PASCAL-XSC
И СОВРЕМЕННАЯ ЧИСЛОВАЯ ОБРАБОТКА

Предисловие редакторов перевода

Чем замечателен PASCAL-XSC?

Постараемся сразу сказать о предмете, рассмотренном в данной книге, самое главное.

- PASCAL-XSC — это современный язык для программирования численных алгоритмов, разработанный преимущественно немецкими специалистами.
- В основе PASCAL-XSC лежит концепция получения численных результатов с максимально высокой точностью, причем точность вычисления оценивается в процессе самого вычисления.
- Транслятор языка реализован для целого ряда компьютерных платформ и операционных систем; имеются бесплатные и свободно доступные через Интернет реализации для IBM PC.
- Написанное на PASCAL-XSC численное программное обеспечение высокоустойчиво относительно перемещений из одного компьютерного окружения в другое.
- Существуют написанные на PASCAL-XSC готовые библиотеки, реализующие основные методы вычислительной математики в варианте с автоматической оценкой точности получаемых результатов.
- PASCAL-XSC используется специалистами в качестве удобного и строгого языка описания новых численных алгоритмов.
- В зарубежных (и отчасти отечественных) вузах накоплен значительный опыт обучения численному программированию с привлечением PASCAL-XSC в качестве базового языка; этот опыт нашел отражение во множестве публикаций.
- Знакомство с PASCAL-XSC существенно облегчает понимание идеологии так называемых достоверных (или надежных) вычислений, а также освоение других современных языков численного программирования, основанных на данной идеологии.

Подробнее об этих и других связанных с PASCAL-XSC фактах рассказывается ниже.

Концепция достоверных численных вычислений

За время своего развития классическая математика накопила достаточно большой арсенал численных методов. Формулировались эти методы в терминах абстрактных математических объектов, важнейшим из которых было вещественное число. Однако попытки прямого применения таких методов всегда наталкивались на одну и ту же трудность: в практических вычислениях можно использовать только числа, имеющие представление в виде конечной цепочки цифр. Понятно, что не всякое вещественное число имеет подобное представление. Это относится как к исходным данным, так и к результатам численных вычислений. Следовательно, возникает проблема соотнесения вычисленного приближенного результата с истинным абстрактным решением поставленной задачи.

В настоящее время применяются два основных подхода к разрешению данной проблемы. Первый из них состоит в том, что вместо абстрактных математических объектов, таких как вещественные числа, векторы и матрицы, в качестве приближений используются объекты того же класса, но имеющие точное конечное представление. Чаще всего в такой роли выступают рациональные числа из некоторого конечного множества и составленные из них векторы и матрицы. При этом вычисление сопровождается ручным оцениванием погрешностей (то есть отклонений приближенных значений от точных) для каждой из входящих в вычисление операций.

Желание распространить описанный подход на решение численных задач с помощью компьютера привело к попыткам построить для каждого численного метода аналитическую зависимость погрешности результата от погрешностей исходных данных и округлений. Хотя на этом пути и были достигнуты определенные успехи (преимущественно в линейной алгебре — см. [1]), но в целом эти попытки не достигли намеченной цели. В качестве причин неудачи можно указать следующие: вычисления по формулам, связывающим погрешность исходных данных с погрешностью результата, сами неизбежно производятся с погрешностью; в них должны учитываться особенности системы команд того конкретного процессора, на котором будет происходить вычисление, что не позволяет применить эти формулы для любого процессора; в них невозможно учесть индивидуальную точность операндов — отсюда грубость оценки конечного результата и т. д. Вообще, для многих достаточно нетривиальных алгоритмов получить сколько-нибудь реалистичную оценку погрешности вычислений оказывается чрезвычайно трудной математической задачей, по сложности зачастую сопоставимой с разработкой самого численного метода.

Как следствие, разработчики систем программирования для численных приложений, основанных на замене точного результата одним приближенным, обычно попросту «машут рукой» на оценку погрешностей, и пользователь таких систем не может уверенно сказать, с какой точностью будет подсчитана, например, даже такая величина, как $exp(0.0001)$.

Кроме того, серьезный изъян имеется в самой идее замены точного значения одним приближенным. Дело в том, что в этом случае не гарантируется, вообще говоря, корректность проверки числовых соотношений: если у вас в программе имеется, например, оператор

if x < 0 **then** A **else** B,

а x вычисляется приближенно, то вы никогда не сможете быть уверены в правильности ветвления, поскольку, скажем, вычисленное значение 0.013 может заменять точное значение x, равное –0.02.

Недостатки данного, привычного, подхода можно перечислять и далее. К ним относится, например, отсутствие ассоциативности в цепочке операций сложения и умножения. Выходит, что результат операций типа скалярного умножения будет разным в зависимости от особенностей компьютерного окружения — транслятора, процессора, выбранной разрядности, способов округления и т. д. Как следствие, выполнение одного и того же алгоритма в разных компьютерных окружениях приводит к различным, порой совершенно непохожим друг на друга результатам.

Постепенно у критически настроенных исследователей все чаще стал возникать вопрос, вынесенный в заголовок обобщающей статьи немецкого математика проф. К. Никеля: «Can we trust the results of our computing?» («Можем ли мы доверять результатам наших вычислений?») [2]. Действительно, беспристрастный анализ традиционного подхода к численным вычислениям и соответствующего инструментария (алгоритмов, языков программирования и аппаратного обеспечения), проведенный специалистами в области вычислительной математики, привел к неутешительному выводу о том, что алгоритм, сформулированный в столь привычных нам терминах, попросту недоопределен и потому обладает, вообще говоря, непредсказуемыми свойствами. (На одной из научных конференций, где присутствовали авторы настоящего предисловия, ректор Технического университета Вены проф. П. Скалички наполовину с юмором, а наполовину всерьез заявил, что с тех пор, как подробнее узнал о принятых способах выполнения машинных вычислений, очень опасается ходить по мостам и оказываться внутри других сложных инженерных сооружений...)

Подводя итоги, можно сказать, что первый подход оказался во многих случаях полезен (особенно в линейной алгебре) для исследования сравнительной точности и устойчивости численных методов, но не позволил гарантировать корректность вычислений по ветвящимся алгоритмам (заметим, что большинство алгоритмов являются ветвящимися!), а также не дал возможности получать надежные ответы на самые простые и естественные вопросы вроде следующего: сколько верных цифр присутствует в полученном численном результате?

