мотиви, пам'ять та створюють умови для розвитку математичної компетентності учнів ПТНЗ.

БІБЛІОГРАФІЯ

1. Жалдак М.І. Математика з комп'ютером. Посібник для вчителів / М.І. Жалдак, Ю.В. Горошко, Є.Ф. Вінниченко. – К.: Вид-во НПУ імені М.П. Драгоманова, 2015. – 310 с.
2. Литвин А. Впровадження педагогічних програмних засобів у будівельних ПТНЗ / Андрій Литвин // Педагогіка і психологія професійної освіти : науково-метод. журнал. — 2011. — № 2. — С. 23-35.
3. Раков С.А. Математична освіта: компетентнісний підхід з використанням ІКТ: монографія / С.А. Раков. - Х.: Факт, 2005. - 360 с.
4. Робоча програма з математики для 10-11 класів загальноосвітніх навчальних закладів [Електронний ресурс]: Режим доступу: http://mon.ua
5. Чебернина Г.М. Збірник задач: Математика в житті людини / Г.М. Чебернина. – К.: 2011р. – 27 с.

ВІДОМОСТІ ПРО АВТОРА

Тінькова Дар'я Сергіївна – викладач математики, ДНЗ «Бердянський машинобудівний професійний ліцей».

Коло наукових інтересів: реалізація компетентнісного підходу на уроках математики засобами інформаційно-комунікаційних технологій.

ВИКОРИСТАННЯ РЕКУРСІЇ ПРИ ВИВЧЕННІ МОВИ ЛОГІЧНОГО ПРОГРАМУВАННЯ TURBO PROLOG

Сергій ШАРОВ, Дмитро ЛУБКО

Робота присвячена визначенню окремих питань щодо використання рекурсії у мовах логічного програмування під час викладання дисципліни «Інтелектуальні системи» у вищому навчальному закладі. Розглядаються типові задачі (знаходження факторіалу, обчислення чисел Фібоначчі, родинні зв'язки та ін.), які дозволять студентам краще опанувати дисципліну на початковому рівні.

Recursion in the language logic programming. The article is devoted to defining specific issues regarding the use of recursion in a logic programming languages while teaching discipline "Intelligent systems" in higher education. We consider the typical tasks (finding factorial, calculating Fibonacci numbers, relatives) That will allow students to better master the discipline at the primary level.

Постановка проблеми. Сучасний розвиток різноманітного програмного забезпечення та інформаційно-комунікаційних технологій визначається значним розширенням сфери застосування інформаційних систем, які засновані на знаннях. Їх можна зустріти в освіті, бізнесі, управлінській діяльності тощо. Робота будь-якої інтелектуальної системи передбачає обробку бази знань, яка складає її основу та має властивість поступово накопичувати нові знання. Природним середовищем для накопичення бази знань вважаються інтелектуальні системи, побудовані за допомогою сімейства мов логічного програмування (Lisp, Turbo Prolog, Visual Prolog тощо). Одним із основних методів обробки даних у мові Prolog є рекурсія, яка обов'язково вивчається студентами у ВНЗ при ознайомленні з мовами логічного програмування.

Аналіз останніх досліджень. У напрямку розвитку саме мови Turbo Prolog цікавими та пізнавальними є праці О.Гущіна, С.Хабарова, П.Шрайнера. Також перспективною мовою є Visual Prolog, який докладніше розглядається у роботах Е.Акчуріна та Е.Кости.

Формування мети статті. Метою статті є огляд основних питань, які повинні вирішуватися при викладанні теми «Рекурсія» під час вивчення дисципліни «Інтелектуальні системи» студентами напряму підготовки 040302 «Інформатика» освітньо-кваліфікаційного рівня «Бакалавр».

