що міститься в правій частині правила.

Розглянемо поведінку підсистеми пояснення. В процесі консультації користувач може захотіти отримати пояснення. Для цього, в процесі діалогу, в той момент, коли система ставить чергове запитання, запитати її чому вона задала саме таке запитання. Тобто, замість відповіді на запитання системи, користувач ставить своє зустрічне запитання, чому система це питає?

Оте “Чому?” система інтерпретує у своїх термінах дерева виводу. Вона піднімається по дереву на один ярус вище і знаходить правило, для застосування якого система ставить це запитання. А користувачу система видає інформацію про це правило, про стан обчислення його умов про висновок даного правила.

Така модель пояснення застосовується в системі MYCIN. До переваг її можна віднести – можливість одержання пояснення будь-якого кроку роботи системи, недоліком – жорстка прив'язка до дерева виводу. Для користувача на рівні «чайника» такий спосіб пояснення може здаватися надмірним і дуже формальним. Варто було б пояснення давати на “вищому рівні” без занурення в деталі.

Одним з важливих питань, що виникають при проектуванні керуючої компоненти систем, заснованих на знаннях, є вибір методу пошуку рішення, тобто стратегії виводу. Від вибраного методу пошуку залежатиме порядок застосування і спрацьовування правив. Процедура вибору зводиться до визначення напряму пошуку і способу його здійснення. Процедури, що реалізовують пошук, зазвичай вбудовані в механізм виводу, тому в більшості систем інженери знань не мають до них доступу і, отже, не можуть в них нічого змінювати по своєму бажанню.

При розробці стратегії управління виводом необхідно відповісти на два питання:

1. Яку точку в просторі станів прийняти як початкову? Річ у тому, що ще

до початку пошуку рішення система, заснована на знаннях, повинна певним чином вибрати початкову точку пошуку – в прямому або зворотному напрямі.

2. Як підвищити ефективність пошуку рішення? Щоб добитися підвищення ефективності пошуку рішення, необхідно знайти евристики рішення конфліктів, пов'язаних з існуванням декількох можливих шляхів для продовження пошуку в просторі станів, оскільки потрібно відкинути ті з них, які свідомо не ведуть до шуканого рішення.

Механізм логічного виводу складається з двох компонент: одна компонента реалізує висновок, інша керує цим процесом. Компонента висновку розглядає присутні правила та факти з робочої множини і додає до висновку нові факти при спрацьовуванні якого-небудь правила – тобто вона виконує першу задачу. Друга компонента, та що керує, визначає порядок застосування правил, а також визначає, чи є ще факти, які можуть бути змінені у разі продовження консультації.

Компонента, що керує, виконує 4 функції зіставлення, вибору, спрацювання, дії.

1) зразок правила порівнюється з наявними фактами;

2) якщо в деякій ситуації можуть застосуватися відразу декілька правил, то з них вибирається одне, те найбільше підходить до заданого критерію (розв’язування конфлікту);

3) якщо зразок правила при зіставленні збігається з деякими фактами з робочої множини, то правило спрацьовує.

4) робоча множина змінюється. До неї додається висновок правила, що спрацювало. Якщо в правій частині правила міститься вказівка на якусь дію, то вона виконується (як, наприклад, в системах забезпечення безпеки інформації).

Інтерпретатор правил працює циклічно. У кожному циклі він перебирає всі правила, для виявлення серед них тих посилок, які збігаються з відомими на той момент фактами з робочої множини. Також інтерпретатор визначає порядок застосування правил. Після вибору правило спрацьовує, його висновок заноситься в робочу множину, і потім цикл повторюється спочатку.

У одному циклі може спрацювати тільки одне правило. Якщо зіставлення з фактами пройшло успішно за декількома правилами, то інтерпретатор проводить вибір за певним критерієм єдиного правила, яке й спрацьовує в даному циклі. На рисунку 3.1 схематично представлено цикл роботи інтерпретатора.

