УДК 51.681.3

ФОРМАЛЬНО-ЛОГІЧНИЙ ПІДХІД ДО ПОБУДОВИ СИСТЕМ АНАЛІЗУ ЗНАНЬ В РІЗНИХ ПРЕДМЕТНИХ ОБЛАСТЯХ

О.В. Палагін, С.Л. Кривий, Д.С. Бібіков, В.Ю. Величко, К. Марков, К. Іванова, І. Мітов

Інститут кібернетики імені В.М. Глушкова НАН України,
Київ-187, МСП, 03680, проспект Академіка Глушкова, 40,
e-mail: palagin_a@ukr.net, факс: +38044 5263348
Київський національний університет імені Тараса Шевченка, Київ-01, МСП, 01601, вул. Володимирська, 64,
e-mail: krivoi@i.com.ua, факс: +38044 2590439
Інститут математики і інформатики Болгарської академії наук;
Болгарія, Софія-1113, вул. Академіка Г. Бонтчева, 8, e-mail: info@foibg.com

Запропоновано формальну постановку задачі добування знань з природномовних об'єктів. Для маніпуляції, аналізу та трансформації текстів введено поняття алгебраїчної системи спискових структур. Для представлення та зберігання семантичних мереж запропоновано використовувати реалізацію багатовимірного методу доступу в інструментальному комплексі ArM32.

We proposed a formal statement of the problem of knowledge extraction from natural language objects. For manipulation, analysis and transformation of texts the notion of an algebraic system of list structures is introduced. For presentation and storage of semantic networks it is offered to use realization of multidimensional access method in an instrumental system ArM32.

Вступ

Комп'ютерна технологія розробки сенсорних систем за допомогою віртуальної лабораторії автоматизованого проектування [1] дозволяє фахівцям різних предметних областей таких як хімія, біологія, біохімія, фізика самостійно перевірити можливість створення вимірювального пристрою й здійснити проектування нового приладу включно до етапу розробки конструкторської документації на виготовлення дослідного зразка. Віртуальна лабораторія розробки сенсорних систем (ВЛРСС) створюється на базі формалізованого подання теоретичних знань, принципів організації, методів і засобів автоматизованого проектування й тестування інформаційно-вимірювальних систем і приладів з використанням методології системної інтеграції.

ВЛРСС є складною системою, що складається із взаємодіючих програмних та апаратних модулів. Для формалізованого опису ВЛРСС, структурування й подання знань у машинній формі зручно використовувати онтологічне подання. Базова онтологія розподіленої віртуальної лабораторії проектування сенсорних систем [2] містить основні поняття, що описують предметну область ВЛРСС і віртуальних методів проектування, а також відносини, семантично значимі для цієї предметної області.

З точки зору розробника електронного пристрою, основною метою функціонування онтології ВЛРСС є активна допомога користувачеві при створенні сенсорної системи з спілкуванням на обмеженій природній мові. Відповідь системи повинна включати описи процедур, методів і компонентів віртуальної лабораторії, що дозволяють вирішити завдання, які поставлені в запиті користувача. Для організації такого діалогу необхідно вирішувати завдання аналізу й синтезу природно-мовних текстів (ПМТ). Іншою важливою задачею, яка повинна вирішуватись при використанні ВЛРСС, є поповнення онтології віртуальної лабораторії у сфері професійних інтересів користувача на основі автоматизованого аналізу ПМТ.

Формальна постановка задачі аналізу ПМТ

Нехай $T = t_1 t_2 ... t_n$ – ПМТ в алфавіті X, тобто $T \in L(X)$, де $L(X)$ – мова в алфавіті X, а $t_i \in T$ – речення цього тексту, $i = 1,2,..., n$. Кожне речення $t_i \in T$, в свою чергу, має структуру $t_i = t_{i1} t_{i2} ... t_{im}$, де t_{ij} змістовно означають граматичні одиниці, з яких побудоване речення $t_i \in T$. Якщо $t_{ij} \in t_i$, то $C_L(t_{i_j}) = t_{i_1} \ldots t_{i_{j-1}}$ і $C_R(t_{i_j}) = t_{i_{i+1}} \ldots t_{i_m}$ будемо називати лівим і правим контекстом слова t_{i_j} відповідно у реченні t_i.

З текстом T зв'яжемо такі об'єкти:

S – словник мови $L(X)$, де знаходяться слова t_{i_j} зі своїми означеннями;

$\gamma \subseteq T \times S$ – відношення, яке визначає можливі значення і типи слів у словнику S;

$A = (D, \Pi)$ – предметна модель, на якій інтерпретується текст T;

$\varphi \subseteq T \times A$ – відношення інтерпретації тексту T на моделі $A = (D, \Pi)$.

