ПРОГРАМУВАННЯ ТА ШТУЧНИЙ ІНТЕЛЕКТ

DOI 10.31891/2307-5732-2020-287-4-264-267
УДК 004.43

Т. М. КОРОТУН
Заклад вищої освіти "Міжнародний науково-технічний університет імені академіка Юрія Бугая"

ВЛАСТИВОСТІ ПАРАДИГМ ПРОГРАМУВАННЯ

Р. Флойд заклав основи теорій аналізу і верифікації програм, створив ряд ефективних методів обробки. Розглядаючи приклад структурного програмування в якості домінуючої методології програмування було визначено, що, по-перше, націленість парадигми на низхідне проектування, покрокове поліпшення і зведення завдання до простіших підзадач; по-друге, перехід від конкретних об'єктів і функцій машинного рівня до абстрактніших об'єктів і функцій, дозволяють продумувати модулі, що виділяються при низхідному проектуванні. Вибір між так протиставленими характеристиками в тексті програми виражається синтаксично, нерідко за допомогою специфікаторів мови програмування.

Ключові слова: парадигма програмування, паралельне програмування.

TETYANA MIKHAILOVNA KOROTUN
Academician Yuriy Bugay International Scientific and Technical University

PROPERTIES PROGRAMMING PARADIGM

R. Floyd laid the foundations of the theories of analysis and verification of programs, created a number of effective processing methods. Considering an example of structural programming as a dominant programming methodology, it has been determined that, firstly, the targeting paradigms for downlinking design, step-by-step improvement and construction of a task to simpler subspaders; Secondly, the transition from specific objects and functions of the machine level to abstract objects and functions, allow to think the modules allocated under downward design. The choice between so opposed characteristics in the text of the program is expressed syntactically, often with the programming language specifiers.

A mock-up of a solution to a new problem is workable when the author presents a small set of relevant data. An experimental debugging ground for research solutions requires adaptability to process almost any data that can be specified as input. The practical version may be limited to the processing of data that actually occur in the field of its application. Effective implementation requires specially selected data that show its advantage over less skillfully executed solutions.

Key words: programming paradigm, parallel programming.

Вступ

Альтернативні підходи до обробки інформації, що склалися при створенні і застосуванні мов і систем програмування, прийнято називати парадигмами програмування (ПП). Незалежно від області додатка, ПП можуть відрізнятися рівнем абстрагування від можливостей апаратури, мірою вивченості постановок завдань, мірою організованості і рангом працездатності програм рішення завдань. При визначенні ПП зазвичай поляризуються наступні характеристики МП :

- програмовані рішення представляються в імперативно-процедурній або в декларативній формі;
- оброблювані елементи даних позиціонуються як блоки пам'яті, що адресуються, або незалежно розміщувані значення;
- програма може бути захищена від змін в процесі її виконання або допускати програмовані модифікації по ходу отримання результатів обчислення;
- програма може бути цілісною або збиратися з типових компонент і шаблонів;
- представлені в програмі функції можуть бути частковими, такими, що типізуються, оброблювальними значення заданого домена або універсальними, такими, що дають розумну реакцію на будь-який елемент даних;
- управління обчисленнями виконується послідовно або паралельно;
- порядок дій може бути визначеним або недетермінованим;
- обчислення можуть бути енергійними або ледачими;
- зони видимості імен можуть бути глобальними або локалізованими за ієрархією конструкцій з можливістю відновлення контексту;
- розподіл і повторне використання пам'яті може бути дією в програмі або виконуватися автоматично системою програмування (СП);
- ініціація пам'яті спочатку розміщуваними значеннями може вимагати програмованих дій або виконуватися в СП за умовчанням;
- домени значень можуть бути незалежними або допускати перетину;
- результат виконання програми може бути розосереджений по ряду змінних або сконцентрований в спеціальному регістрі;
- контроль правильності може виконуватися статично - при аналізі тексту програми або динамічно - при виконанні коду програми.

Саме парадигмам програмування Роберт Флойд присвятив свою Тьюринговскую лекцію, в якій звернув увагу на значущість цього поняття в контексті проблеми навчання програмістів. Авторитетний

учений, що заклав основи теорій аналізу і верифікації програм, автор ряду ефективних методів обробки цих вважав необхідним підкреслити вплив ПП на успіх проектів програмістів, на особливості вивчення різних ПП і на те, як вони мають бути підтримані в мовах програмування. Розглядаючи приклад структурного програмування в якості домінуючої методології програмування, Р. Флойд відмітив, по-перше, націленість цієї парадигми на низхідне проектування, покрокове поліпшення і зведення завдання до простіших підзадач; по-друге, перехід від конкретних об'єктів і функцій машинного рівня до абстрактніших об'єктів і функцій, що дозволяють продумувати модулі, що виділяються при низхідному проектуванні.