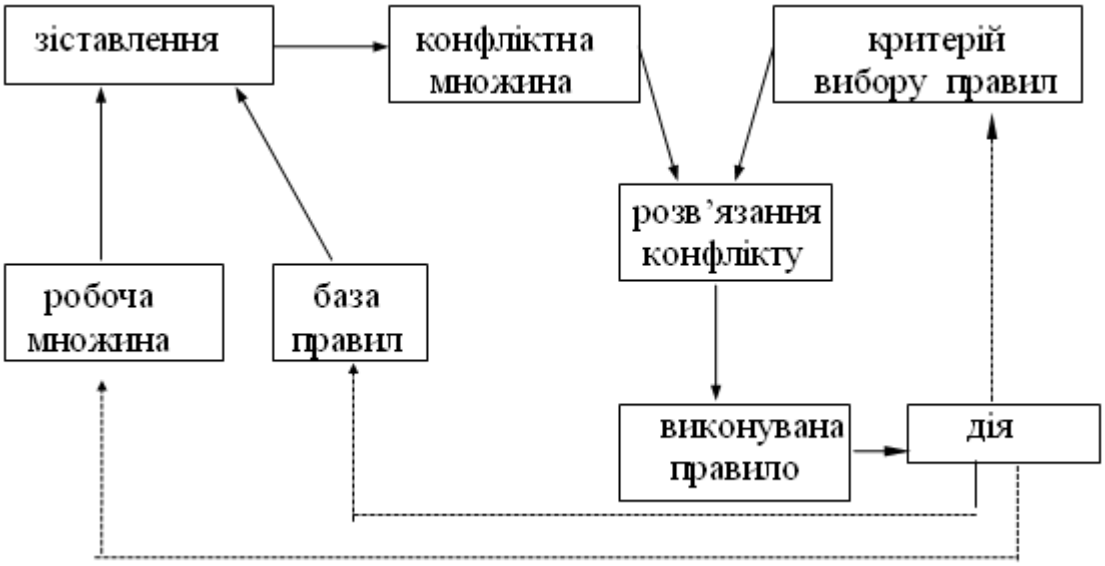


Рисунок 3.1 – Цикл роботи інтерпретатора

Для виявлення успішного зіставлення інформація з робочої множини послідовно порівнюється з посилками правил. Сукупність відібраних правил утворює так звану конфліктну множину. Для розв'язку конфлікту інтерпретатор має критерій, за допомогою якого він вибирає єдине правило після чого воно спрацьовує. Факти, які створюють висновок правила, заносяться в робочу множину або змінюється критерій вибору конфліктуючих правил. Якщо у висновок правила входить назва якої-небудь дії, то вона відповідно виконується.

Нові дані, введені в систему правилом, що спрацювало, у свою чергу можуть змінити критерій вибору правила. Зміна критерію грунтується на висновках, отриманих після аналізу стану в робочій множині системи, а також правил гри (статичних структурних знань) і структурних динамічних знаннях (евристиках).

При розробці компоненту механізму (підсистеми) виводу, що управляє, необхідно вирішити питання про те, по якому критерію слід вибирати правило, яке буде застосовано в конкретному циклі. Вже на ранній стадії розробки ЕС необхідно знати, що вводитиме кінцевий користувач. Це потрібно для того, щоб переконатися, чи буде система достатньо практична і чи зможе вона ужитися в середу, в якій їй належить працювати.

Участь користувача виражається в наступному:

– конкретні задачі. Користувач, стикаючись з конкретними проблемами, може пояснити виникнення проблем і запропонувати можливі варіанти їх рішення;

– спілкування. Інтерфейс користувача повинен відповідати словнику користувача і рівню його підготовки;

– встановлення зв'язків. Знайомство користувача з причинами і наслідками, що викликають те або інша дія в процесі функціонування системи, неоцінимо у визначенні взаємозв'язків фактів в базі знань;

– зворотний зв'язок. Відмітною особливістю зручною у використанні ЕС є її здатність пояснити кінцевому користувачеві хід своїх міркувань.

3.2 Компоненти системи

Правила виводу (породжуючі правила, продукції) являють собою умовні вирази в різних формах:

ЯКЩО виконуються умова ТО висновок;

ЯКЩО ситуація ТО дія;

ЯКЩО виконуються умови 1 І умова 2 ТО умова 3 не виконується.

Правила виводу дозволяють надати пояснення послідовності міркувань та надати пояснення передумов конкретної дії.

Перш ніж сформувати правило потрібно описати факти, що будуть

використовуватись в умові правила. Потрібно виділити об'єкти і їх властивості, або якщо дано ситуацію то виділити стани.

Наступним кроком після формування бази знань потрібно сформувати процедури виводу висновку. Для реалізації правил виводу використовуємо операторний запис мови Prolog [18, 32-36]. Створюємо логічні конструкції "if", "then", "and" й "or" як оператори, оголошені відповідним чином, як показано нижче. Спостережувані факти будуть вводитися як завдання для експертної системи за допомогою процедури symptom.

```
% Оголошення операторів
:- op( 800, fx,  if) .
:- op( 700, xfx, then).
:- op( 300, xfy, or).
:- op( 200, xfy, and).
% Спостережувані факти
:- dynamic fact/1.          % можливість додавати нові факти в БД
% задаємо базу занньь у вигляді правил виводу
 symptom("загальне нездужання").
 symptom("сухість, подразнення").
 symptom("кашель спочатку сухий, потім з мокротою").
 symptom("голос хрипкий або беззвучний").
 symptom("іноді біль при ковтанні").
 symptom("головний біль").
 symptom("підвищення температури тіла").
 symptom("швидка стомлюваність голосу").
 symptom("періодичний кашель з мокротою").

if      symptom("загальне нездужання") and
        symptom("сухість, подразнення") and
        symptom("кашель спочатку сухий, потім з| мокротою") and
```

symptom ("голос хрипкий або беззвучний") and
symptom("іноді біль при ковтанні") and
symptom("головний біль")
and symptom("підвищення температури тіла")
then "ларингіт гострий".

if symptom(1, "загальне нездужання") and
symptom(2 "сухість, подразнення") and
symptom (4 "голос хрипкий або беззвучний") and
symptom(8 "швидка стомлюваність голосу") and
symptom(9 "періодичний кашель з мокротою")
then "ларингіт хронічний катаральний".

```
% Інтерпретатор правил зворотного логічного виводу
% можна визначити як процедуру is_true( P)
is_true( P):-
   symptom ( P).
is_true( P):-
   if Condition then P,  %P правило, що задовольняє
   is_true( Condition).  %
is_true( P1 and P2) :-
   is_true( P1) , is_true( P2).
is_true( P1 or P2) :-
   is_true( P1) ; is_true( P2).
```

Процедура is_true складається з чотирьох правил. Перше правило має за умову тільки один факт (Це умова виходу з рекурсії). Друге правило відшукує істинну умову і є входом в рекрусію. Третє правило істинне якщо істинні обидві його умови (логічне І). Четверте правло істинне якщо істинна одна з його умов (логічне АБО). Тепер цей інтерпретатор можна викликати на виконання за допомогою такого питання:

?- is_true(switch_is_plugged_in).

