Результати, наведені в [11], показали, що відносно критерію, який оцінює якість відновленого після трирівневого ВП зображення без використання детальної інформації, запропоновані в даній роботі фільтри забезпечують ефективність на рівні кращих аналогічних конструкцій.

1. *S.Malllat*, A theory for multiresolution signal decomposition: The wavelet representation, IEEE Trans. Patt. Anal. Mach. Intel., vol. 11, no. 7, pp.674-693, 1989.
2. *M.Vetterli* and *C. Herley*, Wavelets and filter banks: Theory and design, IEEE Trans. on Signal Proc., vol. 40, no 9, pp. 2207-2232, Sept. 1992.
3. *W. Sweldens*, The construction and application of wavelets in numerical analysis, Ph.D. dissertation, Katholieke Universiteit Leuven, Belgium, May 1995.
4. *Новиков И.Я., Стечкин С.Б.* Основы теории всплесков // Успехи мат. наук. 1998. Т.53. № 6(324). С. 53-128.
5. *G.Strang*. Wavelets and dilation equation: A brief introduction, SIAM Rev., vol.31, pp. 614-627, Dec. 1989.
6. *I. Daubechies*, Orthonormal bases of compactly supported wavelets, Commun. Pure and Appl. Math., vol. XLI, pp. 909-996, 1988.
7. *Шенен П., Коснар Г., Гардан И., Робер Ф., Витомски П., Кастельжо П.* Математика и САПР: в 2-х кн. Кн. 1. М., 1988.
8. *Баскаков С.И.* Радиотехнические цепи и сигналы: Учеб. для вузов по спец. "Радиотехника". 2-е изд., перераб. и доп. М., 1988.
9. *K. R.Castleman*, Digital Image Processing, Prentice Hall, 1996.
10. *O. Rioul*, Simple regularity criteria for subdivision schemes, SIAM J. Math. Analysis, vol. 23, pp. 1544-1576, Nov.1992.
11. *Коваль О., Русин Б.* Вибір вейвлету та аналіз степені регулярності біортогональних фільтрів, побудованих на основі теорії полюсів, для стиску зображень. Праці 5-ої Всеукр. Міжнар. конф. "УкрОБРАЗ - 2000", 27 лист. - 1 гр. 2000 р. С. 205-208.

УДК. 681.3

ПОБУДОВА ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМ НА ОСНОВІ ОБ'ЄКТНО-ОРІЄНТОВАНОЇ МОДЕЛІ ПРЕДСТАВЛЕННЯ ЗНАНЬ

© В. Литвин
Національний університет "Львівська політехніка"

Розглядається об'єктно-орієнтована модель представлення знань. Вводиться поняття схеми об'єктів. На основі запропонованої моделі описано процес логічного виведення.

This paper describes object-oriented model of knowledge representation. It introduces the concept of schema object. The paper consider the logical inference process based on proposed model.

Об'єктно-орієнтовані системи керування даними (*OODMS*) – це підхід, який, на думку багатьох вчених, забезпечує вирішення проблеми розробки складних систем. Однак у даному напрямку недостатньо формальних досліджень і загальноприйнятих понять; не існує універсальної об'єктно-орієнтованої моделі даних. Натомість існує багато альтернативних рішень відносно *OODMS*, прийняття яких багато в чому залежить від множини потреб проблемної області прикладної програми.

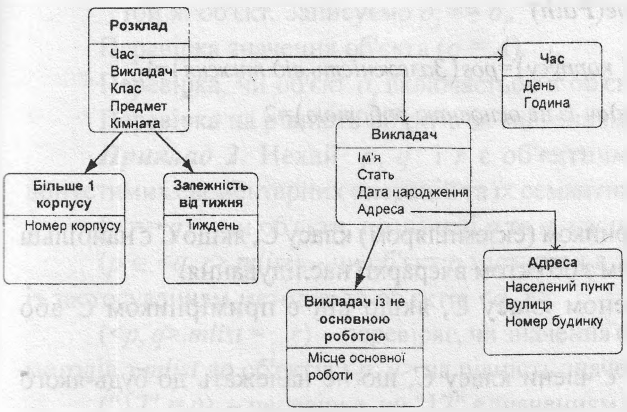

Рис. 1. Приклад об'єктно-орієнтованої схеми даних.

