УДК 681.3.06

*Зубенко В. В.*

# АЛГЕБРАЇЧНІ ЗАСОБИ СПЕЦИФІКАЦІЇ ІНФОРМАЦІЙНИХ МОДЕЛЕЙ. II

*Робота продовжує [1]. Розглядаються рекурсивні засоби специфікації інформаційних систем. Аналізуються варіанти індуктивних визначень множин та функцій, на які спираються такі специфікації. Розглядаються питання однозначності та процедуризації індуктивно визначених функцій.*

**1. Індуктивні визначення множин, систем множин та функцій.** Рекурсивні засоби специфікації інформаційних систем базуються на індуктивних визначеннях об'єктів та функцій, які вже згадувались у першій частині роботи [1]. Зупинимося детальніше на деяких з них та сформулюємо нові. Нагадаємо, що індуктивне визначення **(ІВ)** певної множини об'єктів $M$ має вигляд:

**(Б) База індукції.** Фіксується певна підмножина $M_0 \subset M$ апріорі вибраних базових (атомних) об'єктів.

**(І) Індуктивний перехід.** Вибирається певна сукупність $F$ конструкторів – операцій на $M$, замкнених відносно $M$. Замкненість конструктора відносно $M$ (і навпаки множини $M$ відносно конструктора) означає, що результат його застосування гарантовано належить множині $M$ за умови, що він був застосований до аргументів з $M$.

**(П) Повнота.** Конструктори дають змогу за скінченну кількість кроків отримати з базових елементів кожен з об'єктів множини $M$ і тільки їх.

Оскільки вся сукупність об'єктів в індуктивних визначеннях повністю описується пунктами (Б) і (І), то пункт (П) Повнота може явно не формулюватись. Індуктивно визначені множини називатимемо індуктивними і позначатимемо $M = M(M_0, F)$. Конструктори множини $F$ іноді називають правилами виводу. Рівність $x = c(x_1,...,x_n)$ позначають $x_1,...,x_n \underset{c}{\Rightarrow} x$. При цьому елементи $x_1,...,x_n$ називають гіпотезами, $x$ – висновком правила $c$, а базові елементи з $M_0$ – аксіомами. Виводом елемента $a$ з аксіом називається послідовність елементів $a_1,...,a_m = a$, така, що для всіх $0 \le i \le m$ $a_i$ є або аксіомою, або висновком певного правила виводу з гіпотез, що належать сукупності $a_1,...,a_{i-1}$. Індуктивно визначену множину $M = M(M_0, F)$ утворюють саме ті елементи, що можуть бути виведені з аксіом. Зауважимо, що $M$ є найменшою замкненою надмножиною сукупності базових елементів $M_0$. Справді, 1) всі елементи $M$ належать будь-якій замкненій надмножині $M_0$ і 2) $M$ – замкнена надмножина $M_0$. Отже, вона є найменшою серед таких множин.

Нехай $P(x)$ – довільний предикат на $M$. У загальному випадку пункт **(П)** Повнота в індуктивних визначеннях множини $M$ може замінюватись на пункт:

**(ПВ) Повнота Відносна.** Конструктори дають змогу за скінченну кількість кроків отримати з базових елементів кожен з об'єктів множини $M$, що задовольняє даний предикат-фільтр $P(x)$. Назвемо такі фільтри **термінаторами.** Елементи, які їх задовольняють,– термінальними, а всю сукупність термінальних елементів в $M$ позначимо $T(M_0, F, P)$.

У випадку Повноти Відносної деякі або навіть і всі базові елементи можуть не належати термінальній сукупності. Наприклад, одне з можливих **ІВ** множини $\Sigma^*$ всіх слів над алфавітом $\Sigma = \{a_1, a_2, ..., a_n\}$ має такий вигляд:

**(Б$_1$)** $\varepsilon \in \Sigma^*$ – порожнє слово та букви алфавіту $\Sigma$ є базовими.

(І$_1$) Конструктор $cons: \Sigma \times \Sigma^* \to \Sigma^*$ визначимо так:

$$cons\left(a, \langle b_1,...,b_n \rangle\right) = \langle a, b_1,...,b_n \rangle.$$

**(ПВ$_1$)** Фільтр-термінатор $P(x) = (x \in \Sigma^*)$.

Як бачимо, базові елементи – символи алфавіту $\Sigma$ – не входять до $\Sigma^*$ (односимвольні слова мають вигляд $< a >$). Слово $\langle b,c,d \rangle$ будується так: $cons\left(b, cons\left(c, cons(d, \varepsilon)\right)\right)$. Зауважимо, що для довільної множини $T$ та послідовностей над нею конструктор $cons: T \times T^* \to T^*$ визначається аналогічно.

