УДК 004.415.28

М.С. Львов

Херсонський державний університет, Херсон

ВЕРИФІКАЦІЯ ІНТЕРПРЕТАТОРІВ АЛГЕБРАЇЧНИХ ОПЕРАЦІЙ В РОЗШИРЕННЯХ БАГАТОСОРТНИХ АЛГЕБР

Розроблення алгоритмів виконання алгебраїчних обчислень є однією з основних задач, що виникають при реалізації математичних систем, основаних на символьних перетвореннях. Математичною моделлю цієї задачі є багатосортні алгебраїчні системи. У даній роботі розглянуто підхід до верифікації інтерпретаторів багатосортних алгебраїчних операцій за їх специфікаціями, оснований на конструктивному уточненні поняття розширення багатосортної алгебраїчної системи та доведенні аксіом алгебраїчної системи. Цей підхід проілюстровано прикладами верифікації інтерпретаторів операцій поля раціональних чисел, багаточленів однієї змінної та алгебри висловлень. Практика використання цього підходу при розробленні математичних систем навчального призначення показала його ефективність і навіть універсальність.

Ключові слова: *математичні системи, символьні перетворення, багатосортні алгебраїчні системи, розширення алгебраїчних систем, верифікація, інтерпретатори алгебраїчних операцій.*

Вступ

Розроблення алгоритмів виконання алгебраїчних обчислень є однією з основних задач, що виникають при реалізації математичних систем, основаних на символьних перетвореннях [6].

Математичною моделлю цієї задачі є багатосортні алгебраїчні системи (надалі БАС) [9].

Практика розроблення навіть достатньо простих математичних систем навчального призначення [1 – 3] показала, що реалізація алгебраїчних обчислень потребує ретельного попереднього проектування БАС шляхом розроблення ієрархій сортів БАС та специфікацій інтерпретаторів багатосортних алгебраїчних операцій [8, 9].

В силу цілого ряду причин [9] для реалізації обчислень, основаних на символьних перетвореннях, ми використовуємо систему алгебраїчного програмування APS [4 – 6], адаптовану В. Песчаненко [10] для наших цілей. APS використовує технології алгебраїчного програмування, основані на системах правил переписувань та стратегіях переписувань. Таким чином, інтерпретатор алгебраїчної операції визначається системою правил переписувань термів (rewriting rules system).

У даній роботі ми пропонуємо підхід до верифікації інтерпретаторів багатосортних алгебраїчних операцій за їх специфікаціями, оснований, по-перше, на конструктивному уточненні поняття розширення багатосортної алгебраїчної системи і, по-друге, на алгоритмі доведення аксіом БАС як тотожностей. Основна ідея є достатньо простою: якщо інтерпретатори операцій БАС задано і кожна аксіома БАС є рівністю або умовною рівністю, вона є тотожністю або умовною тотожністю у даній конструктивній реалізації БАС, тобто її можна доводити.

Цей підхід проілюстровано прикладами реалізації інтерпретаторів операцій поля раціональних чисел, багаточленів однієї змінної та алгебри висловлень.

1. Багатосортні алгебри як математична модель алгебраїчних обчислень

1.1. Сигнатури багатосортної алгебри

Означення 1.1 Нехай $\mathbf{U}=\{u_1,\ldots,u_k\}$ – скінчена множина символів, яка називається сигнатурою сортів. Символи u_l, $l\in\{1,...,k\}$ називаються іменами сортів або просто сортами.

Далі ми будемо користуватися, зокрема, такими сортами – елементами сигнатури сортів:

Variable – ім'я сорту – множини змінних;

Bool – ім'я сорту – множини логічних значень;

Nat – ім'я сорту – множини натуральних чисел;

Int – ім'я сорту – множини цілих чисел;

Real – ім'я сорту – множини дійсних чисел.

Інші імена сортів ми будемо вводити при означенні відповідних алгебраїчних понять.

Означення 1.2 Нехай, далі, $\mathbf{S}=\{S_{u_1},...,S_{u_k}\}$ – скінчене сімейство множин, індексованих іменами сортів, які називаються областями значень відповідних сортів.

$S_{Variable}$ – множина змінних;

S_{Bool} – множина {False, True};

S_{Nat} – множина натуральних чисел;

S_{Int} – множина цілих чисел;

S_{Real} – множина дійсних чисел.

Означення 1.3 Багатосортною операцією f над сімейством **S** називається відображення $f: S_{u_1}\times S_{u_2}\times\ldots\times S_{u_m}\to S_v$, де $u_1,...,u_m,v\in\mathbf{U}$ – сорти аргументів та області значень операції f, відповідно, а m – арність операції f.

Тип операції визначається переліком імен сортів її аргументів та іменем сорту області її значень. Тип операції f будемо позначати через $(u_1,...,u_m)\to v$. Сигнатурою Σ операцій називається скінчена множина символів операцій разом з відображенням, яке кожному символу $\varphi\in\Sigma$ співставляє багатосортну операцію f_φ разом з її типом (якщо φ символ операції, то вираз $\varphi:(u_1,...,u_m)\to v$ означає, що цьому символу співставлено операцію типу $(u_1,...,u_m)\to v$).

Багатосортною операцією, є, наприклад, операція множення у векторному просторі. Якщо VectorSpace – ім'я сорту-множини векторів на полем Real дійсних чисел, то операція множення **Mult** "*" задає відображення

Mult : Real × VectorSpace → VectorSpace

Надалі ми будемо користуватися більш звичними, тобто традиційними математичними позначеннями операцій. Оскільки множення вектора на скаляр є бінарною операцією з інфіксною формою запису, маємо:

Real * VectorSpace → VectorSpace

Означення 1.4. Визначимо сорт Bool з областю значень $S_{Bool}=\{True,False\}$. Багатосортним предикатом P називається відображення $P: S_{u_1}\times\ldots\times S_{u_m}\to S_{Bool}$, де $u_1,...,u_m\in\mathbf{U}$, послідовність $u_1,...,u_m$ визначає тип предикату, а число m – його арність. Сигнатура Π багатосортних предикатів визначається аналогічно сигнатурі операцій як множина операцій символів предикатів, яким поставлені у відповідність багатосортні предикати разом з їх типами.

Означення 1.5. Багатосортною алгебраїчною системою **A** називається четвірка $\mathbf{A}=<\mathbf{S},\mathbf{U},\Sigma,\Pi>$, де **S** – множина сортів, індексованих символами множини **U**; $\Sigma=\{\varphi_1,...,\varphi_l\}$ – сигнатура багатосортних операцій; $\Pi=\{\pi_1,...,\pi_p\}$ – сигнатура багатосортних предикатів.

Зауваження. Оскільки сорт Bool можна включити до множини сортів, предикати можна розглядати як багатосортні операції. Тому, об'єднавши сигнатури операцій та предикатів, замість розгляду багатосортних алгебраїчних систем далі ми будемо

розглядати багатосортні алгебри.