Виклад основного матеріалу. Активне застосування систем з елементами штучного інтелекту істотним чином перетворює сучасну реальність та формує особливий тип світовідчуття. Техніка, оснащена ознаками штучного інтелекту, що є засобом підвищення рівня комфорту і безпеки, стає характерною рисою повсякденного побуту [2, с. 109]. Зазначені закономірності призвели до появи інформаційних систем нового типу, з назвою інтелектуальні інформаційні системи (ІІС). Основу будь-якої інтелектуальної системи складають база знань і закладений в систему механізм виведення рішень. Ці компоненти визначають дві основні інтелектуальні характеристики системи: здатність зберігати знання про щось і вміння оперувати цими знаннями [4, с. 211].

До інструментальних засобів розробки інтелектуальних систем відносять основні базові логічні мови – це Lisp та Prolog, а також різноманітні оболонки та генератори експертних систем. З іншого боку, завдяки базовому принципу мови логічного програмування Turbo Prolog, а саме рівнозначності представлення програми і даних, Turbo Prolog-системи вважають природним середовищем для накопичення бази знань. У якості основних областей застосування Turbo Prolog можна відзначити: розробку швидких прототипів прикладних програм; управління виробничими процесами; створення динамічних реляційних баз даних; створення природно-мовних інтерфейсів; реалізація експертних систем тощо. На відміну від процедурних мов, де програміст повинен описати процедуру вирішення крок за кроком, при використанні Turbo Prolog достатньо описати задачу та основні правила її вирішення. Також серед відмінностей від процедурних мов можна виділити: вбудований механізм виведення з пошуком і поверненням (відкат та відсікання); механізм зіставлення із зразком (уніфікація); просту, але виразну структуру даних з можливістю її зміни [5, с. 18].

Слід зазначити, що мова Prolog продовжує розвиватися, що знаходить своє відображення у появі сучасних мов логічного програмування, таких як Visual Prolog. Тому саме на Turbo Prolog ми звернули увагу при викладанні дисципліни «Інтелектуальні системи» студентам напряму підготовки 040302 «Інформатика» ступеня вищої освіти «Бакалавр».

Природнім методом програмування у Turbo Prolog є рекурсія, яка вважається фундаментальним поняттям у комп'ютерних науках та математиці. Розглянемо її більш докладно та наведемо типові приклади використання рекурсії у Turbo Prolog при вивченні дисципліни.

За визначенням, рекурсивною називається програма, яка звертається сама до себе. Особливістю рекурсивної програми є наявність умови завершення, оскільки вона не може викликати себе до нескінченності. З огляду на це, програма, яка містить рекурсію, повинна мати як мінімум два шляхи виконання, один з яких передбачає рекурсивний виклик, а другий – виконання програми без рекурсивного виклику [3]. Для реалізації цієї умови будь-яка рекурсивна процедура повинна містити базис і крок рекурсії.

Базис рекурсії - це пропозиція, що визначає якусь початкову ситуацію або ситуацію, яка відбувається у момент припинення. Як правило, в цьому реченні записується якийсь найпростіший випадок, при якому відповідь виходить відразу навіть без використання рекурсії.

Крок рекурсії – це правило, в тілі якого обов'язково міститься в якості підцілі виклик обумовленого предиката. Параметри повинні змінюватися на кожному кроці так, щоб в результаті або спрацював базис рекурсії, або умова виходу з рекурсії, розміщена в самому правилі [6, с. 38].

У загальному вигляді правило, що реалізує крок рекурсії, буде виглядати таким чином:

<им'я правила рекурсії>:-

<список предикатів>, <предикат умови виходу>,

<список предикатів>, <ім'я правила рекурсії>,

<список предикатів>.

Будь-яке рекурсивне визначення містить принаймні одне нерекурсивне правило і одне (або декілька правил) правило з рекурсією. Якщо остання умова в останньому правилі є рекурсивною, то вважається, що використовується хвостова рекурсія. Така рекурсія має перевагу перед нехвостовою рекурсією, оскільки дозволяє обмежити зростання стека та строго контролювати процес повернення.

