В. Песчаненко
Херсонський державний університет

Розширення стандартних модулів системи алгебраїчного програмування APS для використання у системах навчального призначення

У даній статті розглядаються проблеми використання та розширення стандартних модулів APS, з допомогою яких можна ефективно застосовувати технології символьних перетворень та методи комп'ютерної алгебри для побудови програмних засобів навчального призначення з математики, фізики, інших природничих дисциплін. Удосконалена таким чином система АПС була використана при розробці шкільної системи комп'ютерної алгебри ТерМ [1].

Система алгебраїчного програмування APS розроблена в Інституті кібернетики імені В.М.Глушкова Національної академії наук України. Алгебраїчне програмування в APS підтримується як програмування з алгебраїчними об'єктами, що використовуються для представлення даних та предметних галузей. Алгебраїчні вирази різноманітних алгебр даних використовуються як головний тип об'єктів алгебраїчного програмування. Ці вирази представляють об'єкти предметної галузі або твердження про них. Система APS була розроблена як інструментальна утиліта для створення прикладних систем алгебраїчного програмування.

Головний інструмент для перетворення виразів – це система, в якій розглядаються правила переписувань, перетворення даних, або, інакше кажучи, рівності системи переписувань визначені як тотожності, що виражають такі властивості предметної галузі, які зберігаються при перетвореннях.

Метод правил переписувань – це область комп'ютерної алгебри, що активно розроблюється [2,3,4]. Різні способи екваціонального програмування, що базуються на правилах переписувань та алгебраїчних специфікаціях, зараз підтримуються в різних системах програмування. Значно складніше створити та об'єднати парадигми алгебраїчного, функціонального та логічного програмування [5]. APS відрізняється від систем, описаних в літературі, багатьма рисами.

Головна особливість APS в тім, що системи правил переписувань можуть бути інтерпретовані APS з різними обчислювальними стратегіями. Цей підхід дозволяє використовувати не тільки канонічні (вироджені та не вироджені), але й будь-які інші системи рівностей, причому алгебраїчні програми можуть бути створені комбінуванням правил переписувань з різними стратегіями їх застосування. Інша особливість APS – це можливість комбінувати процедурні та алгебраїчні методи програмування. Мова програмування системи – APLAN. Її використання дозволяє описувати алгебраїчне середовище (компоненти, операції, рівності алгебри даних), писати процедури та викликати їх з програм (систем правил

переписувань). Процедурні інструменти роблять можливим для користувача маніпулювати зі стандартною бібліотекою стратегій та писати свої власні ефективні стратегії застосування правил переписувань.

Різні властивості алгебри даних можуть бути використані з обчислювальними стратегіями, які визначають деякі конгруенції алгебри даних. З цією метою використовується механізм канонічних форм алгебраїчних виразів. Цей механізм використовується завжди, коли застосовується деяке правило переписувань. Вибір канонічних форм обумовлений об'єктом предметної галузі та контролюється користувачем.

Для ефективного застосування обчислювальних стратегій правил переписувань, інтерпретаторів та процедурних інструментів система APS представляє можливість використання різних рівнів програмування. Низький рівень системи – це описи мовою C та включення структур даних для представлення алгебраїчних об'єктів, інструментів для зв'язку з високим рівнем, менеджером віртуальної пам'яті, бібліотек функцій, які підтримують алгебраїчне програмування мовою C. Для детальної специфікації APS див. [6,7].

Взагалі APS було розроблено як експериментальну систему алгебраїчного програмування, яку пропонувалося використовувати з метою побудови прототипів, експериментального дослідження алгоритмів та удосконалення їх реалізації. Однак, багаторічний досвід використанні APS та деякі особливості проекту шкільної комп'ютерної алгебри дозволили висунути гіпотезу про те, що APS можна використовувати як виробничу систему алгебраїчного програмування, якщо адаптувати її до конкретних виробничих вимог.