Сигнатура предикатів $\Pi = \{\pi^{k_1}, \ldots, \pi^{k_r}\}$ включає атомарні предикати, з яких можна будувати складні формули. Зараз неможливо зафіксувати цю сигнатуру, так як вона залежить від предметної моделі. Оскільки модель не фіксується, то і її сигнатуру уточнити неможливо. Зауважимо тільки те, що кожний атомарний предикат має тип (тобто, це буде деяка типізована сигнатура).

Визначимо правила обчислення відношень γ і φ. Відношення γ має досить простий спосіб обчислення:

$$\gamma(t_{i_j}) = \{(a_1, \tau_1), (a_2, \tau_2), \ldots, (a_s, \tau_s)\},$$

де a_i можливі значення слова t_{i_j}, τ_i – його можливі типи. Може трапитися, що $\gamma(t_{i_j}) = \varnothing$. У цьому випадку значення цього слова вважається невизначеним (і ця ситуація вимагає поповнення словника S).

Відношення φ визначається дещо складніше. Якщо модель $A = (D, \Pi)$ визначена, то

$$\varphi(T) = \varphi(t_1) \ldots \varphi(t_n),$$

де

$$\varphi(t_i) = \{\varphi(\gamma(t_{i_1})\gamma(C_R(t_{i_1}))), \varphi(\gamma(C_R(t_{i_2}))\gamma(t_{i_2})\gamma(C_R(t_{i_2}))), \ldots, \varphi(\gamma(C_L(t_{i_n}))\gamma(t_{i_n}))\}$$, при цьому

$$\varphi(\gamma(t_{i_j})) = \gamma(\varphi(t_{i_j}));$$

$$\varphi(\gamma(C_L(t_{i_j}))) = C_L(\gamma(\varphi(t_{i_j}))));$$

$$\varphi(\gamma(\pi_r^k(p_1, \ldots, p_k))) = \gamma(\varphi(\pi_r^k))(\varphi(\gamma(p_1), \ldots, \gamma(p_2))),$$

де $\gamma(\varphi(\pi_r^k))$ – ім'я предиката, тип якого узгоджений з аргументами $\gamma(p_1), \ldots, \gamma(p_k)$.

Задачі, що випливають з формальної постановки

З вищенаведеної формалізованої постановки проблеми аналізу ПМТ випливає, що основні задачі її розв'язання зводяться до таких:

– побудувати предметну модель A; ця задача є основною і найбільш складною в зв'язку з тим, що предметна модель по суті є базою знань (побудова такої бази полягає в тому, щоб визначитися з об'єктами, які добуваються з тексту, з формальною логічною мовою, правилами виведення, аксіоматикою тощо);

– показати обчислюваність відношень γ і φ на предметній моделі A і побудувати алгоритми обчислення відношень γ і φ;

– при обчисленні відношень γ і φ контролювати відповідність типів аргументів і предикатів;

– визначити взаємодію алгоритмів обчислення γ і φ з системами синтаксичного і семантичного аналізу тексту.

Другорядними, але теж важливими, є задачі пов'язані з

– визначенням структури даних для словників;

– визначенням інформації, яка повинна знаходитися в словниках;

– визначенням режиму взаємодії з користувачем (автоматичний, автоматизований, діалоговий);

– визначенням мови інтерфейсу користувача й алгоритмів логічного виведення.

Обмеження на вхідний текст

Розглянемо коротко структурні обмеження, які накладаються на вхідний текст.

Нормалізація ПМТ. Більшість систем як інформаційного пошуку, так і обробки текстової інформації включають як основну компоненту систему аналізу, яка служить для виявлення "змісту" або "значення" заданої одиниці інформації. В звичайних системах такого типу аналіз може виконувати людина. При цьому аналіз використовує завчасно розроблені таблиці або шаблони для визначення того, який ідентифікатор змісту за сенсом більше підходить для заданої одиниці інформації. Відомі системи так званого автоматичного індексування, в яких ідентифікатори змісту приписуються автоматично, виходячи із структури тексту документа і запиту.

У зв'язку з тим, що природна мова включає різного роду нерегулярні явища (зокрема, неоднозначності), які зустрічаються як в синтаксисі, так і в семантиці, то система смислового аналізу повинна приводити вхідні тексти до деякого нормалізованого вигляду, перетворюючи різні, можливо неоднозначні, структури на вході у фіксовані, стандартні ідентифікатори змісту. Такого типу процедури нормалізації мови часто використовують словники і списки слів, які включають допустимі ідентифікатори змісту, причому для кожного ідентифікатора наводиться відповідне означення з тим, щоб регулювати і контролювати його використання. Слід зауважити, що до появи поняття "онтологія" (та розробки "онтологізованих" систем обробки ПМТ) процедури аналізу ПМТ рідко виходили за рамки аналізу одного речення. Це пояснюється тим, що проблема аналізу ПМТ досить складна і приходиться сильно обмежувати "свої запити" при спробі автоматизації такого роду аналізу, виконуючи деякий спрощений аналіз тексту (або накладати обмеження на стилі оброблюваних текстів).