Таким образом, широкое использование компьютеров в вычислительной практике показало очевидную ограниченность первого подхода. По этой причине математиками стал разрабатываться другой, альтернативный, подход, получивший название интервального.

Суть данного подхода состоит в том, что неизвестное точное значение заменяется не единственным элементом того же класса, как при первом подходе, а конечно-представимым множеством элементов, содержащим в себе этот неизвестный элемент. Название «интервальный» данный подход получил в связи с тем, что интервал, представляемый обычно парой рациональных чисел-границ, является простейшим видом конечно-представимого множества, локализующего простейший абстрактный объект — вещественное число.

В рамках интервального подхода исходные данные и промежуточные результаты представляются граничными значениями, над которыми и производятся все операции. При этом сами операции (прежде всего арифметические) определяются таким образом, что результат соответствующей точной операции обязательно лежит внутри вычисляемых границ.

Приведем правила выполнения операций вещественной интервальной арифметики:

$$[a,\ b] + [c,\ d] = [a + c,\ b + d];$$
$$[a,\ b] - [c,\ d] = [a - d,\ b - c];$$
$$[a,\ b] \times [c,\ d] = [\min(ac,\ ad,\ bc,\ bd),\ \max(ac,\ ad,\ bc,\ bd)];$$
$$[a,\ b] \ / \ [c,\ d] = [\min(a/c,\ a/d,\ b/c,\ b/d),\ \max(a/c,\ a/d,\ b/c,\ b/d)],$$
$$\text{где } 0 \notin [c,\ d].$$

Возникающая при вычислении границ погрешность учитывается с помощью направленных округлений: левая вычисленная граница получается округлением влево до ближайшего машинного числа, а правая — вправо.

Интервальный подход позволяет единообразным способом учесть все виды погрешностей вычислительного процесса: приближенно известные исходные данные заключаются в гарантированно содержащие точное значение границы, погрешности округлений лишь несколько расширяют границы промежуточных результатов, а сам численный метод строится так, чтобы погрешность метода (вызванная, например, остановкой бесконечно сходящегося процесса) также включалась в вычисленные границы конечного результата.

Главные преимущества интервального подхода — автоматический учет всех видов погрешностей в процессе самого вычисления и гарантированная точность результата (если, скажем, вы получили результат [1.71718, 1.71901], то тем самым вам гарантируется верность его первых трех цифр). Корректным образом, через соотнесение границ, теперь могут быть выполнены операции сравнения.

Итак, интервальный подход лишен главных недостатков, присущих первому, более традиционному, подходу. Можно указать и на важное дополнительное преимущество: если одна и та же задача решается различными численными методами, то результаты вычислений по этим методам можно пересекать в теоретико-множественном смысле — с тем чтобы: 1) отобрать более точный метод, 2) сделать окончательный результат (собственно пересечение) более точным, чем полученный любым из

методов в отдельности, 3) проконтролировать разработку и программирование методов (пустота пересечения свидетельствует об ошибке).

Как видим, второй подход обладает многими несомненными преимуществами перед первым.

На сегодняшний день интервальные вычисления имеют уже достаточно богатую историю. Хотя идея приближения абстрактного математического объекта с помощью заключения его в конечно-представимые границы уходит своими корнями в глубокую древность, тем не менее ее систематическая разработка началась лишь в связи с широким применением компьютеров в вычислительной практике. Первые попытки использования интервального подхода в решении конкретных задач численного анализа относятся к рубежу 50–60-х гг. Решающим шагом на пути оформления идеи в самостоятельное математическое направление стала написанная в 1966 г. основополагающая монография американского математика Рамона Мура [3]. Первоначально это направление получило название «интервальный анализ», а затем чаще стал употребляться термин «интервальная математика».

В настоящее время интервальная математика — это прежде всего основанный на интервальном подходе численный анализ. Интервальный численный анализ включает в себя такие области, как линейная алгебра, решение разнообразных уравнений и систем уравнений, оптимизация и т. д., то есть по названиям разделов в основном дублирует традиционный численный анализ. Существуют, впрочем, и специфические разделы: например, большое число работ посвящено методам вычисления рациональных интервальных функций. Некоторые задачи традиционного численного анализа имеют различные интервальные постановки. Приведем простейший пример: задача нахождения корня x уравнения

$$a + x = b,$$

где a и b — вещественные числа, в интервальной постановке может быть сформулирована двумя способами (A и B — интервалы):

1) найти интервал X такой, что $A + X = B$;
2) найти интервал X такой, что $X = B - A$.

(Решение первой задачи является подмножеством решения второй.)

В отличие от традиционного подхода, в интервальных алгоритмах широко используются теоретико-множественные операции: например, приближение, полученное на $(n-1)$-й итерации, с целью уточнения пересекается с приближением, полученным на n-й итерации. Операндами этих, а также арифметических операций могут быть представленные своими границами множества, построенные из математических объектов разнообразных классов: замкнутых, открытых, полуоткрытых и бесконечных интервалов на вещественной оси; кругов, прямоугольников, секторов и других фигур на комплексной плоскости; шаров, параллелепипедов, конусов в многомерном случае; из всех названных объектов могут составляться векторы и матрицы и т. д. Точность метода теперь характеризу-

ется некоторой величиной, соответствующей «размеру» результирующего множества. Точность может быть оценена и до реального вычисления — подобно тому, как это делается при первом, традиционном, подходе.

Классическими монографиями по интервальной математике являются [3–5], множество сборников и монографий вышли в последующие годы. Общее число публикаций по данной тематике, включая журнальные статьи, насчитывает сегодня несколько тысяч. О том, как ознакомиться с современным состоянием «интервальной науки», будет сообщено ниже.

Для полноты картины надо сказать о том, что внутри интервальной математики существует тенденция к комбинированию в одном методе традиционных и интервальных вычислений. Осуществляется такое комбинирование следующим образом; сначала из интервалов с исходными данными берутся числа-представители и с помощью традиционных вычислений находится приближенное (в классическом смысле) решение задачи, а затем это решение с помощью интервальных вычислений расширяется на всю область определения исходных данных. Тем самым подобные методы совмещают в себе свойства обоих упомянутых выше подходов. В западной литературе такие методы получили название inclusion methods — «методы включения». В отечественной монографии [6], где эти методы являются основным предметом исследования, они названы двусторонними.