Перша версія Терм 7 підтримувала вивчення алгебри у 7 класі середньої школи. Аналіз задач курсу алгебри за 7 клас та технічних вимог до Терм-7 виявив, що розв'язування більшості математичних задач не потребує великих машинних ресурсів, тому при розробленні програмних модулів сервера алгебраїчних обчислень ТерМ 7 алгебраїчні модулі APS були реалізовані на верхньому рівні. Але подальшій розвиток системи (Терм 7-9) показав, що на розв'язування деяких алгоритмічних задач система використовувала більший час та об'єм пам'яті, ніж у попередній версії. Детальний аналіз цієї проблеми показав, що кількісне розширення APLAN-модулів призводить до зниження ефективності їх застосування. Для розв'язання цієї проблеми було вирішено, реалізувати ієрархію алгебраїчних типів даних та числових алгебр ТерМ 7 на нижньому рівні. Результат, який було отримано після цього ми вирішили назвати – розширення стандартних модулів APS , яке включає в себе:

1) Об'єкт COM, надання стандартного інтерфейсу.
2) Реалізацію функцій зберігання та читання модулів APS з бінарного файлу;
3) введення у мову APLAN поняття відмітки асоціативної операції;

4) прототипування та реалізацію на нижньому рівні числових типів даних, які фактично необхідні для реалізації алгебраїчних задач шкільного типу;
5) прототипування та реалізацію на нижньому рівні множинних типів даних, які фактично необхідні для реалізації алгебраїчних задач шкільного типу;
6) прототипування та реалізацію на нижньому рівні алгебраїчних типів даних, які фактично необхідні для реалізації алгебраїчних задач шкільного типу.

Початкова версія інтерпретатора APS розповсюджувалась під Windows як DOS-додаток, що робило неможливим його ефективне використання для серверу обчислень TepM. Тому з метою використати інтерпретатор APS для реалізації символьних перетворень у TepM, була написана спеціалізована оболонка для інтерпретатора, яка реалізовує початкове завантаження ядра та виклик окремого APLAN-модуля. Далі в якості зовнішньої оболонки інтерпретатора APS було реалізовано COM-об'єкт, який використовується для обміну даними з іншими додатками у системі шкільної комп'ютерної алгебри Терм та може бути використаний іншими. Така реалізація оболонок інтерпретатора APS дозволяє ефективно переходити від технології COM до будь-якої іншої технології, яка є сумісною з мовою C.

Висхідна версія APS під Windows не підтримувала технології зберігання об'єктних алгебраїчних модулів на жорсткому диску, але технологія роботи з бінарними алгебраїчними модулями була реалізована в ранній версії системи під Unix. Об'єктні модулі формувалися в оперативній пам'яті як результат компіляції висхідного модуля.

Таким чином, по-перше, файлова система дистрибутиву програмного засобу містила вихідні тексти алгебраїчних модулів (APLAN-модулів), а, по-друге, будь-яке завантаження програмного засобу потребувало компіляції APLAN-модулів.

Цей недолік було усунуто з реалізацією функцій зберігання та читання модулів APS з бінарного файлу, що значно скоротило час завантаження ПЗ та дозволило не розповсюджувати разом з дистрибутивами ПЗ, вихідні текст APLAN-модулі. Файлова система дистрибутиву при цьому якісно спростилася.

Відомо, що в математиці існують як асоціативні, так і не асоціативні операції, та кожне середовище для програмування, математичне середовище підтримують обчислення виразів які містять в собі такі операції. Але особливістю APS є те, що правила інтерпретації таких відміток в APS дещо відрізняються від стандартних математичних правил інтерпретації. Наприклад, при розв'язуванні алгебраїчних задач дуже часто інтерпретуються наступні вирази:

$$x - y - z \text{ та } x - (y - z).$$

Оскільки в APS внутрішня структура дерева бездужкового виразу є правоорієнтованою (це пов'язано з алгоритмом переписування), а дужки використовуються лише для визначення пріоритетів підвиразів і в дереві не представлені, то для обох цих

виразів інтерпретатор APS побудує одне й те саме дерево, яке має наступний вигляд (рис. 1)

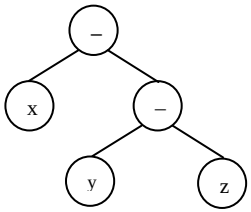

Рис. 1

Як бачимо, обидва вирази є синтаксично рівними, що для другого виразу математично некоректно. Таким чином, ця проблема не може бути розв'язана засобами APS.