Розглянемо тепер індуктивні визначення систем множин (**ІВС**). Нехай $P_1(x_1),...,P_k(x_k)$ – фільтри на довільному універсумі $U$ певних елементів, що виділяють у ньому певні сорти елементів $U_1,...,U_k$. При цьому області істинності сортових фільтрів можуть і перетинатися. **ІВС** системи множин $< M_1,...,M_k >$, $M_i \subseteq U_i$ має вигляд:

**(БС) База індукції.** Фіксується підмножина $U_0 \subset U$ базових об'єктів.

**(ІС) Індуктивний перехід.** Вибирається певна сукупність $F$ конструкторів – багатосортних операцій на $U$. Кожний аргумент конструктора та його результат належать певному фіксованому сорту. Результат застосування конструктора гарантовано належить сорту-результату за умови, що він був застосований до аргументів відповідного сорту (замкненість конструктора).

**(ПС) Повнота.** Конструктори дають змогу за скінченну кількість кроків вивести з аксіом кожен з об'єктів множин $M_i$, $i=1, n$, і тільки їх.

Система множин $< M_1,...,M_k >$ замкнена відносно певних конструкторів, якщо кожна з компонет $M_i$ замкнена відносно них. Як і у випадку звичайних **ІВ,** система $< M_1,...,M_k >$ є найменшою замкненою відносно даних конструкторів системою множин, породжених базою $U_0 \subset U$. Порівняння здійснюється покомпонентно.

Нехай $Q_i(x)$, $i=1, n$,– фільтри-термінатори, визначені на сортах $U_i$. Пункт **(ПС)** Повнота в індуктивних визначеннях систем множин теж може замінюватись на пункт:

**(ПВС) Повнота Відносна.** Конструктори дають змогу за скінченну кількість кроків отримати з базових елементів кожен з об'єктів множин $M_i$, $i=1, n$, що задовольняють фільтри-термінатори $Q_i(x)$.

Покажемо, як, наприклад, за допомогою **ІВС** можна задавати контекстно-вільні мови у даному алфавіті $\Sigma = \{a_1, a_2, ..., a_n\}$ (КВ-мови [2]). Нехай $V = \{A_1, A_2, ..., A_m\}$ – певний алфавіт допоміжних символів. Універсум $U$ утворюють натуральні числа та сукупність $(\Sigma \cup V)^*$ усіх слів в алфавіті основних та допоміжних символів. Фільтр $P_i(x)$, $i=1,m$, виділятиме сорт слів, що можуть бути виведені з аксіоми $A_i$. Розглянемо систему мов $< L_1,...,L_m >$, $L_i \subseteq (\Sigma \cup V)^*$, що визначається таким **ІВС**:

**(БС$_2$) База індукції.** Усі натуральні числа є базовими. Кожному допоміжному символу $A_i \in V$, $i=1, m$ поставимо у відповідність скінченну не порожню підмножину слів $R_i = \{u_1, ..., u_{r_i}\} \subset (\Sigma \cup V)^*$. Слова з $R_i$, $i=1, m$ – базові. За визначенням $P_i(u_j)$ для всіх $j=1, r_i$.

**(ІС$_2$) Індуктивний перехід.** Єдиний конструктор $Sub: \Sigma^* \times N \to \Sigma^*$ кожному числу $k \geq 0$ та слову вигляду $u = vA_iw$, де $v$ – префікс $u$ довжини $k$, $A_i$ – допоміжний символ, ставить у відповідність слова вигляду $vu_jw$, $j=1,r_i$. Конструктор залишає слово $u$ без змін у разі відсутності в ньому префікса довжини $k$ і наступного за ним допоміжного символу. За визначенням, конструктор не змінює фільтр слова. Тобто слово $u$ передає у спадок результатам $vu_jw$ свій фільтр.

Не складно перевірити, що мова $L_i$, $i=1, m$ збігається з сукупністю всіх словоформ в об'єднаному алфавіті КВ-граматики з правилами виводу $A_i \to u_j$, $i=1, m$, $j=1, r_i$, що можуть бути виведені в ній з аксіоми $A_i$.

**(ПВС$_2$) Повнота Відносна.** Фільтри-термінатори – $P_i(x) \& (x \in \Sigma^*)$, $i=1, m$. Такі термінатори задають мови $L(A_i)$ в основному алфавіті, що породжуються наведеною вище граматикою, якщо аксіомами в ній вибрати відповідно символи $A_1,...,A_m$.

Перейдемо тепер до індуктивних визначень функцій. Нехай $M = M(M_0, F)$ – довільна індуктивна множина. Індуктивне визначення унарної функції **(ІВФ)** $f: M \to W$ спирається на **ІВ** множини $M$ і має вигляд:

**(БФ) База індукції.** $\forall x \in M_0 : f(x) = g(x)$, де $g(x)$ – «відома» функція, що задана апріорі на підмножині $M_0$.