Означення 1.6. Отже, нехай $\mathbf{A} =< \mathbf{S}, \mathbf{U}, \Sigma >$ – багатосортна алгебра. Нехай $u, v \in \mathbf{U}$ – символи сортів цієї алгебри. Будемо казати, що сорт v залежить від сорту u, якщо одна з операцій сигнатури Σ має тип $u_1 \times ... \times u \times ... \times u_m \to v$. Через $\mathbf{U}_v$ позначимо підмножину сортів, від яких залежать сорт v. Підмножину елементів Σ, що мають тип $u_1 \times ... \times u \times ... \times u_m \to v$ позначимо через Σ_v, а сімейство областей значень сортів $\mathbf{U}_v$ позначимо через $\mathbf{S}_v$. Обмеженням $\mathbf{A}_v$ багатосортної алгебри $\mathbf{A}$ на сорт v називається багатосортна алгебра $\mathbf{A}_v =< \mathbf{S}_v, \mathbf{U}_v, \Sigma_v >$.

Таким чином, багатосортна алгебра $\mathbf{A}$ може бути представлена набором обмежень (алгебр) $\mathbf{A}_v, v \in \mathbf{U}$, тобто $\mathbf{A} =< A_{u_1}, ..., A_{u_k} >$.

Приклад 1.

Завдання полягає у тому, щоб побудувати програмну систему, яка підтримує спрощення цілих алгебраїчних та тригонометричних виразів. Таким чином, ядро цієї системи має підтримувати обчислення у кільці поліномів та кільці тригонометричних виразів багатьох змінних над полем раціональних чисел. Отже специфікаціям підлягають такі алгебри – обмеження на указані сорти:

1. MultiPolynom – кільце поліномів багатьох змінних;
2. MultiTrig – кільце тригонометричних поліномів багатьох змінних;
3. LinComb – векторний простір лінійних комбінацій багатьох змінних (аргументи тригонометричних поліномів);
4. Rat – поле раціональних чисел (коефіцієнти поліномів та тригонометричних поліномів).

Відношення залежності сортів породжує структуру залежності на множині алгебр $\mathbf{A}_u, u \in \mathbf{U}$: алгебра $\mathbf{A}_v$ залежить від алгебри $\mathbf{A}_u$, якщо сорт v залежить від сорту u. Якщо відношення залежності не має циклів, то багатосортну алгебру можна будувати крок за кроком (інкрементно), будуючи алгебру $\mathbf{A}_v$, якщо алгебри, від яких $\mathbf{A}_u$ залежить, вже побудовані.

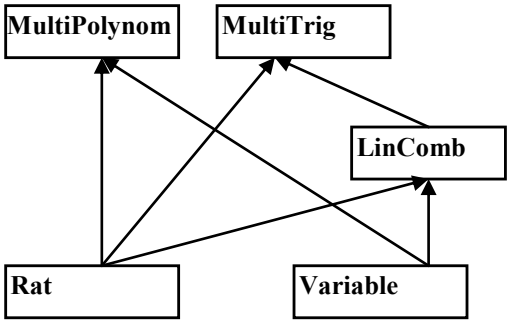

Рис. 1. Діаграма відношення залежності алгебр прикладу 1

1.2. Аксіоми та конструкції багатосортної алгебри

Для побудування алгебр $\mathbf{A}_u$ ми використовуємо їх аксіоматичні та конструктивні описи (означення).

Означення 1.7. Аксіомою алгебри $\mathbf{A}_u$ називається рівність або умовна рівність у сигнатурі Σ_u. Аксіоматичним описом алгебри $\mathbf{A}_u$ є, за означенням, є скінчений набір аксіом (система аксіом) алгебри $\mathbf{A}_u$.

В математиці, зокрема, в алгебрі, аксіоматичний метод широко застосовується для визначення властивостей різноманітних предметних областей. Алгебри, аксіоми яких є рівностями, називають многовидами. Алгебри, аксіоми яких є рівностями або умовними рівностями, називають квазімноговидами. Ми будемо користуватися алгебраїчною термінологією та відповідними системами аксіом з підручника [13].

Для визначення конкретної алгебри разом з аксіомами часто використовують так звані постулати мінімальності. Наприклад, поле раціональних чисел можна визначити як мінімальне поле, яке містить множину натуральних чисел. Аналогічно, кільцем многочленів $F[x]$ над полем F є мінімальне комутативне кільце з одиницею, яке містить поле F та змінну x.

Конструктивний опис алгебри $\mathbf{A}_u$ – це означення конструктору сорту $\mathbf{S}_u$ (тобто визначення термів, множина яких утворює сорт $\mathbf{S}_u$) та множини інтерпретаторів операцій Σ_u.

Означення 1.8. Сигнатурою конструкторів $\mathbf{T}_u$ називається скінчена множина символів операцій разом з відображенням, яке кожному символу $\tau \in \mathbf{T}_u$ співставляє символ сорту u разом з переліком символів сортів його аргументів (якщо τ – символ операції, то вираз $u = \tau(u_1, ..., u_m)$ означає, що цьому символу співставлено символ сорту u та символи сортів його аргументів $u_1, ..., u_m$).

Конструктором сорту $\mathbf{S}_u$ алгебри $\mathbf{A}_u$ називається система рівностей**,** яка визначає синтаксично елементи сорту $\mathbf{S}_u$ як терми у сигнатурі T_u. Отже, сорт $\mathbf{S}_u$ – це множина термів у (власній) сигнатурі T_u конструктора сорту $\mathbf{S}_u$.

Означення 1.8 є ключовим у нашому підході до специфікації алгебраїчних обчислень. Тому ми надамо необхідні приклади та пояснення.

Приклад 2.

Поле Rat раціональних чисел. Елементи цього поля – раціональні числа, які представляють у вигляді звичайних дробів. Конструктор сорту визначає стандартну форму представлення елементу цього сорту. Частіше за все це канонічна форма. Таким чином,

$$S_{Rat} = \{\frac{p}{q} : p \in S_{Int}, q \in S_{Nat}, GCD(p,q) = 1\}. \quad (1)$$

Роль конструктора сорту грає символ – горизонтальна риска. Цей же символ математики використовують для позначення операції ділення, зокрема, у Rat. Це явище у математиці є розповсюдженим. Таку ж роль грає, наприклад, символ кореня, символи додавання та множення ($a + b*\sqrt{2}$). І ці приклади можна продовжувати. Виявилося, однак, що для задачі специфікацій алгебраїчних обчислень це не зовсім зручно. Конструктори сортів, з нашої точки зору, мають бути позначеними окремо. Тому ми вводимо окремо поняття сигнатури операцій Σ та сигнатури конструкторів T. Зокрема, для конструктора сорту Rat ми використовуємо подвійну нахильну риску:

$$S_{Rat} = \{p // q : p \in S_{Int}, q \in S_{Nat}, GCD(p,q) = 1\}. \quad (2)$$

Ще однією важливою обставиною є те, що у стандартних формах представлення елементів сортів синтаксичні аспекти означення, які визначаються символами конструкторів, практично завжди поєднуються з семантичними аспектами, що задаються у вигляді контекстних умов, тобто предикатів. У нашому прикладі таким предикатом є рівність $GCD(p,q) = 1$.

Приклад 3.