Р. Флойд виразив упевненість, що можна налагодити ясне навчання ряду таких семантичних методів для усіх рівнів програмного проекту, щоб у навчених студентів вистачало голови на рішення повного спектру проблем від упрощѐнной учбової постановки завдання до класу практичних завдань, що безперервно розширюється.

Слід зазначити, що за тридцять років, що пройшли з часу Тьюринговой лекції Р. Флойда, число різних мов і систем програмування зросло з декількох сотень до десятків тисяч. При цьому число парадигм не таке велике. У різних джерелах називають від двадцяти до сорока парадигм, нерідко включаючи в їх перелік окрему техніку і методи.

Вибір між так протиставленими характеристиками в тексті програми виражається синтаксично, нерідко за допомогою специфікаторів мови програмування (МП). Визначення операційної семантики МП зазвичай використовує поняття абстрактної машини, яка у разі багатопроцесорних конфігурацій природно стає конструкцією з абстрактних процесорів, - абстрактним комплексом, що дозволяє виділяти схемний рівень і спрощувати включення в схему розробки програм механізмів верифікації (подібність моделі або відповідність аксіомам), а на їх основі підключати перевірку програм на правдоподібність, логічний висновок властивостей, виконання індуктивних і дедуктивних побудов [4].

Крім того, межі між областями практичного прояву різних парадигм програмування можна виразити типовими схемами постановок завдань на програмування.

Стандартне імперативно-процедурне програмування:

«Визначений алгоритм рішення актуальної задачі. Необхідно отримати програму реалізації алгоритму з практичними просторово- тимчасовими характеристиками на доступному устаткуванні«.

Функціональне програмування: «Відома предметна область. Слід вибрати символьне представлення даних для цієї області і відлагодити систему універсальних функцій, придатних для створення прототипів рішення актуальних завдань з цієї області».

Логічне програмування: «Дана колекція фактів і стосунків, що характеризує актуальне завдання. Потрібно привести цю колекцію до форми, що демонструє відповіді на практичні запити відносно цього завдання».

Об'єктно-орієнтоване програмування: «Доступна ієрархія класів об'єктів з працездатними методами рішення ряду завдань деякої сфери додатка ИТ. Вимагається без зайвих трудовитрат уточнити цю ієрархію, щоб пристосувати її до рішення нових затребуваних завдань цієї області, її розширення або їй подібною».

Багато завдань включають такі формулювання в якості підзадач, що приводить при створенні МП і розробці СП до підтримки різних парадигм одночасно. Так, наприклад, при цілеспрямованій розробці монопарадигматичної мови Haskell, що позиціонується як чисто функціональна мова, автори прийшли до концепції монад, що дозволяє притягати механізми інших парадигм. Потреба в підтримці парадигм, відсутніх в МП, що реалізовується, може бути забезпечена в СП за допомогою бібліотечних процедур, асемблерних вставок, макрогенераторів або організації виходу на рівень операційної системи.

Таблиця 1

Назви деяких ЯВУ, що відносяться до різних парадигм програмування

Імперативне (стандартне / системне) програмування	Функціональне / апплікативне програмування	Логічне / декларативне програмування	об'єктно-орієнтоване програмування	учбові мови програмування
Fortran	Lisp	Planner	Simula-67	Pascal
Algol	Рефал	Snobol	Smalltalk-80	Basic
Альфа	ML	Prolog	C++	Grow
C	Cmucl	Conniver	Eiffel	Logo
Modula	Erlang	QLisp	Clos	Oberon
Эльбрус	Interlisp	Mercury	Java	Робик
Occam	MuLisp	Oz	C#	Рапира
BLISS	Scheme		Scala	Oz
ЯРМО	Hope			
YACC	Clean			
Lex	Dylan			
	Miranda			
	Python			
	Haskell			
	Rubi			
	F#			

У практиці визнають основними парадигми імперативно-процедурного, функціонального, логічного і об'єктно-орієнтованого програмування, підтримувальні механізми зниження трудомісткості повного життєвого циклу програм, з тенденцією просування до паралельного програмування. Нині по суті розрізняють більше двох десятків парадигм. Багато МП відносять до п'яти-восьми парадигм. Частину згадуваних в різних джерелах парадигм можна характеризувати як стилі або методики, що відбивають пошук шляхів зниження трудомісткості програмування і підвищення надійності програм на базі доступних СП. Наприклад, аспектно-орієнтоване програмування підтримується як макророзширення ТОП. Структурне програмування фактично зводиться до ряду рекомендацій по стилю представлення імперативно-процедурних програм. Метапрограмування є технікові компіляції програм в комплекті з типовими елементами даних. Зустрічаються прецеденти розгляду недетермінізму як основній реалізації паралелізму.