Собственно интервальные и комбинированные методы можно было бы собирательно назвать локализационными, поскольку и те и другие отличаются от прочих методов тем, что локализуют (ограничивают) точное решение соответствующей задачи классической математики. Часто применяют и другие названия: self-validating numerics — «самоверифицирующие вычисления», verified computations — «вычисления с верификацией» и reliable computations — «надежные вычисления», или «достоверные вычисления».

За прошедшие два-три десятилетия эти вычисления широко применялись в различных областях науки и технологии. Прежде всего здесь необходимо упомянуть саму математику. С помощью интервальных вычислений удалось доказать целый ряд важных теорем, предполагающих в процессе своего доказательства сравнение сложных числовых выражений. Поскольку из-за свойства гарантированности включения точного результата в вычисленный интервал подобные вычисления обладают доказательной силой, они и были использованы для автоматизации процесса доказывания.

Что же касается внематематических приложений, то они оказались возможны всюду, где адекватным способом формализации неопределенности, содержащейся в исходных данных и/или результате, оказывалось указание разумных границ неизвестной величины. «Исходные данные лежат в таких-то границах и связаны такими-то формулами с результатом. Можно ли указать разумные границы, в которых лежит результат?» — такой способ математической формализации оказался во многих случаях значительно более простым и естественным, нежели, скажем, базирующийся на вероятностно-статистическом подходе или теории нечетких множеств.

В настоящее время уже почти невозможно уследить за распространением «надежных» вычислений по прикладным областям. Так, авторам настоящей статьи известны публикации, связанные с использованием таких вычислений в машиноведении и метрологии, химии и биологии, радиоэлектронике и электротехнике, геодезии и геофизике, экономическом планировании и прогнозировании, теории автоматического управления и космических исследованиях, автоматизированном проектировании и машинной графике, разработке программного обеспечения общего назначения и даже в организации пожаротушения и огранке драгоценных камней. Список, разумеется, далеко не полный.

Стремление к надежности численных расчетов привело не только к разработке нового математического аппарата, появлению многочисленных приложений, но и к развитию соответствующих компьютерных средств. К краткому рассказу о них мы сейчас и перейдем.

От первых библиотек — к SC-языкам

Еще на заре развития интервальной математики были осуществлены первые эксперименты по реализации интервальных вычислений на ЭВМ. Реализация осуществлялась обычно следующим способом: интервальные операции программировались на языке типа Алгола или Фортрана и оформлялись в виде библиотеки подпрограмм. Так были созданы сначала примитивные коллекции процедур, а затем и весьма «продвинутые» пакеты, содержавшие полный набор подпрограмм, реализующих арифметические, теоретико-множественные и логические операции над интервалами и составленными из них векторами и матрицами; как правило, такие пакеты включали в свой состав также подпрограммы для вычисления элементарных функций от интервального аргумента. В некоторых из них поддерживались даже переменная разрядность и работа с числами, представленными в десятичной системе счисления (см., например, [7]). Отдельные библиотеки вышли на уровень коммерческих продуктов [8].

Однако, при всех их достоинствах, пользоваться подобными библиотеками было крайне неудобно из-за отсутствия в доступных на тот момент языках возможности конструировать произвольные типы данных и определять над элементами этих типов инфиксные операции — значительная часть текста основной программы представляла собой последовательности вызовов подпрограмм. Кроме того, такой способ реализации приводил к значительному (до двух порядков) замедлению выполнения программы по сравнению с обычными вычислениями с плавающей точкой.

Впоследствии был опробован другой подход — основанный на использовании языков с богатыми средствами конструирования типов и определения операций, таких как Алгол-68 и Ада. Лаконичность и выразительность программного текста значительно возросли; замедление в этом случае не превосходило одного порядка. Основными препятствиями на пути широкого использования таких языков стало отсутствие их компи-

ляторов для большинства компьютерных платформ, а также невозможность эффективной реализации на этих языках операций с оптимальным направленным округлением результатов, необходимым для наиболее точного учета погрешностей округления. Впоследствии, однако, с появлением языка C++, реализованного для множества платформ, указанный подход получил «второе дыхание» — и на C++ стали создаваться библиотеки (см., например, [9]), которые оказалось возможным достаточно удобно и без значительного ущерба для эффективности вызывать из основной программы.

И все же на сегодняшний день преимущество у еще одного подхода — третьего. Он состоит в том, что запись численного алгоритма осуществляется на языке программирования, специально спроектированном с учетом требований интервальных вычислений. Такие языки были названы SC-языками — от английского словосочетания «Scientific Computations», т. е. «Научные Вычисления» (разумеется, обозначение «SC» условно — не в меньшей степени эти языки пригодны и для программирования инженерных расчетов).

За 70—90-е гг. было разработано (преимущественно в Германии и Швейцарии) целое семейство таких языков: FORTRAN-SC и -XSC, PASCAL-SC и -XSC, MODULA-SC, OBERON-XSC. Введение в базовый язык дополнительных конструкций и реализация специальных численных модулей позволили создать новые системы программирования на основе Фортрана, Паскаля, Модулы-2 и Оберона — популярных и хорошо изученных множеством пользователей языков. Основными отличительными чертами расширенных подобным образом систем служит их ориентация на полную математическую корректность, интервально-локализационный подход, достижение максимальной точности численных результатов и высокой воспроизводимости расчетов в различных компьютерных окружениях. Объединяют такие системы следующие особенности: строгая спецификация точности всех численных операций, включая вычисление элементарных функций (причем специфицируемая точность обычно является предельно достижимой для данной операции), возможность выполнить любую арифметическую операцию над числами с плавающей точкой с направленным округлением или округлением к ближайшему машинно-представимому числу, наличие предопределенных интервальных типов данных и операций над ними (включая комплексно-интервальные и векторно-матричные), возможность записывать арифметические и логические выражения над переменными определенных пользователем типов в традиционной (инфиксной) форме, ввод-вывод чисел с управляемым округлением, возможность вычисления с наивысшей достижимой точностью выражений типа скалярного произведения и некоторые другие, менее существенные свойства.