**(ІФ) Індуктивний перехід.** Для кожного конструктора $c: M^n \to M$ елементів з $M$ існує конструктор значень функції $h_c\ M^n \times W^n \to W$, такий, що $f(c(x_1,...,x_n)) = h_c(x_1,...,x_n, f(x_1),..., f(x_n))$ для будь-яких $x_1,...,x_n \in M$.

**(ПФ) Повнота.** Функція $g$ та конструктори $h_c$ дають змогу за скінченну кількість кроків отримати кожне із значень функції $f$ на $M$ і тільки їх.

Зазначимо, що деякі аргументи в конструкторі $h_c$ можуть бути фіктивними, а як конструктор $h_c$ може використовуватись і сама функція $f$. Індуктивні визначення приводять у загальному випадку до багатозначних часткових функцій. Справді, кожному виводу елемента $x \in M$ в $M(M_0, F)$, що закінчується певним правилом $x_1,...,x_n \underset{c}{\Rightarrow} x$, відповідає дерево з конструктором $c$ в його корені й піддеревами, що відповідають підвиводам елементів $x_1,...,x_n$. Сам елемент $x$ може бути побудований у результаті проходу дерева виводу, наприклад, зліва направо і знизу вгору (тобто в напрямку від листків до кореня). Кожне дерево виводу елемента $x \in M$ породжує певне значення $f(x)$ індуктивно визначеної функції $f$. І якщо існує два таких неізоморфних дерева виводу для даного $x$, то вони можуть породжувати різні значення. У випадку відсутності подібних неізоморфних дерев для всіх елементів множини $M$ називатимемо її індуктивне визначення **строгим.** Строгість індуктивного визначення гарантують такі дві умови (але вони не є необхідними): 1) області значень будь-яких двох конструкторів $c_1$ та $c_1$ не перетинаються і 2) для кожного $x \in M$ є тільки одна сукупність складових $x_1,...,x_n$ така, що $x = c(x_1,...,x_n)$. Остання умова виконується, наприклад, у випадку взаємно однозначних конструкторів. Індуктивне визначення слів $\mathbf{IB}_1$ є строгим. Наступне $\mathbf{IB\Phi}_3$ спирається на це індуктивне визначення слів $\mathbf{IB}_1$ і задає функцію $// : \Sigma^* \to N$, що обчислює довжину слова:

$(\mathbf{Б\Phi}_3)\ /\varepsilon/ = 0\ \ (\mathbf{I\Phi}_3)\ /cons(a,w)/ = 1 + /w/$.

Як і у випадку індуктивних множин, пункт Повноти **(ПФ)** може замінюватись на Повноту Відносну. А саме,

**(ПВФ) Повнота Відносна.** Функція $g$ та конструктори $h_c$ дають змогу за скінченну кількість кроків отримати кожне із значень функції $f$ на $M$, що задовольняє певний фільтр-термінатор $P(x,y)$.

За допомогою Повноти Відносної можна детермінізувати багатозначну індуктивно визначену функцію, відкидаючи фільтром-термінатором «зайві» значення для даного аргументу.

Аналогічний вигляд мають індуктивні визначення функції від кількох змінних та систем взаємно індуктивних функцій. Останні особливо важливі в застосуваннях. Обмежимося для простоти випадком бінарних функцій. Для $n$-арних функцій визначення будуть аналогічними. Нехай $M_1$ та $M_2$ – довільні індуктивні множини. Домовимося послідовність вигляду $a_{11}, a_{12}, ..., a_{1m}, \ldots, a_{n1}, a_{n2}, ..., a_{nm}$ записувати скорочено $[a_{ij}]_{i,j=1}^{n,m}$. **ІВФ** функції $f : M_1 \times M_2 \to W$ має вигляд:

**(БФ) База індукції.** $\forall < x, y > \in M_1^{(0)} \times M_2^{(0)}$: $f(x,y) = g(x,y)$, де $g(x,y)$ – «відома» функція, що задана на множині пар базових елементів множин $M_1$ та $M_2$. Функція $g(x,y)$ може задавати значення функції $f(x,y)$ і на ширшій області $D_0 \subset M_1 \times M_2$, за умови, що вони є надмножинами множини $M_1^{(0)} \times M_2^{(0)}$.

**(ІФ) Індуктивний перехід.** Для кожної пари конструкторів $c : M_1{}^n \to M_1$, $d : M_2{}^m \to M_2$ елементів множин $M_1$ та $M_2$ існує конструктор $h_{<c,d>} : M_1{}^n \times M_2^m \times W^{n \times m} \to W$ значень функції такий, що для будь-яких $x_1,...,x_n \in M_1$, $y_1,...,y_n \in M_2$:

$$f\left(c_1(x_1,...,x_n), c_2(y_1,...,y_m)\right) =$$
$$= h_c(x_1,...,x_n, y_1,...,y_m, [f(x_i,y_j)]_{i=1,j=1}^{n,m}).$$

**(ПФ) Повнота.** Функція $g$ та конструктори $h_{<c,d>}$ дають змогу за скінченну кількість кроків отримати кожне зі значень функції $f$ на $M_1 {\times} M_2$ і тільки їх.

