6. *Затылкин А.В.* Синтез системы управления интеллектуальной компьютерной обучающей системой / Затылкин А.В., Кемалов Б.К., Юрков Н.К. // Новые промышленные технологии. – 2011. – № 2. – С. 24-29.

Статью рекомендовал к опубликованию д.т.н., профессор А.А. Зори.

Затылкин Александр Валентинович
Пензенский государственный университет.
E-mail: oldalez@yandex.ru.
440026, г. Пенза, ул. Красная, 40.
Тел. 88412368212.

Буц Виктор Петрович
E-mail: butsvp@mail.ru.

Юрков Николай Кондратьевич
E-mail: yurkov_nk@mail.ru.

Zatylkin Alexander Valentinovich
Penza State University.
E-mail: oldalez@yandex.ru.
40, Krasnaya Street, 440026, Penza, Russia.
Phone: +78412368212.

Buts Viktor Petrovich
E-mail: butsvp@mail.ru.

Yurkov Nikolay Kondratievich
E-mail: yurkov_nk@mail.ru.

УДК 681.3.06

Б.А. Державец

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ ДЛЯ МАТЕМАТИКОВ ИЛИ МАТЕМАТИКА ДЛЯ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Более 35 лет ведутся активные дискуссии с участием представителей академической науки и ИТ-практиков: является ли математика основой программирования? Их точки зрения подчас очень сильно разнятся – одни считают, что в основу информационных технологий легла математика, другие утверждают, что информационные науки – это самостоятельное направление и математика является лишь равноценным партнером. В статье рассматривается вопрос и прослеживается процесс взаимопроникновения математики и информационных технологий.

Математика; программирование; информационные технологии.

B.A. Derzhavets

INFORMATION TECHNOLOGIES FOR MATHEMATICIANS OR THE MATHEMATICS FOR INFORMATION TECHNOLOGIES

More than 35 years are conducted active discussions with participation of representatives of the academic science and IT-experts : whether is the mathematics a programming basis? Their points of view it is sometimes very strongly separated – one consider that in a basis of information technology the mathematics has laid down, others assert that information sciences is an indepen-

dent direction and the mathematician is only equivalent partner. In article the question is considered and process interosculation of mathematics and information technology is traced.

Matematics; computer programming; IT.

Уже более 35 лет не прекращаются попытки любой ценой заложить математические основы в науку программирования и схемотехнические разработки (создание принципиальных схем электронных узлов, блоков, приборов и комплексов). По убеждению профессора механико-математического факультета МГУ им. М.В. Ломоносова А. Михалева, понимание и знание информационных технологий напрямую зависят от объема фундаментальных математических знаний.

Но, как известно, в областях инженерной деятельности процесс математизации не только не является самоцелью, но и не рассматривается как панацея в борьбе с растущей сложностью решаемых инженерных задач. Математический аппарат является лишь одним из инструментов, нередко весьма важным, но, как правило, не основным при решении проектных, конструкторских и технологических задач [1].

Чтобы убедиться, что это действительно так, достаточно, например, рассмотреть ситуацию, сложившуюся в радиотехнике, электронике и связи – исторически новых инженерных дисциплинах, которые, с одной стороны, ненамного старше программирования, а с другой – «родились из уравнения» [1] Максвелла.

Один из наиболее близких к программированию аспектов этих новых инженерных дисциплин – схемотехнические разработки. Преобразование сигналов активными и пассивными компонентами электронных цепей: тип преобразования; последовательность выполнения различных процедур преобразования; ветвления, в том числе условные; циклы (регенеративные процессы) и многие другие операции, выполняемые «электронными операторами» над носителем информации – электрическим сигналом – отражаются в принципиальной схеме так же, как в тексте программы отражаются операции над цифровыми аналогами этого сигнала.

Конечно, тут нет полной аналогии. Кроме разработки топологии схемы, радиоинженер должен еще рассчитать энергетические нагрузки на «операторах программы» (компонентах схемы), решить вопросы их унификации и стандартизации и многие другие, связанные с преобразованием принципиальной схемы (текста программы на «языке схем») в технологичную в производстве, надежную и экономичную в эксплуатации инженерную конструкцию (то, что в программировании стали называть: отчужденный от разработчика программный продукт).

Все эти фазы технологического цикла разработки электронного устройства – материального объекта для преобразования информации и программы – информационного объекта (нередко создаваемого для той же цели) выполняются пока по существенно различным для этих отраслей законам промышленного производства.