Подчеркнем, что в SC-языках все приближенные операции выполняются с оптимальным специфицированным округлением. Как следствие, написанное с их помощью программное обеспечение высокомобильно: при использовании стандартизованной машинной арифметики (о стандарте см. ниже) и соблюдении договоренностей о режимах компиляции в любом компьютерном окружении будут получаться тождественные результаты. Присутствие в этих языках операций высокоточного вычисления скалярных произведений позволяет добиться в ответственных случаях полной ассоциативности сложения и умножения даже независимо от режимов компиляции — это еще одно из средств достижения мобильности.

Одни SC-языки стали предметом лишь теоретического обсуждения, для других появились экспериментальные системы программирования, а некоторые, среди которых можно назвать PASCAL-XSC, «доросли» и до коммерческих версий. В качестве методов реализации использовались как построение специального компилятора в машинный код, так и прекомпилятора, конвертирующего программу с SC-языка в стандартное подмножество какого-либо языка общего назначения. Преимуществом последнего метода (он был выбран и для PASCAL-XSC) является сравнительная легкость реализации и возможность достичь высокой независимости программ от свойств компьютерного окружения (платформы, операционной системы и т. д.).

На аппаратном уровне SC-языки могут быть поддержаны как специальными процессорами, так и любыми процессорами, следующими известному стандарту на арифметику чисел с плавающей точкой ANSI/IEEE-754 [10] (стандарт предусматривает операции с направленными округлениями, поддержку бесконечных интервалов и др.). Заметим, что в соответствии с данным стандартом разработаны все современные процессоры для персональных компьютеров, содержащие команды арифметики чисел с плавающей точкой. По сути, лишь программы на SC-языках и позволяют по-настоящему использовать возможности аппаратных средств, имеющихся сегодня фактически на любом персональном компьютере.

В настоящее время с помощью SC-языков разработаны различные пакеты численных методов, написанные на этих языках программы активно используются для решения научно-технических задач. Наиболее же известен на сегодня среди SC-языков PASCAL-XSC: он тщательно описан, имеет реализации для самых различных платформ, широко применяется в учебном процессе многих западных университетов, накоплен определенный опыт его применения и в ряде научных и высших учебных заведений СНГ (Санкт-Петербургский государственный университет, Красноярский ВЦ СО РАН, Львовский государственный университет, Московский институт стали и сплавов, Саратовский государственный университет и др.). Поэтому именно учебник по PASCAL-XSC и был выбран нами для перевода и издания в качестве первой книги, знакомящей уже более широкий круг русскоязычных читателей с идеями SC-языков. Второе, исправленное и дополненное издание, является на сегодня наиболее представительным и современным описанием языка, поскольку содержит Добавление, отсутствующее как в немецком, так и в английском изданиях настоящей книги, а также ссылки на самые свежие публикации и на релевантную информацию в Интернете.

PASCAL-XSC: специфика, применение и реализация

Стандартный Паскаль и PASCAL-XSC

Знаменитый язык программирования Паскаль был разработан известным швейцарским «изобретателем языков» Никласом Виртом. Первое, предварительное, описание языка было опубликовано в 1968 г., в 1970 г. был создан первый транслятор, после чего язык получил широчайшее распространение по всему миру. Лаконичность, выразительность, концептуальная выдержанность — эти качества позволили Паскалю завоевать огромное число приверженцев. Уже к 1979 г. существовало свыше 80 трансляторов языка для самых различных платформ и операционных систем.

Столь бурный процесс распространения языка привел к появлению множества его версий и породил проблему совместимости программ, написанных на отличающихся, порой значительно, диалектах Паскаля. В результате специалисты по компьютерному делу предприняли специальные усилия по разработке международного стандарта, который и появился в 1982 г. Этот стандарт был принят Международной организацией по стандартизации — ISO (International Organization for Standardization). С тех пор все диалекты Паскаля обязательно должны включать ISO-стандарт Паскаля в качестве своего подмножества. Нормативное описание языка с учетом требований ISO-стандарта можно найти в книге [11].

PASCAL-XSC является одним из расширений стандартного Паскаля. «Изобретен» PASCAL-XSC был в Институте прикладной математики (II) Университета г. Карлсруэ (Германия) под руководством проф. У. Кулиша. Наиболее активный период развития языка и разработки соответствующего программного обеспечения пришелся на 80-е — начало 90-х гг.

Каковы же особенности PASCAL-XSC, отличающие его от стандартного Паскаля? Если говорить очень кратко (а для подробного ознакомления и предназначена настоящая книга), то это модульная структура программы, межмодульный экспорт-импорт данных различных типов, возможность введения собственных и переопределения уже существующих знаков операций с последующим их использованием в обычных выражениях, разрешение совмещать имена процедур, а также знаки операций (так, что они становятся применимыми к объектам различных типов), распространение основных операций на все структурированные типы, добавление комплексных чисел и интервалов, векторов и матриц в набор предопределенных типов, наличие динамических массивов и развитых средств их размещения и переразмещения, управление округлениями для всех числовых операций и операций ввода-вывода, максимальная (в строгом математическом смысле) точность таких операций, поддержка высокоточного вычисления операций типа скалярных произведений. Приведенный список не исчерпывает всего набора дополнительных по отношению к ISO-стандарту языковых средств PASCAL-XSC, но он дает представление о направленности произведенной доработки.

PASCAL-XSC как средство обучения

Паскаль всегда рассматривался экспертами как один из лучших языков для обучения программированию. И PASCAL-XSC в полной мере сохраняет это свойство — с тем уточнением, что речь идет о программировании численном.

Действительно, оригинальный инструментарий PASCAL-XSC, созданный для решения задач числовой обработки, продуман весьма тщательно. Однако и в целом PASCAL-XSC столь аккуратно и экономно укомплектован действительно необходимыми средствами, что трудно назвать лучший язык для первоначального обучения программированию и уж точно невозможно найти что-либо более подходящее для обучения программированию численному. Несомненно, язык покажется находкой для преподавателей компьютерного дела, особенно на технических и естественнонаучных факультетах вузов.