Ітерпретатор прямого логічного виводу. Іноді більш природним способом є формування міркувань у напрямку від частини "if" до частини "then". Прямий логічний вивід починається не з гіпотез, а з деяких підтверджених фактів. Виявивши, що блимає індикатор “Link” на мержевому концентраторі, можна зробити висновок, що порт коректно підключено до мережі.

В якості *умови* може бути заданий будь-який вираз І/АБО. Інтерпретатор починає свою роботу з того, що відомо (задано у відношенні fact), виводить всі *висновки*, які випливають із цього відношення, і додає (за допомогою предиката assert) ці висновки до відношення fact. В наведенії нижче программі використана наступна термінологія: Forward – вперед, Derived – виведений, Condition – умова, Conclusion – висновок.

```
forward :-
  new_derived_fact( P),        % Новий виведеней факт
  !,
  write( 'Derived: '), write(P), nl,
  assert( fact( P)),           % Додати новий факт в БД
  forward                      % Продовжити роботу
  ;
  write( 'No more facts').  % Процес породження правил завершений

new_derived_fact( Concl):-
  if Cond then Concl,          % Деяке правило
  not( fact( Concl)),          % Висновок правила щн не є фактом БД
  composed_fact( Cond).              % Довести істинність умови

composed_fact( Cond):-
  fact( Cond).                                  % Простий факт в БД
composed_fact( Cond1 and Cond2):-  % Обидві умови істинні - логічне І
  composed_fact( Cond1),
  composed_fact( Cond2).
```

```
composed_fact( Cond1 or Cond2):-   % Одна з умов істинна - логічне АБО
  composed_fact( Cond1);
  composed_fact( Cond2).
```

Система користувацького інтерфейсу забезпечує взаємодію між експертною системою і користувачем. Ця взаємодія зазвичай включає декілька функцій:

1. Обробка даних, отриманих з клавіатури, і винесення вхідних і вихідних даних на екран монітора.

2. Підтримка діалогу між користувачем і системою.

3. Розпізнавання ситуації нерозуміння між користувачем і системою.

4. «Дружні» відносини з користувеч.

Для забезпечення найдоступнішої роботи з програмою розроблена ієрархічна структура меню. Система інтерфейсу з користувачем зобов'язана ефективно обробляти ввід і вивід. Для цього необхідно обробляти дані, що вводяться і виводяться, швидко, ясно і виразно. Необхідно також включити можливість роботи з додатковими засобами: магнітними дисками, USB-флеш, додатковими файлами даних.

Окрім цього, система інтерфейсу має підтримувати відповідний діалог між користувачем і системою.

Консультація повинна завершуватися ясним висновком, що видається системою, і поясненням послідовності виводу, що привела до цього висновку.

Структурна схема системи меню представлена на рисунку 3.2.

Розробка процедур оболонки.

1. Процедура "Завантаження": завантаження БЗ проводиться шляхом завантаження файлу з ім'ям, вказаним користувачем, в оперативну пам'ять для використання його вмісту в програмі.

Рисунок 3.2 – Структурна схема меню

2. Процедура "Збереження": збереження БЗ означає запис її вмісту з оперативної пам'яті у файл з ім'ям, вказаним користувачем. Ця процедура дозволяє зберегти зміни, проведені з БЗ під час роботи.

3. Процедура "Додавання": додавання ділиться на додавання симптомів і додавання хвороб. У першому випадку запрошується назва хвороби, до якої потрібно додати симптом, потім вводиться симптом, і відбувається додавання. Є можливість додати відразу декілька симптомів.

Додавання хвороб здійснюється шляхом введення з клавіатури нової назви хвороби і її симптомів (для закінчення введення симптомів необхідно набрати 'end'). Додавання відбувається в кінець БД.

4. Процедура "Перегляд": здійснюється проглядання всієї БД, що міститься в даний момент в оперативній пам'яті.

5. Процедура "Логічний вивід": консультація здійснюється таким чином: за допомогою задавання питань користувачеві про те, чи є у нього якийсь симптом, програма визначає діагноз. В процесі роботи процедура в оперативній пам'яті формує тимчасові динамічні бази даних: БЗ-yes (БД, що містить симптоми, на які користувач відповів "Так") і БЗ-no (БД, що містить симптоми,

на які користувач відповів "Ні"). Для того, щоб процедура вивела діагноз, необхідно, щоб користувач відповів "Так" на всі питання про симптоми, що характеризують цю хворобу.

6. Процедура "Пояснення логічного виводу" : дана процедура виводить список симптомів хвороби, яка була отримана за допомогою процедури логічного виводу, пояснюючи цим її діагноз.

7. Процедура "Видалення" : видалення ділиться на видалення симптомів і видалення хвороб. При видаленні симптому необхідно ввести назву хвороби, якою належить симптом, що підлягає видаленню, а також назва самого симптому. Є можливість видалити декілька симптомів у хвороби (для закінчення введення ознак, що видаляються, необхідно набрати 'end').

При видаленні хвороби запрошується її назва. Разом з нею з БД віддаляються і всі її симптоми.

8. Процедура "Редагування": редагування ділиться на редагування симптомів і редагування хвороб. У першому випадку вводиться назва хвороби, симптом якої потрібно виправити, сам симптом і його відредаговане значення. З БД видаляється старе значення симптому і додається нове.

При редагуванні хвороби необхідно ввести її назву і потім нове виправлене значення. Процедура видалить старе значення і додасть нове.