Кільце Polynom поліномів однієї змінної над полем Rat. Елементи цього поля – поліноми, які представлені звичайно у вигляді суми мономів, упорядкованих у порядку спадання степенів.

$$S_{Polynom} = \{M_0 + M_1 + ... + M_k : M_i \in S_{Monom},$$
$$\deg(M_i) > \deg(M_{i+1})\}.$$

У специфікаціях сорту Polynom це означення має бути рекурсивним, оскільки треба видалити з формули символ “три точки” та визначити окремо випадок, коли поліном містить лише один моном. Як ми побачимо, при такому підході окремо треба визначити ще поняття степеня полінома.

$$S_{Polynom} = \{Q : Q = M + +P, M \in S_{Monom},$$
$$P \in S_{Polynom}, \deg Q \overset{df}{=} \deg M; \quad (3)$$
$$\deg(M) > \deg(P)\} \cup S_{Monom}.$$

Для визначення носіїв сортів ми будемо користуватися спеціальною мовою специфікацій, яка допускає нерекурсивні та рекурсивні синтаксичні визначення елементів сортів, визначення функцій доступу (зокрема, для формулювання контекстних умов) та контекстні умови.

Наведемо визначення сортів наших прикладів цією мовою:

```
Rat r ={
  (Int a)//(Nat b);          //
```
Конструктор сорту
```
    Num(r) = a, Den(r) = b;        //
```
Функції доступу (селектори)
```
    GCD(a, b) = 1                  //
```
Контекстна умова
```
  };

  Monom M =      {
          (Rat  c)$(Const  Variable
x)^^(Nat n); // Конструктор сорту
          Coef(M) = c,
                // Функції доступу
          Var(M) = x,
          Deg(M) = n
         };

  Polynom P = {
         (Monom   M)++(Polynom   Q);
                //Конструктор сорту
           LeadMon(P) = M,
                // Функції доступу
        LeadCoef(P) = Coef(M),
        Deg(P) = Deg(M);
        Deg(P) > Deg(Q)
                // Контекстна умова
         };
```

Для того, щоб реалізувати обчислення в деякій алгебрі $\mathbf{A}_v, v \in \mathbf{U}$, потрібно реалізувати алгоритми виконання кожної її операції таким чином, щоб виконувались аксіоми цієї алгебри.

Означення 1.9. Інтерпретатором операції сигнатури Σ_u називається функція, яку реалізовано алгоритмом виконання відповідної операції.

Інтерпретатори операцій визначаються засобами мови APLAN системи алгебраїчного програмування APS. Отже, ми включаємо цю мову у мову специфікацій.

Таким чином, для аксіоматичного та конструктивного опису алгебри $\mathbf{A}_v$ до її означення ми включаємо скінчену множину аксіом Ax_v та скінчену множину інтерпретаторів I_v. Тоді багатосортна алгебра $\mathbf{A}_v$ визначається наступним чином: $\mathbf{A}_v =< \mathbf{S}_v, \mathbf{U}_v, T_v, \Sigma_v, Ax_v, I_v >$.

2. Розширення алгебр

Означення 2.1. Нехай $\mathbf{A}_u$ та $\mathbf{A}_v$ – багатосортні алгебри. Багатосортна алгебра $\mathbf{A}_v$ називається розширенням багатосортної алгебри $\mathbf{A}_u$, якщо $S_u \subseteq S_v$ та для будь-якої пари операцій f_1 та f_2 типів відповідно $\varphi : (u_1,...,u_m) \to u$ та $\varphi : (v_1,...,v_m) \to v$, якщо виконуються входження $S_{u_1} \subseteq S_{v_1}, ..., S_{u_m} \subseteq S_{v_m}$, то $\forall (a_1,...,a_m) \in S_{u_1} \times ... \times S_{u_m}$, має місце рівність $f_1(a_1,...,a_m) = f_2(a_1,...,a_m)$.

Вкладенням називається ізоморфне відображення $\mathrm{Red}: S_u \to S'_v$, яке відображає S_u на підмножину $S'_v \subset S_v$. Обмеження алгебри $\mathbf{A}_v$ на підмножину $E_1(x),...,E_k(x)$, ізоморфне $\mathbf{A}_u$, визначається системою умовних тотожностей $E_1(x),...,E_k(x)$: $S'_v = \{a \in S_v \mid E_1(a),...,E_k(a)\}$. Застосування системи $E_1(x),...,E_k(x)$ як системи переписувань «спрощує» терм $a \in S'_v$ до терму $a' \in S_u$:

$\mathrm{Red}^{-1}(a) = a'$.

Конструктивний опис розширення $\mathbf{A}_v$ полягає в описі конструктора $\mathbf{A}_v$ та вкладення $\mathbf{A}_u$ в алгебру $\mathbf{A}_v$.

На рис. 2 подвійною стрілкою позначено той факт, що алгебра $\mathbf{A}_v$ є розширенням алгебри $\mathbf{A}_u$, в алгебрі $\mathbf{A}_v$ визначено конструкцію $v = \tau(u_1,...,u,...,u_m)$ та вкладення $\mathrm{Red}_{u,v}$.

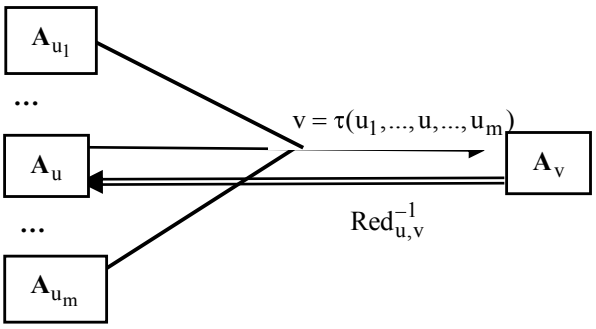

Рис. 2. Фрагмент діаграми розширень як конструкцій та вкладень

Довідка.

Відношення розширення БАС, в іншій термінології, відношення «сорт-підсорт» є основним у даній роботі. Багатосортні алгебри, частково упорядковані цим відношенням, називають упорядковано-сортними. Основи теорії упорядковано-сортних алгебр у її застосуванні до теорії програмування закладені у [11]. Російською мовою вони викладені у [12].

Приклад 4.

Розглянемо конструктор поля Rat (приклад 2). У відповідності до означення він визначає конструкцію Rat, аргументами якої є сорти Int та Nat. Доповнимо специфікації сорту Rat вкладенням $\mathrm{Red}: \mathrm{Rat} \to \mathrm{Int}$, яке визначено рівністю $\mathrm{Red}(a\,//\,1) = a$. Таким чином сорт Rat визначено як розширення сорту Int.

Розглянемо тепер конструктор кільця Polynom (приклад 3). Він визначає рекурсивно конструкцію Polynom, аргументом якої є сорт Monom. Доповнимо специфікації сорту Polynom вкладенням $\mathrm{Red}: \mathrm{Polynom} \to \mathrm{Monom}$, яке визначено рівністю $\mathrm{Red}(M + +0) = M$. Отже, сорт Polynom визначено як розширення сорту Monom.

У свою чергу, сорт Monom є розширенням сорту Degree з функцією Red, визначеною рівністю **`1$x^^k = x^^k`**, розширенням сорту LinMonom з функцією Red, визначеною рівністю **`a$x^^1 = a$x`** та розширенням сорту Rat з функцією Red, визначеною рівністю **`a$x^^0 = a`**. Сорти Degree та LinMonom є розширенням сорту Variable з функціями редукції, заданими відповідно рівностями **`x^^1 = x та 1$x = x.`** Отже, діаграма розширень має вигляд, як на рис. 3.