При викладанні теми «Рекурсія» у межах дисципліни «Інтелектуальні системи» потрібно розглянути декілька типових прикладів, що показують роботу рекурсивного виклику. Опанування роботи з рекурсією можна почати з класичного прикладу на основі створеної на перших заняттях зі студентами програми «Родинні зв'язки». Припустимо, що у базі знань даної програми є набір фактів, що описує родинні зв'язки людей через ставлення «быть родителем». Предикат «родитель» має два аргументи. У якості його першого аргументу вказується ім'я батька, в якості другого – ім'я дитини. Потрібно створити правило «быть предком», використовуючи предикат «родитель».

Для того щоб одна людина була предком іншої людини, потрібно, щоб він або був його «родителем», або бути «родителем» іншого його предка. Цю ідею можна записати таким чином:

```
предок(Предок,Нащадок):–
родитель(Предок,Нащадок). /* предком є «родитель» */
предок(Предок,Нащадок):–
родитель(Предок,Людина),
предок(Людина,Нащадок). /* предком є «родитель» предка*/
```

Наступний нижченаведений типовий приклад використання рекурсії передбачає обчислення факторіалу натурального числа. Ця задача допускає рекурсивне рішення на багатьох мовах програмування, а також має рекурсивний математичний вигляд:

```
fact (1,1):- !. / * Умова припинення рекурсії * /
fact (N, F):- N1 = N-1, fact (N1, F1), / * F1 дорівнює факторіалу числа, на одиницю меншого вихідного числа * /
F = F1 * N. / * Факторіал опис: числа дорівнює добутку F1 на саме число * /
```

Варто відзначити, що метод рекурсії має свої переваги перед методом ітерації, який використовується в імперативних мовах програмування набагато частіше. З іншого боку, рекурсія має великий недолік: їй, взагалі кажучи, може не вистачати для роботи стека. При кожному рекурсивному виклику предикату відбувається запам'ятовування всіх проміжних змінних, які можуть знадобитися, у спеціальному стеці. Максимальний розмір цього стека у випадку роботи програми на мові TurboProlog складає всього всього 64 Кб. Звісно, що при великих вхідних значеннях об'єму стека може не вистачити.

Для вирішення проблеми переповнення стуку рекомендується використовувати хвостову рекурсію. Для її здійснення рекурсивний виклик предиката повинен здійснюватися в останній підцілі у тілі рекурсивного правила і до моменту рекурсивного виклику не повинно залишитися точок повернення. Тобто у підцілей, розташованих ліворуч рекурсивного виклику визначається предикат, та не повинно залишатися жодних неперевірених варіантів. Крім того, у рекурсивній процедурі не повинно бути пропозицій, розташованих нижче рекурсивного правила. При дотриманні цих умов Turbo Prolog розпізнає хвостову рекурсію і усуває пов'язані з нею додаткові витрати. Цей процес називається оптимізацією хвостової рекурсії або оптимізацією останнього дзвінка [6, с. 42].

Ще один класичний приклад використання рекурсії пов'язаний з обчисленням чисел Фібоначчі, які можна визначити таким чином: перше і друге число дорівнюють одиниці, а кожне наступне число є сумою двох попередніх. З точки зору програми на Prolog першим базисом є твердження, що перше число Фібоначчі дорівнює одиниці. Другий базис - аналогічне твердження про друге число Фібоначчі. Крок рекурсії: для обчислення числа Фібоначчі з номером N спочатку потрібно обчислити і скласти числа Фібоначчі з номерами N-1 і N-2. Записати ці міркування можна так:

```
fib (1,1): - !. / * Перше число Фібоначчі дорівнює одиниці * /
fib (2,1): - !. / * Друге число Фібоначчі дорівнює одиниці * /
fib (N, F): -
N1 = N-1, fib (N1, F1), / * F1 це N-1-е число Фібоначчі * /
N2 = N-2, fib (N2, F2), / * F2 це N-2-е число Фібоначчі * /
F = F1 + F2. / * N-е число Фібоначчі дорівнює сумі N-1-го і N-2-го чисел Фібоначчі *
```

