УДК 681.3:622.276

В.Шекета, канд.техн.наук
Івано-Франківський національний технічний університет нафти і газу

ВИКОРИСТАННЯ ДОМЕНУ ЛОГІЧНИХ ПРОГРАМ ДЛЯ ПРЕДСТАВЛЕННЯ АБСТРАКТНИХ ТИПІВ ДАНИХ МОДИФІКАЦІЙНИХ ЗАПИТІВ

Запропоновано спосіб використання домену логічних програм для представлення абстрактних типів даних модифікаційних предикатних запитів для інформаційних інтелектуальних систем на основі баз даних і знань. Відмінними рисами пропонованого підходу є використання представлення інформації в формі логічних заперечень, явне задання поліморфних залежностей між абстрактними типами даних і побудова абстрактних доменів на формально-логічній основі теорій послідовного уточнення доменів та абстрактної інтерпретації .

Умовні позначення

LP_W^n-домен логічних програм; $\Re$ - скінченний інформаційний простір; $\Delta_X^{SO_w}$ – узагальнення операції вилучення Шрьодера; K_B – база знань; Q_M – модифікаційний предикатний запит; $K_{B+}()$; $K_{B-}()$ - модифікаційні літерали; << - дескриптор модифікації; W – множина змінних; M_F – множина функцій; T – множина термів; ST – система абстрактних типів даних; SO – система обмежень; F – функторний символ; ψ – прототип функторного відображення; r_f – абстрактний оператор; $w \in \Gamma_f(w_1)$ – множина обмежень Гербранда; $list\_var$ – поліморфний абстрактний тип даних; $a\_type()$ – абстрактна процедура для роботи із поліморфними типами даних.

1. Вступ. В даний час нафтогазова галузь має автоматизовані мікропроцесорні системи різних рівнів і типів, а також різні програмні комплекси для прогнозу нафтогазоносності, пошуку та розвідки родовищ нафти і газу, але масштаби та ефективність застосування ЕОМ для кожного рівня різні. Впровадження нових інформаційних технологій на базі мікропроцесорних систем в нафтогазовій справі дає можливість суттєво змінити економіко-екологічний підхід до пошуково-розвідувальних робіт на нафту і газ. Суттєве місце в ринку програмних продуктів для нафтогазової предметної області займають інформаційні інтелектуальні системи на основі баз даних і знань. Основою таких систем є програмні модулі, побудовані на основі концепцій логічного *Prolog* – програмування.

Виконання аналізу абстрактних типів даних для логічних програм є важливим як і для програми (виконання дебаггінгу і верифікації), так і для компілятора (оптимізація програмного коду).

Порівняння різних технік і методик аналізу абстрактних типів даних в термінах точності, ефективності і загальності є досить важким завданням, оскільки такі техніки часто використовують різні методи аналізу, які базуються на протилежних вихідних гіпотезах. Виконаний нами аналіз публікацій дозволяє виділити наступні методи: методи аналізу абстрактних типів даних і побудови на його основі процедури логічного висновку подібні до тих, які використовуються у функціональних мовах програмування високого рівня [1,2]; техніки аналізу на основі методів верифікації програм [3]; аналіз абстрактних типів на основі типованих графів [4]; техніки аналізу абстрактних типів на основі абстрактної інтерпретації [5–12].

Вихідним кроком підходу до аналізу абстрактних типів даних на основі абстрактної інтерпретації є вибір абстрактного домену, який визначає, як буде виконано присвоєння абстрактних типів термам. Формальні засоби опису базових типів

не дозволяють обробляти залежності між типами. Деякі залежності абстрактних типів серед попарно різних аргументів процедур можуть бути виражені через використання змінних типу в мовах програмування, що оперують із структурними типами даних. Це є стандартне рішення, описане зокрема в [3,5,8]. Подібну техніку використано також в рамках підходу щодо регулярної апроксимації множин успішних рішень [7]. Проте слід відзначити, що використання змінних типу не дозволяє виразити залежності між абстрактними типами даних із точки зору позиціювання аргументів процедур. Тільки в роботах [5,6] показано приклад виключення випадку, який явним чином визначає залежності між типами. В [5] виконано оголошення абстрактного домену для аналізу базових типів. В даній роботі введено поняття типізованих програм, що значно звужує можливості застосування такого підходу. Крім того, оскільки змінні типів не використовуються, практичне застосування розроблених доменів є досить складним завданням. В роботі [6] автори виконують узагальнення техніки побудови домену *POS* для аналізу базовості до випадку абстрактних типів. Проте коректність такого підходу не була підтверджена формальним доведенням. Більше того, використання змінних типу є в більшій мірі наслідком вибраного способу практичної реалізації, чим формальності теорії, що лежить в основі підходу.