Обмеження задачі аналізу. Конкретизація задачі аналізу спрощується до такої.

Словник S, про який говорилося вище, є тлумачним словником мови $L(X)$ (це може бути словник російської, української, англійської або будь-якої іншої природної мови).

Текст T складається з речень мови $L(X)$ і являє собою текст, який не включає ніяких символів, крім символів алфавіту X (тобто, T не включає формул, графіків, рисунків і т. п.).

Відношення γ складається із суперпозиції двох відношень $\gamma_1 * \gamma_2$, які виконуються послідовно. Відношення γ_1 означає розпізнавання належності слова до даної мови і перевірку правильності написання слова $t_{i_j} \in t_i$, де $t_i \in T$ у відповідності з написанням його в тлумачному словнику, тобто

$$\gamma_1(t_{i_j}) = \begin{cases} 1, & \text{якщо } t_{i_j} \in S; \\ 0, & \text{якщо } t_{i_j} \notin S. \end{cases}$$

Якщо слово $t_{i_j} \in t_i$ розпізнано в словнику S, то воно заноситься в словник T' правильних слів, а якщо це не так, то передбачається сигналізація про те, що дане слово відсутнє у словнику S і приймається рішення про додавання даного слова у словник або його виправлення (слово може бути неправильним, наприклад, внаслідок сканування тексту T).

Словники S і T' є вхідними даними для відношення γ_2. Змістовний сенс відношення γ_2 зводиться до того, що коли $\gamma_1(t_{i_j}) = 1$, то $\gamma_2(t_{i_j})$ визначає його граматичну одиницю мови (іменник – ім'я власне, іменник – загальний, займенник тощо), а також можливі флексії слова $t_{i_j} \in t_i$.

Областю інтерпретації тексту T є модель $A = (D, \Pi)$, де T – вихідний текст, можливо розширений деякою додатковою інформацією, а сигнатура Π визначається виходячи з тексту T внаслідок використання інформації про різні входження слова t_{i_j} в речення $t_i \in T$. При цьому обчислення відношення φ обмежується окремо взятим реченням $t_i \in T$, яке визначається кожним входженням слова t_{i_j} в текст T. У випадку трудності визначення предиката $\pi_i \in \Pi$, передбачається діалоговий режим обчислення $\varphi(\pi_i)$ і $\gamma(\varphi(\pi_i))$.

Розглянемо питання про засоби маніпуляції з текстовою інформацією. Представимо формальну алгебро-логічну мову, орієнтовану на обробку такого типу інформації.

Алгебраїчна система спискових структур

Побудова та опис такої мови виконується в два етапи. На першому описується алгебра спискових структур, а на другому – доповнення цієї алгебри до алгебраїчної системи спискових структур (АССС).

Алгебра спискових структур. Нехай $F(X)$ – вільна напівгрупа з одиницею над деяким скінченним алфавітом $X = \{x_1, x_2, ..., x_n\}$. Роль одиниці відіграє пусте слово e. Нагадаємо, що словом в алфавіті X називається довільна скінченна послідовність символів цього алфавіту. Довільне слово $p = y_1y_2 \dots y_m$ із $F(X)$ будемо називати **списком** елементів $y_1y_2 \dots y_m$, а самі елементи $y_i \in X$, $i = 1, 2, ..., m$, – складовими цього списку. При цьому елемент y_1 називається *початком*, а елемент y_m – *кінцем списку*. Якщо $p \in F(X)$, то число складових списку p називається його довжиною і позначається $l(p)$. Якщо p, q – два списки, то список (слово) q називається *початком* (*кінцем*) списку (слова) p, коли існує таке слово p', що $p = qp'$ ($p = p'q$). Два списки $p = s_1s_2...s_k$ і $q = t_1t_2...t_l$ рівні між собою, якщо $l = k$ і $s_i = t_i$, $i = 1, 2, ..., k$.

З теорії відомо, що $F(X)$ є алгеброю з однією бінарною операцією конкатенації (*conc*) і однією нульарною операцією (пусте слово e). Введемо в розгляд ще декілька функцій і операцій над списками, тобто над елементами множини $F(X)$ [3].