Однако процесс сближения этих двух ветвей информационной технологии неотвратимо развивается. Во всяком случае, нельзя, видимо, считать случайным тот факт, что аппарат блок-схем, модульность, структурирование и многие другие используемые в программировании для борьбы с «проклятием размерности» [1] технологические приемы и средства были навеяны аналогичными решениями, найденными для той же цели радиоинженерами в начале века и активно используемыми в схемотехнических задачах радиотехники и электроники.

С началом микропроцессорной революции граница между аппаратными и программными средствами преобразования информации начала расплываться. Инженер-электронщик, который еще в начале 60-х годов, когда программисты уже пользовались преимуществом языка высокого уровня, писал свои программы (принципиальные схемы) на языке электронных схем уровня ниже Ассемблера (дискретные компоненты), к концу 70-х гг. с появлением интегральных схем (ИС) повысил этот уровень до эквивалента ассемблера. Затем уже к середине 70-х гг.

стали широко доступны большие интегральные схемы (БИС), которые поднимают уровень языка схем разработчика электронной аппаратуры до сопоставимого с языком программирования высокого уровня, и, наконец, на рубеже 80-х годов массовыми компонентами электронного оборудования, какими в начале 60-х гг. были транзисторы, становятся микропроцессорные наборы, а затем и однокристальные микроЭВМ.

И в начале 80-х гг. исчез барьер, отделяющий технологию создания средств аппаратного преобразования информации от программирования, когда массовыми компонентами электронного оборудования становятся микропроцессорные наборы и по мере того как основные функции нижнего уровня компонентов электронных схем – микропроцессоров оказывается необходимым задавать программным путем. И уже только конкретные условия поставленной разработчику инженерной задачи определяют какую часть схемотехнического решения «паять», а какую «программировать».

Каким математическим аппаратом реально располагает инженер-электронщик, разрабатывающий цифровые системы на элементной базе от интегральных схем до микропроцессорных наборов, т.е. на том уровне, где до сих пор раздельно развивавшиеся технологии электронных схем и программирования начинают, наконец, пересекаться и непосредственно взаимодействовать еще в процессе проектирования средств электронной обработки информации?

Э. Клингман, президент исследовательской фирмы Cybernetic Micro Systems и автор монографии «Проектирование микропроцессорных систем», подчеркивает, что цифровые блоки на интегральных схемах «трудно описать математически. При проектировании комбинационных схем можно использовать полуматричные методы такие, как метод карт Карно или методы Квайна и Мак-Класки, но область их применения в значительной степени ограничена. При введении сложных временных соотношений эти методы становятся бесполезными. Диаграммы переходов являются превосходным инструментом анализа систем, однако с точки зрения проектирования цифровых стандартных блоков, они оставляют желать много лучшего. Были разработаны классические методы минимизации количества вентилей, но стоимость вентилей сейчас невелика и продолжает снижаться так, что целесообразность работы в этом направлении становится сомнительной... Проработав в течение десяти лет в области математической физики, я могу себе представить, что это утверждение может обескуражить некоторых читателей, полагающих, что всякая область знаний, не имеющая математической основы, сомнительна. Среди тех, кто понимает красоту и мощь строгого математического утверждения, существует тенденция упускать из виду некоторые ограничения, присущие математическому языку в отличие от естественного. Во многих областях математика становится настолько трудной для понимания, что в качестве вспомогательного средства в интуитивном анализе используют диаграммные методы. В течение длительного периода это было характерно для таких областей, как химия и электротехника, но в последнее время и в теоретической физике появились диаграммы Фейнмана и Голдстоуна. В большинстве случаев при проектировании цифровых систем достаточно использовать диаграммы логических схем, дополняемые описаниями на естественных языках. Некоторые фирмы и университеты используют в системах машинного проектирования более мощные полуматематические методы, которые, по всей видимости, получат дальнейшее развитие» [3].

Математика является самой абстрактной из наук и стоит особняком. Но по убеждению А. Михалева, математика помогает информатике формировать ее теоретическое ядро. Хотя, многие специалисты в области прикладной математики

неоднократно подчеркивали уязвимость концепции глобальной математизации науки программирования.