В роботах [13,14] база знань інформаційної системи розглядається як набір інформаційних сутностей атомарних предикатів з деякого скінченного інформаційного простору $\Re$. Всі зміни, що відбуваються в базі знань, розглядаються як наслідок модифікаційних предикатних запитів Q_M, що генеруються інтелектуальною інформаційною системою відповідно до вказівок користувача. Основою самих запитів є набір модифікаційних предикатних правил. Розглядаються два типи правил:

$$Q_M \longleftrightarrow (K_B)^{<<}\left\| K_{B+}(o\ ) << K_{B+}(o_1),...,\ K_{B+}(o_l), K_{B-}(p_1),...,\ K_{B-}(p_m)\right., \qquad (1)$$

$$Q_M \longleftrightarrow (K_B)^{<<}\left\| K_{B-}(o) << K_{B+}(o_1),...,\ K_{B+}(o_l), K_{B-}(p_1),...,\ K_{B-}(p_m)\right., \qquad (2)$$

де $o, o_i, p_i \in \Re$. Основна ідея такого запису правил полягає в тому, що $K_{B+}(o\ )$ означає, що атомарний предикат o повинен бути включений в базу знань K_B, а K_{B-} означає, що o – повинен бути виключений з бази знань, а $(K_B)^{<<}$- означає модифікацію бази знань на рівні логічної зв'язаності предикатних правил як наслідок виконання операцій додавання і вилучення правил; $<<$ – розглядається як комплексна стрілка, дослідження властивостей якої буде виконано пізніше *засобами теорії категорій*. **Недослідженим** залишається питання введення абстрактних типів даних для модифікаційних предикатних запитів(які побудовані нами як розширення множини логічних *Prolog*-програм [13]) та побудови коректного домену для їх аналізу .

Таким чином, метою даного дослідження є побудова способу використання домену логічних програм для представлення абстрактних типів даних модифікаційних предикатних запитів для інформаційних інтелектуальних систем на основі баз даних і знань.

2. Дослідження домену абстрактних типів даних

Нехай задано множину змінних W і множину функцій M_F, причому кожній функції присвоєна певна розмірність . Для $l \in N$ означимо множину термів T :

$$1)\ \mathrm{T}^{\varnothing}\left(M_F^1, W\right)W;\ 2)\ \mathrm{T}^{l+1}\left(M_F^2\ , W\right) = \mathrm{T}^{l}\ \left(M_F^1, W\right)W;$$

$$3)\left\{ f(t_1,...t_n)\middle|\ f^n \in M_F^1\ \{t_1,...t_n\} \subseteq T^l\ \left(M_F^1, W\right)\ \right\};\ 4)\ \mathrm{T}\left(M_F, W\right) = \bigcap_{c\geq 0} \mathrm{T}^c\left(M_F, W\right).$$

Ми виходимо з припущення, що M_F містить принаймні один символ арності $\varnothing$.

Означення 1. (Система обмежень для абстрактних типів даних). Нехай $ST=<M_F^1, M_F^2, \psi>$ – система абстрактних типів даних. Системою обмежень для системи абстрактних типів даних ST є система обмежень $SO=\{SO_w\}_{w\in r_{f(w_1)}}$, де $SO_w \subseteq F_{M_F^1,w}$ є замкнутою відносно операції $\wedge$ і містить діагональні елементи

$$(\theta_1)^{SO_w}_{<x_1,...,x_k>,<y_1,...,y_k>}=\wedge t\in T(M_F^1,\varnothing), i=1,...,k\{(x_i\in t\Rightarrow y_i\in t)\wedge(y_i\in t\Rightarrow x_i\in t)\} \quad (3)$$