Об'єктно-орієнтована модель даних має дві важливі характеристики, які надають значних переваг при використанні цієї моделі. Одна з них – безпосередня можливість моделювання, будування складних об'єктів з більш простих за допомогою конструкторів. Розрізняють такі конструктори: кортежів, множин, мультимножин, списків, масивів.

Друга – організація класів (типів) в ієрархію наслідування. Крім того, щоб об'єктно-орієнтована модель була гнучкою у використанні, вона повинна підтримувати декларативну мову запитів високого рівня.

В об'єктно-орієнтованій концепції яскраво виділяються дві її складові: структура моделі, що базується на атрибутах об'єкта (база даних) та її поведінка (база знань).

Основні означення

Розглянемо основні поняття, що відображають структурну суть моделі. Для цього введемо такі позначення: C - клас; C^* - множина класів, яка наслідується із C.

Нехай задана деяка схема об'єктів (наприклад, рис. 1). На даному рисунку тонкими лініями зображено вкладення класів (агрегація), жирними лініями - наслідування між класами.

Шлях *Path* до атрибуту A_n визначимо як послідовність $C_1A_1A_2\ldots A_n$, $n\geq 1$ де C_1 - верхній клас у схемі (корінь схеми); A_1 - атрибут класу C_1; A_i - атрибут класу C_i, такого що C_i є доменом атрибуту A_{i-1} класу A_{i-1}, $1 < i \leq n$.

Значення n назвемо довжиною шляху та позначимо $len(Path) = n$.

Означення 1. Класом шляху назвемо множину класів, через які проходить заданий шлях, тобто $class(Path) = C_1 \cup \{ C_i | C_i$ є доменом атрибуту A_{i-1} з класу C_{i-1}, $1 < i \leq n\}$.

Означення 2. Границею шляху назвемо множину класів, до якої входять класи із множини класу шляху $class(Path)$ разом із усіма класами, що наслідуються з них, тобто $scope(Path) = \cup_{C_i \in class(Path)} C_i^*$; клас C_1 є коренем $scope(Path)$.

Кожному класу ІС із $scope(Path)$ визначимо номер позиції i (позначаємо pos).

Позицію i визначимо як номер класу C_i, до якого згідно з ієрархією наслідування належить клас C, де $C_i \in class(Path)$.

Приклад 1.Для прикладу розглянемо шлях, зображений на рис. 1.,

$$Path = Розклад.Викладач.Адреса.Вулиця$$

$$len(Path) = 3$$

$$classPath = \{Розклад, Викладач, Адреса\}$$

$$scope(Path) = \{Розклад, Більше\ 1\ корпусу, Залежність\ від\ тижня, Викладач, Викладач\ із\ не\ основною\ роботою, Адреса\}$$

Клас *Розклад* є коренем $scope(Path)$

$$pos(Розклад)=pos(Більше\ 1\ корпусу)=pos(Залежність\ від\ тижня)=1$$

$$pos(Викладач) = pos(Викладач\ із\ не\ основною\ роботою)=2$$

$$pos(Адреса) = 3$$

Означення 3. Об'єкт o є примірником (екземпляром) класу C, якщо C є найбільш спеціалізованим класом асоційованим з об'єктом в ієрархії наслідування.

Означення 4. Об'єкт o є членом класу C, якщо він є примірником C або примірником деякого підкласу C.

Тобто примірниками класу C є члени класу C, що не належать до будь-якого підкласу C.