При написанні ПЗ Терм нам довелося розв'язати зазначену проблему за допомогою заміни x – y на x + (-1) *y під час перетворення виразів з формату математичного редактора у формат APS представлення алгебраїчних даних. Але такий підхід не є оптимальним, оскільки розміри систем переписувань значно розширювалися за рахунок перевірки додаткових умов. У даному розширенні APS цю проблему було вирішено введенням поняття асоціативної відмітки. В стандарті APS за замовчуванням опущені дужки для бінарних операцій відновлюються праворуч. Нововведення допускає розглядати бінарні операції опущені дужки для яких відновлюються ліворуч. Синтаксис їх введення наступний:

<mark description> ::= MARK <послідовність mark description elements separated by "," >;
<mark description element> ::=<mark symbol>(<arity>)
| <mark symbol>(<arity>, <priority>, "<infix notation>")
<mark symbol> ::= <identifier>

В новій нотації до цього має бути додані наступні рядки:
<mark symbol>(<arity>, <priority>, "<infix notation>"»,<assio>)
<assio> ::= 0 або 1 (за замовчуванням 1).

Якщо визначена відмітка є неасоціативною, то відповідний підтерм будується як лівостороннє дерево, і тому розглянуте вище протиріччя знімається.

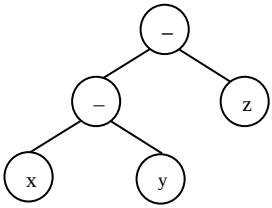

Рис. 2

Система алгебраїчного програмування APS підтримувала обчислення з числами необмеженої розрядності лише в ранній версії, у наступний версіях ця властивість біла вилучена. В початковій версії були реалізовані лише стандартні для С числові типи даних. Терм 7-9 використовує наступні числові алгебри: цілі, дійсні, раціональні, комплексні та радикальні числа, причому як технічне завдання, так і фактичний стандарт на системи комп'ютерної алгебри вимагають реалізації обчислень з необмеженою розрядністю.

Цілі, дійсні та раціональні числові алгебри реалізовані з використанням бібліотеки GNU GMP, що є вільною для розповсюдження та використовується для обчислень з необмеженою кількістю знаків.

Цілі, дійсні та раціональні числа зберігаються у внутрішньому форматі GNU GMP, тобто для цілих чисел – mpz_t, для дійсних – mpf_t та для раціональних - mpq_t. Комплексні та радикальні числа у внутрішньому представленні мають вигляд бінарного дерева.

Слід відмітити, що цілі числа необмеженої розрядності підключаються в розширеній версії АПС лише за необхідностю, тобто якщо для обчислень стандартного типу long(C++) недостатньо. Синтаксис запису дійсних чисел наступний: <int>.<int>).

За технічним завданням ПЗ Терм має підтримувати такі форми запису раціональних чисел: звичайні дроби, мішані дроби та періодичні десяткові дроби

Періодичні числа можна використовувати за допомогою відмітки операції PeriodNum("<ціла частина>","<число до періоду>","<період>").

Для їх перетворення до дійсного або раціонального числа використовується відмітка операції RealNum("<ціле>","<ціле>") та функції rat2period та period2rat. Цей метод може бути використаний при написанні АПЛАН-модуля.

Синтаксис запису раціональних чисел наступний: RAT(<int>,<int>), радикальних: RAD(<int>,<int>). Але слід зауважити, що при діленні цілих чисел за замовчуванням інтерпретація проходить в алгебрі раціональних чисел, а при добуванні квадратного кореня – в алгебрі радикальних чисел. Для того, щоб обчислення проходили в полі дійсних чисел, необхідно викликати метод use_float(1) (за замовчуванням use_float(0)) або в файлі конфігурації поставити відповідний прапорець.