Зауважимо, що на практиці конструктори $h_{<c,d>}$, як правило, досить прості і більшість їх аргументів фіктивні. Наприклад, наступне $\mathbf{IB\Phi}_4$ задає операцію множення натуральних чисел:

$$(\mathbf{Б\Phi}_4)\,(x \times 0) = 0)\,,$$

$$(\mathbf{I\Phi}_4)\,(x \times s(y)) = x + (x \times y)\,.$$

Єдиним конструктором $h$ тут виступає операція додавання. Розглянемо іншу задачу – про Ханойські вежі [3]. Нехай $T = \{1,2,3\}$ – номери кілочків для розташування веж, а $Hanoi : N \times T \times T \to (T \times T)^*$ – функція, що задає послідовність дій з переміщення вежі висотою $n$ з одного кілочка на інший. Нагадаємо, що дія полягає у взятті верхнього кільця однієї з веж та перекладанні його на іншу. При цьому менше кільце не кладеться на більше. Для подання однієї такої дії достатньо вказати пару чисел вигляду $<$ звідки, куди $>$, що належить $T \times T$. Нехай «$\circ$» – операція конкатенації двох послідовностей. **ІВФ**$_5$ функції *Hanoi* має вигляд:

$$(\textbf{БФ}_5)\ Hanoi(1,x,y) = cons(<x,y>,\varepsilon),$$

де $\varepsilon$ – порожня послідовність.

$$(\textbf{ІФ}_5)\ Hanoi(s(n),x,y) = Hanoi(n,x,6-x-y) \circ$$
$$\circ\ cons(<x,y>, Hanoi(n,6-x-y,y))$$

Коректність **ІВФ**$_5$ просто довести структурною індукцією по $n$.

Як і у випадку унарних функцій, конструктором $h_{<c,d>}$ може використовуватись і сама функція $f$. На відміну від унарних функцій, така можливість значно розширює можливості **ІВФ.**

Індуктивні визначення систем функцій (**ІВФС**) розглянемо на прикладі бінарних функцій вигляду $f: M \times M \to M$, де $M = M(M_0, F)$. **ІВФС** системи функцій $<f_1,...,f_k>$ має структуру:

**(БФС) База індукції.** $\forall <x,y> \in M_0 \times M_0$: $f_r(x,y) = g_r(x,y)$, $r = 1,k$, де $g_r(x,y)$ – «відомі» функції, задані у загальному випадку на надмножинах пар базових елементів.

**(ІФС) Індуктивний перехід.** Для кожної функції $f_r$, $r = 1,k$, та пари конструкторів $c: M^n \to M$, $d: M^m \to M$ існує конструктор $h_{<c,d>,r}: M^n \times M^m \times M^{n \times m \times k} \to M$ значень функції такий, що для будь-яких $x_1,...,x_n$, $y_1,...,y_n \in M$:

$$f_r\big(c(x_1,...,x_n), d(y_1,...,y_m)\big) =$$
$$= h_{<c,d>,r}(x_1,...,x_n,y_1,...,y_m,[f_1(x_i,y_j)]_{i,j=1}^{n,m},...$$
$$...,[f_k(x_i,y_j)]_{i,j=1}^{n,m}).$$

**(ПФС) Повнота.** Функції $g_r$ та конструктори $h_{<c,d>,r}$ дають змогу за скінченну кількість кроків отримати кожне зі значень функцій $f_r$, $r = 1,k$, на парах аргументів з $M^2$ і тільки їх.

Усі наведені **ІВФ** мають одну спільну рису – значення функції на своїх аргументах при індуктивному переході визначаються за допомогою значень функції на складових її аргументів. Таку індукцію називатимемо індукцією **типу згортки**. Іншим, дуальним, варіантом індуктивних визначень функцій є індуктивні визначення функцій **типу розгортки**, коли значення функції на аргументах при індуктивному переході визначаються за допомогою значень функції на заданих або нових аргументах, отриманих із заданих за допомогою конструкторів. У цьому випадку База індукції задає умову завершення індуктивних переходів по всьому фронту області її визначення. Прикладом **ІВФ** типу розгортки є визначення **ІВФ**$_6$ відомої числової функції Аккермана $B(x,y)$ [5]:

$$(\textbf{БФ}_6)\ B(s(x),0) = sg(x),\ B(0,y) = 2 + y.$$

$$(\textbf{ІФ}_6)\ B(s(x),s(y)) = B(x, B(s(x),y)).$$

Як бачимо, у пункті (**ІФ**$_6$) значення функції $B$ залежить по першому аргументу від нього самого, а не тільки від його попередника.