Об'єкт o подамо у вигляді трійки $o = (id, C, val)$, де id – ідентифікатор об'єкта; C – клас, до якого належить об'єкт; val – значення об'єкта. Значення може бути атомарним (цілі числа, символи, символьні рядки, логічні змінні та дійсні числа); множина (набір ідентифікаторів об'єктів); список або кортеж (набір атрибутів). Множини представляють набір об'єктів реального світу; кортежі – властивості об'єктів, а списки - порядок, який є в реальному світі.

Модель проблемної області інтерпретує класи. Визначення інтерфейсу класу відбувається через методи, що є шаблоном для всіх об'єктів, які є екземплярами класу. Методи носять ім'я функцій, параметрами і результатом якої є об'єкти.

Метод – це відображення деякої підмножини аргументів об'єктів в новому об'єкті. Кожний метод має форму

$$C_1 \times C_2 \times ... \times C_n \rightarrow C_{result},$$

де C_1, C_2,..., C_n класи, до яких належать об'єкти-аргументи; C_{result} – клас об'єкта-результату.

Всі класи формують дерево, де вершина кореня представляє найбільш загальний клас об'єктів, і будь-який інший клас може мати батьків. Підкласи наслідують поведінку від своїх батьків, а також можуть визначати власні додаткові методи. Отже, дерево класів забезпечує поліморфізм включення, який дозволяє об'єкту класу C використовуватись у будь-якому контексті, що визначається суперкласом C.

Об'єкти інкапсулюють стан і поведінку. Стан об'єкта зафіксовано його значенням.

Методи, визначені на класі, об'єкт якого є екземпляром, визначають поведінку об'єкта. Поведінку виявляють, застосовуючи метод до об'єкта. Результат прикладної програми методу – деякий інший об'єкт.

Запис $<o_1, o_2,\ldots, o_n>m_1m_2\ldots m_m$ використовується для визначення методів над об'єктами. Запис $<o_1, o_2,\ldots, o_n>mlist$ буде використовуватись, коли список імен методів незначний.

Елементарні операції над об'єктами

Визначимо елементарні операції над об'єктами, які повертають логічний результат:

Перевірка, чи два об'єкти o_i та o_j однакові за *id*. Тобто вони визначають один і той ж об'єкт. Записуємо $o_i == o_j$.

Перевірка значення об'єкта ($o = A$).

Перевірка, чи об'єкт o_i включається в об'єкт o_j $o_i \subseteq o_j$ ($o_i \in o_j$).

Перевірка на рівність об'єктів за значеннями. Записуємо $o_i =_{\{\}} o_j$.

Приклад 2. Нехай p, q і r є об'єктними змінними. Наведемо приклади допустимих елементарних операцій та їх семантики:

($p == q$) – чи об'єкти, позначені p та q, насправді один і той самий об'єкт?

($p \in <q, r>.mlist$) – чи об'єкт p міститься в значенні набору об'єкта, отриманого із застосуванням методів до об'єктів $<q, r>$.

($<p, q>.mlist =_{\{\}} r$) – перевіряє, чи значення об'єкта, отриманого із застосуванням методів з *mlist* до об'єктів $<p, q>$ на рівність значенню об'єкта, позначеного r.

("17" = p) – перевірка, чи "17" є значенням об'єкта, позначеного p.

Алгебра об'єктів

Для підтримки складних об'єктів необхідні відповідні оператори для роботи з такими об'єктами. Тобто операції над складними об'єктами повинні транзитивно розповсюджуватись на всі його компоненти. Прикладами можуть бути вибірка або знищення складного об'єкта загалом. Користувач повинен мати змогу визначати додаткові операції із складними об'єктами. Для реалізації цієї можливості необхідні деякі засоби, які забезпечує система, такі як два типи посилань ("є частиною" і "загальна").

Нехай θ - оператор з алгебри. Для визначення алгебри використовуємо нотацію $P\theta <Q_1\ldots Q_k>$. P і Q_i визначають множину об'єктів. Бінарну операцію будемо записувати у вигляді $P\theta Q$.