Комплексні числа є шаблонними типами даних, функціональність яких не залежить від аргументів, але потребує від них повної функціональності числових полів. Синтаксис комплексних чмсел наступний: COMPLEX(<re>,<im>). Комплексні та радикальні числа у внутрішній структурі представлені бінарними деревами.

В результаті аналізу математичних задач 7 класу було виявлено наступні алгебраїчні типи даних (рис.3.).

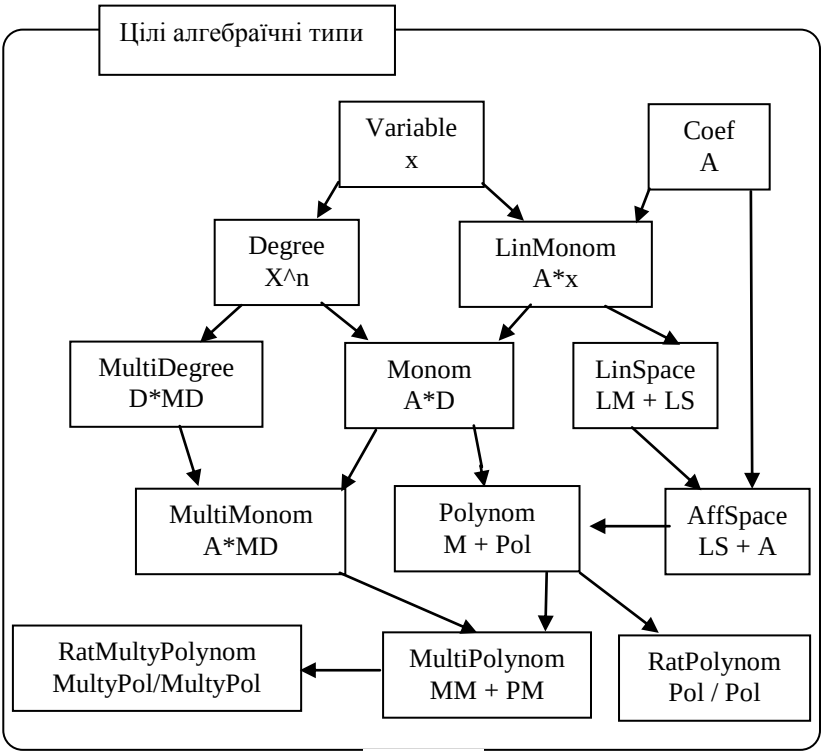

Рис. 3

Але в початковій версії інтерпретатора APS такі алгебраїчні дані, які необхідні для розв'язування алгебраїчних задач, були відсутні. Тому виникла необхідність у Терм 7 реалізувати їх на верхньому рівні.

Розглянемо більш детально кожен з цілих алгебраїчних типів даних. На малюнку спочатку зображено ім'я алгебраїчного типу, під ім'ям зображено тип елементів, над якими виконуються арифметичні дії.

Degree – цілий алгебраїчний тип даних (ЦАТД), у якому реалізовано операції множення, ділення, НСД, НСК між елементами типу та піднесення до степен над елементами типу x^^n та x^^m. **Degree** є напівгрупою з одним твірним елементом.

LinMonom – ЦАТД, у якому реалізовано операції додавання між елементами ЦАТД, множення та ділення на число над елементами виду A*X, де A – коофіціент, X – атом. LinMonom є лінійним мономом.

MultyDegree – ЦАТД, який реалізує операції множення, ділення, НСД, НСК між елементами ЦАТД, піднесення до степеня над елементами виду: Degree*MultyDegree. MultyDegree – є розширенням Degree щодо операції множення та напівгрупою степенів від багатьох змінних.