Нехай N – множина натуральних чисел і $p = y_1y_2 \dots y_m$ – довільне слово із $F(X)$, тоді

$head(p) = y_1$ ($head: F(X) \to F(X)$).

Іншими словами, функція $head(p)$ дає перший символ слова p. Безпосередньо з визначення цієї функції випливають такі її властивості:

$head(e) = e$, $head(y) = y$, якщо $y \in X$, $head(head(p)) = head(p)$.

$tail(p) = y_2 \dots y_m$ ($tail: F(X) \to F(X)$).

Очевидно, що $tail(e) = e$, $tail(y) = e$, якщо $y \in X$.

Зміст наведених нижче функцій випливає з їх визначення.

$add(p, i, x) = y_1 \dots y_i x y_{i+1} \dots y_m$, $0 \le i \le l(p)$.

$sub(p, i) = y_1 \dots y_{i-1} y_{i+1} \dots y_m$, $1 \le i \le l(p)$.

$dist(p, i) = (p_1, p_2)$, де $p_1 = y_1 \dots y_i$, $p_2 = y_{i+1} \dots y_m$, $0 \le i \le l(p)$.

$hl(p, i) = y_1 \dots y_i$, $0 \le i \le l(p)$.

$tr(p, i) = y_{i+1} \dots y_m$, $0 \le i \le l(p)$.

$push(p, x) = px = add(p, l(p), x)$.

$pop(p) = y_1 \dots y_{m-1} = sub(p, l(p))$.

Універсальна алгебра $G = (F(X), \Omega = \{conc, head, tail, e\})$ розширена операторами рекурсії та суперпозиції називається алгеброю спискових структур. Розширену алгебру спискових структур предикатом рівності та умовним оператором будемо називати алгебраїчною системою спискових структур.

Алгоритм морфологічного аналізу, записаний в АССС. Розглянемо приклад використання АССС для представлення алгоритму морфологічного аналізу слів природної мови. Нехай X означає алфавіт деякої природної мови, а $F(X)$ – множину слів скінченної довжини в алфавіті X. Вхідними даними алгоритму морфологічного аналізу є такі дані: D – словник мови в алфавіті X, F – словник закінчень, $\hat{S}$ – словник суфіксів. Морфологічний аналіз виконується для слова $w \in F(X)$ і при цьому вважається, що належність слова w до певного лексико-граматичного розряду (іменників, прикметників, дієслів тощо) вже відома, як тільки встановлено, що це слово або його корінь належать до словника D, в якому цей клас явно вказаний.

```
МОРФОЛОГІЯ (w,D, F, Ŝ)
begin
D: словник мови в алфавіті X;
F: словник закінчень;
Ŝ: словник суфіксів;
w : слово, що аналізується.
  if subword(D,w) = 1 then print(w,e,e) else if l(w) > 1 then okonch(w,1,e,e) else print(<<w>>).
end

okonch (p,i,q,r)
begin
 if subword(F,head(conv(p)))=1 then
   if subword(D, hl(p,l(p)-i)) = 1 then print(hl(p,l(p)-i),q,head(conv(p))r)
   else if l(hl(p,l(p)-i))=1 then print(<<p>>)
        else if continue(hl(p),l(p)-i))=1 then  /* закінчення має продовження */
                okonch (hl(p,l(p)-i-1),i+1,q,head(conv(hl(p,l(p)-i))r)
             else sufiks(hl(p,l(p)-i),i,q,r)
 else if subword(F,hl(r,l(p)-i))=1 then
         sufiks(hl(p,l(p)-i),i-1,q,hl(r,l(r)-i))
      else sufiks(p,1,e,e)
end

sufiks(p,i,s,f)
begin
 if subword(Ŝ,head(conv(p)))=0 then print(<<p>>)
 else if subword(D, hl(p,l(p)-i)) = 1 then print(hl(p,l(p)-i),s,f)
      else if l(hl(p,l(p)-i))<=1 then print(<<p>>)
           else if continue(hl(p),l(p)-i))=1 then  /* суфікс має продовження */
                   sufiks(hl(p,l(p)-i-1),i+1,q,head(conv(hl(p,l(p)-i))s,f)
                else print(<<p>>)
end.
```

Словники та ідентифікація слів при обробці текстової інформації. Розглянемо вільну напівгрупу $F(X)$ над деяким скінченним алфавітом $X = \{x_1, x_2, ..., x_m\}$. Нехай $p, q \in F(X)$ – довільні слова, де $l(p) < l(q)$. Необхідно знайти одне (перше) входження слова p в слово q або всі входження слова p в слово q. Задачі такого типу носять назву *задач ідентифікації* (у даному випадку задачі ідентифікації слів). Задачі ідентифікації – складова частина проблем, пов'язаних із редагуванням текстів, пошуком даних у базах даних і символьними обчисленнями. Розглянемо проблему пошуку слова p в слові q, одну з основних при розв'язуванні перелічених задач.