Именно учебная ориентация настоящего издания обусловила повышенное внимание к «отделке» его текста. Прежде всего, переводчики и редакторы данной книги, постоянно испытывая самые неприятные чувства от лексики большинства современных изданий по языкам программирования, решили максимально тщательно подойти к отработке терминологии с позиций как академической точности, так и общеязыкового благозвучия. В итоге мы отказались от многого из того, чем пестрит современная литература по программированию, — всевозможных «массивовых и записных типов» (у нас — просто «типы массива и записи»), «перегруженных операций» (у нас они, в соответствии со смыслом, названы «совмещенными»), «вещественных чисел» (это вообще математически некорректно — единственно возможный перевод: «числа типа *real*»), «целых выражений» (мы предпочли «целочисленных»), «анонимных типов» (избавляясь от субъективации, мы перевели «безымянных») и т. д. и т. п. Немало возникло и специфических проблем, связанных с особенностями числовой обработки. Так, не без труда нам удалось «развести» в переводе английские accurate, precise и exact, каждое из которых чаще всего передается по-русски просто как «точный», тогда как оттенки их значений играют немаловажную роль в контексте заложенной в PASCAL-XSC идеологии. Но, конечно же, о достигнутом качестве редактирования судить надлежит читателю — мы лишь сообщаем, что старались подготовить книгу именно как учебник, то есть добиваясь одновременно легкости чтения и однозначности. Следует, впрочем, сознаться, что немецкий университетский стиль изложения сильно облегчил нашу задачу.

Язык, однако, может быть использован не только для обучения программированию, но и в качестве инструментального средства в курсе численного анализа. Нотация PASCAL-XSC может быть успешно применена для детального описания численных алгоритмов. В качестве примера приведем описание твинной арифметики (одной из модификаций обычной арифметики вещественных интервалов) в виде набора записанных на PASCAL-XSC программных модулей [12].

Прекрасным материалом для построения курса современного численного анализа может послужить глава 5 настоящей книги, содержащая 28 упражнений с их решениями. Упражнения направлены на формирование определенной культуры, в рамках которой связь между абстрактными математическими объектами и конечно-представимыми машинными числами устанавливается всегда корректно. Для закрепления и развития приобретенных начальных навыков может служить более серьезный материал, представленный в книгах [5, 6, 13]. В совокупности настоящее издание и упомянутые книги дают возможность сформировать курс численного анализа, опирающийся на концепцию «надежных» вычислений и оснащенный соответствующим программным инструментарием. Разумеется, речь идет лишь о книжных изданиях на русском языке; о том, как получить доступ к другим публикациям, мы скажем ниже.

Реализации PASCAL-XSC

Знакомиться с описанием языка без возможности получить в свое распоряжение транслятор — это все равно что читать поваренную книгу без шансов попробовать описанные в ней блюда. К счастью, в данном случае эта проблема отсутствует.

Как мы уже говорили, PASCAL-XSC имеет трансляторы для различных платформ (в том числе для IBM PC) и операционных систем (полный список реализаций *см.* на сайте `www.xsc.de`, о котором будет сказано ниже). Достигнуто это было благодаря выбранному разработчиками методу реализации: программа на PASCAL-XSC обрабатывается прекомпилятором, написанным на строго стандартизированном подмножестве языка Си, а результатом обработки является код в том же самом подмножестве Си. Этот код транслируется с помощью какого-либо Си-компилятора из числа имеющихся сейчас в большинстве компьютерных окружений. Такая же технология была применена и к другим компонентам системы программирования на PASCAL-XSC. В итоге ядро системы программирования может быть без чрезмерных усилий перенесено из одного компьютерного окружения в другое. Поскольку же исходные тексты всей системы программирования PASCAL-XSC свободно доступны через Интернет, существует принципиальная возможность самостоятельной адаптации этой системы к любому новому окружению, в котором транслируются программы на языке Си.

В большинстве существующих реализаций выполнение арифметических операций производится с аппаратной поддержкой: используются процессорные операции с плавающей точкой и управлением округлениями, предусмотренные стандартом ANSI/IEEE-754 [10]. Достигаемая благодаря этому производительность программ на PASCAL-XSC близка к производительности программ, написанных на других современных языках.

PASCAL-XSC в Интернете

Ресурсы, связанные с использованием PASCAL-XSC, достаточно полно представлены в Интернете.

Прежде всего приведем ссылку на сайт Института прикладной математики (II) Университета г. Карлсруэ:

```
http://www.uni-karlsruhe.de/~iam/lehrstuhl2-e.html
```

На сайте размещен ряд материалов, посвященных концепции достоверных вычислений и ее воплощению в соответствующее программное обеспечение; специальный раздел содержит сведения о языке PASCAL-XSC.

Значительное количество файлов, относящихся к PASCAL-XSC, находится на ftp-сервере Университета г. Карлсруэ. Эти файлы содержат, в частности, трансляторы для IBM PC; полные исходные тексты программ на языке Си, реализующих систему программирования на PASCAL-XSC; библиотеку написанных на PASCAL-XSC численных методов (решение систем линейных и нелинейных уравнений, обращение матриц, нахождение собственных значений, оценивание арифметических выражений, вычисление нулей полиномов, начальные и краевые задачи в решении дифференциальных уравнений, автоматическое дифференцирование, глобальная оптимизация и др.); полный текст руководства по системе программирования на PASCAL-XSC. Последнее из перечисленного настоятельно рекомендуем получить в свое распоряжение всем тем, кто собирается заниматься трансляцией написанных на PASCAL-XSC программ: настоящая книга это руководство не включает.

Итак, вот ссылка на соответствующий каталог ftp-сервера:

```
ftp://ftp.iam.uni-karlsruhe.de/pub
```

Есть также способ получить упомянутые выше файлы без прямого обращения к ftp-серверу. Этой цели служит следующий сайт:

```
http://www.xsc.de
```

Информация на нем удобно структурирована; помимо PASCAL-XSC, имеются также материалы, относящиеся к другим SC-средствам. Мы рекомендуем этот сайт как предоставляющий все необходимое для того, чтобы приступить к практической работе с PASCAL-XSC.