Monom – ЦАТД, який реалізує операції множення, додавання, ділення, НСД, НСК між елементами ЦАТД та піднесення до степеня над елементами виду: Koof*Degree.Monom – э розширенням Degree щодо операції множення та ділення на число та розширенням LinMonom щодо операції множення. Monom є напівгрупою мономів з одним твірним елементом.

LineSpace – ЦАТД, який реалізує операції додавання між елементами ЦАТД, множення, ділення на число над елементами виду: LinMonom+LineSpace. LineSpace - є розширенням LinMonom операції додавання та лінійним векторним простором.

MultyMonom – ЦАТД, який реалізує операції множення, ділення, НСД, НСК між елементами ЦАТД та піднесення до степеня над елементами виду: A*MultyDegree. MultyMonom - є розширенням MultyDegree щодо операції множення на число та напівгрупою мономів багатьох змінних.

Polynom – ЦАТД, який реалізує операції множення, ділення між елементами ЦАТД та піднесення до степеня над елементами типу: Monom+Polynom. Polynom – є розширенням Monom щодо операції додавання та кільцем многочленів однієї змінної над полем.

Affspace – ЦАТД, який реалізує операції множення на число, ділення на число, додавання між елементами ЦАТД над елементами виду: Affspace+Numeric. AffSpace – є розширенням LinSpace відносно операції додавання числа та лінійним векторним простором.

MultyPolynom – ЦАТД, який реалізує операції множення, ділення, НСД, НСК та піднесення до ступеня над елементами типу: MultyMonom+MultyPolynom. MultyPolynom – є розширенням MultyMonom відносно операції додавання та кільцем многочленів багатьох змінних над полем.

RatPolynom – РАТД, який реалізує операції множення, ділення між елементами ЦАТД та піднесення до степеня над елементами типу: Polynom/ Polynom та є розширенням Polynom щодо операцій ділення та полем раціональних функцій однієї змінної над полем.

RatMultyPolynom – РАТД, який реалізує операції множення, ділення між елементами ЦАТД та піднесення до степеня над елементами типу: MultyPolynom/ MultyPolynom та є розширенням MultyPolinom щодо операцій ділення та полем раціональних функцій багатьох змінних над полем.

Реалізація ієрархії алгебраїчних типів даних за даною схемою дозволили повністю реалізовувати стандартні алгоритми символьних обчислень, але вона має кілька недоліків: по-перше рекурсивний спуск для обчислення значень виразів проходить біля 7-8 правил переписувань, по-друге в деяких випадках неможливо було реалізувати повне використання шаблонних алгоритмів, тобто технології параметричного програмування.

Перший недолік полягає у тому, що швидкість виконання правил переписувань є дуже низькою, оскільки інтерпретація полягає у рекурсивному спуску на кілька рівнів, а лише потім – у

здійсненні необхідного переписування, це вимагає і часу і досить великого об'єму пам'яті. Наслідком другого недоліку була необхідність фактично копіювання фрагментів вже написаного коду та вставляння його в інше місце програми.

Поява цих проблем спонукала до реалізації розширення APS у Терм 7-9, яке позбавлене цих недоліків. Розглянемо детальніше ієрархії алгебраїчних типів даних APS для Терм 7-9 (аналогічно ПМК Терм 7).(рис. 4).