для кожного $<x_1,...,x_k>$ і $<y_1,...,y_k>$ в w і є замкнутим щодо операцій:

$$\varphi_1 *^{SO_w} \varphi_1=\varphi_1\wedge\varphi_2 \quad (4)$$

$$\Delta_x^{SO_w}\varphi=\bigcup\left\{\varphi[Q/x] \text{ Існують } \left|\begin{array}{l} t\in T(M_F^2,W) \text{ такі, що} \\ Q=\{t_1\in T(M_F^1,\varnothing)\,|\, t\in[t_1]\varphi\}\end{array}\right.\right\}, \quad (5)$$

причому, для довільного $Q\in r(T(M_F^1,\varnothing))$, матимемо , що

$$(x\in t)[Q/x]=\begin{cases}\text{істина, } L^{\max}_{q\in Q}[q]\varphi\subseteq[t]\varphi\\ \text{хиба, в інших випадках}\end{cases}, \quad (6)$$

крім того, справедливо, що:

1) $(y\in t)[Q/x]=(y\in t)$, якщо $x\neq y$; 2) $(\wedge M_F)[Q/x]=\wedge\{\varphi[Q/x]\,|\,\varphi\in M_F\}$;

3) $(\varphi_1\Rightarrow\varphi_2)[Q/x]=(\varphi_1[Q/x])\Rightarrow(\varphi_2[Q/x])$;4) $(\vee M_F)[Q/x]=\vee\{\varphi[Q/x]|\varphi\in M_F\}$;

5) $(\neg\varphi)[Q/x]=\neg(\varphi[Q/x])$.

Введена нами операція $\Delta_x^{SO_w}$ є узагальненням операції вилучення Шрьодера [3], яка означається як

$\exists_x(\varphi)=\varphi[\text{істина}/X]\vee\varphi[\text{хиба}/X]$.

В загальному випадку, якщо множина $T(M_F^1,\varnothing)$ є скінченною, тоді і множина $F_{M_F^1,w}$ теж є скінченною, і, таким чином, результуюча система обмежень для абстрактних типів даних є скінченною. Більше того, операції кон'юнкції і циліндрифікації стають ефективними, хоча вони і не можуть бути обчислені за допомогою скінченної послідовності кроків.

Означення 2. Для заданої системи абстрактних типів $ST=<M_F^1,M_F^2,\psi>$, задамо відображення $[\ ]_{ST}: F_{M_{F,w}^1}\to(\sigma_w\to\{0,1\})$, через послідовність кроків:

1) $[x\in t]_{ST}\theta=\begin{cases}1, \text{ якщо } \theta(x)\in[t]\psi\\ \varnothing, \text{ в інших випадках}\end{cases}$;

2) $[\vee(M_F)]_{ST}\theta=\begin{cases}1, \text{ якщо існує } \varphi\in M_F, \text{ таке, що } [\varphi]_{ST}\theta=1\\ \varnothing, \text{ в інших випадках.}\end{cases}$;

3) $[\wedge M_F]_{ST}\theta=\begin{cases}1, \text{ якщо } [\varphi]_{ST}\theta=1, \text{ для кожного } \varphi\in M_F\\ \varnothing, \text{ в інших випадках.}\end{cases}$;

4) $[\varphi_1\Rightarrow\varphi_2]_{ST}\theta=\begin{cases}1, \text{ якщо } [\varphi_1]_{ST}\theta=1, \text{ тоді } [\varphi_2]_{ST}\theta=1\\ \varnothing, \text{ в інших випадках.}\end{cases}$; 5) $[\neg\varphi]_{ST}\theta=1-[\varphi]_{ST}\theta$.

Для заданих $w\in r_f(w_1)$ і $\varphi_1,\varphi_2\in F_{M_{F,w}^1}$ матимемо $\varphi_1\le_{ST,w}\varphi_2$, якщо для кожного $\sigma\in\sigma_w$ $[\varphi_1]_{ST}\sigma=1$, то тоді $[\varphi_2]_{ST}\sigma=1$. Будемо вважати, що $\varphi_1\equiv_{ST,w}\varphi_2$, тоді і тільки тоді, коли справедливо, що $\varphi_1\le_{ST,w}\varphi_2$ і $\varphi_2\le_{ST,w}\varphi_1$. Таку еквівалентність будемо називати (ST,w) – еквівалентністю.