Слід особливо відмітити трудомісткість паралельного програмування, що призводить до тенденції включати його у будь-яку парадигму на правах узагальнення окремих конструкцій і тим самим відкладати на майбутнє необхідність розробки методів верифікуючою компіляції і оптимізації програмних компонентів, засобів масштабованої генерації коду і трансформацій програм, що автоматизуються, з посвідченням збереження властивостей при їх комплексації з раніше відлагоджених компонентів [8]. Парадигма паралельного програмування займає нішу, пов'язану з реалізацією програм виконання обчислень на багатопроцесорних системах для організації високопродуктивних обчислень [3]. Ця ніша обтяжена різким підвищенням складності відладки програм, викликаною комбінаторикою виконання фрагментів асинхронних процесів. Якщо перший бар'єр до успіху в паралельному програмуванні обумовлений послідовним стилем мислення, то другий бар'єр залежить від концептуально не здоланої трудомісткості відладки програм. Хоча основна трудомісткість життєвого циклу програм пов'язана саме з тестуванням і відладкою програм, цей напрям за півстоліттям професійного програмування прогресує переважно за рахунок «силових« характеристик устаткування, концептуально не виходячи за межі можливостей екстракодів відлагодження програм в домовну епоху. Доцільність методів верифікації програм, що вже досягли практичних характеристик, ще не отримала довіри наших практиків, можливо із-за кваліфікаційного цензу на створення формальних специфікацій [2].

Інструментальне ядро ЯВУ можна обмежити однією арифметичною системою. Елементарні рівні опорних ЯВУ різних парадигм можна описати як розширення або конкретизації такого ядра. При аналізі парадигм ЯВУ слід зазначити віхи вдосконалення системної програмотехники :

– перший ЯВУ Fortran ввѐл в практику роздільну компіляцію, що понизила трудовитрати на відладку програм завдяки виділенню техніки зборки модулів;

– універсальна мова Lisp дала життя машинно-незалежному стилю символьної обробки даних, що привело до парадигми функціонального програмування;

– популярність мови С пов'язана з рішенням проблем машинно-залежного перенесення програм, що одночасно підтримало перенесення на безліч архітектури;

– логічне програмування на мові Prolog дозволило працювати з недовизначеними постановками завдань, підвищуючи міру їх вивченості;

– поява С++ і ТОП розширило сфери додатка інформаційних систем, при конструюванні яких потрібне повторне програмування.

Теоретично відмінність ПП досить ясно виражається на рівні операційної семантики, що представляє деталі структур пам'яті і механізми виконання укрупнѐнных дій. Для вироблення практичних рекомендацій потрібно детальніше формулювання відмінностей на усіх рівнях визначення МП, включаючи реалізаційну прагматику, що задає межі обчислюваності і ефективності програмованих рішень, набір синтаксично помітних понять, що показує зручність відображення термінології області додатка в програмні конструкції, і експлуатаційну прагматику, що визначає працемісткість програмування і живучість програм, що розробляються.

У методиці програмування конкретизація відповідає низхідним методам зверху «вниз». (Усупереч лінгвістичній асоціації немає причин вважати низхідні методи зворотними до висхідних. Узагальнення психологічно не симетрично конкретизації. Top_Down - Bottom_Up.)

По мірі вивченості істотно розрізняються наступні категорії постановок завдань, мислення, що впливають на стиль, і вибір методів рішення завдань :

- нові;
- дослідницькі;
- практичні;
- ефективні.

Висновки

Макетний зразок рішення нової задачі працездатний при пред'явленні автором невеликого набору відповідних даних. Для експериментального полігону відладки рішень дослідницьких завдань потрібно пристосованість до обробки майже будь-яких даних, які можуть бути задані в якості вхідних. Практична версія може бути обмежена обробкою даних, що реально зустрічаються у сфері еѐ застосування. Для ефективної реалізації потрібні спеціально підібрані дані, що показують еѐ перевагу над менш майстерно виконаними рішеннями задачі.