Оскільки системи, засновані на знаннях, реалізуються на комп'ютерах, то і вхідна інформація сприймається у вигляді, зрозумілому комп'ютеру. Проте для того, щоб з системою міг взаємодіяти непідготовлений користувач, в неї потрібно включити засоби спілкування на природній мові. Переважна більшість систем, що грунтуються на знаннях, володіють інтерфейсом на природній мові – допустимі вхідні повідомлення користувача обмежені набором понять, що містяться в базі знань. На рисунку 3.3 приведена розроблена діаграма діяльності користувача при використанні меню експертної системи.

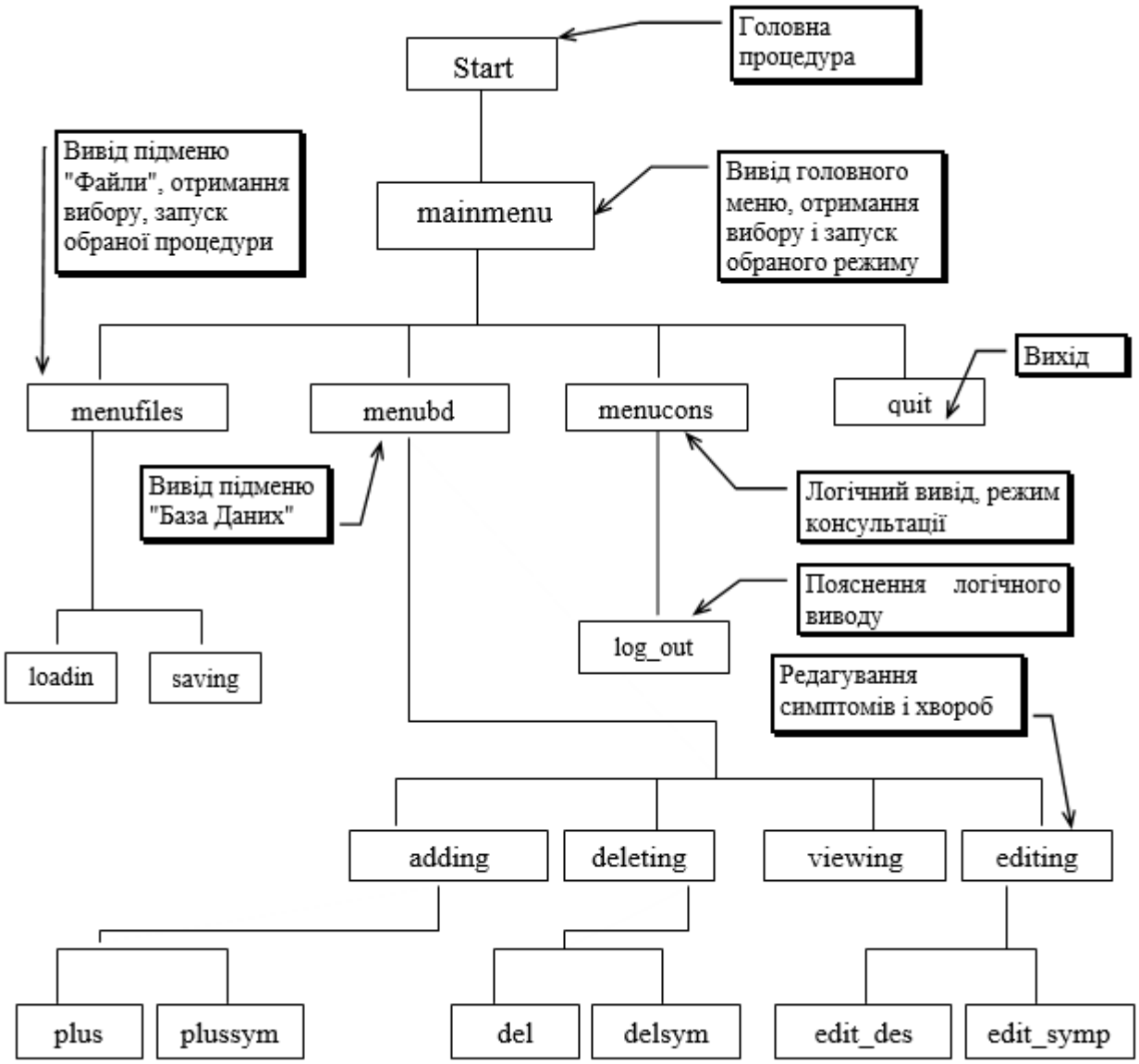


Рисунок 3.3 – Діаграма діяльності при використанні меню системи

3.3 Тестування та приклад роботи системи

Для забезпечення вимог до надійності і апаратної сумісності експертної системи та враховуючи останні здобутки у технологіях побудови інтелектуальних інформаційних систем запропоновано клієнт-серверну реалізацію системи. Для програмної реалізації бази знань, машини виводу обрано середовище Прологу SWI-Prolog. Для подолання недоліку малої

швидкодії Прологу застосовано зберігання бази знань і використанням БД. Для доступу до БД використано бібліотеку PrologSQL.

PrologSQL [37] це міст між SWI - Prolog та SQL, програма що перетворює виклики предикатів Prolog у SQL запити і відправляє їх у ORACLE або ODBC. Пролог - Oracle інтерфейс забезпечує програміста двома рівнями взаємодії. По-перше, інтерфейсу реляційного рівня. По- друге, інтерфейсу рівня перегляду, що може перевести всі твердження Пролога в один запиті SQL до БД, в тому числі і об'єднання (join) і операції агрегування (aggregate operations). Цей інтерфейс дозволяє отримати доступ з середовища Прологу до таблиць БД як якби вони існували, як факти. Весь доступ БД робиться "на льоту". Інтерфейс PrologSQL дає програмісту БД всі інструменти Пролог як мови запитів, в тому числі інтенсіональні специфікації бази даних, рекурсію, здатність обробляти неповні знання, керування виводом висновку та ін.