Оскільки слово p задане, а слово q може бути довільним, то задача ідентифікації слова p в слові q зводиться до побудови скінченного автомата A, який представляє регулярну мову, що має вигляд $L = \{X\} \cdot p = \{x_1 \vee ... \vee x_m\}p$. Очевидно, що дана мова регулярна, і тоді за теоремою синтезу існує автомат A, який представляє цю мову деякою множиною станів F. Зауважимо, що $p \in L$, оскільки $e \in \{X\}$. Більше того, слово p – найкоротше слово із L, яке необхідно знайти в слові q.

Побудова автомата A за словом p виконується за допомогою функції "відмов" g шляхом використання спрощеного алгоритму синтезу скінченного автомата [3]. Детально такий алгоритм описаний в [4]. Користуючись даним методом розв'язання задачі ідентифікації слів, можна виконувати пошук і заміну відразу кількох слів, тобто побудувати автомат, який акцептує мову $\{X\} \cdot (p_1 \vee p_2 \vee ... \vee p_k)$.

Деякі системи підготовки та редакції текстів використовують так звані spell-checking підсистеми. Опишемо коротко принцип побудови та роботу такої підсистеми. Якщо довільна скінченна множина слів $\check{S}$ вхідної напівгрупи $F(X)$ представляється скінченним автоматом, то цей факт можна застосувати для перевірки правопису слів. Дійсно, нехай множина $\check{S}$ відповідає множині слів словника (наприклад, англійської мови). Тоді, якщо є сумнів відносно правильності написання деякого слова p із $\check{S}$, то достатньо слово p подати на вхід автомата A, який представляє множину слів $\check{S}$ деякою множиною станів F. Якщо автомат A під дією слова p досягає одного із станів множини F, то слово p написане правильно, а якщо ні, то в слові є помилка. Зауважимо, що така перевірка слова p вимагає часу, пропорційного довжині $l(p)$ слова p. Розглянемо на прикладі роботу такого автомата.

Нехай $\check{S}$ = {abase, abash, abat, abbey} – підмножина слів англійської мови (словник). Відповідний автомат, який представляє слова із $\check{S}$, показаний на рис. 1. В цьому автоматі 0 – початковий стан, а 6, 7, 10 – заключні стани.

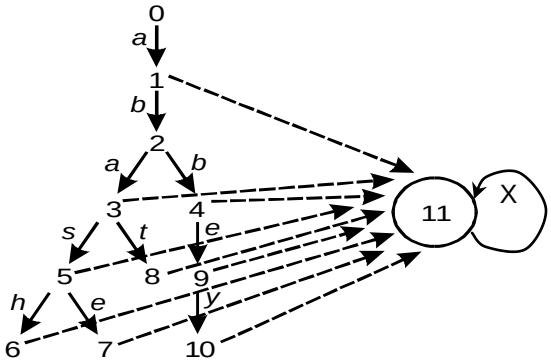

Рис. 1. Автомат A, який представляє слова із $\check{S}$

Для того, щоб застосувати автомат A для перевірки правопису слів із множини $\check{S}$ та ідентифікації помилок у словах, введемо ще один стан – стан 11 і переходи (показані пунктирами) в цей стан з кожного іншого стану, за винятком станів із F, під дією букв англійського алфавіту, для яких немає переходів в A.

Тепер для перевірки правильності написання конкретного слова, наприклад *abbei*, подаємо його на вхід автомата A. Під дією цього слова попадаємо в стан 11, який не належить множині заключних станів F, отже, слово *abbei* написане неправильно. Перехід у стан 11 був виконаний зі стану 9 під дією вхідного символу *i*, тому цей символ ідентифікується як помилковий.

Зауважимо, що подання словника у вигляді скінченного автомата дає значну економію пам'яті обчислювальної системи, що позитивно позначається на всьому процесі роботи з таким словником.

Представлення та обробка природномовних текстів

Знання, які одержані внаслідок аналізу природномовного тексту, подаються у вигляді

– відношень, занесених до бази знань (БЗ);

– онтологій, які відображають залежності між поняттями (концептами) та самими відношеннями в базі знань.