Те, що $o \in P$, будемо записувати у вигляді $P(o)$.

Означення 5. Об'єднанням двох множин об'єктів P та Q назвемо множину $P \cup Q$, таку що містить об'єкти, які належать до множини P або Q, тобто $P \cup Q = \{o | P(o) \vee Q(o)\}$.

Означення 6. Перетином двох множин об'єктів P та Q назвемо множину $P \cap Q$, таку що містить об'єкти, які належать до множини P та Q одночасно, тобто $P \cap Q = \{o | P(o) \wedge Q(o)\}$.

Означення 7. Різницею двох множин об'єктів P та Q назвемо множину $P - Q$, таку що містить об'єкти, які належать до множини P, але не належать до множини Q, тобто $P - Q = \{o | P(o) \wedge \neg Q(o)\}$.

Операцію перетину можна визначити через різницю так $P \cap Q = P - (P - Q)$.

Операція вибору: $P\sigma_{I}<Q_1\ldots Q_k>$ повертає множину об'єктів p з P, які входять у кожний вектор вигляду $<p,q_1\ldots q_k> \in P\times Q_1\times\ldots\times Q_k$, і задовольняють відношення F. Це еквівалентно наступному: $\{p|\ P(p)\wedge Q_1(q_1)\wedge\ldots\wedge Q_k(q_k)\wedge F(p,q_1\ldots q_k)\}$.

Тобто оператор вибору завжди повертає підмножину з P.

Приклад 3.Покажемо процес виконання операцій на прикладі схеми об'єктів, зображеної на рис. 2.

"*Знайти всіх осіб, які живуть в місті, де знаходиться гімназія?*". Операція вибору запишеться у вигляді: *Особа*σ_{F}<*Гімназія*>, де F відношення вигляду F = (<*p*>.*адреса* == <*a*>.*адреса*); *a* вказівник на клас *Гімназія*, *p* вказівник на клас *Особа*.

Генерація $Q_1\gamma'_{F}<Q_2\ldots Q_k>$. Результатом є об'єкти t, що разом з набором об'єктів $<q_1\ldots q_k> \in Q_1\times\ldots\times Q_k$, задовольняють відношення F. Тобто $\{t|\ Q_1(q_1)\wedge\ldots\wedge Q_k(q_k)\wedge F(t,q_1\ldots q_k)\}$.

Приклад 4.$Q\gamma_{p\in\ q,r\ \ mlist}<R>$. Результат: $\{p|\ Q(q)\wedge\ldots\wedge R(r)\wedge p\in<q,r>.mlist\}$.

Приклад 5.Розглянемо приклади генерації для рис. 2. "*Повернути всі предмети, що викладає директор гімназії*". *Гімназіяγ'*$_{F}$<>, де F = ($t\in$<*a*>.*директор.викладає*). *a* вказівник на клас *Гімназія*; шлях <*a*>.*директор.викладає* визначає множину предметів.

"*Знайти всі міста, де є гімназія і в яких живуть викладачі цієї гімназії*". *Гімназіяγ'*$_{F}$<> де F = ($x\in$<*a*>.*викладачі* $\wedge\ t$==<*x*>.*адреса*). У даному прикладі *a* знову вказівник на клас *Гімназія*, шлях <*a*>.*викладачі* визначає множину викладачів. Шлях <*x*>.*адреса* визначає адресу, за якою живе викладач.