Таким чином, матимемо, що якщо $SO=\{SO_w\}_{w\in r_f(w_1)}$ є системою обмежень для ST, тоді, для кожного $w\in r_f(w_1)$, кожна трансфінітна формула в SO_w позначає відповідний клас (ST,w) – еквівалентності. Крім того, для кожного $w\in r_f(w_1)$, SO_w є повною структурою відносно операції $\leq_{ST,w}$ і, згідно введених означень, вона є замкнутою відносно операції $\wedge$ і має фіксований верхній елемент (істина = $\wedge\varnothing$). Множина SO_w має фіксованим верхнім елементом логічну константу "істина", фіксованим нижнім елементом – логічну константу "хиба", $\vee$ – оператор обчислення найменшої верхньої границі, $\wedge$ – оператор обчислення найбільшої нижньої границі.

Властивість 1. Нехай $SO=\{SO_w\}_{w\in r_f(w_1)}$ є системою обмежень для системи абстрактних типів $<M_F^1,M_F^2,\psi>$, тоді для кожного $w\in r_f(w_1)$ і $\{\varphi_2,\varphi_1\}\subseteq SO_w$, такої, що $\varphi_1\equiv\varphi_2$, ми матимемо, що $\varphi_1[Q/x]\equiv\varphi_2[Q/x]$ для кожного $x\in w$ і $Q=\{t_1\in T(M_F^1,\varnothing)\mid t\in[t_1]\psi\}$, де $t\in T(M_F^2,w)$.

Означення 3. Для заданої системи обмежень $SO=\{SO_w\}w\in r_f(w_1)$ для системи абстрактних типів $<M_F^1,M_F^2,\psi>$ і $\varphi\in SO_w$, означимо $\beta_{SO_w}(\varphi)=\{\sigma\in\sigma_w\}$. Для всіх $\theta\in\sigma_w$, таких, що $\theta\in\sigma$, ми матимемо, що $[\varphi]_{ST}\theta=1\}$. Причому $\beta_{SO_w}(\varphi)\in NS_w$ для кожного $\varphi\in SO_w$. Для заданих $\{\varphi_1,\varphi_2\}\subseteq SO_w$, буде справедливо також $\varphi_1\equiv_{\beta_{SO_w}}\varphi_2$ тоді і тільки тоді, коли $\beta_{SO_w}(\varphi_1)=\beta_{SO_w}(\varphi_2)$. Таке відношення еквівалентності будемо називати β_{SO_w} – еквівалентністю.

Обробка негативної інформації

Нехай $ST=<M_F^1,M_F^2,\psi>$ є системою абстрактних типів. Для заданих $w\in r_f(w_1)$, $x\in w$, і $l\in T(M_F^1,\varnothing)$ матимемо, що:

$$\beta(x\notin l\Rightarrow\text{хиба})=\{\sigma\in\sigma_w(\text{для всіх }\theta\leq\sigma\text{ ми маємо }[x\in l]\theta=\varnothing)\}. \qquad (7)$$

Це означає, що $\beta(x\in l\Rightarrow\text{хиба})$ містить множину підстановок, які відображають x в терм, який не є і не може бути ініціалізований до терму типу l. Наведене твердження є прикладом інтуїтивного заперечення. На можливість представлення фрагментів інформації в формі заперечень в нашому дослідженні буде звертатися особлива увага, оскільки таке представлення є особливо корисним з точки зору побудови практичних реалізацій.

Якщо низспадно замкнута множина підстановок міститься в конкретизації деякої трансфінітної формули φ над скінченною множиною змінних W, тоді її розширення до множини змінних W', $W\subseteq W'$ буде міститися в конкретизації тієї самої формули над W'. Це означає, що ми не повинні утримувати кількість змінних, явним способом на низькому рівні при дослідженні абстрактних типів даних. Тому застосування операції перейменування в даному випадку буде корисним, якщо перейменування не виконується через використання діагональних елементів і операції циліндрифікації [10]. Наступний оператор буде оптимальним наближенням, в даній ситуації:

Означення 4. Для заданої системи обмежень $SO=\{SO_w\}_{w\in\Gamma_f(w_1)}$ і для $w\in\Gamma_f(w_1)$ і $x\notin W$ означимо операцію переіменування $PN_{x\to k}^{SO_w}(\varphi)=\varphi[k/x]$.