Загальне **ІВФ** функції $f: M_1 \times M_2 \to W$ типу розгортки має вигляд:

**(БФ) База індукції.** $\forall <x,y> \in D_0 : f(x,y) = g(x,y)$, де $g(x,y)$ – «відома» функція, що задана на деякій підмножині пар $D_0 \subseteq M_1 \times M_2$.

**(ІФ) Індуктивний перехід.** Для кожної пари конструкторів $c: M_1{}^n \to M_1$, $d: M_2{}^m \to M_2$ елементів множин $M_1$ та $M_2$ існує конструктор $h_{<c,d>}: M_1{}^n \times M_2^m \times W^{k^2} \to W$, $k \geq 0$, значень функції $f$ такий, що для будь-яких $x_1,...,x_n \in M_1$, $y_1,...,y_m \in M_2$: $f\big(c(x_1,...,x_n), d(y_1,...,y_m)\big) = h_{<c,d>}(z_1,...,z_k,[f(z_i,z_j)]_{i=1,j=1}^{k})$, де $z_r$, $r = 1,k$, або збігається з одним із $x_i$, $y_j$, або виведений з них за допомогою конструкторів.

**(ПФ) Повнота.** Функція $g$ та конструктори $h_{<c,d>}$ дають змогу за скінченну кількість кроків отримати кожне із значень функції $f$ на $M_1 \times M_2$ і тільки їх.

Як і раніше, конструктори $h_{<c,d>}$ можуть збігатися з функцією $f$ і містити велику кількість фіктивних параметрів. Останнє особливо актуально у випадку загальних *n*-арних функцій. Щоб уникнути фіктивних аргументів, є сенс переходити від звичайних *n*-арних операцій до так званих розширених *I*-арних операцій та відповідних їм алгебраїчних структур [6]. Але це потребує вже переходу до нижчого рівня абстракції.

Аналогічний вигляд мають індуктивні визначення систем функцій типу розгортки. Розглянемо два приклади. Перше визначення задає систему функцій $<f,g>$, $f: N \to N^*$, $g: N \times N \to N^*$, в якій $f$ продукує розкладання натурального числа на прості множники, а $g$ є допоміжною функцією. Наприклад, $f(24) = =<2,2,2,3>$. Позначимо «%» операцію взяття залишку від цілочислового ділення двох чисел.

**(БФС$_7$)** $f(1) = g(1,x) = \varepsilon$ для всіх $x \in N$, $g(x,y) = cons(x,\varepsilon)$ для всіх $1 < x < y^2$, і в $x$ немає дільників менших від $y$.

**(ІФС$_7$)** Для будь-яких $x$, $y > 1$:

$$f(x) = (x\%2 = 0 \to cons(2, f(x/2)))\#g(x,3)),$$
$$g(x,y) = (x\%y = 0 \to cons(y, g(x/y, y))\#g(x, y+2)).$$

Визначення **ІВФС$_7$** є справді індуктивним визначенням типу розгортки, оскільки виклики $g$ не зменшують $x$, а $y$ навіть збільшують на 2.

Нехай $Z = \{0,1,...,n\}$, $W = \{\overline{0},\overline{1},...,\overline{n-1}\}$, $Bool = =\{\overline{0},\overline{1}\}$. У наступному прикладі функція $Queen: N \to (W^*)^*$ генерує всі розв'язки відомої комбінаторної задачі про $n$ ферзів [4]. Інші функції в системі: $BackTrack: W^* \times Z \times (W^*)^*$, $Q: Z \times W^* \times W \to Bool$, $P: W^* \times W \to Bool$, $\exists u(\prec u): W^* \times W \to Bool$, $Ju(\prec u): W^* \times W \to W$, $h: W^* \to W^*$, $L: W^* \to Z$, $[]: W^* \times W \to Z$ є допоміжними. У задачі необхідно знайти всі позиції $n$ ферзів на шаховій дошці розміром $n \times n$, в яких вони не загрожують один одному. Конфігурації ферзів подаються словами $\overline{a}_1\overline{a}_2...\overline{a}_r$ в алфавіті $W$. Слово $\overline{530}$ подає конфігурацію трьох ферзів. Вони розміщені у перших трьох вертикалях дошки: на 1-й вертикалі ферзь стоїть на 6-й клітці (нумерація кліток вертикалі починається з 0), на 2-й вертикалі – на 4-й позиції і на третій вертикалі – на 1-й позиції. Функція *BackTrack* реалізує бектрекінг і генерує найближчу (у лексикографічному порядку) до $x$ правильну конфігурацію ферзів, розміщених у перших кількох підряд вертикалях. Якщо довжина чергової правильної конфігурації є $n$, то вона долучається до списку. Відповідне ІВФС має вигляд:

**(БФС$_8$)** $BackTrack(\varepsilon,k,y) = y$, $h(\varepsilon) = \varepsilon$, $x \underset{k}{\prec} u = \overline{0}$ для $k = n$, $Q(j,x,\overline{k}) = \overline{1}$ для $j > |x|$, $\overline{a}x[1] = a$ для довільного $x$, $|\varepsilon| = 0$.