Еще один источник полезных текстов — личный сайт д-ра Г. Болендера из Университета г. Карлсруэ:

```
http://www.uni-karlsruhe.de/~iam/html/literatur/download.html
```

Особо подчеркнем, что в настоящее время все материалы, относящиеся к языку PASCAL-XSC, включая трансляторы и документацию, могут быть получены совершенно бесплатно (ранее платными были, например, трансляторы для рабочих станций с операционной системой UNIX).

Если у вас возникнут какие-либо вопросы по самому языку или по системе программирования, вы можете обратиться (на немецком или английском языке) к разработчикам. Писать в этом случае следует по адресу:

pascal-xsc@math.uni-karlsruhe.de

Необходимо сказать также, что в Интернете имеется масса связанных с использованием PASCAL-XSC материалов, размещенных на сайтах, созданных без участия разработчиков языка. Это и научные статьи, и готовые программы, и шлейфы разнообразных дискуссий. Так, обратившись в какой-то момент с запросом по ключевому слову «PASCAL-XSC» к одной из поисковых машин Интернета, мы получили порядка 400 ссылок. Надеемся, что, действуя таким же способом, вы сможете подобрать себе материалы, непосредственно относящееся к сфере ваших личных интересов.

О том, как получать информацию по более широкому кругу вопросов из сферы «надежных» вычислений, не ограниченному одним лишь языком PASCAL-XSC, мы расскажем ниже.

Другие современные компьютерные средства для выполнения достоверных вычислений

Настоящая книга открывает русскоязычному читателю дверь в мир современных компьютерных средств для выполнения достоверных, или «надежных», вычислений. PASCAL-XSC занимает среди них почетное место, но, разумеется, в этом мире есть и многое другое.

90-е годы уходящего столетия были временем бурного развития компьютерных средств для выполнения вычислений с автоматической верификацией точности, именуемых также «надежными», или «интервальными». Несомненно, интерес к этим вычислениям в ближайшее время лишь усилится, ведь пока интервальный подход в наиболее полной мере удовлетворяет практическим нуждам потребителей численных результатов, и конкурентов ему, по сути, нет.

Пока же вкратце (для подробного рассказа нужно написать отдельную книгу) расскажем о том, что делается в области разработки «интервального» программного обеспечения.

Первое направление — дополнение уже известных языков, компьютерно-алгебраических и других систем средствами работы с числовыми объектами интервального типа. К таким средствам относятся прежде

всего арифметические и теоретико-множественные операции, операции сравнения, преобразования и ввода-вывода, вычисления элементарных функций и специальные (например, оценивание ширины интервала). Как мы уже упоминали, имеется целый ряд библиотек на различных диалектах языка Си, реализующих достаточно представительный набор подобных операций. Вместе с тем продолжается разработка программных пакетов, реализующих всевозможные обобщения интервальной арифметики, вычисления с переменной разрядностью и нестандартными числовыми представлениями. Программируются эти пакеты на самых различных языках — от Си до Пролога.

В последние годы набирает силу прогрессивная тенденция объединения численно-интервальных и символьно-аналитических (компьютерно-алгебраических) систем программирования. Такое объединение позволяет совместить достоинства обоих подходов: производить аналитические выкладки, сопровождая их получением численных результатов с гарантированной и управляемой точностью. Использование символьно-аналитических преобразований дает возможность в ряде случаев улучшить точность самого интервального вычисления. Не случайно интервальные типы данных появились в среде Maple и Mathematica — известных систем компьютерной алгебры.

Операции над интервалами были определены даже в таком, казалось бы, имеющем весьма отдаленное отношение к числовой обработке, языке программирования, как Лисп. Средства интервальных вычислений оказалось возможным ввести в Microsoft Excel. Среди популярных языков, над «интервализацией» которого работают специалисты, находится Java. Подобные примеры можно множить. Однако наиболее серьезная на сегодняшний день разработка, выделяющаяся из общего ряда, — это Forte Fortran/HPC (дата появления — май 2000 г.).

Forte Fortran/HPC представляет собой интегрированную вычислительную среду, предназначенную для функционирования на мощных рабочих станциях типа Sparc с операционной системой Solaris. Как и станции Sparc, Forte Fortran/HPC относится к коммерческой продукции известной американской фирмы Sun Microsystems. В состав Forte Fortran/HPC входят компиляторы языков Fortran 95, Fortran 77 и C++. Предполагается, что все новое численное программное обеспечение для компьютеров платформы Sparc будет создаваться преимущественно на языке Fortran 95 (компилятор языка Fortran 77 нужен для совместимости со старым программным обеспечением). Forte Fortran 95 — это язык принципиально интервальный, в котором интервальные типы данных полностью «уравнены в правах» с типами REAL и INTEGER. Согласно идеологической концепции Forte Fortran 95, тип INTERVAL может появляться во всех контекстах, где традиционно для Фортрана мог появляться тип REAL. Разумеется, язык содержит и специфические средства, предназначенные для работы с объектами интервальных типов.

Forte Fortran 95 — первый и пока единственный «интервализованный» язык численного программирования с промышленным компилятором, созданным фирмой мирового уровня. «Интервализации» своего основного

языка для численных приложений Sun Microsystems придает большое значение. Как сказано в информационном сообщении, помещенном на официальном сайте фирмы и обращенном ко всем потенциальным пользователям системы Forte Fortran/HPC, «поддержка интервальной арифметики в Forte Fortran 95 способна изменить тот подход, который вы применяете, когда думаете о вычислениях».

В настоящей статье уместно было бы сравнить PASCAL-XSC и Forte Fortran 95, но, увы, их полноценное сопоставление выходит за рамки нашей задачи. Поэтому ограничимся общим утверждением и его иллюстрацией. Утверждение состоит в следующем: у каждого из этих двух языков имеются уникальные средства и возможности, отсутствующие у другого. Так, например, PASCAL-XSC позволяет с предельной точностью выполнять операции типа скалярных произведений (заметим, что в числе прочего этот инструмент позволяет моделировать арифметику произвольной точности) — аналога этому в Forte Fortran 95 нет. В свою очередь, Forte Fortran 95 реализует расширенную интервальную арифметику, допускающую работу с бесконечными интервалами и деление на интервал, содержащий ноль, — таких возможностей PASCAL-XSC не предоставляет.