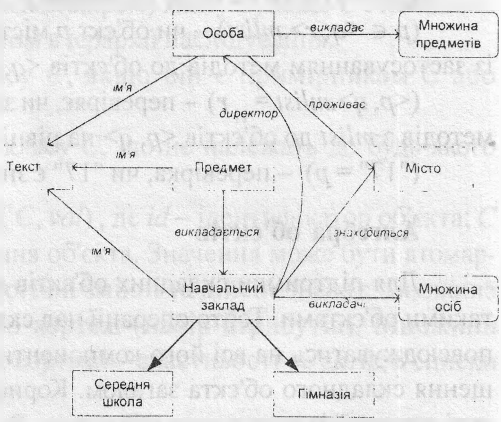

Рис. 2. Приклад простої схеми об'єктів

Відображення $Q_1\rightarrow_{mlist}<Q_2\ldots Q_k>$. Визначає послідовність методів з *mlist* для кожного об'єкта $q_1\in Q_1$, використовуючи об'єкти з $<Q_2\ldots Q_k>$ як параметри до методів в *mlist*. Результатом є множина об'єктів, отримана від кожної послідовності. Якщо нема методів, що використовують якісь параметри, то замість $<Q_2\ldots Q_k>$ будемо писати < >.

Відображення є спеціальним випадком оператору генерації, який можна записати у вигляді: $\{t|Q_1(q_1)\wedge\ldots\wedge Q_k(q_k)\wedge t==<q_1\ldots q_k>.mlist\}$.

Поняття схеми об'єктів

Об'єкти та класи співвідносяться між собою, утворюючи єдину структуру, яку будемо називати схемою та позначатимемо *Sch*. Дане співвідношення є ієрархічним (агрегація або наслідування) і (або) утворюється через визначення поведінки об'єктів різних класів.

Означення 8. Схема, в якій використовуються лише об'єкти, називається мікросхемою. Її позначатимемо $oSch = \{o_1, o_2, \ldots, o_k\}$.

Означення 9. Схема, в якій використовуються лише класи, називається макросхемою. Її позначатимемо $CSch = \{C_1, C_2, \ldots, C_k\}$.

Найчастіше вживаються змішані схеми, які є композицією мікро- та макросхем. У цих схемах, щоби не розділяти окремо клас C та об'єкт o з цього класу $o \in C$, будемо позначати їх однією буквою z. Тоді множину усіх елементів, що входять до схеми, запишемо у вигляді $Z = \{z_1, z_2, \ldots, z_s\}$.

Операції над схемами добре виконувати, оперуючи з їх аналітичним виразом. Розглянемо множину $Q = \{q_1, q_2, \ldots, q_p\}$ усіх відношень, що визначені у схемі. Тобто $Q = \{m_{11}, \ldots, m_{1m_1}, \ldots, m_{s1}, \ldots, m_{sm_s}, hier_1, \ldots, hier_p\}$, де $M = \{m_{i1}, \ldots, m_{im_i}$ - множина методів визначена для z_i, $Hier = \{hier_1, \ldots, hier_p\}$ - множина усіх ієрархічних відношень, визначена у схемі. Кількість елементів у множині Q дорівнює:

$$p = m_1 + \ldots + m_s + hier_p.$$

Також Q будемо записувати у вигляді: $Q = \{M_1, M_2, \ldots, M_k, Hier\}$.

Тоді будь-яку схему можна представити як

$$Sch = \{z_1, \ldots, z_s, Fz_1, \ldots, Fz_s\},$$

де Fz_i - підмножина елементів із Z, з якими елемент z_i зв'язаний відношеннями із Q.

В загальному вигляді схему можна розглядати як пару $Sch = (Z, F)$, де F - відображення Z в Z, зважене відношеннями із Q:

$$F : Z \underset{Q}{\rightarrow} Z.$$

Останній короткий запис будемо використовувати для визначення операцій над схемами.

Об'єднання схем. Нехай задані схеми $Sch_1 = (Z_1, F_1)$, $Sch_2 = (Z_2, F_2)$, …, $Sch_n = (Z_n, F_n)$. Схема $Sch = (Z, F)$ називається об'єднанням схем Sch_i, $i = 1, 2, \ldots, n$ і позначається $Sch \cup Sch_i$, якщо $Z \cup Z_i$ і для будь-якого $z \in Z$, $Fz = \cup F_i z$, $i = 1, 2, \ldots, n$.