Подання знань у вигляді відношень БЗ та засоби пошуку наслідків із деяких фактів, притаманні БЗ, дають можливість формальними методами виконувати перевірки несуперечності чи виконуваності множини формул. У системах гільбертовського типу використання формальних методів (тобто, систем, в яких доведення будуються формальним способом з аксіом шляхом застосування правил виведення) не має якого-небудь задовільного розв'язання. Справа в тому, що системи гільбертовського типу не є структурованими, а це означає, що для збору необхідної інформації по одному єдиному об'єкту необхідно переглянути всю множину логічних формул, яка знаходиться у системі (як правило такою системою є база даних). На жаль, цим недоліком страждають всі формальні логічні системи гільбертовського типу. З метою ліквідації цього недоліку було запропоновано графічне представлення формул та їх аргументів, яке служить глобалізації й структурованості інформації. Основою графічного представлення є *концептуальні графи* (*КГ*) і більш складні структури – *семантичні мережі* (*СМ*). Таке представлення дає можливість візуалізувати модель природномовної картини світу, до якої належить проблема, що розглядається. Крім того, ця візуалізація дозволяє отримувати, в разі необхідності, весь процес доведення.

Концептуальні графи і семантичні мережі. Неформальні означення і приклади. Введемо неформальні означення вищенаведених понять КГ і СМ, а також наведемо приклади для ілюстрації цих понять [5, 6].

Концептуальним графом називається позначений дводольний орієнтований граф $G = (V_1 \cup V_2, E)$, *де* $V_1 \cap V_2 = \varnothing$, *вершини із* V_1 *позначені іменами предикатів, а вершини із* V_2 *– іменами аргументів. Дуги графа з'єднують вершини, позначені іменами предикатів, з вершинами, які позначені іменами аргументів. Вершини із множини* V_1 *називаються вузлами-предикатами, вершини із* V_2 *– вузлами-концептами, а самі предикати – концептуальними предикатами.*

З наведеного означення випливає, що КГ повинен задовольняти таким умовам:

– число дуг, які зв'язують вузли-предикати з вузлами-концептами, дорівнює арності предиката (тобто, дорівнює числу його аргументів);

– всі вузли-предикати, які позначені символом одного і того ж концептуального предиката, мають однакову арність;

– всі дуги, які з'єднують вузли-предикати і вузли-концепти в КГ, упорядковані від 1 до n, де n – арність предиката.

Якщо логічна мова, що використовується, є типізованою, то його об'єктам приписані певні типи і з кожним концептуальним предикатним символом пов'язується кортеж типів *<a,b,...,c>*, який називається сигнатурою цього предиката.

КГ не випадково вибрані як структури даних для представлення предикатів та їх аргументів. Ці структури пройшли тривалу апробацію у системах баз даних та знань і зарекомендували себе з найкращого боку. Це проявляється перш за все в ефективності виконання операцій на таких структурах даних. Операції на множині КГ виконуються з метою ефективної побудови більш складних структур – семантичних мереж.

Семантичною мережею (*СМ*) *називається об'єднання заданої множини концептуальних графів разом з описом їх взаємозв'язків та зануренням в контекст області суджень.*

Для підкріплення цих, не зовсім формальних означень розглянемо приклади, які ілюструють поняття концептуального графа та семантичної мережі.

Приклад. Розглянемо речення "КГ описує семантику деякого висловлювання". Концептуальний граф $G^1 = (V_1 \cup V_2, E)$, що відповідає цьому реченню, показаний на рис. 2.

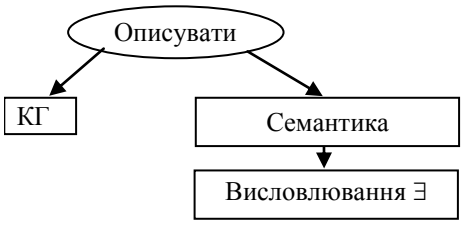

Рис. 2. Графічне зображення концептуального графа G^1

У цьому графі вершини "КГ" і "Висловлювання" належать до множини V_1, а вершина "Описувати" належить до V_2. Вершина "Семантика" є допоміжною, а елементами множини ребер даного КГ є ("Описувати", "КГ"), ("Описувати", "Висловлювання"). Подання цього речення у вигляді формули логіки предикатів має вигляд: P(a,b), де P – символ предиката з іменем "Описувати", a, b – аргументи предиката P з іменами "КГ" і "Висловлювання" відповідно.

Зауважимо, що подання КГ може мати різні реалізації, які можуть включати певні уточнення, характеристики контексту чи уточнення семантики (як у попередньому прикладі вершина "Семантика").

В той час як КГ представляють одну логічну формулу, СМ представляє сукупність КГ, занурених у взаємозв'язки та спільний контекст області суджень. СМ будуються із КГ за правилами кон'юнкції та спрощення.