Як зазначалось вище, рекурсія досить часто використовується для здійснення обчислень. Типовою задачею використання рекурсії вважається обчислення ступеня числа. Рекурсивне визначення функції, що обчислює ступінь числа, можна представити у такому вигляді: число X у ступені У є число X у ступені У-1, яке помножене на власне число X. Наприклад, два у ступені п'ять - це два у четвертому ступені, помножене на два. На мові Prolog цей вираз можна представити таким чином:

```
st(X,Y,Z):- YY=Y-1, st(X,YY,ZZ), Z=ZZ*X.
```

Ми використовуємо предикат st, який має три параметри, де перший аргумент (вхідний) - це число, яке буде возводитися у ступінь (X); другий аргумент (вхідний) – це ступінь, в який зводиться число (Y); третій аргумент (вихідний) – результат (Z).

Обмежити хід рекурсії можна додаванням певної пропозиції, при якій ціль могла бути уніфікована без додаткових викликів. Наприклад, два в першій - це два або «будь-яке

число X у ступені 1 дорівнює власне числу X». На мові Prolog - цей вираз можна представити таким чином: st(X,1,X).

У зв'язку з тим, що процес уніфікації завжди проходить послідовно зверху-вниз і зліва-направо, то як правило базис рекурсії розміщують раніше, ніж крок. У нашому випадку це виглядає наступним чином.

```
st(X,1,X).
st(X,Y,Z):- YY=Y-1, st(X,YY,ZZ), Z=ZZ*X.
```

Однак, виконання програми на цьому не закінчено і після завантаження програми на виконання користувач дізнається про переповнення стеку.

Додаткова умова не дозволить проводити уніфікацію цілі у випадку, якщо другий аргумент менше або дорівнює одиниці. Це може виглядати таким чином:

```
st(X,1,X):-!.
st(X,Y,Z):- YY=Y-1, st(X,YY,ZZ), Z=ZZ*X.
```

У даному випадку в при уніфікації першого речення спрацьовує безумовний стандартний предикат відсікання "!", який означає видалення з пам'яті альтернативних шляхів уніфікації мети. Це призводить до того, що після отримання першого можливого рішення - Prolog не буде намагатися знайти інші рішення та не перейде в область негативних значень [1].

Декілька слів слід сказати про динамічне програмування. Це загальний підхід для реалізації рекурсивних програм, який дає можливість отримувати ефективні та елегантні рішення для великого класу задач.

Сама технологія має назву – висхідне динамічне програмування (bottom-up dynamic programming). Вона заснована на тому, що значення рекурсивної функції можна визначити, обчислюючи всі значення цієї функції, починаючи з найменшого, використовуючи на кожному кроці раніше обчислені значення, для підрахунку поточного значення.

Спадаюче динамічне програмування (top-down dynamic programming) є ще простішою технологією. Вона дозволяє виконувати рекурсивні функції при тій же кількості ітерацій, що і висхідне динамічне програмування [3].

Висновки. Отже, мова логічного програмування Turbo Prolog та її похідні часто використовуються для розробки інтелектуальних систем різної складності. Одним із головних методів обробки знань та здійснення обчислень є рекурсія, яка виконує функції організації повтору певних дій. Викладання теми «Рекурсія» під час вивчення дисципліни «Інтелектуальні системи» студентами напряму підготовки 040302 «Інформатика» освітньо-кваліфікаційного рівня «Бакалавр» дозволяє студентам краще пізнати основи роботи з логічними мовами програмування та поглибити свої знання для вирішення різних практичних та інженерних задач у житті.

БІБЛІОГРАФІЯ

1. Логическое программирование: [Електронний ресурс]. – Режим доступу: http://proprolog. narod.ru/razd_rs.htm.