Еще в 70-х гг. Р.В. Хемминг «заложил два краеугольных камня в основание науки программирования» [1]:

1) цель машинной обработки – понимание, а не числа;
2) прежде чем решать задачу, подумай, что делать с ее решением.

В своей книге «Числовые методы» Р. Хемминг указывает на две классические работы, посвященные постановке математических задач: «Метод» Архимеда и «Как решать задачу» Пойа и подчеркивает причину принципиальной несводимости прикладного программирования к решению математических задач: «Однако оба этих автора занимались решением точно сформулированных задач, тогда как нам интересно, что делать, когда задача поставлена нечетко и столь же неопределены условия, которым должны удовлетворять результаты» [4].

В эпоху расцвета «культа центрального процессора» [1], когда только формировалось ядро элиты профессиональных программистов – системные программисты, которым предстояло разработать операционные системы и организовать пакетный режим обработки данных, Хемминг убеждал: «Если ставится задача понять физическое явление, то автор задачи должен понимать и контролировать процесс обработки данных... Опыт показывает, – разъясняет он эту точку зрения, – что обычно и легче и лучше научить специалиста в конкретной области и математике, и программированию, чем наоборот. Но если мы требуем этого от заказчиков, то долг вычислителей приложить все усилия к тому, чтобы уменьшить для них трудности обучения. Произвольные правила, особый жаргон, бессмысленный формализм, изменения в методах и обозначениях, препятствия в получении времени – все это должно быть сведено к минимуму» [4]. Думается, и сегодня трудно дать более четкое обоснование необходимости организации режима персональных вычислений и условий для его реализации.

Хотя, по своему содержанию книга Р. Хемминга в основном посвящена, казалось бы, лишь вычислительным, т.е. наиболее математическим, аспектам программирования, однако всякий раз, когда Хемминг как математик ставит в ней ключевые вопросы организации вычислительного процесса, он подчеркивает, что ответы на них «должны быть найдены в первоначальной задаче, а не в математических трактатах или даже книгах по численному анализу» [4].

Сейчас идет сложный процесс преобразования ИТ не просто в отдельную область человеческой деятельности, но и в самостоятельное научное направление. Но проектирование, разработка конструкции и технологии изготовления материальных и информационных объектов до сих пор были и все же останутся областью инженерного искусства. Возможно, что область приложений математических методов в инженерных задачах проектирования и конструирования информационных объектов (программировании) будет расширяться. Как и представители других инженерных дисциплин, программисты получат формальные способы автоматического синтеза некоторых типов программных конструкций, т.е. получат типовой для инженера-конструктора традиционных отраслей техники набор формальных методов решения рутинных задач конструирования. Однако не следует ждать появления формального «ответа на вопрос о том, как написать любую программу, – ответа на этот вопрос вообще не существует» [2].

После того как выполнены трудоемкие исследования предметной области и завершился сложнейший этап постановки задачи, программист, как и любой инженер-разработчик, «снова и снова остается один на один со своей собственной задачей: ему нужно составить программу! Выбрать, как именно следует расположить и связать данные в памяти, понять, какая именно последовательность операторов – спо-

собных сделать все, что угодно, и оттого одновременно и податливых, и опасных – выполнит поставленную задачу. И как организовать эти операторы в цикл, который будет с каждым шагом приближать машину к намеченной цели» [2].

Э. Любимский вывел универсальную для всех без исключения инженерных дисциплин формулу инженерного творчества: «Выбрать, понять, изобрести, проверить, усомниться и повторить все сначала». Таким образом, нет и не может быть формальной теории, которая сведет работу программиста, занятого конструированием широкого класса информационных объектов, к решению формально поставленных математических задач. То есть происходит основное направление эволюции программирования как инженерной дисциплины – от математических задач первого этапа информационной технологии к широкому кругу инженерных задач проектирования, конструирования, технологии изготовления и промышленной эксплуатации информационных объектов. Примером проникновения в программирование принципов, методов и средств инженерного искусства является САПР (системы автоматизации проектирования).

Одно из направлений такого влияния исследуется, естественно, наиболее активно. Это характер влияния инструмента – САПР на работу проектировщика. Влияние задачи (САПР) на инструмент (программирование) привлекает пока значительно меньшее внимание. Между тем, как заметил Э. Дейкстра, «сфера применения может быть такой же революционизирующей, как и средство» [5].