Правило кон'юнкції (*ПК*): Якщо вузол-концепт c_1 в G^1 ідентичний вузлу-концепту c_2 в G^2, то G отримується шляхом вилучення c_2 і з'єднанням з c_1 всіх вузлів, що зв'язували c_2 в G^2.

Правило спрощення (*ПС*): Якщо КГ G після з'єднання включає два ідентичних (з'єднаних з одними і тими ж вузлами-концептами) вузли, то можна вилучити один з них разом з дугами, що зв'язують його з цими вузлами.

Приклад. Візьмемо перший КГ (G^1), наведений у попередньому приклад, а другий КГ (G^2) нехай відповідає реченню "КГ описує семантику речення з однією граматичною основою". Графічне зображення цього КГ представлене на рис. 3.

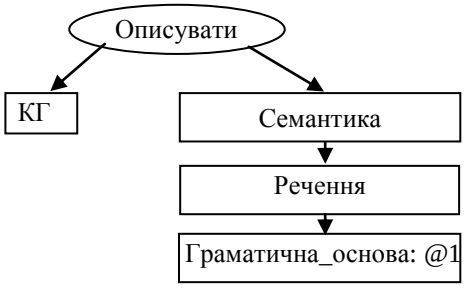

Рис. 3. Графічне зображення концептуального графа G^2

Приклад. Розглянемо СМ, яка побудована з КГ G^1 і G^2 за правилами ПК і ПС. Внаслідок застосування ПК отримуємо $СМ^{1,2}$ для даних графів, яка показана на рис. 4. Інколи таку СМ називають об'єднаним КГ.

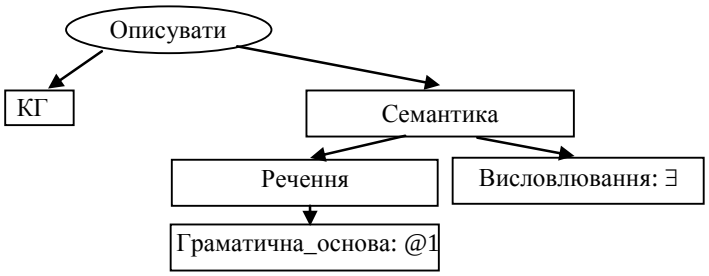

Рис. 4. Графічне зображення семантичної мережі СМ[1,2]

Подання семантичної моделі ПМО в залежності від типів задач, що потребують розв'язання (і відповідних КГ) може мати різний ступінь деталізації. При цьому розрізняють дві задачі формально-логічного подання ПМО:

перша належить до внутрішньомовної обробки, за якої результат семантичного аналізу представляється найбільш повними логічними виразами у відповідному логічному базисі. Їх формування виконується паралельно з процедурою зняття лексичної неоднозначності, яка потребує деталізації КГ і експліцитного представлення відповідних контекстних залежностей.

друга належить до позамовної обробки, до етапу побудови бази знань предметної області (БЗ ПдО), а точніше – до побудови бази правил логічного виведення. Така база знань повинна мати короткі правила, які дають можливість реалізувати ефективне виведення (з точки зору швидкодії та пам'яті). Для цієї задачі слід використовувати максимально спрощені КГ і відповідні їм СМ.

Для представлення та зберігання семантичних мереж доцільно використовувати багатовимірний метод доступу [7], який реалізовано в інструментальному комплексі ArM32. Цей метод доступу базується на основі Багатодоменної Інформаційної Моделі (БІМ) [8]. Елементи ArM32 організовані в ієрархії нумерованих інформаційних просторів зі змінними рангами. Немає ніяких обмежень на ранг простору. До кожного елемента можна отримати доступ за допомогою відповідної багатовимірної просторової адреси, яка представлена масивом координат. Багатодоменна інформаційна модель є кроком у процесі розвитку інструментів для організації баз даних. Її головна ідея полягає у можливості фактично необмеженого доступу до багатовимірних інформаційних структур. В БІМ існують дві головні компоненти - нумеровані інформаційні простори й базові інформаційні елементи.

Базовий інформаційний елемент представляє собою довільну довгу послідовність (рядок) машинних кодів (байти). В ArM32 довжина рядка може змінюватися від нуля до 1GB. Немає ніякої межі для кількості рядків в архіві, але їх повна довжина плюс внутрішні індекси не може перевищувати межу для довжини єдиного файлу операційної системи.

Основні інформаційні елементи об'єднані в нумеровані набори, які називаються нумерованими інформаційними просторами рангу 1. Нумерований інформаційний простір рангу n – множина, елементами якої є впорядковані інформаційні простори рангу n-1. ArM32 дозволяє використовувати інформаційні простори різних рангів в одному архіві (файлі). Головними операціями ArM32 є читання, запис, додавання, вставка, переміщення, заміна й видалення основного інформаційного елемента, або будь-якої його частини. Нумеровані інформаційні простори в ArM32 впорядковані і головні операції з просторами враховують їх впорядкованість. ArM32 підтримує багатопотоковий паралельний доступ до інформаційної бази в режимі реального часу.