Опишемо програмний комплекс експертної системи медичної діагностики. Серверна частина включає: сервер СУБД, WEB-сервер, REFRESH-сервіс, DATA Converter-сервіс, які можна розміщувати на різних комп'ютерах.

Сервер СУБД є порівняно потужним комп'ютером на якому встановлена СУБД Microsoft SQL Server 2005, він виконує функції зберігання даних системи. SQL Server - це комплексне, інтегроване, закінчене вирішення обробки даних, що надає всім користувачам організації найбільш безпечну, надійну і продуктивну платформу для даних підприємства [38]. Платформа даних SQL Server включає наступні інструменти:

– реляційна база даних: безпечне, надійне, масштабоване ядро з покращеною продуктивністю і підтримкою структурованих і неструктурованих даних;

– Replication Services: реплікація даних для розподілених і мобільних застосувань обробки даних, висока доступність систем, масштабований паралелізм з вторинними сховищами даних для звітних вирішень підприємства і інтеграція, з різнорідними системами, включаючи існуючі бази даних Oracle.

– Notification Services: розвинені можливості повідомлень для розробки і впровадження масштабованих застосувань, здатних доставляти персоналізоване, своєчасне оновлення інформації, множині сполучених і мобільних пристроїв;

– Integration Services: можливості витягання, перетворення і завантаження для сховищ даних і інтеграції даних в масштабі підприємства;

– Analysis Services: аналітична обробка в реальному часі для швидкого, складного аналізу великих і змішаних наборів даних, що використовує багатовимірне зберігання;

– Reporting Services: вичерпне рішення для створення, управління і доставки як традиційних паперових звітів, так і інтерактивних, заснованих на технології WWW звітів;

– Інструменти управління: SQL Server включає засоби управління для розвиненого управління і налаштування баз даних, також як і тісну інтеграцію з такими інструментами, як Microsoft Operations Manager (MOM) і Microsoft Systems Management Server (SMS). Стандартні протоколи доступу до даних істотно зменшують час, необхідний для інтеграції даних SQL Server з існуючими системами. На додаток, підтримка Web служб вбудована для забезпечення взаємодії з іншими застосуваннями і платформами;

– Інструменти розробки для ядра бази даних, витягання, трансформації і завантаження даних, витягання інформації, OLAP і звітності, які тісно інтегровані з Microsoft Visual Studio® [39] для надання наскрізних можливостей розробки застосувань;

– Підтримка 64 -х бітних систем Intel Itanium2 і x64.

WEB-сервер призначений для попередньої обробки даних і створення користувацького інтерфейсу до функцій експертної системи [40]. На нього встановлюється Web-сервер Microsoft Internet Information Server 5.1 (MS IIS 5.1), що включає інтерпретатор ASP. На WEB-сервер може бути покладена функція забезпечення спільної безпеки – запити користувача потрапляють саме сюди, тут перевіряються права користувачів.

Фізично WEB-сервер і сервер СУБД можна об'єднати на одній машині, що рекомендується, якщо СУБД використовується лише даною системою. У разі, коли WEB- і СУБД-сервер розділені, рекомендується організувати високопродуктивне мережеве з'єднання (наприклад, можна використовувати двоточкове з'єднання 100MBit Ethernet).

Клієнт системи працює з користувацьким інтерфейсом по протоколу HTTP. Прийнятна швидкість роботи може досягатися при швидкості з'єднання в 33.6 KBit/s.

REFRESH-сервіс відповідає за своєчасне оновлення даних, а так само включає функції зберігання локальних файлів, працює як системний сервіс. DATAConverter-сервіс відповідає за передобробку даних.

Клієнтська частина розділена на чотири модулі: UsersResourcesManager, ResourcesManager, ResourcesViewer, XMLMetaDataEditor.

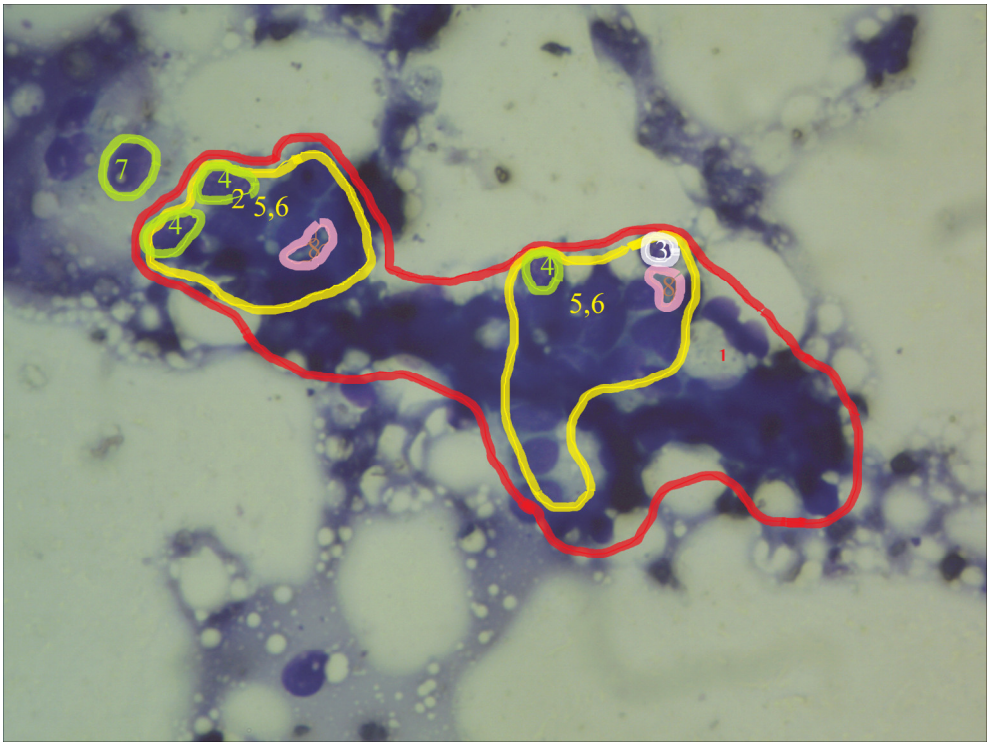
Модуль UsersResourcesManager призначений для управління користувачами і ресурсами системи, а так само зіставлення користувачів і ресурсів, використовується користувачами типа Admin.