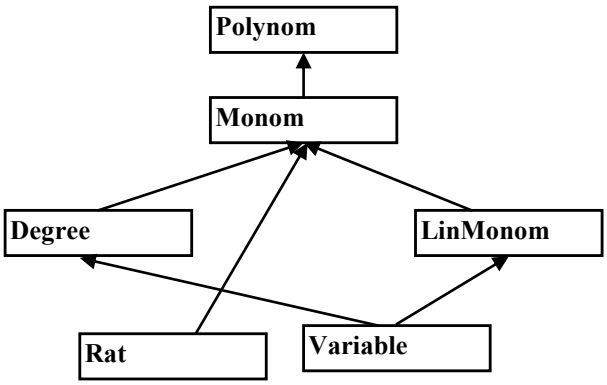

Рис. 3. Діаграма розширень прикладу 4

Механізм розширень є одним з основних методів специфікацій багатосортних алгебр. Зокрема, він дозволяє визначити перевантажені алгебраїчні операції та функції приведення алгебраїчних типів.

2.1. Статичні та динамічні розширення

Означення 2.2. Розширення B алгебри A називається статичним (нерекурсивним), якщо у його конструкторі $B = \varphi(A_1,...,A,...,A_n)$ жоден з аргументів не співпадає з B.

Приклади статичних розширень:

1. Поле раціональних чисел Rat є статичним розширенням кільця цілих чисел Int, оскільки

$$\mathrm{Rat}\quad R = (\mathrm{Int}\quad A)\,//\,(\mathrm{Nat}\quad B).$$

2. Півгрупа мономів Monom однієї твірної є статичним розширенням поля коефіцієнтів Coef, оскільки

$$\mathrm{Monom}\quad M = (\mathrm{Coef}\quad a)\$(\mathrm{Var}\quad x)\,\hat{}\,(\mathrm{Nat}\quad N).$$

Означення 2.3. Розширення B алгебри A називається динамічним (рекурсивним), якщо у його конструкторі $B = \varphi(A_1,...,A,...,A_n)$ щонайменше один з аргументів співпадає з B.

Конструктор динамічних розширень є рекурсивним визначенням, отже, місить як базову, так і рекурентну частини. Рекурентна частина визначається за допомогою конструктора сорту. Базова частина визначається вкладенням, тобто співвідношенням редукції.

Приклади динамічних розширень:

3. Векторний простір LinComb лінійних комбінацій багатьох змінних над полем Coef є лінійним динамічним розширенням одновимірного лінійного простору LinMonom, елемент якого має вигляд a$x. Елемент $w \in \text{LinComb}$ має вид $w = a_1\$x_1 + +a_2\$x_2 + +... + +a_m\$x_m$. Таким чином, його конструктор є рекурсивним означенням:

$\text{LinComb } w = (\text{Coef } a)\$(\text{Variable } x) + +(\text{LinComb } w)$,

$u + + 0 = u$

або

$\text{LinComb } w = (\text{LinMonom } u) + +(\text{LinComb } w)$,

$u + + 0 = u$.

4. Кільце многочленів однієї змінної Polynom над полем Coef. Це кільце в алгебрі традиційно позначається через $F[x]$.

$\text{Polynom } w = (\text{Monom } M) + +(\text{Polynom } w)$

$M + + 0 = u$

Практика використання динамічних розширень показала корисність їх подальшої класифікації як лінійних та бінарних (динамічних) розширень.

Означення 2.4. Динамічне розширення B алгебри A називається лінійним, якщо у його конструкторі $B = \varphi(A_1,...,A,...,A_n)$ у точності один з аргументів співпадає з B.

Динамічне розширення B алгебри A називається бінарним, якщо у його конструкторі $B = \varphi(A_1,...,A,...,A_n)$ у точності два аргументи співпадають з B.

Приклади 3, 4 розширень алгебр, наведені нами при його означенні, є прикладами лінійних динамічних розширень. Розглянемо приклад бінарного динамічного розширення:

5. Числове поле Rad, елементами якого є лінійні комбінації квадратних коренів натуральних чисел, вільних від квадратів, з раціональними коефіцієнтами можна представити як бінарне розширення поля Rat за допомогою наступної конструкції. Нехай $p_1, p_2, ..., p_n, ...$ – послідовність усіх простих чисел, розташованих у порядку зростання. Позначимо також через Q поле раціональних чисел. Введемо наступні позначення:

$$\text{Rad}_0 = Q, \quad \text{Rad}_n = \{r : r = a + b*\sqrt{p_n}, \; a, b \in \text{Rad}_{n-1}, \; n = 1,2,...\}.$$

Поле Rad є нескінченним об'єднанням зростаючої послідовності полів Rad_n:

$$\text{Rad} = \bigcup_{n=0}^{\infty} \text{Rad}_n, \qquad \text{Rat} = \text{Rad}_0 \subset \text{Rad}_1 \subset ... \subset \text{Rad}_n \subset ... \tag{4}$$

Отже, конструктор Rad має вид

$$\text{Rad } r = (\text{Rad } a) + (\text{Rad } b)*\sqrt{\text{Nat } p}. \tag{5}$$

Зауважимо, що послідовність розширень (4) є послідовністю скінчених алгебраїчних розширень полів коренями поліномів $x^2 - p_n = 0$.

Зверніть увагу на те, що у представлення (5) включені як опис елементу базової алгебри Rat q, так і опис механізму розширення – конструктор $(\text{Rad } a) + (\text{Rad } b)*\sqrt{\text{Nat } p}$. Така специфікація у точності відповідає математичному означенню (4). З іншого боку, опис базового елементу є зайвим, якщо його можна отримати з співвідношення вкладення. Дійсно, специфікація векторного простору LinComb з включенням до неї опису елемента базової алгебри LinMonom має вигляд

$$\text{LinComb } w = (\text{LinMonom } u) + +(\text{LinComb } w) \mid (\text{LinMonom } u).$$

Однак, включення $\text{LinMonom} \subset \text{LinComb}$ визначено рівністю $u + + 0 = u$. Тому окремий опис LinMonom u є зайвим. Ми допускаємо обидва способи специфікацій, не примушуючи користувача аналізувати різницю між ними у кожному окремому випадку.

Формули (4) безпосередньо узагальнюються на довільні динамічні розширення. Якщо алгебра B є динамічним розширенням алгебри A з конструктором $B = \varphi(A_1,...,B,...,A_n)$, зростаючу послідовність $B_0 \subset B_1 \subset ...B_n \subset ...$ визначимо таким чином:

$$1.\ B_{(0)} = A, \tag{6}$$

$$2.\ B_{(n+1)} = \varphi(A_1,...,B_{(n)},...,A_n). \tag{7}$$

Вкладення $\text{Red} : A \to B$ визначає вкладення $\text{Red}_i : B_{i+1} \to B_i$, звідки безпосередньо випливає представлення A у вигляді об'єднання зростаючої послідовності алгебр, кожна з яких є статичним розширенням попередньої.

$$B = \bigcup_{n=0}^{\infty} B_n, \quad B_0 \subset B_1 \subset ... \subset B_n \subset .... \tag{8}$$

Таким чином, динамічні розширення алгебр є послідовностями статичних розширень. Цей факт дозволяє, як ми побачимо далі, використовувати загальну схему реалізації динамічних розширень, виводячи відповідні системи переписуючих правил.