Множина трансфінітних формул, може бути використана як обчислювальний домен, якщо множина введених абстрактних типів є скінченною. В такому випадку кожна трансфінітна формула є ізоморфною до деякої визначеної формули [10].

Хоча використання скінченної множини абстрактних типів є доцільним з точки зору аналізу властивості базовості і зв'язаності, але це тим не менше не дозволяє нам

виконувати ефективний аналіз абстрактних поліморфних типів даних. У випадку, коли $T\left(M_F^2,\varnothing\right)$ є нескінченною, то множина трансфінітних формул теж не може бути представлена скінченним способом. Хоча повне використання всієї потужності трасфінітних формул є рідко доцільним. В багатьох випадках скінченні формули із змінними типу можуть представляти ту саму інформацію, що й трансфінітні формули [9-10]. Виконаємо застосування абстрактних поліморфних типів даних для випадку модифікаційних предикатних запитів. Припустимо, що ми маємо поліморфний абстрактний тип *list* _ var і нам потрібно використати операцію кон'юнкції

$$\wedge_{t\in T\left(M_F^2,\varnothing\right)}\left(x\in list\_\mathrm{var}(t)\Leftrightarrow\left(c\in t\wedge\imath\in list\_\mathrm{var}(t)\right)\right) \tag{8}$$

для того, щоб виразити взаємозв'язок між змінними в накладеному обмеженні $\{x=[c/\imath]\}$. Через використання змінних типу, та сама інформація може бути виражена за допомогою скінченної формули

$$x\in list\_\mathrm{var}(ST)\Leftrightarrow\left(c\in ST\wedge\imath\in list\_\mathrm{var}(ST)\right). \tag{9}$$

Відмітимо, що дана формула може бути також записана як запит

$$x\left(list\_\mathrm{var}(ST)\right)<<c\ (ST),\imath\ (list\_\mathrm{var}(ST)), \tag{10}$$

$$c\ (ST)<<x\left(list\_\mathrm{var}(ST)\right),\imath\ (list\_\mathrm{var}(ST))<<x\left(list\_\mathrm{var}(ST)\right). \tag{11}$$

В даному запиті змінні, що представляють інтерес *x, c, ι*, будуть інтерпретуватися як константи в тілі запиту, в той час як змінні типів є дійсними змінними.

Розглянемо деякий запит із накладеними обмеженнями щодо змінних z і x. Оскільки LP_W^n і SO_W є повними структурами, то ми можемо оголосити ко-адитивну функцію $\beta^n: LP_W^n\to SO_W$, яка дозволить нам розглядати LP_W^n як абстрактну інтерпретацію для SO_W.

Таким чином, для запиту $Q\in LP_W^n$ матимемо :

$$x(list\_\mathrm{var}(ST))<<y(ST),\ z(list\_\mathrm{var}(ST)) \tag{12}$$

$$y(ST)<<x(list\_\mathrm{var}(ST)),\ z(list\_\mathrm{var}(ST))<<. \tag{13}$$

Для випадку $\Delta_Z^{LP_W^1}$, Q матимемо :

$$x(list\_\mathrm{var}(ST))<<y(ST),\ y(ST)<<x(list\_\mathrm{var}(ST)); \tag{14}$$

і для $\Delta_Z^{LP_W^1}Q$ отримаємо :

$$z(list\_\mathrm{var}(ST))<<. \tag{15}$$

Відмітимо, що в $\Delta_x^{LP_W^n}Q$, ми задали твердження для $y\left(ST\right)$, оскільки ми маємо справу із класами еквівалентності обмежень.

Виконаємо застосування абстрактної процедури *a _ type*() .Одержимо :

$$a\_type(Y,list\_\mathrm{var}(X_1))<<Y=[\ ], \tag{16}$$

$$a\_type(Y,list\_\mathrm{var}(X_1))<<Y=[C|ST],a\_type(C,X_1), \tag{17}$$

$$a\_type(ST,\ list\_\mathrm{var}(X_1)). \tag{18}$$

Відмітимо, що в загальному випадку означення процедури *a _ type*() може бути одержано автоматично із означень абстрактних типів даних, і таке означення буде композиційним щодо додавання нових типів даних до існуючої системи абстрактних типів.