Модуль ResourcesManager призначений для редагування параметрів ресурсів системи, використовується користувачами типа Expert і Admin.

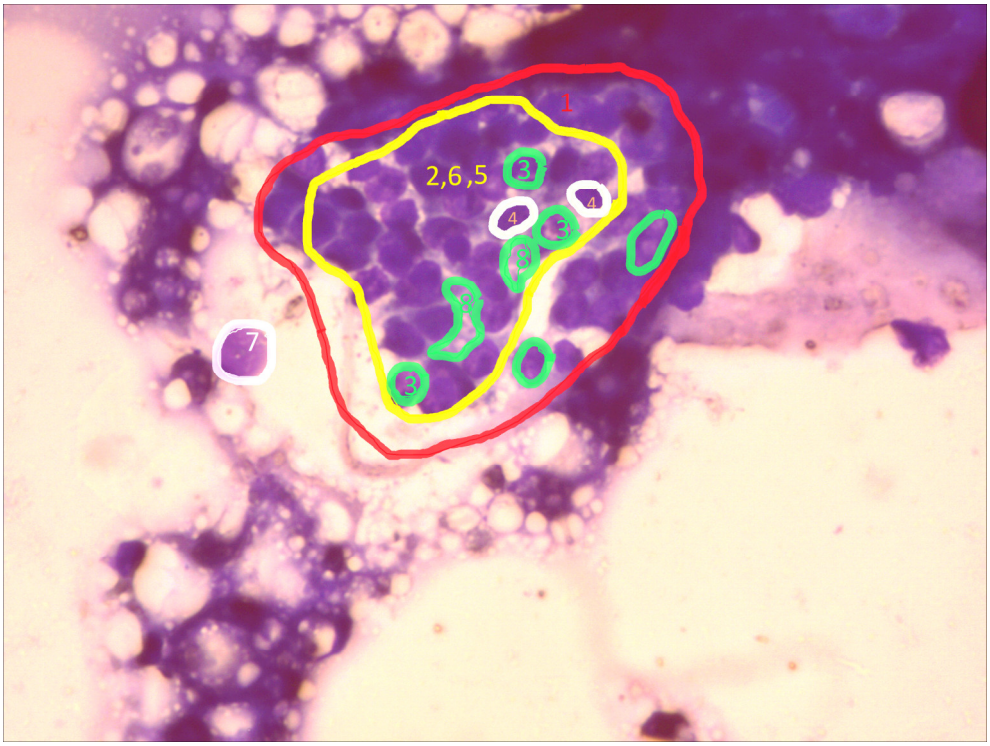
Модуль ResourcesViewer призначений для перегляду і пошуку метаданих про ресурси даних, використовується користувачами типа Patient.

Модуль XMLMetaDataEditor являє собою редактор файлів метаопису ресурсів, представлених у вигляді XML-файлів, використовується користувачами типа Manager.

На рисунку 3.4 а,б, виділено виділено якісні діагностичні ознаки в цитологічних препаратах.



а)



б)

Рисунок 3.4 – Виділення якісних діагностичних ознак в цитологічних препаратах

В результаті зробленого аналізу морфологічних ознак визначено правила для поставлення діагнозу кістозної непроліферативної мастопатії [22, 41, 42]:

ЯКЩО *клітини розміщені пластами*

І *зустрічаються кубічні та призматичні елементи, сосочкові та округлі комплекси*

І *у фоні багато фагоцитів та гістіоцитів*

І *в клітинах ядро, круглого виду, яке розташовується по центру*

І *зустрічаються клітини із апокриновою секрецією, які мають 2 зони* (*базальну та апікальну*),

ТО *кістозна непроліферативна мастопатія* (*80%*).

Сформульовано правила для діагностування фіброзної непроліферативної мастопатії:

ЯКЩО *невелика кількість гіперхромних мономорфних клітин*

І *вузький обідок інтенсивно пофарбованої цитоплазми*

І *гіперхромні ядра, круглого виду*

ТО *фіброзна непроліферативна мастопатія* (*70%*).

Вихідний текст експертної системи представлено в додатку А.

Світлокопії виданих публікацій розміщені в додатку Б.

В додатку В розміщена довідка про практичне використання результатів.

ВИСНОВКИ

Результатом даної кваліфікаційної роботи є розробка продукційної бази знань експертної системи. Отримані такі основні результати:

1. Проведений аналіз предметної області інтелектуальних медичних систем, визначено переваги і недоліки технологій їх реалізації. Сформульовано загальну концепцію та вимоги до створюваної компоненти експертної системи.

2. Розроблено алгоритми аналізу природно-мовних медичних текстів для автоматизованого наповнення бази знань.