В настоящее время САПР относится к числу наиболее быстро растущих секторов индустрии ЭВМ, уступая по темпам лишь сектору персональных ЭВМ. Поэтому можно ожидать, что воспринимаемые здесь законы и технология промышленного производства информационных объектов в самом недалеком будущем начнут оказывать обратное влияние на процесс создания программ, в том числе и за пределами этой области приложений.

Ежегодно во всем мире выдаются десятки тысяч патентов и авторских свидетельств на схемотехнические изобретения в различных областях применений электрорадиотехники, промышленной электроники, автоматики и связи. Общее число вновь создаваемых принципиальных схем, определяющих функционирование различных устройств преобразования информации, измеряется миллионами. Однако до сих пор в этих страстях информационной техники значительно реже можно услышать такие страстные призывы к созданию формального аппарата для автоматического синтеза новых принципиальных схем, какие постоянно раздаются по поводу синтеза программ из храма большой науки программирования.

Математическая теория связи, как известно, позволяет радиоинженеру оценить предельную пропускную способность канала связи, надежность передачи сообщений в условиях заданного отношения сигнал/шум и другие технические характеристики проектируемой системы. Однако разработка необходимого способа кодирования сообщений, не говоря уже о принципиальной схеме устройства, остается и в этой «густоматематизированной» отрасли целиком в зоне ответственности инженерного искусства. Теория показывает здесь границы достижимого, но, как правило, не даст практических рекомендаций по достижению этих границ.

Проектирование, разработка конструкции и технологии изготовления материальных и информационных объектов до сих пор были и останутся, все же областью инженерного искусства.

Как видим, стоит говорить о не математической основе программирования, а о взаимопроникновении математики и информационных технологий. На современном этапе одним из основных постановщиков задач для математики является информатика (как в свое время это делали физика, география, химия, астрономия в периоды своего становления). Примером проблемы из области ИТ, представляющей большой интерес с точки зрения математики, служат параллельные вычисления.

Идея параллельных вычислений, как известно, возникла с появлением первых ЭВМ. В начале 50-х гг. американский математик Дж. фон Нейман кроме архитектуры последовательной ЭВМ, разработал также принцип построения процессорной матрицы, в которой каждый элемент был соединен с четырьмя соседними и имел 29 состояний. Он теоретически показал, что такая матрица может выполнять все операции, поскольку она моделирует поведение машины Тьюринга. Практическая реализация основных идей параллельной обработки стала возможна только тогда, когда инженерные науки достигли определенного уровня развития, т.е. с появлением в 60-х годах транзистора, который благодаря малым размерам и высокой надежности по сравнению с электронными лампами позволил строить машины, состоящие из большого количества логических элементов, что принципиально необходимо для реализации любой формы параллелизма. Появление параллельных ЭВМ с различной организацией (конвейерные ЭВМ, процессорные матрицы, многопроцессорные ЭВМ) вызвано различными причинами. Совершенствование же этих ЭВМ происходило по внутренним законам развития инженерных наук.

На сегодняшний день параллельные вычисления стали доминирующей парадигмой в архитектуре компьютеров, в основном в форме многоядерных процессов [6]. После 2004 г., когда стало ясно, что дальнейший быстрый род тактовой частоты ЦПУ проблематичен, началось быстрое развитие технологии многоядерных процессоров. Техника параллельных вычислений стала крайне актуальной в связи с широким внедрением виртуализации и cloud computing.

Ресурс увеличения плотности активных элементов на чипе еще сохранялся и именно это сделало многоядерные процессоры привлекательными для повышения производительности вычислений. В настоящее время число ядер удваивается каждые 18 месяцев.

Многоядерность упрощает проблему работы машин с сетевыми интерфейсами 10-100 Гбит/с, так как имеется возможность разделить обработку потока на несколько субпотоков, поступающих на отдельные ядра. Если 100 гигабитный поток разделить на 10 субпотоков, а процессор имеет 64 разряда, то будет достаточно иметь тактовую частоту > 350 МГц. Существует несколько подходов в применении многоядерных процессоров:

1. **Ппараллельность на уровне инструкций.** Обычно неэффективно для бизнес-приложений, но вполне приемлемо для тяжелых вычислительных задач.
2. **Виртуализация.** Техника хороша для объединения серверов, работающих в рамках ОС Linux или Windows.
3. **Hadoop (Apache).** Эта технология распараллеливания по данным весьма хорошо масштабируема для больших объемов однородной информации, пригодна для работы на большом числе процессоров. В настоящее время использует мультиядерность не в полной мере.
4. **DataRush.** Этот программный продукт предлагает достаточно гибкий и эффективный подход для одиночных серверов с большим числом чипов и ядер. Применим для объемов данных в десятки терабайт, легко осваивается программистами.