Перетин схем. Нехай задані схеми $Sch_1 = (Z_1, F_1)$, $Sch_2 = (Z_2, F_2)$, …, $Sch_n = (Z_n, F_n)$. Схема $Sch = (Z, F)$ називається перетином схем Sch_i, $i = 1, 2, \ldots, n$ і позначається $Sch \cap Sch_i$, якщо $Z \cap Z_i$ і для будь-якого $z \in Z$, $Fz = \cap F_i z$, $i = 1, 2, \ldots, n$.

Означення 10. Схема $Sch^* = (Z^*, F^*)$ називається підсхемою $Sch = (Z, F)$ і позначається $Sch \subseteq Sch^*$, якщо $Z \subseteq Z^*$ і для будь-якого $z \in Z^*$, $Fz \subseteq Fz^*$.

Доповнення схеми. Нехай задані схеми $Sch_1 = (Z_1, F_1)$ і $Sch_2 = (Z_2, F_2)$ такі, що Sch_2 є підсхемою Sch_1, тобто $Sch_2 \subseteq Sch_1$. Схема $Sch_3 = (Z_3, F_3)$ називається доповненням схеми Sch_2 до схеми Sch_1, якщо $Sch_1 = Sch_1' \cup Sch_2'$, де $Sch_1' = (Z_1', F_1')$, $Sch_2' = (Z_2', F_2')$ і виконуються такі умови:

$$Z_1' = Z_1 \setminus Z_2,\ \forall z \in Z_1',\ F_1' z \subseteq F_1 z,\ Z_2' = Z_1'' \cup Z_2'';$$

де $Z_1'' \subseteq Z_1'$, $Z_2'' \subseteq Z_2'$, причому кожний елемент $z \in Z_2''$ зв'язаний хоча б одним відношенням з об'єктами із Z_2:

$$\forall z \in Z_2',\ F_2' z \subseteq F_1 z.$$

Включення схем. Визначимо операцію включення мікросхеми до макросхеми. Спочатку зробимо це для простіших схем, що складаються із двох елементів і одного відношення між ними. Нехай $CSch_0 = (C_1, C_2, F_0C_1, F_0C_2)$, де $F_0C_1 = \{q_1C_2\}$; $F_0C_2 = \{\varnothing\}$.

Мікросхема $oSch_0 = (o_1, o_2, F_0o_1, F_0o_2)$ де $F_0o_1 = \{q_1o_2\}$; $F_0C_2 = \{\varnothing\}$ називається включеною в $CSch_0$, якщо

$$o_1 \in C_1,\ o_2 \in C_2,\ F_1o_1 \subseteq F_0C_1,\ F_1o_2 \subseteq F_0C_2\ .$$

Довільна мікросхема *oSch* називається включеною в макросхему *CSch*, якщо будь-яка підсхема мікросхеми, що складається з двох об'єктів і одного відношення між ними, включена в макросхему. В результаті виконання операції включення відношення, що існують між класами макросхеми, можуть бути перенесені з конкретизацією на об'єкти мікросхеми.

Означення 11. Дві схеми $Sch_i = (Z_i, F_i)$ і $Sch_j = (Z_j, F_j)$ називаються структурно-еквівалентними, якщо вони відрізняються одна від однієї тільки перепозначенням елементів або відношень.

Конкретизація схем. Ця операція за своїм характером близька до операції включення схем із збереженням відношень. В її основі лежить механізм генерації мікросхем структурно-еквівалентних макросхемі (або її фрагменту), що включаються в неї. У простішому випадку генерація полягає у виділенні елементів із множини об'єктів і відношень макросхеми і побудова з них мікросхеми, структурно-еквівалентній макросхемі, що включаються в неї. Аналітично ця операція визначається таким чином. Нехай $CSch = (C_1, \ldots, C_i, \ldots, C_n, F_0C_1, \ldots, F_0C_i, \ldots, F_0C_n)$ задана макросхема. Мікросхема $oSch = (o_1, \ldots, o_i, \ldots, o_n, F_0o_1, \ldots, F_0o_i, \ldots, F_0o_n)$ називається результатом конкретизації $CSch$, якщо