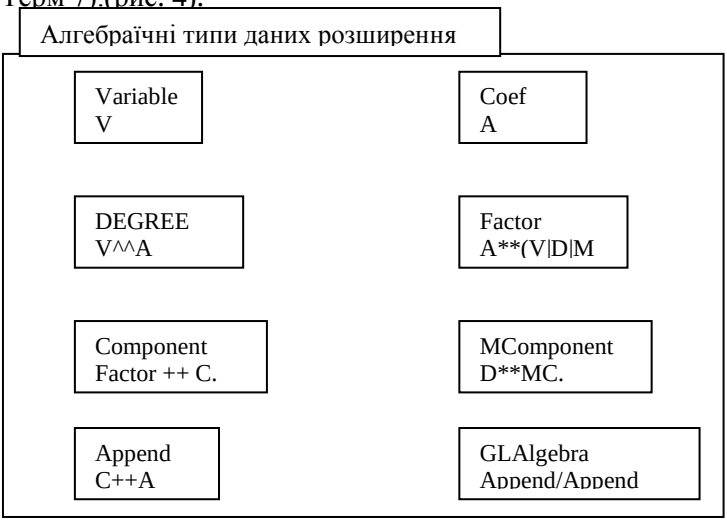

Рис. 4

Кожен з цих шаблонних типів не є розширенням один одного, а реалізовує нову функціональність. Але якщо комбінувати використання цих типів між собою, то можна побудувати точну відповідність типам даним спроектованих в ПМК Терм 7. Зауважмо, що імена, типів зображених на рис.4, відповідають іменам відміток вершин дерев що представляють даний тип.

Для зручності побудуємо таблицю:

АТД ПМК Терм	АТД розширення APS
Degree	Degree
LinMonom	Factor(A*V)
MultyDegree	MComponent(Degree*MC.)
Monom	Factor(A*Degree)
LinSpace	Component(Factor(A*V)+C.)
MultyMonom	Factor(A*MComponent(Degree*MC))
Polinom	Component(Factor(A*V)+C)+Append

AffSpace	Component(Factor(A*V)+C)+Append
MultyPolinom	Component(Factor(A*MComponent(Degree*MC))+C)+Append
RatPolinom	GLAlgebra(Component(Factor(A*V)+C)+Append/Component(Factor(A*V)+C)+Append)
RatMultyPolinom	GLAlgebra(Component(Factor(A*MComponent(Degree*MC))+C)+Append/Component(Factor(A*MComponent(Degree*MC))+C)+Append)

З поданої таблиці можна зробити висновок, що побудовані шаблонні типи даних повністю реалізують алгебраїчні типи даних ПМК Терм за допомогою інстанціювання побудованих шаблонних класів, що значно спрощує їх використання для вирішення алгебраїчних задач у інших системах.

Таким чином, побудоване розширення стандартних модулів APS повністю відповідає вимогам систем учбового призначення з природничих наук та може бути використано для комерційних проектів в даній області знань.

Досвід та традиції розробок з використанням APS акумулюються та використовується в Інституті кібернетики імені В.М.Глушкова Академії наук України, Херсонському державному університеті, інших наукових установах. Цей досвід включає в себе детальну розробку, реалізацію та використання мови і системи для рекурсивних перетворень структур даних, що використовуються як допоміжні в комп'ютерному обладнанні та програмному забезпечені [8].

ЛІТЕРАТУРА

1.М.С. Львов. Терм VII – шкільна система комп'ютерної алгебри. Комп'ютер у школі та сім'ї. – 2004. – №7.- С. 27-30

2.Huet G., Oppen D.C. Equations and rewrite rules: a survey. In: Formal language theory - N.-Y., Acad. Press, 1987, p.349-406

3.Rewriting techniques and applications. Journal of symbolic and algebraic computations,vol.3,W1-2 (special issue), Acad. Press, 1987.

4.Rewriting techniques and applications. LNCS, vol.256, Springer 1987.

5.D.de Groot, G.Lindstrom, eds. Logic programming: functions, relations and equations, Englewood, Clif., 1986.

6.J. V. Kapitonova, A.A. Letichevsky, M. S. L'vov, and V. A. Volkov. Tools for solving problems in the scope of algebraic programming. LNCS V.958, pages 31-46, Springer-Verlag,1995.

7.A. Letichevsky, J. Kapitonova, V. Volkov, A. Chugajenko, V. Chomenko Algebraic programming system APS (user manual) Glushkov Institute of Cybernetics, National Acad. of Sciences of Ukraine, Kiev, Ukraine, 1998.

8.Ю.В.Капитонова, А.А.Летичевский, Математическая теория разработки вычислительных систем, Москва, Наука 1988.