3. Розроблено конценптуальну модель діагностичних знань у вигляді кортежу побудованого з опису вихідного стану об'єктів задачі, опису результуючого стану об'єктів, опису оператора для відображення вихідного стану у результуючий, опис цілей і обмежень, опис ресурсів, потрібних або виділених для реалізації задачі.

4. Розроблена концептуальна модель дозволяє побудувати продукційну базу знань ЕС медичної діагностики, тобто таку модель, яка ефективно реалізується сучасними програмними засобами.

5. Розроблено структуру програмного модуля та проведено тестування на цитологічних зображеннях раку молочної залози.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Викторов В.А. О развитии медико-технической науки / В.А. Викторов // Вестник РАМН. 2011. - №5 - С. 3-7.

2. Глибовець М. М. Штучний інтелект / М. М. Глибовець, О. В. Олецький – К.: Видавничий дім "КМ Академія", 2002. - 366 с.

3. Гаврилова Т.А. Базы знаний интеллектуальных систем / Т.А. Гаврилова, В.Ф. Хорошевский – СПб: Питер, 2000. – 384 с.

4. Тейз А. Логический подход к искусственному интеллекту: От модальной логики к логике баз данных: Пер. с фр. /Тейз А., Грибомон П., Юлен Г. – М.: Мир, 1998. – 494 с.

5. Осипов Г.С. Приобретение знаний интеллектуальными системами: Основы теории и технологии. – М.: Наука, 1997. – 112 с.

6. Поспелов Д.А. Логико-лингвистические модели в системах управления. – М.: Энергоиздат, 1981. – 232 с.

7. Искусственный интеллект: в 3-х кн. Кн. 2. Модели и методы: справочник; под ред. Д. А. Поспелова. – М.: Радио и связь, 1990. – 304 с.

8. Корнеев В. В. Базы данных. Интеллектуальная обработка информации / В. В. Корнеев, А. Ф. Гареев, С. В. Васютин, В. В. Райх. – М.: «Нолидж», 2000. – 352 с.

9. Представление и использование знаний / под ред. Х. Уэно, М. Исидзука; пер. с япон. к. т. н. И. А. Иванова. – М.: Мир, 1989. – 220 с.

10. Аверкин А. А. Нечеткие множества в моделях управления и искусственного интеллекта / Под ред. Д. А. Поспелова. / А. А. Аверкин, И. З. Батыршин, А. А. Блишун, В. Б. Силов, В. Б. Тарасов – М.: Наука, 1986. - 312 с.

11. Девятков В.В.. Системы искусственного интеллекта / В.В. Девятков – М.: МГТУ им. Н. Э. Баумана, 2001. - 352 с.

12. Гаврилов А.В. Гибридные интеллектуальные системы / А.В. Гаврилов, Ю.В. Новицкая // Международная конференция "Информационные системы и технологии", 22-26 апреля 2003, НГТУ, Новосибирск – С. 98-101

13. Вступ у сутність експертних систем. – Режим доступу: https://uadoc.zavantag.com/text/38290/index-1.html?page=4.

14. Мачуляк М. В., Галан В. Ю., Іпіроті В. О., Николин І. П. Системи підтримки прийняття рішень в медичній діагностиці. Інтелектуальні комп'ютерні системи та мережі : тези доп. V Наук.-практ. конф. молодих вчених і студентів (2 груд. 2021 р.). Тернопіль : ЗУНУ, 2021. С.

15. Галан В. Ю., Мачуляк М. В., Іпіроті В. О., Николин І. П. Моделювання знань в системах медичної діагностики. Інтелектуальні комп'ютерні системи та мережі : тези доп. V Наук.-практ. конф. молодих вчених і студентів (2 груд. 2021 р.). Тернопіль : ЗУНУ, 2021. С.

16. Березький О.М., Дубчак Л.О., Мельник Г.М. Методичні рекомендації до виконання кваліфікаційної роботи з освітнього ступеня "Магістр". Спеціальність: 123 - Комп'ютерна інженерія. Магістерська програма - Комп'ютерна інженерія". Тернопіль: ЗУНУ, 2021. 32 с.

17. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп'ютерна інженерія» / І.В. Гураль, Л.О. Дубчак / Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2019. 33 с.

18. Братко И. Алгоритмы искусственного интеллекта на языке PROLOG. Третье издание / Иван Братко - Москва: Вильямс, 2004. - 640 с.

19. Гуц А. К. Математическая логика и теория алгоритмов / А. К. Гуц - Омск: Наследие. Диалог-Сибирь, 2003. - 108 с.

20. Giarratano J. Expert Systems Principles and Programming, Third Edition / Joseph Giarratano - USA: PWS Publishing Company, 1998. - 602 p.

21. Mycin [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Mycin.

22. Березький О. М., Батько Ю.М., Березька К.М., Вербовий С.О., Дацко Т.В., Дубчак Л.О., Ігнатєв І.В., Мельник Г.М., Николюк В.Д., Піцун О.Й. Методи, алгоритми і програмні засоби опрацювання біомедичних зображень. Тернопіль: Економічна думка, ТНЕУ, 2017. 330 с.

23. Berezsky O., Verbovyy S., Dubchak L., Datsko T. Fuzzy System of Diagnosing in Oncology Telemedicine. Sensors & Transducers. 2017. Vol. 208, Issue 1. P. 32-38.

24. Berezsky O., Dubchak L., Batryn N., Datsko T., Berezska K., Pitsun O., Batko Y. Fuzzy system for breast disease diagnosing based on image analysis. Proceedings of the 1st International Workshop on Informatics & Data-Driven Medicine, Lviv, Ukraine, 11-13 november 2019.