$$o_i \in C_i,\ Fo_i \subseteq FC_i,\ i = 1,\ldots,n\ .$$

Процес розв'язування задачі

Основу процесу розв'язування задачі становлять три елементи: інформаційна дошка, знання та інтерпретатор. Інформаційна дошка потрібна для того, щоб зберігати дані про хід і стан задачі, що розв'язується. Дошка містить об'єкти із схеми об'єктів. Ці об'єкти ієрархічно групуються за рівнями аналізу (рис. 3) і разом із своїми атрибутами утворюють словник простору рішень.

Об'єкти на інформаційній дошці утворюють ієрархію, яка відображає ієрархічність різних рівнів абстракції знань. Інтерпретатор користується загальною інформацією про зроблені в процесі розв'язку кроки (плануючі пари). Тому до об'єктів інформаційної дошки включимо клас: плануюча пара.

Інформаційна дошка є суперклас для класів, що є на дошці. З точки зору зовнішньої поведінки визначимо для цього класу два оператори:

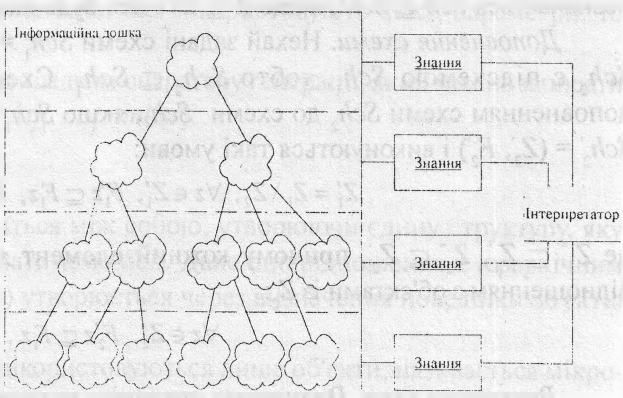

Рис. 3. Інформаційна дошка

register добавити об'єкт на дошку.

resign видалити об'єкт із дошки.

Об'єкти інформаційної дошки з'являються або щезають з неї залежно від процесу розв'язання задачі.

Необхідні для розв'язання задачі знання про предметну область розділені на кілька незалежних джерел знань. Кожне джерело намагається запропонувати інформацію, яка корисна для розв'язування задачі. Поточна інформація з кожного джерела знань поміщається на дошці і модифікується залежно від знань.

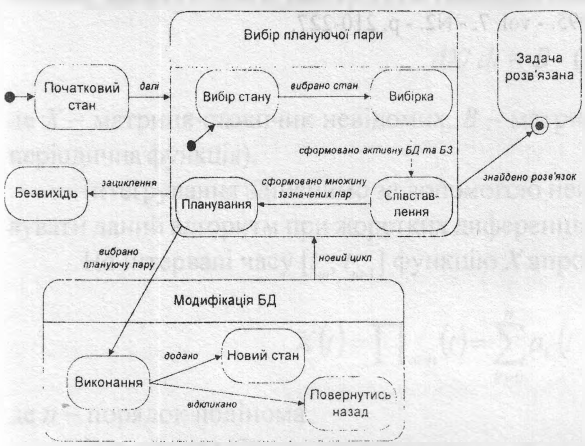

Рис. 4. Подання роботи інтерпретатора у вигляді скінченного автомата

У загальному випадку джерела знань відповідають ієрархічній структурі об'єктів, що розміщуються на інформаційній дошці. Більше того, кожне джерело використовує об'єкти одного рівня ієрархії як вхідні дані, а як вихід генерує або змінює об'єкти на іншому рівні.

На інформаційній дошці ми отримуємо деякий зріз мікросхеми.