За более подробной информацией о системе Forte Fortran/HPC, языке Forte Fortran 95 и его «интервализации» отсылаем на сайт фирмы:

`http://www.sun.com/forte/fortran` — общие сведения и дальнейшие ссылки;
`http://www.sun.com/forte/fortran/interval` — «интервализация».

Для обсуждения именно интервальных возможностей Forte Fortran 95 служит специальный электронный рассылочный список. С его помощью вы можете получать всю актуальную информацию и принять участие в дискуссиях специалистов. Для подписки достаточно послать письмо с единственной строкой:

subscribe sun-interval-fortran-questions

на следующий адрес:

`majordomo@interval.louisiana.edu`

Детальные инструкции можно получить, послав по тому же адресу электронное письмо со словом help в качестве содержимого.

К информации о Forte Fortran 95 добавим, что идеи, заложенные в нем, PASCAL-XSC и других «интервализованных» языках, оказывают самое непосредственное влияние на разработку Fortran-2000 — нового стандарта языка Фортран. Введение в этот стандарт интервальных типов данных можно считать уже практически согласованным решением.

Скажем кратко и о втором направлении разработки «интервального» программного обеспечения. Оно представляет собой создание развитых специализированных средств, предназначенных для решения задач какого-либо класса с гарантированной и автоматически вычисляемой оценкой точности. Чаще всего они реализованы как пакеты, иногда с собственным входным языком и интерактивным интерфейсом. Таких пакетов существует уже достаточно много. Среди них можно упомянуть следующие: GlobSol и VerGO — для решения задач глобальной оптимизации, FADBAD/TADIFF — для выполнения автоматического дифференцирования, INTBLAS — для решения задач линейной алгебры, AWA и COSY — для решения обыкновенных дифференциальных уравнений (с помощью COSY могут решаться и другие задачи, в частности глобальной оптимизации), SvLis — для решения задач геометрического моделирования, UniCalc — для решения систем алгебраических уравнений и неравенств, DAEPACK — для комбинированных вычислений, включающих численные операции и символьные преобразования, и др. Сведения об этих и других специализированных программных средствах, основанных на применении аппарата интервальной математики, можно найти в Интернете на следующей Web-странице:

`http://www.cs.utep.edu/interval-comp/intsoft.html`

Развитие достоверных вычислений и международное сообщество специалистов

Еще один вопрос, который может возникнуть у читателя, желающего серьезней вникнуть в концепцию надежных математических вычислений, воплощенную в PASCAL-XSC: как ознакомиться с публикациями по данной тематике, следить за текущими результатами и установить связь с ведущими специалистами? Иначе говоря, как включиться в профессиональную жизнь сообщества специалистов?

Для отражения текущего состояния исследований служат прежде всего периодические издания. Публикации по интервальным вычислениям присутствуют во многих журналах, посвященных вычислительной математике и использованию численных методов в прикладных областях, однако есть, разумеется, и специализирующиеся на интервальной тематике. Так, с конца 70-х гг. до 1988 г. во Фрайбурге (ФРГ) выходило единственное в мире периодическое издание, посвященное исключительно интервальным вычислениям: «Freiburger Intervall-Berichte». В 1991 г. эстафету от него принял журнал «Интервальные вычисления/Interval Computations», выходивший в России первоначально на русском и английском языках. В 1995 г. журнал был переименован в «Reliable Computing», и языком публикаций стал исключительно английский. В 1997 г. журнал перешел под управление крупнейшего голландского научного издательства

Kluwer Academic Publishers и постепенно превратился в ведущее издание международного профессионального сообщества. Подробнее о журнале можно узнать, обратившись к соответствующему разделу сайта Kluwer Academic Publishers. Вот начальная страница с описанием журнала:

http://www.wkap.nl/journalhome.htm/1385-3139

Обратим внимание, что через сайт можно заказать и бесплатно получить в качестве образца один из выпусков. Кроме того, следует иметь в виду, что в связи с российским происхождением журнала для подписчиков из России (и СНГ) установлены особые, сниженные, цены, делающие журнал доступным для отечественных организаций и частных лиц. Подробнее об условиях подписки (и о получении прежних номеров) можно узнать, написав по одному из адресов:

rc-journal@mtu-net.ru, nest@into.nit.spb.su

Можно также воспользоваться обычной почтой, послав письмо по следующему адресу:

Россия, 195256, С.-Петербург, а/я 52,
редакция журнала «Reliable Computing».

Существует также академическое периодическое издание, публикующее статьи преимущественно на русском языке и систематически отражающее тематику интервальных вычислений, — журнал «Вычислительные технологии» (Новосибирск). Познакомиться с этим журналом можно с помощью следующего сайта:

http://www-sbras.nsc.ru/win/mathpub/comp-tech

Развитие интервального подхода и оформление его в самостоятельное направление на стыке вычислительной математики и компьютерного дела породили потребность в проведении специально посвященных ему научных мероприятий. Первое международное мероприятие по данной тематике — *Symposium on Interval Analysis* — состоялось в 1968 г. в Великобритании. В дальнейшем крупнейшими мероприятиями подобного рода были проходившие в 1975, 1980 и 1985 гг. в ФРГ симпозиумы под одинаковым названием *International Symposium on Interval Mathematics*. В 1982 г. международный симпозиум по данной тематике провела фирма IBM: он проходил в США и назывался *A New Approach to Scientific Computation*. После 1985 г. международные мероприятия, в значительной степени или целиком посвященные «надежным» вычислениям, стали проводиться примерно ежегодно. Из их числа отметим *Reliability in Computing; International Workshop on the Role of Interval Methods in Scientific Computing* (США, 1987), *International Symposium on Computer Arithmetic and Self-Validating Numerical Methods* (Швейцария, 1989), *International Symposium on Computer Arithmetic,*

Scientific Computation and Mathematical Modelling (Болгария, 1990). С 1992 г. раз в два года проводятся международные конференции *INTERVAL'XX*; с той же периодичностью проходят симпозиумы *SCAN-XX* (*IMACS/GAMM International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics*).