Дуже важлива особливість ArM32 – можливість не займати дисковий простір для порожніх структур (елементи або простори). Дійсно, тільки непорожні структури необхідно зберігати у зовнішній пам'яті. Можливості ArM32 є прийнятними для побудови інформаційної основи ВЛРСС. ArM32 реалізований у вигляді DLL з використанням середовища розробки программ DELPHI 2003. Наведемо короткий перелік основних класів ArM32: TZeroBlock – указує на блоки вільного дискового простору в архіві; TSpaceBlock містить прямі й непрямі покажчики на інші простори; TIndexBlock містить 169 трьохбайтових покажчиків на блоки з архіву; TDomainBlock включає інформацію про список даних і чотири області непрямих покажчиків TIndexBlock, які ведуть до блоків покажчиків з елементами; TelementBlock містить 126 чотирьохбайтових покажчиків на елементи (номер блоку + логічний зсув); TDataBlock містить 503 байти даних елементів областей. Кожен елемент складається з довжини (від 1 до 4 байтів) і значення елемента (від 1B до 1GB) заданої довжини.

Багатовимірні інформаційні простори уможливлюють ефективне створення складних інформаційних структур, використовуючи маленьку кількість ресурсів, що є дуже важливим для ВЛРСС. Побудова онтологій предметних областей і метаонтологій на основі автоматизованого аналізу ПМТ служить основою у роботі користувачів віртуальних лабораторій зі знаннями у різних предметних областях.

Висновки

В роботі наведена формальна постановка задачі добування знань з природномовних об'єктів. Для маніпуляції, аналізу та трансформації текстів введено поняття алгебраїчної системи спискових структур, за

допомогою якої виконується оброблення текстів на рівні фізичного представлення. Для представлення та зберігання семантичних мереж запропоновано використовувати реалізацію багатовимірного методу доступу в інструментальному комплексі ArM32. Знання, описані експертом та одержані внаслідок аналізу ПМТ, подаються у вигляді бази знань та онтології предметної області. Це дозволяє запровадити віртуальну організацію роботи користувача зі знаннями з різних предметних областей.

Подяка

Робота виконана за фінансової підтримки Міністерства освіти і науки України в рамках спільного Українсько-Болгарського проекту № 145 / 23.02.2009 "Розробка розподілених віртуальних лабораторій на основі прогресивних методів доступу для підтримки проектування сенсорних систем" і Болгарського національного наукового фонду в рамках спільного Болгарсько-Українського проекту D 002-331 / 19.12.2008 з тією ж назвою.

1. *Oleksandr Palagin, Volodymyr Romanov, Krassimir Markov, Vitalii Velychko, Peter Stanchev, Igor Galelyuka, Krassimira Ivanova, Ilia Mitov.* Developing of Distributed Virtual Laboratories for Smart Sensor System Design Based on Multi-dimensional Access Method. // International Book Series "INFORMATION SCIENCE & COMPUTING". Number 8 FOI ITHEA, Sofia, Bulgaria. – 2009. – P. 155–161.
2. *Палагин А., Романов В., Марков К., Величко В., Галелюка И., Иванова К., Станчев П., Митов И., Станева М.* Базовая онтология распределенной виртуальной лаборатории проектирования сенсорных систем // Knowledge – Dialogue – Solution: International book series "INFORMATION SCIENCE & COMPUTING". Number 15 FOI ITHEA, Sofia, Bulgaria. – 2009. – P. 19–23.
3. *Кривий С.Л.* Дискретна математика: вибрані питання. – Київ: Видавничий дім «Києво-Могилянська академія», 2007. – 570 с.
4. *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. – М.: Мир, 1979. – 535 с.
5. *Логический* подход к искусственному интеллекту. От классической логики к логическому программированию / А. Тейз, П. Грибомон, Ж. Луи и др. – М.: Мир, 1990. – 429 с.
6. *Логический* подход к искусственному интеллекту. От модальной логики к логике баз данных / А. Тейз, П. Грибомон, Г. Юлен и др. – М.: Мир, 1998. – 494 с.
7. *Kr. Markov.* A Multi-domain Access Method. // Proceedings of the International Conference on Computer Based Scientific Research. Plovdiv, 1984. – P. 558–563.
8. *Markov K.* Multi-Domain Information Model. // J. Information Theories and Applications. – 2004. – 11, N 4. – P. 303–308.