Для опису роботи інтерпретатора використаємо схему скінченного автомата. Розглянемо діаграму станів та переходів на рис. 4. Інтерпретатор може перебувати в одному із п'яти основних станів: початковий стан, вибір плануючої пари, модифікація БД, безвихідь і розв'язок задачі. Найбільший інтерес становить перехід із стану вибору плануючої пари до модифікації БД. Спочатку інтерпретатор вибирає стан, потім вибирає знання, на основі яких здійснюється співставлення, і нарешті на етапі планування вибирається плануюча пара.

Вибравши плануючу пару, інтерпретатор переходить в стан модифікації БД, тобто змінюється стан інформаційної дошки. Зміна стану відбувається двома способами: додається новий стан на інформаційну дошку або інформаційна дошка повертається у попередній стан.

Кінцевими станами роботи інтерпретатора є розв'язок задачі або безвихідь.

Висновок

Використання об'єктно-орієнтованого підходу в системах інженерії знань виводить на перший план можливість природної декомпозиції задачі на ряд підзадач, що представляються достатньо автономними агентами, що працюють із знаннями. Це єдина практична можливість роботи в умовах експоненціального росту складності (кількість взаємозв'язків), характерного для систем, що використовують знання. Наведена математична модель повністю відображає дані переваги об'єктно-орієнтованої моделі. Вона базується на схемі об'єктів, яка утворюється на інформаційній дошці під час процесу логічного виведення.

1. Bertino E., Foscoli P. Index Organizations for Object-Oriented Database Systems. - Knowledge and engineering. - April. 1995. - vol. 7. - N2. - p. 193-209.
2. Буч Г. Объектно-ориентированный анализ и проектирование. М., 1998.
3. Клыков Ю.И., Горьков Л.Н. Банки данных для принятия решений. М., 1980.
4. Литвин В.В. Реалізація множинного наслідування в об'єктно-орієнтованих базах даних // Вісник ДУ "Львівська політехніка" Львів. №383. 1999. С. 140-144.
5. Ойхман Е.Г., Попов Э.В. Реинжиниринг бизнеса: реинжиниринг организаций и информационные технологии. М., 1997.
6. Представление и использование знания / под ред. Х.Уэно, М.Исидзука. М., 1989.
7. Jacobson I., Ericsson M., Jacobson A. The Object Advantage: Business Process Reengineering with Object Technology // ACM Press. - Addison-Wesley Publishing, N-Y: 1995.
8. Straube D.D., Ozcu M.T. Query Optimization and Execution Plan Generation in Object-Oriented Database Systems. - Knowledge and engineering. - April. 1995. - vol. 7. - N2. - p. 210-227.

УДК 621.3.01

РОЗРАХУНОК ПАРАМЕТРИЧНОЇ ЧУТЛИВОСТІ ЕЛЕКТРОМАҐНЕТНИХ КІЛ НЕЯВНИМИ МЕТОДАМИ

© В. Самотий, А. Павельчак, В. Мінкіна*
Національний університет "Львівська політехніка"
*Ченстоховська політехніка

Запропоновано методику аналізу параметричної чутливості електромагнетних кіл з використанням неявних методів числового інтегрування. Аналіз параметричної чутливості зводиться до обчислення часткової похідної вектора змінних стану за вектором параметрів кола. Чутливість вектора змінних стану кола до зміни параметрів визначається в усталеному режимі, тому тут побічно вирішується ще й завдання прискореного пошуку усталених режимів.

The method of analysis of the parametric sensitivity by using explicit integration methods is developed. The proposed method comes to the calculation of the vector of circuit parameters partial derivative of the state variables vector. The parameter variation sensitivity of the state variables vector is calculated in steady-state therefore the problem of steady-states quick search is also solved.

Параметрична чутливість – це задача, яка пов'язана з аналізом чутливостей схеми (пристрою, системи) до зміни її параметрів. Розв'язання цієї задачі дасть змогу інженеру-конструктору оптимально спроектувати пристрій. Іншими словами, визначити, які параметри впливають більше на вихідний сигнал, а які менше.