2.2. Верифікація алгебраїчних обчислень

У [11] показано, що поняття розширення алгебр дозволяє здійснювати синтез інтерпретуючих правил операцій та їх оптимізацію, спираючись на правила «загального випадку», які мають бути визначеними у специфікаціях відповідного сорту. Таким чином, *проблема верифікації алгебраїчних обчислень зводиться до проблеми обґрунтування правильності правил «загального випадку»*. Які властивості притаманні «правильним» специфікаціям опера-

цій? Як відрізняти «правильні» означення від «неправильних»? Для відповіді на ці питання ми маємо перевірити виконання наступних умов:

1. *Правила специфікації алгебраїчних операцій, визначених конструктивно в специфікаціях алгебри, мають задовольняти системі аксіом цієї алгебри.*

2. *Результат виконання операції має відповідати синтаксичним означенням та контекстним умовам конструктора сорту*.

2.3. Верифікація алгебраїчних обчислень: приклади

Приклад 5. Верифікація сорту BoolAlg.

Елементами сорту BoolAlg є формули логіки висловлювань багатьох змінних. Нехай $F(x_1, x_2, ..., x_n)$ – довільна формула формули логіки висловлювань від n змінних. Позначимо через **O, I** логічні значення відповідно істина та хибність. Тоді

$$F(x_1, x_2, ..., x_n) = x_n \,\&\, F(x_1 x_2, ..., x_{n-1}, I) \vee \vee \neg x_n \,\&\, F(x_1, x_2, ..., x_{n-1}, O).$$

Якщо позначити

$$A(x_1, ..., x_{n-1}) = F(x_1, ..., x_{n-1}, I),$$
$$B(x_1, ..., x_{n-1}) = F(x_1, ..., x_{n-1}, O),$$

отримаємо представлення

$$F(x_1, x_2, ..., x_n) = x_1 \,\&\, A(x_1, ..., x_{n-1}) \vee \vee \neg x_1 \,\&\, B(x_1, ..., x_{n-1}). \quad (9)$$

Тепер здійснимо послідовно такі ж перетворення формул **A**, **B** відносно змінних $x_{n-1}, ..., x_1$. В результаті отримаємо рекурсивне представлення формули логіки висловлювань. Насправді, через $BoolAlg_m$ позначимо множину формул логіки висловлювань від змінних $x_1, ..., x_m$. Тоді

$$BoolAlg_n = \{F : F = x_n \,\&\, A \vee \vee \neg x_n \,\&\, B,\ \ A, B \in BoolAlg_{n-1}\}.$$

Отже, сорт BoolAlg є об'єднанням зростаючої послідовності алгебр $BoolAlg_m$:

$$BoolAlg_0 = Bool,$$
$$BoolAlg_0 \subset BoolAlg_1 \subset \ldots \subset BoolAlg_m \subset \ \ldots (10)$$
$$BoolAlg = \cup BoolAlg_m.$$

Розширення такого типу ми назвали бінарними (означення 2.3). Зауважимо, що формула (9) задає канонічну форму формули алгебри висловлювань. Незважаючи на те, що ця канонічна форма менш відома, ніж ДКНФ та ДДНФ, на наш погляд, вона має властивості, що дозволяють достатньо просто визначити обчислення. Дійсно, позначимо

$$BF(A, B, x) \overset{df}{=} x \,\&\, A \vee \neg x \,\&\, B. \quad (11)$$

Таким чином, $BF(A, B, x)$ – конструктор сорту BoolAlg.

Тоді інтерпретатори логічних операцій визначаються формулами:

$$BF(A_1, B_1, x) \,\&\, BF(A_2, B_2, x) = = BF(A_1 \,\&\, A_2, B_1 \,\&\, B_2, x), \quad (12)$$

$$BF(A_1, B_1, x) \vee BF(A_2, B_2, x) = = BF(A_1 \vee A_2, B_1 \vee B_2, x), \quad (13)$$

$$\neg BF(A, B, x) = BF(\neg A, \neg B, x). \quad (14)$$

Ми бачимо, що основні логічні операції виконуються поаргументно! Нарешті, легко перевірити, що функція вкладення визначається рівністю

$$BF(A, A, x) = A. \quad (15)$$

Розглянемо одну з аксіом сорту Bool, наприклад, правило де Моргана

```
~(A&B) = ~A|~B.
```

Підставимо замість A, B їх конструктивні означення :

```
A = BF(A1,B1,x), B = BF(A2,B2,x).
   ~(BF(A1,B1,x)& BF(A2,B2,x)) =
   ~BF(A1,B1,x)|~BF(A2,B2,x).
```

Виконаємо обчислення у лівій та правій частинах рівності та випишемо відповідні контекстні умови:

```
BF(~(A1&A2),~(B1&B2),x) =
  BF(~A1|~A2,~B1|~B2,x),                (16)
   Arg(x)>Arg(~(A1&A2)),
  Arg(x)>Arg(~(B1&B2)),                 (17)
   Arg(x)>Arg(~A1|~A2)),
  Arg(x)>Arg(~B1|~B2)).                 (18)
```

Відповідно до означення конструктору `BF(A,B,x)`, рівність (16) означає, що її аргументи або синтаксично рівні, або зводяться до синтаксично рівних функцією вкладення `BF(A,A,x)=A.`

Оскільки інших контекстних умов нема, рівності аргументів (16) мають бути тотожностями алгебри висловлювань

```
~(A1&A2)≅~A1|~A2,~(B1&B2)≅~B1|~B2.
```

Висновок: перевірка виконання аксіоми де Моргана на елементах сорту зводиться до перевірки її виконання на аргументах конструктора сорту. Тому доведення має здійснюватися індукцією по індексам башти алгебр (8). Крок індукції полягає у доведенні умовної тотожності

```
~(A1&A2)≅~A1|~A2,~(B1&B2)≅~B1|~B2
```

→

```
~(BF(A1,B1,x)&BF(A2,B2,x))=
 ~BF(A1,B1,x)|~BF(A2,B2,x),
```

та доведенні нерівностей (17), (18).

Приклад 6. Верифікація сорту Polynom.

Розглянемо одну з аксіом сорту Polynom, наприклад, аксіому дистрибутивності

```
x*(y + z) = x*y + x*z.
```

Як і у попередньому прикладі, підставимо замість `x, y, z` їх конструктивні означення :

```
x = a++A, y =b++B, z = c++C.
```

Отримаємо:

```
(a++A)*((b++B)+(c++C)) =
```

(a++A)*(b++B)+(a++A)*(c++C).

Виконаємо обчислення за основними правилами з специфікації операцій сорту Polynom у лівій та правій частинах рівності та випишемо відповідні контекстні умови :

a(b+c)++((a(B+C)+A(b+c))+A(B+C))=
=(ab+ac)++((aB+Ab)+AB)+((aC+Ac)+AC)),
Deg(b)=Deg(c), Deg(a)>Deg(A),
Deg(b)>Deg(B), Deg(c)>Deg(C).