**(ІФС$_8$)** Для будь-яких $n > 3$, $x \in W^*$, $y \in (W^*)^*$, $0 \le k \le n-1$,:

$$Queen(n) = BackTrack(\overline{1},3,\varepsilon),$$

$$BackTrack\ (x,k,y) =$$
$$= \begin{pmatrix} |x| = n \to BackTrack(h(x), L(x)+1, cons(x,y)) \\ |x| < n \to (\exists u(x \underset{k}{\prec} u) \to BackTrack(Ju(x \underset{k}{\prec} u), 0, y)\# \\ \#BackTrack(h(x), L(x)+1, y)) \end{pmatrix}$$

$|\overline{a}x| = 1 + |x|$, $h(app(z,\overline{a})) = z$, $L(app(z,\overline{a})) = a$,

$$P(x,\overline{k}) = Q(1,x,\overline{k}),\ \overline{a}x[k] = x[k-1],$$

$$\exists u(x \underset{k}{\prec} u) = \left(P(x,\overline{k}) \to \overline{1}\,\#\,\exists u(x \underset{k+1}{\prec} u)\right),$$

$$Ju(x \underset{k}{\prec} u) = \left(P(x,\overline{k}) \to x \circ \overline{k}\,\#\,Ju(x \underset{k+1}{\prec} u)\right),$$

$$Q(j,x,\overline{k}) = \big((x[j] = k \vee (|x[j]-k| = |x|+1-j)) \to \\ \to \overline{0}\,\#\,Q(j+1,x,\overline{k})\big).$$

Коректність визначень ІВФС$_7$ та ІВФС$_8$ можна перевірити структурною індукцією.

**2. Процедуризація індуктивно визначених функцій.** Формально індуктивні визначення функцій типу згортки є частковим випадком індуктивного визначення типу розгортки. Тому, розглядаючи проблему процедуризації індуктивно визначених функцій, обмежимося тільки визначеннями типу розгортки. Процедуризація таких функцій здійснюється, як і у випадку звичайних конструктивних функцій, розглянутих у першій частині роботи [1], але спирається вже на синтаксичне подання індуктивно визначених функцій за допомогою термів 2-го ступеня. Нехай $\Omega = \langle S,C,F \rangle$ – довільна сигнатура, а $\Phi$ – довільний алфавіт типізованих функціональних змінних сигнатури $\Omega$. Тобто кожному символу $G \in \Phi$ поставлено у відповідність певне слово $w \in S^+$ та певний сорт $s \in S$. Слово $w$ визначає сорти аргументів, а $s$ – сорт результатів функціональної змінної $G$. Вираз $w \mapsto s$ називатимемо **типом** змінної $G$ в сигнатурі $\Omega$. Нехай $X$ – довільний алфавіт типізованих предметних змінних сигнатури $\Omega$. Зафіксуємо певний сорт $s \in S$. Визначимо індуктивно сукупність $T^s_\Omega(X,\Phi)$ термів 2-го ступеня типу $s$ сигнатури $\Omega$ над алфавітами $X$ та $\Phi$ (**ІВ$_9$**):

**(Б$_9$)** Константи $c \in K$ типу $s$ та змінні $x \in X$ типу $s$ – базові терми типу $s$.

**(І$_9$)** Нехай $\alpha \in T^{Bool}_\Omega(X)$ – предикатний терм, $r,t \in T^s_\Omega(X,\Phi)$ – довільні терми типу $s$, $f \in F$ та $G \in \Phi$ – відповідно функціональний символ та функціональна змінна типу $s_1...s_k \mapsto s$,

$t_i \in T_{\Omega}^{s_i}(X,\Phi)$ – терм типу $s_i$ для $1 \le i \le k$. Тоді $(\alpha \to r\#t)$, $f(t_1,...,t_k)$, $G(t_1,...,t_k) \in T_{\Omega}^{s}(X,\Phi)$. Усі ці терми називаються структурними.

Терми вигляду $(\alpha \to r\#t)$ називаються **умовними**. Позначимо $T_{\Omega}(X) = \bigcup_{s \in S} T_{\Omega}^{s}(X,\Phi)$ множину всіх $\Omega$-термів 2-го ступеня. $\Omega$-терми з сукупності $T_{\Omega} = T_{\Omega}(\varnothing,\varnothing)$, що не містять предметних та функціональних змінних, будемо називати **замкненими** або **базовими.**