3. Висновки та перспективи наступних досліджень.

Таким чином, в даній статті виконано побудову способу використання домену логічних програм для представлення абстрактних типів даних модифікаційних

предикатних запитів для інформаційних інтелектуальних систем на основі баз даних і знань. Відмінними рисами пропонованого підходу є використання представлення інформації в формі логічних заперечень, явне задання поліморфних залежностей між абстрактними типами даних і побудова абстрактних доменів на формально-логічній основі теорії послідовного уточнення доменів.

Темою наступних досліджень буде питання можливості виконання представлення системи абстрактних типів засобами диз'юнктивних логічних програм замість використа-них нами традиційних логічних програм.

In given paper the method of logical programs domain use is offered for the representation of abstract data types of predicate queries modification for the information intellectual systems on the basis of databases and knowledgebases. The use of representation of information in the form of logical negations, explicit representation of polymorphic dependencies between abstract data types, and the construction of abstract domains on the formal-logical basis of the theories of sequentially improving of domains and of abstract interpretation is the offered approach distinguishing features.

Література

1. Barbuti R., Giacobazzi R. A Bottom-up Polymorphic Type Inference in Logic Programming. Science of Computer Programming, 19(3).– 1992.–P.281–313.
2. Apt K. R., Marchiori E. Reasoning about Prolog Programs: from Modes through Types to Assertions. Formal Aspects of Computing, 6(6A).–1994.P–743–765.
3. Yardeni E. , Shapiro E. A Type System for Logic Programs. Journal of Logic Programming, 10.–1991.–P.125–135.
4. Hentenryck P. , Cortesi A., Charlier B. Type Analysis of Prolog using Type Graphs. Journal of Logic Programming, 22(3).–1995.–P.179–209.
5. Codish M. , Demoen B. Deriving Polymorphic Type Dependencies for Logic Programs Using Multiple Incarnations of Prop. In Proc. of the first International Symposium on Static Analysis, volume 864 of Lecture Notes in Computer Science. Springer-Verlag.–1994–P.281–296.
6. Codish M., Lagoon V. Type Dependencies for Logic Programs Using ACI Unification. In Proceedings of the 1996 Israeli Symposium on Theory of Computing and Systems, IEEE Press,June 1996.–P.136–145.
7. Gallagher J., Waal D. A. Fast and Precise Regular Approximation of Logic Programs. In Pascal Van Hentenryck, editor, Proceedings of the Eleventh International Conference on Logic Programming,Santa Margherita Ligure, Italy, The MIT Press.–1994.–P.599–613.
8. Janssens G., Bruynooghe M. Deriving Descriptions of Possible Values of Program Variables by means of Abstract Interpretation. Journal of Logic Programming, 13(2 & 3), 1992.–P.205–258.
9. Kifer M. , Wu J. A First Order Theory of Types and Polymorphism in Logic Programming. In IEEE Symposium on Logic in Computer Science.– 1991.
10. Lu Lunjin. A Polymorphic Type Analysis in Logic Programs by Abstract Interpretation. Journal of Logic Programming, 36(1).–1998.–P.1–54.
11. Papadimitriou C. Computational Complexity. Addison-Wesley.–1994.
12. Smaus J.-G., Hill P., King A. Mode Analysis for Typed Logic Programs. In Proc. of the LOPSTR'99 Workshop, Venice,Italy, September 1999.–P.163–170.
13. Шекета В.І. Модифікаційні предикатні запити, як інструмент підтримки діалогу з користувачем в інформаційних системах на основі баз даних і знань // Вісник Тернопільського державного технічного університету.- 2003-Технічні науки.-Том 8.-№4. –2003.- С.113-119.
14. V.Sheketa. Predicate queries modification, as an tool to work with the knowledgebases of oil and gas subject domain. In proceedings of 6-th International scientific conference "Modern problems of Radio engineering, telecommunications and Computer science – TCSET'2004" .-Lviv-Slavsko, February 24-28.-2004.–P.315–319.

Одержано 27.02.04 р.