По специфіці програмування дуже істотно розрізняються наступні категорії завдань :

- багаторівневе абстрагування для особливо важливих і складних завдань із спеціальними методами рішення;
- бізнес-додатки, залежні від динаміки виробничої діяльності людини;
- збір фактів і накопичення знань для не цілком певних завдань;
- дослідження і розробка нових алгоритмів і структур даних, включаючи створення нових мов;
- реалізація добре вивчених алгоритмів для коректних завдань.

Література

1. Бек, К. Экстремальное программирование — Питер, 2002

2. Бурдонов И.Б., Косачев А.С., Кулямин В.В. Теория соответствия для систем с блокировками и разрушениями − М.Ж Физматлит, 2008. − 412 с.

3. Воеводин В. В. Параллельные вычисления / В. В. Воеводин, Вл. В. Воеводин. — СПб.: БХВ-Петербург, 2002. — 608 с.

4. Городняя Л.В. О проблеме автоматизации параллельного программирования // В сборнике Международной суперкомпьютерной конференции «Научный сервис в сети Интернет: многообразие суперкомпьютерных миров » — URL: http://agora.guru.ru/abrau2014

5. Городняя Л.В. Парадигмы параллельного программирования в университетских образовательных программах и специализации // Всероссийская научная конференция "Научный сервис в сети Интернет: решение больших задач − Новороссийск-Москва, 2008. − с. 180-184. Л.В.

6. Городняя Л.В. Парадигмы программирования Часть 1 Сравнение парадигм программирования Препринт 172 URL: - http://www.iis.nsk.su/files/preprints/172.pdf

7. Палмер С.Р., Фелсинг. Дж.М. Практическое руководство по функционально ориентированной разработке ПО. − М.: Вильямс, 2002. − 299 с.

8. Степанов Г.Г. Пути обеспечения переносимости программ и опыт использования системы СИГМА // Трансляция и преобразование программ. – Новосибирск: ВЦ СО АН СССР, 1984. – 9 с.

9. Хорстман К. Scala для нетерпеливых. – ДМК пресс, 2013. – 408 с. – 300 экз. – ISBN 978-5-94074-920-2, 978-0-321-77409-5.

Referenses

1. Bek, K. Ekstremalnoe prohrammyrovanye — Pyter, 2002

2. Burdonov Y.B., Kosachev A.S., Kuliamyn V.V. Teoryia sootvetstvyia dlia system s blokyrovkamy y razrushenyiamy − M.Zh Fyzmatlyt, 2008. − 412 s.

3. Voevodyn V. V. Parallelnыe vыchyslenyia / V. V. Voevodyn, Vl. V. Voevodyn. — SPb.: BKhV-Peterburh, 2002. — 608 s.

4. Horodniaia L.V. O probleme avtomatyzatsyy parallelnoho prohrammyrovanyia // V sbornyke Mezhdunarodnoi superkompiuternoi konferentsyy «Nauchnыi servys v sety Ynternet: mnohoobrazye superkompiuternыkh myrov » — URL: http://agora.guru.ru/abrau2014

5. Horodniaia L.V. Paradyhmы parallelnoho prohrammyrovanyia v unyversytetskykh obrazovatelnыkh prohrammakh y spetsyalyzatsyy // Vserossyiskaia nauchnaia konferentsyia "Nauchnыi servys v sety Ynternet: reshenye bolshykh zadach − Novoro ssyisk-Moskva, 2008. − s. 180-184. L.V.

6. Horodniaia L.V. Paradyhmы prohrammyrovanyia Chast 1 Sravnenye paradyhm prohrammyrovanyia Preprynt 172 URL: - http://www.iis.nsk.su/files/preprints/172.pdf

7. Palmer S.R., Felsynh. Dzh.M. Praktycheskoe rukovodstvo po funktsyonalno oryentyrovannoi razrabotke PO. − M.: Vyliams, 2002. − 299 s.

8. Stepanov H.H. Puty obespechenyia perenosymosty prohramm y opыt yspolzovanyia systemы SYHMA // Transliatsyia y preobrazovanye prohramm. – Novosybyrsk: VTs SO AN SSSR, 1984. – 9 s.

9. Khorstman K. Scala dlia neterpelyvыkh. – DMK press, 2013. – 408 s. – 300 ekz. – ISBN 978-5-94074-920-2, 978-0-321-77409-5.

Рецензія/Peer review : 10.09.2020 р. Надрукована/Printed : 04.11.2020 р.