Кроме параллельных вычислений, для математики представляют большой интерес проблемы из области ИТ такие, как эффективность загрузки многопроцессорного комплекса, поиск математических алгоритмов, поддающихся распараллеливанию, разработка программного обеспечения для реализации параллельных вычислений – все это серьезные научные проблемы. Имеется еще много других примеров взаимопроникновения математики и ИТ и поэтому необходимо сохранять неразрывную связь этих двух областей – если эти дисциплины разойдутся, думается, серьезные потери понесут обе.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Громов Г.Р.* Математика в программировании. – М., 1994.
2. *Катков В.Л., Любимский Э.З.* Программирование. – Минск: Высшая школа, 1999.
3. *Клигман Э.* Проектирование микропроцессорных систем / Пер. с англ. – М.: Мир, 1985.
4. *Хемминг Р.В.* Числовые методы. – М.: Наука, 1972.
5. *Дейкстра Э.* Дисциплина программирования / Пер. с голланд. – М.: Мир, 1978.
6. Krste Asanovic et al. The Landscape of Parallel Computing Research: A View from Berkeley. University of California, Berkeley. Technical Report No. UCB/EECS-2006-183. December 18, 2006: «Old [conventional wisdom]: Increasing clock frequency is the primary method of improving processor performance. New [conventional wisdom]: Increasing parallelism is the primary method of improving processor performance… Even representatives from Intel, a company generally associated with the „higher clock-speed is better“ position, warned that traditional approaches to maximizing performance through maximizing clock speed have been pushed to their limit».
7. *David A.* Patterson and John L. Hennesy. Computer Organization and Design (Second Edition) Morgan Kaufmann Publishers, 1998. ISBN 1558604286. – P. 715.
8. Asanovic et al: Old [conventional wisdom]: Power is free, but transistors are expensive. New [conventional wisdom] is [that] power is expensive, but transistors are «free».
9. *Yale P.* The Microprocessor Ten Years From Now: What Are The Challenges, How Do We Meet Them? (wmv). Distinguished Lecturer talk at Carnegie Mellon University, April 2004. Retrieved on November 7, 2007.

Статью рекомендовал к опубликованию д.т.н., профессор В.А. Петраков.

Державец Борис Абрамович
Ростовский институт Российского государственного торгово-экономического университета.
E-mail: dba477@list.ru.
344002, г. Ростов-на-Дону, ул. Тургеневская, 49.
Тел.: 88632675287.

Derzhavets Boris Abrumovich
The Rostov Institute of the Russian State Trade and Economic University.
E-mail: dba477@list.ru.
49, Turgenevskya Street, Rosotov-on-Don, 344002, Russia.
Phone: +78632675287.

УДК 616-71; 616.711-007.55

С.Л. Щербин, З.А. Коков, С.А. Синютин, А.Т. Коков, С.М. Щербина

ОПРЕДЕЛЕНИЕ РАЗНИЦЫ ДЛИН НИЖНИХ КОНЕЧНОСТЕЙ И АСИММЕТРИЧНОГО ПОЛОЖЕНИЯ СЕДАЛИЩНЫХ БУГРОВ ПРИ СТАТИЧЕСКОМ СКОЛИОЗЕ ПРИНЦИПИАЛЬНО НОВЫМИ ФИЗИЧЕСКИМИ СПОСОБАМИ

В статье рассматриваются новые функциональные методы определения разницы длин нижних конечностей и асимметричного положения седалищных бугров для определения высоты корректора, назначаемого под стопу или проекцию седалищного бугра при восстановительной коррекции статических сколиозов. Приведены результаты ЭМГ исследования сокращающихся паравертебральных мышц на электронейромиографе. Установлено, что если при электромиографическом исследовании в рефрактерном периоде появляется напряжение паравертебральных мышц в виде осцилляций с амплитудой порядка ±50 и выше мкВ, то это говорит об истинной разнице длин нижних конечностей, а отсутствие осцилляций – о функциональной разнице.

Функциональная диагностика; электромиография; восстановительная коррекция.