Визначимо тепер семантику $\Omega$-термів 2-го ступеня. Нехай $A = \left(S^A, K^A, F^A\right)$ – довільна $\Omega$-алгебра. **Оцінкою** типізованих змінних з $X$ в $\Omega$-алгебрі $A$, як і раніше, називається довільна функція $\sigma : X \to \bigcup_{s \in S} s^A$, що зберігає типи змінних. **Оцінкою** функціональних змінних з $\Phi$ в $\Omega$-алгебрі $A$ називається довільна функція $\sigma : \Phi \to \bigcup_{s \in S} T_{\Omega}$, що зберігає типи. Тобто для кожного символу $G \in \Phi$ будь-якого типу $w \mapsto s$ терм $\sigma(G) \in T_{\Omega}^{s}(X,\Phi)$. Позначимо $\left(\to^A\right) : Bool \times A \times A \to A$ 3-арну операцію вибору в $A$ як таку, що для будь-яких $a,b \in A$: $(tt \to a,b) = a$ і $(ff \to a,b) = b$.

Значення $I[t]_{\sigma}^{A}$ довільного $\Omega$-терму $t \in T_{\Omega}^{s}(X,\Phi)$ в $\Omega$-алгебрі $A$ при оцінці змінних $\sigma$ визначається індуктивно:

$(\mathbf{Б}_{10,a})$ $I[c]_{\sigma}^{A} = c^A$ для всіх констант $c \in C$;

$(\mathbf{Б}_{10,b})$ $I[x]_{\sigma}^{A} = \sigma(x)$ для всіх $x \in X$;

$(\mathbf{Б}_{10,c})$ $I[G]_{\sigma}^{A} = \sigma(G)$ для всіх $G \in \Phi$;

$(\mathbf{I}_{10,a})$ $I[f(t_1,...,t_k)]_{\sigma}^{A} = f^A\left(I[t_1]_{\sigma}^{A},...,I[t_k]_{\sigma}^{A}\right)$ для всіх $f \in F$;

$(\mathbf{I}_{10,b})$ $I[G(t_1,...,t_k)]_{\sigma}^{A} = I[G]_{\sigma}^{A}\left(I[t_1]_{\sigma}^{A},...,I[t_k]_{\sigma}^{A}\right)$ для всіх $G \in \Phi$;

$(\mathbf{I}_{10,c})$ $I[(\alpha \to r\#t)]_{\sigma}^{A} = (I[\alpha]_{\sigma}^{A} \to^A I[r]_{\sigma}^{A}, I[t]_{\sigma}^{A})$.

Як і у випадку звичайних термів, має місце Лема.

**Лема.** Значення терму 2-го ступеня в алгебрі $A$ не залежить від оцінки змінних, що не входять в нього.

**Доведення.** Аналогічне доведенню Леми для звичайних термів [1].

З точки зору семантики кожен з $\Omega$-термів 2-го ступеня $K = K(G_1,...,G_k)$, $G_i : w_i \to s_i$, $1 \le i \le k$, задає в алгебрі $A$ певну композицію $K^A : \prod_{i=1}^{k} s_i{}^A \to s^A$, яка кортежу термів $(t_1,...,t_k)$, такому, що $t_i : s_i$, $1 \le i \le k$, ставить у відповідність функцію $K^A(G_1 \mapsto t_1,...,G_k \mapsto t_k)$ (або скорочено – $K^A(t_1,...,t_k)$), яка визначається процедурно за таким **Правилом.**

Нехай $K = K(x_1,...,x_m)$, де $x_i : r_i$, $1 \le i \le m$ – усі предметні змінні, що входять у терми $t_i$. **Правило** для знаходження значення функції $K^A(x_1,...,x_m)$, що відповідає значенню змінних $x_1 = a_1,...,x_m = a_m$, полягає, як і у випадку звичайних термів, у побудові **обчислення** $K_0, K_1,...,K_n,...$ за термом $K$, такого, що

1) $K_0 = K\left(x_1 \mapsto c_{a_1},...,x_m \mapsto c_{a_m}\right)$;

2) для кожного $i \ge 0$ $\Omega$-терм $K_{i+1}$ отримано з $\Omega$-терму $K_i$ шляхом заміни в ньому крайнього лівого і крайнього внутрішнього його структурного підтерму $t$ на константу $c_a$, де $a$ – значення $I[t]_{\sigma}^{A}$ цього підтерму в алгебрі $A$ при $\sigma(x_i) = a_i$, $1 \le i \le m$ і $\sigma(G_i) = t_i$, $1 \le i \le k$.

Якщо обчислення скінченне і $K_n = c_a$ є останнім його термом для деяких $a \in A$, $n \ge 0$, то за визначенням $a = K^A(a_1,...,a_m)$. Інакше значення функції $K^A(a_1,...,a_m)$ вважається невизначеним.