Многие страны регулярно проводят национальные научные встречи по данной тематике. В России это Интервальные конференции, проводимые в последние годы на базе Института вычислительных технологий СО РАН, — *см.* сайт

http://www.ict.nsc.ru

Кроме того, результаты по исследованию и использованию интервальных вычислений постоянно докладываются на мероприятиях смежных областей, таких как компьютерная алгебра, автоматическое управление и т. д.

Одним из крупнейших в мире центров систематических исследований в области интервальных вычислений, особенно активно занимающимся развитием компьютерного направления, является Институт прикладной математики Университета г. Карлсруэ, в котором, как уже было сказано, разработан и язык PASCAL-XSC.

В России интервальные вычисления как самостоятельное компьютерно-математическое направление начали развиваться с середины 70-х годов. Отечественная школа интервальной математики имеет свои уникальные достижения, международное признание получили некоторые компьютерные разработки (например, пакет UniCalc), есть определенные результаты и в прикладных областях. Библиография опубликованных на русском языке «интервальных» работ [14] уже в 1994 г. насчитывала 547 позиций.

О мировых научных и практических достижениях в сфере «надежных» вычислений, включая информацию о публикациях и программных средствах, можно узнать с помощью сайта, который поддерживает проф. В. Крейнович из Техасского университета в г. Эль-Пасо (США). Начальная страница сайта следующая:

http://www.cs.utep.edu/interval-comp

Наиболее оперативно о профессиональных новостях можно узнавать, подписавшись по электронной почте на рассылочный список reliable_computing. Чтобы это сделать, пошлите сначала сообщение, состоящее лишь из слова help, на адрес:

majordomo@interval.louisiana.edu

После этого достаточно будет воспользоваться полученной инструкцией.

Надеемся, что после обращения к перечисленным источникам недостатка информации вам испытывать не придется.

Благодарности

В заключение хотелось бы выразить глубокую благодарность директору Института прикладной математики (II) Университета г. Карлсруэ проф. д-ру У. Кулишу за всестороннюю и многообразную помощь в издании и переиздании русского перевода настоящей книги. Без этой помощи замысел рассказать русскоязычному читателю об интереснейшем языке программирования PASCAL-XSC и воплощенной в нем культуре достоверных вычислений наверняка остался бы неосуществленным. Мы благодарим также других своих коллег из того же института, способствовавших выходу книги в свет, и прежде всего д-ра Г. Болендера, оказавшего немалое содействие в ее доработке и переиздании. Кроме того, хотелось бы поблагодарить д. ф.-м. н., гл. редактора международного журнала «Relable Computing» В. М. Нестерова и проф. В. Крейновича из Техасского университета в г. Эль-Пасо (США) за внимательное прочтение рукописи настоящей статьи и ценные замечания по ее содержанию, а также выразить признательность д. ф.-м. н., вед. науч. сотр. Института вычислительных технологий СО РАН (Новосибирск) С. П. Шарому за предоставленную полезную информацию.

Д. В. Ширяев,
Кэмпбелл (Калифорния, США)
Dmitri.Chiriaev@eng.sun.com

А. Г. Яковлев,
Москва
Yakovlev@rc-journal.mtu-net.ru

Литература

1. *Wilkinson J. H.* Modern error analysis // SIAM Rev. — 1971. — Vol. 13, № 4. — P. 548—568.
2. *Nickel K.* Can we trust the results of our computing? // Mathematics for Computer Science; Proc. Symposium held in Paris, March 16—18, 1982. — S. l.; Association francaise pour la cybernetique et technique (AFCET), 1982. — P. 167—175
3. *Moore R. E.* Interval analysis. — Englewood Cliffs; Prentice Hall, 1966. — 145 p.
4. *Moore R. E.* Methods and applications of interval analysis. — Philadelphia; SIAM, 1979. — xi, 190 p.
5. *Alefeld G., Herzberger J.* Introduction to interval computations. — New York etc.; Academic Press, 1983. — xviii, 333 p.; Рус. перев.; *Алефельд Г., Херцбергер Ю.* Введение в интервальные вычисления: Пер. с англ. — М.: Мир, 1987. — 356 с.
6. *Добронец Б. С., Шайдуров В. В.* Двусторонние численные методы. — Новосибирск; Наука, 1990. — 208 с.
7. *Yohe J. M.* Portable software for interval arithmetic // Fundamentals of numerical computation (computer — oriented numerical analysis) / Ed.: G. Alefeld, R. D. Grigorieff. — Wien etc.: Springer-Verlag, 1980. — (Computing; Suppl. 2). — P. 211—229.
8. *Rump S. M.* Introduction to ACRITH — accurate scientific algorithms // Computerarithmetic, Scientific Computation and Programming Languages / Ed.: Kaucher E., Kulisch U., Ullrich Ch. — Stuttgart: B. G. Teubner Verlag, 1987. — P. 296—369.
9. *Klatte R., Kulisch U., Wiethoff A., Lawo C., Rauch M.* C-XSC. A C++ class library for extended scientific computing. — Berlin etc.: Springer Verlag, 1993. — xvi, 270 p.
10. *An American national standard. IEEE standard for binary floating-point arithmetic: ANSI/IEEE Std 754—1985.* — Approved March 21, 1985. IEEE Standards Board; Approved July 26, 1985. American National Standards Institute. — New York (N.Y.): IEEE, 1985. — 18 p. — Перепечатано в SIGPLAN Not. — 1987. — Vol. 22, № 2. — P. 9—25.
11. *Йенсен К., Вирт Н.* Паскаль: руководство для пользователя. — М.: Финансы и статистика, 1989. — 255 с.
12. *Нестеров В. М.* Твинные арифметики и их применение в методах и алгоритмах двустороннего интервального оценивания: Автореф. дис. ... докт. физ.-мат. наук. — Санкт-Петербург, 1999. — 34 с.
13. *Калмыков С. А., Шокин Ю. И., Юлдашев З. Х.* Методы интервального анализа. — Новосибирск; Наука, 1986. — 224 с.
14. *Dobronets B. S., Kearfott R. B., Kupriyanova L. V., Yakovlev A. G., Zyuzin V. S.* Bibliography of works on interval computations published in Russian. — St. Petersburg, Moscow: Institute for New Technologies, 1994. — (Reliable Computing; Suppl. 1).