2. Ревко П.С. Искусственные интеллектуальные системы в повседневной жизни человека / П.С. Ревко. – Известия Южного федерального университета. Технические науки. – Вып. № 9-2, 2006. – С. 109 – 110.

3. Рекурсия в программировании и разработка рекурсивных программ : [Електронний ресурс]. – Режим доступу: http://mf.grsu.by/Kafedry/kaf001/ academic_process/048/20.

4. Сахнюк П.А. Интеллектуальные системы и технологии: Учебное пособие. – Ставрополь: Агрус 2012. - 228 с.

5. Хабаров С.П. Интеллектуальные информационные системы. PROLOG-язык разработки интеллектуальных и экспертных систем: учебное пособие / С.П.Хабаров.- СПб. СПбГЛТУ, 2013.– 138 с.

6. Шрайнер П. А. Основы программирования на языке Пролог : курс лекций : учеб. пособие для студентов вузов, обучающихся по специальностям в обл. информ. Технологий / П. А. Шрайнер. - М. : Интернет - Ун-т Информ. Технологий, 2005. – 176 с.

ВІДОМОСТІ ПРО АВТОРІВ

Шаров Сергій Володимирович – кандидат педагогічних наук, доцент, доцент кафедри інформатики і кібернетики Мелітопольського державного педагогічного університету імені Богдана Хмельницького

Коло наукових інтересів: інформаційні системи, програмування, самостійна робота студентів.

Лубко Дмитро Вікторович – кандидат технічних наук, доцент, доцент кафедри інформаційних технологій Таврійського державного агротехнічного університету.

Коло наукових інтересів: експертні системи, інтелектуальні системи, програмування, веб-технології.

ВИКОРИСТАННЯ МЕНТАЛЬНИХ КАРТ У НАВЧАЛЬНОМУ ПРОЦЕСІ

Ірина ШАХІНА, Роман МЕДВЕДЄВ

У статті висвітлено питання щодо поняття ментальна карта, історії її розвитку, переваги і недоліки використання у навчальному процесі, методики створення на прикладі одного з on-line ресурсів.

The article deals with the mental map concept and the history of its development, the advantages and disadvantages of its usage in the educational process, the methods of its creation on the basis of one of on-line resources.

Постановка проблеми. Сучасний період розвитку суспільства, оновлення всіх сфер його соціального і духовного життя потребує якісно нового рівня освіти, який відповідав би міжнародним стандартам. Особливо це стосується спеціальної та вищої освіти. З активним розвитком інформаційно-комунікаційних технологій, активно розвивається й освіта. Головним завданням педагога є вибір різноманітних прийомів, форм і засобів представлення навчального матеріалу. Адже головне – зацікавити учня, змусити його самостійно досліджувати певну галузь, вивчати для себе щось нове, тощо.

Для того, щоб краще і цікавіше донести до учнів навчальний матеріал, педагоги використовують сучасні інформаційні технології. Нині, одним з найцікавіших способів подачі навчального матеріалу, а також систематизації самостійної роботи учня є, так звані, ментальні карти. Проте, на жаль, дана технологія ще не до кінця освоєна, і зрозуміла для вчителів.

Аналіз попередніх досліджень. Проблемам упровадження інформаційно-комунікаційних технологій в освітній процес присвячені праці В. Бикова, Р. Гуревича, М. Жалдака, Ю. Дорошенка, Ю. Запорожченка, І. Захарової, І. Кухаренка, Н. Морзе, Є. Полат, І. Роберт, І. Селевка, П. Стефаненка, В. та І. Трайньових, М. Шишкіної та ін. Використанням соціальних сервісів у навчальному процесі займаються такі науковці, як В. Биков, Р. Гуревич, І. Захарова, Н. Морзе, Є. Патаракін та ін. Питання використання ментальних карт у навчальному процесі відображені у роботах таких закордонних вчених,