Звідки отримаємо дві рівності та нерівність, які треба довести:

a(b+c) = ab+ac, (19)

(a(B+C)+A(b+c))+A(B+C) =
((aB+Ab)+AB)+((aC+Ac)+AC)); (20)

Deg(a(b+c)) >
Deg(((a(B+C)+A(b+c))+A(B+C))). (21)

Розглянемо постановку задачі доведення нерівності (21). Ця нерівність є наслідком правил виконання операцій додавання та множення. Якщо ці правила правильно оперують з степенями, вона виконується автоматично. Тому задача доведення нерівностей, які формулюються як контекстні умови правильності аксіом, зводиться до задач доведення правильності умов у правих частинах правил інтерпретації операцій. У нашому випадку треба доводити такі умовні нерівності:

Для основного правила операції **Add:**

Deg(a)=Deg(b),Deg(a)>Deg(A),Deg(b)
>Deg(B) → Deg(a+b)> Deg(A+B),

Для основного правила операції **Mult:**

Deg(a)>Deg(A), Deg(b)>Deg(B) →
Deg(a*b)> Deg((a*B+A*b)+ A*B).

Перша з них є наслідком загального правила:

Deg(P+Q) = Max(Deg(P), Deg(Q)),
якщо **LeadMon(P+Q)≠0.**

Отже, якщо позначити **Deg(a)=k, Deg(b)=l, Deg(A)=k1, Deg(B)=l1,**

k = l, k > k1, l > l1 → Max(k, l)
> Max(k1, l1).

Аналогічно, для другої загальним правилом є

Deg(P*Q) = Deg(P)+ Deg(Q).

При тих же позначеннях

k = l, k > k1, l > l1 → k+l >
Max(k1+l, l1+k, k+l).

Висновок: доведення правильності контекстних умов в окремій операції зводиться до доведення умовних нерівностей, які формулюються для цієї аксіоми на основі правил обчислення степенів поліномів – результатів виконання операцій сорту. При цьому треба перевіряти не тільки основні правила інтерпретації, а і похідні.

Розглянемо тепер постановку задачі доведення рівностей. Рівності (19), (20), як і у прикладі 5, мають місце с точністю до співвідношень вкладення сорту: **M++0 = M, 0++P = P**. Отже, вони є тотожностями алгебри многочленів. На відмінність від попереднього прикладу, рівності (19), (20) не є аксіомами. Перша з них – це дистрибутивний закон для базового сорту, друга – тотожність у алгебрі поліномів, яка доводиться з аксіом сорту – дистрибутивності, комутативності та асоціативності. Таким чином, доведення може здійснювати методом індукції у такому формулюванні:

1. База індукції:

Для елементів **a, b, c, A, B, C** сорту Monom виконуються усі аксіоми сорту Polynom.

2. Припущення індукції, яке має бути застосованим, має вид:

Для елементів **A, B, C** сорту Polynom та елементів a, b, c сорту Monom виконуються усі аксіоми сорту Polynom.

2. Крок індукції, який треба виконати, полягає у наступному:

Для елементів **X=a++A, Y=b++B, Z=c++C** сорту Polynom також виконуються усі аксіоми сорту Polynom.

Висновок індукції: На сорті Polynom виконуються усі аксіоми сорту Polynom.

Зауважимо, що перевірка бази індукції потребує визначення операції додавання на сорті Monom. Для нашого прикладу можна вважати, що Сорт Monom визначається як статичне розширення сорту Degree конструктором

Monom M = (Field a)$(Degree D),
1$D = D;

На цьому сорті операція Add є частковою:

a&A + b$A = (a+b)$A.

Висновок: перевірка виконання окремої аксіоми (наприклад, дистрибутивності) на елементах сорту Polynom зводиться до перевірки виконання усієї системи аксіом на аргументах конструктора сорту. Тому доведення має здійснюватися у формі доведення усієї системи аксіом сорту індукцією наведеної форми для тільки для основних правил інтерпретації операцій.

Приклад 7. Верифікація сорту Rat.

Розглянемо, як приклад, задачу перевірки виконання аксіоми дистрибутивності для сорту Rat.

Як ми показали у прикладі 4, на множині Int×Nat має бути визначена конгруентність, яка виділяє пари аргументів (чисельник, знаменник) конструктора сорту Rat. Ця конгруентність має вид

$(a = p_1 // q_1,\quad b = p_2 // q_2,\quad a \sim b \leftrightarrow p_1 * q_2 = p_2 * q_1)$.

Отже, для елементів сорту Rat має місце співвідношення

$(p\ !/ q = r // s\ ) \& (p * s = r * q) \& (\mathrm{GCD}(r,s) = 1)$.(22)

Розглянемо доведення аксіоми дистрибутивності. Нехай

a(b+c) = ab+ac, a=p1//q1,
b=p2//q2, c=p3//q3.

Підставивши представлення **a, b, c** в аксіому та здійснивши обчислення за правилами з специфікацій Rat, отримаємо:

$p_1(p_2q_3+q_2p_3)!/(q_1q_2q_3)=(q_1q_3p_1p_2+q_1q_2p_1p_3)!/(q_1q_2q_1q_3)$.

Для спрощення виразів введемо позначення:

$A = p_1(p_2q_3+q_2p_3), B = q_1q_2q_3, C = q_1q_3p_1p_2 + q_1q_2p_1p_3, D = q_1q_2q_1q_3$.

Тоді верифікація аксіоми дистрибутивності зводиться до перевірки тотожності `AC = BD` над сортом `Int` та доведенні (22) на сорті `Rat`. Доведення тотожності рівності верифікує дану аксіому, доведення співвідношення (22) верифікує інтерпретатор конструктора сорту.

2.4. Методи верифікації алгебраїчних обчислень в розширеннях алгебр

Розглянемо алгебри

$$A =< \mathbf{S}_A, \mathbf{U}_A, T_A, \Sigma_A, Ax_A, I_A >,$$
$$B =< \mathbf{S}_B, \mathbf{U}_B, T_B, \Sigma_B, Ax_B, I_B >. \qquad (23)$$

Узагальнимо метод верифікації системи алгебраїчних програм – інтерпретаторів операцій сорту **v** з його специфікацій за умови, що алгебру A_v визначено як статичне розширення алгебри A_u сорту **u**: $A_u \subset A_v$. Нехай $A \overset{df}{=} A_u$, $B \overset{df}{=} A_v$, $A \subset B$. Далі ми будемо використовувати позначення п. 2.2. Специфікації сорту **v** визначають:

Конструктор сорту: $v = \tau(u, W)$. Обов'язковим аргументом τ є сорт **u**.

Контекстна умова: $\pi(u, W)$.

Умова вкладення: $\rho(u, W)$. **Функція вкладення:** $\rho(u, W) \to \tau(u, W) = u$. Предикат $\rho(u, W)$ є функціональним: для кожного значення u існує єдине значення P таке, що $\rho(u, W)$. Цю функцію ми будемо позначати через $F\rho$: $\rho(u, W) = (F_p(u) = W)$.

Інтерпретатор конструктору сорту: $!\tau(x, P) = y$. Виклик інтерпретатора $!\tau(x, P)$ (позначається окличним знаком як префіксом імені функції) робить істинним предикат $\pi(x, P)$.