**Розглянемо приклад.** Покладемо $K = G$ для певної змінної $G \in \Phi$, і нехай $\sigma$ – оцінка в $\Omega_1$-алгебрі $N = \left(S^N, C^N, F^N\right)$, така, що $\sigma(G) = (n = 0 \to 1\#n \cdot G(pred(n)))$, $n \in X$, а 1, 2, … є скороченням відповідно термів *succ*(0), *succ*(*succ*(0)) і т. д. Як ми побачимо, $K^N(n) = n!$. Розглянемо спочатку приклад обчислення значення функції $K^N(n)$ для $n = 2$. За визначенням,

$$K_0 = (2 = 0 \to 1\#2 \cdot G(pred(2)));$$

$$K_1 = (ff \to 1\#2 \cdot G(pred(2)));$$

$$K_2 = 2 \cdot G(pred(2));$$

$$K_3 = 2 \cdot G(1);$$

$$K_4 = 2 \cdot (1 = 0 \to 1\#1 \cdot G(pred(1)));$$

$$K_5 = 2 \cdot \left( ff \to 1\#1 \cdot G\left( pred(1) \right) \right);$$

$$K_6 = 2 \cdot 1 \cdot G\left( pred(1) \right);$$

$$K_7 = 2 \cdot 1 \cdot G(0);$$

$$K_8 = 2 \cdot 1 \cdot \left( 0 = 0 \to 1\#0 \cdot G\left( pred(0) \right) \right);$$

$$K_9 = 2 \cdot 1 \cdot \left( tt \to 1\#0 \cdot G\left( pred(0) \right) \right);$$

$$K_{10} = 2 \cdot 1 \cdot 1.$$

Отже, обчислення закінчилось базовим термом $2 \cdot 1 \cdot 1$, значення якого в $\Omega_1$-алгебрі $N$ є 2. Доведемо за допомогою індукції, що $K^N(n) = n!$ для будь-якого натурального $n$.

**База.** $K^N(0) = 1 = 0!$.

**Індуктивний перехід.** Нехай $K^N(n) = n!$ для деякого $n \ge 0$. Тоді

$$K(succ(n)) \Rightarrow$$

$$\Rightarrow \left( succ(n) = 0 \to 1\# succ(n) \cdot G\left( pred(succ(n)) \right) \right) \Rightarrow$$

$$\Rightarrow \left( n+1 = 0 \to 1\# succ(n) \cdot G\left( pred(succ(n)) \right) \right) \Rightarrow$$

$$\Rightarrow \left( ff \to 1\# succ(n) \cdot G\left( pred(succ(n)) \right) \right) \Rightarrow$$

$$\Rightarrow \left( succ(n) \cdot G\left( pred(succ(n)) \right) \right) \Rightarrow$$

$$\Rightarrow \left( (n+1) \cdot G\left( pred(succ(n)) \right) \right) \Rightarrow$$

$$\Rightarrow \left( (n+1) \cdot G\left( pred(n+1) \right) \right) \Rightarrow$$

$$\Rightarrow \left( (n+1) \cdot G(n) \right) \Rightarrow \ldots \Rightarrow (n+1) \cdot c_{n!}.$$

Але $((n+1) \cdot c_{n!})^N$ дорівнює за визначенням $(n+1)!$.

Як ми не раз наголошували в першій частині роботи [1], всі розглянуті нами алгебраїчні засоби специфікації об'єктів та функцій відповідають рівню семантики «чорного ящика», яка не виходить за межі кортежів фіксованої довжини. Така семантика добре «працює» на ранніх етапах аналізу та проектування інформаційних моделей, але її недостатньо, щоб забезпечити всі етапи життєвого циклу інформаційних моделей. Окремі незручності такої семантики, пов'язані з необхідністю звертатись до фіктивних аргументів функцій, вже згадувались в попередньому параграфі. Третя, заключна, частина роботи присвячена нетрадиційним алгебраїчним структурам, зокрема так званим Прикладним Програмним Алгебрам (ППА), орієнтованим на більш адекватну порівняно з класичними алгебраїчними структурами специфікацію програмних структур та даних [6].


1. *Зубенко В. В.* Алгебраїчні засоби специфікації інформаційних моделей. I.– Наукові записки НаУКМА. Т. 19–20.– Комп'ютерні науки.– 2002.– С. 5–17.
2. *Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л.* Алгебра. Языки. Программирование.– К.: Наукова думка, 1974.– 328 с.
3. *Абрамов С. А.* Математические построения и программирование.– М.: Наука.– 1978.– 191 с.
4. *Глибовець М. М., Олецький О. В.* Штучний інтелект.– К.: ВД «КМ Академія», 2002.– 365 с.
5. *Мальцев А. И.* Алгоритмы и рекурсивные функции.– М.: Наука, 1964.– 388 с.
6. *Zubenko V. V.* Appleid Program Algebras / Technical Report 8504.– Kiel.: CAU, 1985.– 51 p.



*V. V. Zubenko*


## ALGEBRAIC TOOLS FOR INFORMATION SYSTEMS SPESIFICATION. II


*Recursive tools for information systems spesification is discussed.*