25. Березький О. М., Мельник Г.М., Березька К. М. Нечітка база знань інтелектуальної системи діагностування видів раку молочної залози, Вісник Хмельницького національного університету. Технічні науки. 2013. №6. С. 284-292.

26. Berezsky O., Pitsun O., Batryn N., Datsko T., Berezska K., Dubchak L. Modern automated microscopy systems in oncology. Proceedings of the 1st International Workshop on Informatics & Data-Driven Medicine, Lviv, Ukraine, 28-30 november 2018. P. 311-325.

27. Березький О. М., Мельник Г.М., Березька К. М., Дацко Т.В. Інтелектуальна система аналізу зображень ауто- та ксеногенних тканин. Науковий вісник НЛТУ України: зб. наук.-техн. праць. Львів: РВВ НЛТУ України. 2014. Вип. 24.11. С. 323-330.

28. Березький О. М., Мельник Г.М., Батько Ю.М., Дацко Т. В. Інтелектуальна система для діагностування різних форм раку молочної залози на основі аналізу гістологічних і цитологічних зображень. Науковий вісник НЛТУ України: зб. наук.-техн. праць. Львів: РВВ НЛТУ України. 2013. Вип. 23.13. С. 357-367.

29. Березький О. М. Піцун О.Й., Лящинський П.Б., Мельник Г.М. Інтелектуальна система автоматизованої мікроскопії аналізу гістологічних та цитологічних зображень. Штучний інтелект. 2017. №2 (76). С. 128-140.

30. Створення гібридної інтелектуальної системи морфометричної діагностики для верифікації внутрішньопротокового раку молочної залози за цитологічного дослідження / В.А.Дацко, О. М. Березький, Г.М. Мельник та інші, всього 6 осіб. Український журнал клінічної та лабораторної медицини. 2013. Т. 8, № 4. С. 162-165.

31. Березький О. М., Березька К. М. Гібридні інтелектуальні системи медичного діагностування. Матеріали Міжнародної наукової конференції «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» (ISDMCI'2014), м. Залізний Порт, 28–31 травня 2014 р. Херсон: ХНТУ, 2014. С. 246–247.

32. Cloksin W. F. Programming in Prolog. Fifth Edition / William F. Cloksin, Christopher S. Mellish: Springer, 2003. - 205 p.

33. Huntbach M. M. Ringwood G. A. Agent-Oriented Programming-From Prolog to Guarded Definite Clauses / Matthew M Huntbach, Graem A Ringwood: Springer, 1999. - 213 p.

34. Клоксин У. Программирование на языке Пролог / У. Клоксин, К. Меллиш: М.:Наука, 1987. - 336 с.

35. Nugues P. An Introduction to Language Processing with Perl and Prolog / Pierre Nugues: Springer, 2006. - 514 p.

36. Малпас Дж.. Реляционный язык пролог и его применение: Пер. с англ. / Под ред. В.Н. Соболева / Дж. Малпас – М.: Наука, 1990. - 464 с.

37. PrologSQL: SWI-Prolog to SQL bridge [Електронний ресурс]. – Режим доступу: https://www.sikorskiy.net/info/prj/prologsql/.

38. Microsoft SQL Server [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft_SQL_Server.

39. Powers L. Microsoft Visual Studio 2008 unleashed / Lars Powers, Mike Snell - Indianapolis, IN, USA: Sams, 2008. - 1248 p.

40. Молчанов В. П. Засоби систем обробки даних : навчальний посібник / В. П. Молчанов. – Х. : Вид. ХНЕУ, 2013. – 100 с.

41. Березький О. М. Комп'ютерна система аналізу біомедичних зображень / О. М. Березький, Ю.М. Батько, Г.М.Мельник // Вісник Національного університету «Львівська політехніка». Комп'ютерні науки та інформаційні технології. – 2009. – № 570. – С. 84-89.

42. Berezsky O. An Intelligent System for Cytological and Histological Image Analysis / Oleh Berezsky, Grygoriy Melnyk, Tamara Datsko, Sergiy Verbovy // Proceedings of the 13 th International Conference «The Experience of Designing and Application of CAD Systems in Microelectronics» CADSM 2015, 24-27 February 2015, Polyana-Svalyava (Zakarpattya), Ukraine. – 2015. – P. 28-31.

43. Березький О. М. Статистичне оброблення цитологічних зображень / О. М. Березький, К. М. Березька, С. Ю. Попіна // Вісник Хмельницького національного університету: зб. наук.-техн. праць. – Сер.: Технічні науки. – 2012. – № 5. – С. 161–164.

44. Berezsky O. Fuzzy system diagnosing of precancerous and cancerous conditions of the breas / O. Berezsky, S. Verbovyy, L. Dubchak, T. Datsko // Proceedings of the XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT'2016), Lviv, 6-10 September, 2016. – Lviv, 2016. – P. 200–203.

45. Березький О. М., Мельник Г.М., Дацко Т.В., Вербовий С. О. База даних цитологічних та гістологічних зображень ауто- та ксеногенних тканин. Науковий вісник НЛТУ України: зб. наук.-техн. праць. Львів: РВВ НЛТУ України. 2014. Вип. 24.10. С. 338-345.

46. Свідоцтво про реєстрацію авторського права на твір №75359. База даних цифрових гістологічних та цитологічних зображень передракових та ракових станів молочної залози «BPCI2100». О.М. Березький, Г.М. Мельник, С.О. Вербовий, О.Й. Піцун, В.Д. Николюк, Т.В. Дацко. Дата реєстрації 14.12.2017 р.