Конгруенція конструктору сорту: Конгруенція Θ на множині $S_u \times S_{w_1} \times ... \times S_{w_k}$: $(x, p_1, ..., p_k) \overset{\Theta}{\approx} (x', p_1', ..., p_k')$ визначає умову на аргументах конструктору сорту: якщо $\tau(x, P) = \tau(x', P')$, то $(x, P) \overset{\Theta}{\approx} (x', P')$. Неважко побачити, що конструкція елементу сорту є канонічною формою. Тому ця конгруенція визначає класи еквівалентності елементів $S_u \times S_{w_1} \times ... \times S_{w_k}$, які мають єдину канонічну форму.

Сигнатури операцій сортів u: $\Sigma_u =< \varphi_1, ...\varphi_m >$; **сорту v:** $\Sigma_v =< \varphi_1, ...\varphi_m, \psi_1, ..., \psi_1 >$. Ми вважаємо, що сигнатура операцій розширення B є розширенням сигнатури операцій базової алгебри A.

Аксіоми сортів u: $AX_u =< A_1, ...A_s >$; **сорту v:** $AX_v =< B_1, ..., B_{s'} >$. Ми вважаємо, що система аксіом розширення B є «розширенням» системи аксіом базової алгебри A.

Специфікації операції $\varphi_1, ...\varphi_m$ задані системами рівностей.

Верифікація інтерпретаторів операцій алгебри B полягає у:

1. Доведенні тотальної коректності інтерпретатору функції $!\tau(x, P)$.

2. Доведенні тотожностей над алгеброю A, які є наслідками аксіом AX_B та конгруенції Θ конструктору алгебри B.

Доведення часткової коректності інтерпретатора $!\tau(x, P)$ спирається на його представлення у вигляді суперпозиції функцій побудування канонічної форми аргументу (x, P) як елемента носія алгебри B та застосуванні вкладення до цієї форми $\rho(x, P) \to (!\tau(x, P) = x)$.

Позначимо через τ_{can} функцію, яка будує аргумент (x, P) як елемент носія алгебри B, а через τ_{red} – функцію, яка здійснює вкладення $\rho(x, P) \to (!\tau(x, P) = x)$. Тоді

$$!\tau(x, P) = \tau_{red}(\tau_{can}(x, P)). \qquad (24)$$

Тоді часткову коректність $!\tau(x, P)$ можна довести, довівши імплікації

$$(\tau_{can}(x, P) = (x', P')) \to \to ((x, P) \overset{\Theta}{\sim} (x', P')) \,\&\, \pi(x', P')), \qquad (25)$$

$$\rho(x, P) \,\&\, (\tau_{red}(x, P) = x) \to \to ((x, P) \overset{\Theta}{\sim} (x, F_\rho(x)) \,\&\, \pi(x, F_\rho(x))). \qquad (26)$$

Означення 2.5. Розширення алгебри $A \subset B$ назвемо прямим, якщо функція τ_{can} є тотожною.

Елемент носія прямого розширення B є довільним елементом $S_u \times S_{w_1} \times ... \times S_{w_k}$. Конгруенція Θ є одиничною. Таким чином, при прямих розширенням доводити треба лише коректність вкладення $A \to B$.

Розширення $A_i \subset A_{i+1}$ прикладів 10, 11 є прямими. Більшість розширень, які зустрічаються у цій роботі, є прямими. Розширення $Int \to Rat$ (приклад 12) є прикладом розширення, яке не є прямим.

Розглянемо тепер доведення тотожностей. Тотожності над A, які є наслідками аксіом AX_B та

конгруенції Θ, отримуються таким чином: Якщо рівність $B_i \overset{df}{=} (L(x_1,...,x_n,P) = R(x_1,...,x_n,P)) \in AX_B$, причому змінні $x_1,...,x_x$ визначені над B, підстановки $\tau(u_i,P_i)$ замість x_i визначають рівності над $S_u \times S_{w_1} \times ... \times S_{w_k}$. Формальне виконання інтерпретаторів операцій алгебри B у лівій та правій частинах рівності

$$L(\tau(u_1,P_1),...,\tau(u_n,P_n),P) = \\ = R(\tau(u_1,P_1),...,\tau(u_n,P_n),P) \quad (27)$$

приводить до рівності

$$\tau(f,G_1,,,G_k) = \tau(f',G_1',...,G_k'), \quad (28)$$

де

$$f = f(u_1,..,u_n,P_1,..,P_n,P),$$
$$f' = f'(u_1,..,u_n,P_1,..,P_n,P),$$
$$G_i = G_i(u_1,..,u_n,P_1,..,P_n,P),$$
$$G_i' = G_i'(u_1,..,u_n,P_1,..,P_n,P),\ i = 1,..,n.$$

Отже, має місце співвідношення еквівалентності

$$(f,G_1,,,G_k) \overset{\Theta}{=} (f',G_1',...,G_k'). \quad (29)$$

Це співвідношення залежить від змінних $u_1,..,u_n,P_1,..,P_n,P$. Будемо вважати, що конгруентність Θ задано системою тотожних рівностей $V_j = W_j,\ j = 1,..,e$. Тоді верифікація інтерпретаторів операцій алгебри B полягає у доведенні тотожностей

$$V_j(u_1,..,u_n,P_1,..,P_n,P) = \\ = W_j(u_1,..,u_n,P_1,..,P_n,P),\ \ j = 1,..,e \quad (30)$$

над базовими сортами $u,w_1,...,w_k$. Алгоритм доведення (30) має спиратися на аксіоми базових сортів. Отже, цей алгоритм існує, якщо проблема тотожностей над базовими алгебрами є алгоритмічно розв'язуваною.

Якщо розширення $A \subset B$ є прямим, сукупність тотожностей (30) є сукупністю "по координатних" тотожностей. Тому конгруентність (29) визначає тотожності

$$f = f',\ \ G_i = G_i',\ \ i = 1,..,k. \quad (31)$$

Узагальнимо тепер метод верифікації системи інтерпретаторів операцій сорту **v** з його специфікацій за умови, що алгебру A_v визначено як динамічне розширення алгебри A.

Нехай

$$A_0 = A,\ \ A_{i+1} = \tau(A_0,A_i),\ \ B = \bigcup_{i=0}^{\infty} A_i \quad (32)$$

– специфікація конструкції лінійного динамічного розширення $A \subset B$.

$$x \in A_0,\ X \in A_i,\ \ \rho(x,X) \to \tau_{red}(x,X) = X \quad (33)$$

– вкладення, визначені для елементів A_i цього розширення.

Як і для статичних розширень, будемо зараз вважати, що система аксіом B є розширенням системи аксіом базової алгебри A. Це означає, що цій системі мають задовольняти усі алгебри A_i. Тоді:

– верифікація функції $!\tau$ здійснюється індукцією по індексу i та полягає у доведенні імплікацій (25), (26);

– верифікація інтерпретаторів операцій на B за допомогою аксіом здійснюється індукцією по індексу i та полягає у доведенні тотожностей над алгебрами A_{i+1} за індуктивним припущенням, що усі аксіоми доведені як тотожності алгебри A_i. Якщо алгоритм доведення тотожності на $A_0 = A$ спирається тільки на аксіоми A, індукція дозволяє розширити цей алгоритм на B.

Насправді особливістю використання розширень як методів побудування алгебр є те, що на вже побудованій алгебрі B визначаються нові операції. Сорт Int розширюється до Rat з метою визначення операції ділення (приклад 12).

Сорт Polynom розширюється як векторний простір. Таким чином, ми розглядаємо сорт Monom як одновимірний векторний простір, а крок розширення полягає у додаванні нової координати. У цьому прикладі A_i є i-вимірним векторним простором. Операції множення на скаляр, додавання та віднімання та усі аксіоми векторного простору підпадають під розглянутий нами метод. Але операція множення многочленів та усі аксіоми, які містять символ операції множення, мають бути розглянуті окремо.

Нехай

$$P,Q,R = P*Q \in \text{Polynom},\ \deg P = i, \deg Q = j,$$
$$A_k = \{p : \deg\ p \le k\}.$$

Тоді $\deg\ R = i + j$. Отже,

$$P \in A_i,\ \ Q \in A_j,\ \ P*Q \in A_{i+j}.$$

Верифікація операції множення методом математичної індукції, розглянута вище (19), (20) зводить доведення аксіом на A_{i+j} до доведення тотожностей на A_{i+j-1}. Таким чином, базис індукції має містити доведення аксіом на базових алгебрах, тобто на мономах. Наприклад, у (20) треба вважати, що

$$a,\ A,\ b,\ B,\ c,\ C \in \text{Monom}.$$

Узагальнимо ці міркування для довільних алгебр. Метод індукції ми змогли застосувати, оскільки степінь добутку є монотонно зростаючою функцією степенів множників. Очевидно, це можна робити і у загальному випадку. Ми сформулюємо це для бінарних операцій.

Нехай $A_0 = A$, $A_{i+1} = \tau(A_0, A_i)$, $B = \bigcup_{i=0}^{\infty} A_i$ та $\varphi(a,b)$ – бінарна операція, визначена на B. Тоді, якщо існує така монотонно зростаюча по кожному з аргументів функція $s(i,j)$, що для довільних $a \in A_i$, $b \in A_j$ $\varphi(a,b) \in A_{s(i,j)}$ метод індукції зводить доведення аксіом алгебри B, які містять операцію φ, до доведення цих аксіом для змінних, визначених на алгебрі A.

Список літератури

1. *Lvov M. Applied Computer Support of Mathematical Training / M. Lvov, A. Kuprienko, V. Volkov // Proc. of Internal Work Shop in Computer Algebra Applications. – K., 1993. – P. 25-26.*
2. *Lvov M. AIST: Applied Computer Algebra System / M. Lvov // Proc. of ICCTE'93. – K. – P. 25-26.*
3. *Львов М.С. Терм VII – шкільна система комп'ютерної алгебри / М.С. Львов // Комп'ютер у школі та сім'ї. – 2004. – №7. – С. 27-30.*
4. *Algebraic programming system APS (user manual) / A. Letichevsky, J. Kapitonova, V. Volkov, A. Chugajenko, V. Chomenko. – K.: Glushkov Institute of Cybernetics, National Acad. of Sciences of Ukraine, 1998.*
5. *Капитонова Ю.В. Дедуктивные средства системы алгебраического программирования / Ю.В. Капитонова, А.А. Летичевский, В.А. Волков // Кибернетика и системный анализ. – 2000. – № 1. – С. 17-35.*
6. *Tools for solving problems in the scope of algebraic programming / J. Kapitonova, A. Letichevsky, M. Lvov, V. Volkov // Lectures Notes in Computer Sciences. – 1995. – № 958. – P. 31-46.*
7. *Львов М.С. Основные принципы построения педагогических программных средств поддержки практических занятий / М.С. Львов // Управляющие системы и машины. – 2006. – № 6. – С. 70-75.*
8. *Песчаненко В.С. Розширення стандартних модулів системи алгебраїчного програмування APS для використання у системах навчального призначення / В.С. Песчаненко // Науковий часопис НПУ імені М.П. Драгоманова. Серія №2. Комп'ютерно-орієнтовані системи навчання: Зб. наук. пр. – К.:НПУ ім .М.П. Драгоманова, 2005. – №3 (10). – С. 206-215.*
9. *Песчаненко В.С. Об одном подходе к проектированию алгебраических типов данных / В.С. Песчаненко // Проблеми програмування. – 2006. – №2-3. – С. 626-634.*
10. *Песчаненко В.С. Использование системы алгебраического программирования APS для построения систем поддержки изучения алгебры в школе / В.С. Песчаненко // Управляющие системы и машины. – 2006. – №4. – С. 86-94.*
11. *Goguen J. Ordered-Sorted Algebra I: Partial and Overloaded Operations. Errors and Inheritance / J. Goguen, J. Meseguer. – SRI International, Computer Science Lab., 1987.*
12. *Гоген Дж. А. Модели и равенство в логическом программировании / Дж. А. Гоген, Ж. Мезегер // Сб. Математическая логика в программировании. – М.: Мир, 1991. – С. 274-310.*
13. *Ван дер Варден Б.Л. Алгебра / Б.Л. Ван дер Варден; изд. 2-ое. – М. ГРФМЛ, 1979. – 624 с.*

Надійшла до редколегії 3.08.2009

Рецензент: д-р пед. наук, канд. фіз.-мат. наук, проф. О.В. Співаковський, Херсонський державний університет, Україна, Херсон.

ВЕРИФИКАЦИЯ ИНТЕРПРЕТАТОРОВ АЛГЕБРАИЧЕСКИХ ОПЕРАЦИЙ В РАСШИРЕНИЯХ МНОГОСОРТНЫХ АЛГЕБР

М.С. Львов

Разработка алгоритмов выполнения алгебраических вычислений – одна из основных задач, возникающих при реализации математических систем, основанных на символьных преобразованиях. Математическая модель этой задачи - многосортные алгебраические системы. В данной работе рассмотрен подход к верификации интерпретаторов многосортных алгебраических операций по их спецификациям, основанный на конструктивном уточнении понятия расширения многосортной алгебраической системы и доказательстве аксиом алгебраической системы. Этот подход проиллюстрирован примерами верификации интерпретаторов операций поля рациональных чисел, многочленов одной переменной и алгебры высказываний. Практика использования этого подхода при разработке математических систем учебного назначения показала его эффективность и даже универсальность.

Ключевые слова: *математическая система, символьные преобразования, многосортные алгебраические системы, расширения алгебраических систем, верификация, интерпретаторы алгебраических операций.*

VERIFICATION OF INTERPRETERS OF ALGEBRAIC OPERATIONS IN EXPANSIONS OF MULTI-SORTED ALGEBRAS

M.S. Lvov

Development of algorithms of algebraic computations implementation is one of basic tasks, which arise up during realization of mathematical systems, based on symbolic transformations. The mathematical model of this task is multi-sorted algebraic systems. In this work, the approach to verification of interpreters of multi-sorted algebraic operations by their specifications, based on structural clarification of expansion concept of multi-sorted algebraic system and proving the axioms of algebraic system is considered. This approach is illustrated by the examples of verification of interpreters of operations of rational numbers field, polynomial of one variable and propositional algebra. The practice of the use of this approach at the mathematical systems development of the educational destination shows its efficiency and even universality.

Keywords: *mathematical systems, symbolic transformations, multi-sorted algebraic systems, expansions of algebraic systems, verification, interpreters of algebraic operations.*