

М. А. Плаксин,

Национальный исследовательский университет «Высшая школа экономики», Пермский филиал

О ПРОПЕДЕВТИКЕ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ В ШКОЛЬНОЙ ИНФОРМАТИКЕ

Аннотация

Необходимость в методике преподавания темы «Параллельные вычисления» в средней школе назрела. В статье перечислены некоторые теоретические проблемы, связанные с переходом от понятия последовательного алгоритма к параллельному, представлен трехлетний опыт работы автора в этом направлении: методический подход, отбор материала, деловые игры и их апробация на школьниках и взрослых, опыт решения задач на параллельные вычисления на конкурсе «ТРИЗформашка», классы задач, примеры задач, программные исполнители, тексты для пропедевтического учебника информатики.

Ключевые слова: информатика, средняя школа, начальная школа, методика обучения, параллельное программирование, параллельные вычисления, пропедевтика, ТРИЗформатика, ТРИЗформашка.

Каждый понимает, как править четверкой лошадей, но как управлять тысящей цыплят, уже не знает никто — а нам предстоит управлять сотнями миллионов муравьев.

Вл. В. Воеводин* [2]

Современный этап развития computer science связан с массовым распространением параллелизма вычислений. Параллелизм развивается на всех уровнях. В мобильных устройствах и персональных компьютерах работают многоядерные процессоры, на видеокартах — графические ускорители. От-

дельные компьютеры для решения трудоемких задач объединяются в вычислительные кластеры (которые могут включать в себя сотни единиц)**. Растет парк многопроцессорных ЭВМ, вершиной которых являются суперкомпьютеры. По данным одного из ведущих российских специалистов в этой области Вл. В. Воеводина, вычислительные мощности, которые сегодня рассматриваются как суперкомпьютерные, примерно за десять лет становятся мощностями настольных компьютеров, а за следующие десять лет — мощностями мобильных приложений [17].

* В данной статье будут упомянуты два представителя блестящей семьи Воеводиных, являющиеся лидерами «суперкомпьютерной науки»: Валентин Васильевич (В. В.) и Владимир Валентинович (Вл. В.). Третий, самый молодой Воеводин, также занимающийся суперкомпьютерами, — Вадим Владимирович (Вад. В.) в статье упоминаться не будет.

В. В. Воеводин (1934–2007) — доктор физ.-мат. наук, профессор, чл.-кор. АН СССР, академик РАН.

Вл. В. Воеводин (1962 г. р.) — доктор физ.-мат. наук, профессор, чл.-кор. РАН, зав. кафедрой суперкомпьютеров и квантовой информатики факультета вычислительной математики и кибернетики МГУ имени М. В. Ломоносова, зам. директора Научно-исследовательского вычислительного центра МГУ.

Вад. В. Воеводин (1986 г. р.) — канд. физ.-мат. наук, ст. научный сотрудник Научно-исследовательского вычислительного центра МГУ.

** На Летней Суперкомпьютерной Академии факультета вычислительной математики и кибернетики МГУ летом 2015 года была озвучена история о том, как в новогодние каникулы с 31 декабря 2005 по 11 января 2006 года для задачи, относящейся к разработке новых лекарств, были связаны в кластер и вели совместные расчеты простаивающие во время каникул компьютерные классы университетов Москвы, Новосибирска, Петербурга, Томска и Челябинска. Общее время вычислений за двенадцать дней составило примерно четыре с половиной года [17].

Контактная информация

Плаксин Михаил Александрович, канд. физ.-мат. наук, доцент кафедры информационных технологий в бизнесе Пермского филиала Национального исследовательского университета «Высшая школа экономики»; *адрес:* 614070, г. Пермь, ул. Студенческая, д. 38; *телефон:* (342) 205-52-50; *e-mail:* mapl@list.ru

M. A. Plaksin,

National Research University Higher School of Economics, Perm Branch

ABOUT PROPAEDEUTICS OF PARALLEL COMPUTING IN SCHOOL INFORMATICS

Abstract

The need for methodics of teaching the theme "Parallel Computing" in secondary school is on the agenda. The article lists some of the theoretical problems associated with the transition from the concept of the sequential algorithm to the concept of parallel and presents a three-year experience of the author in this direction: a methodical approach, the selection of material, business games and their testing on pupils and adults, experience of solving tasks on parallel computing on "TRIZformashka" competition, classes of problems, examples of tasks, program executors, texts for propaedeutic informatics textbook.

Keywords: informatics, secondary school, elementary school, teaching methods, parallel programming, parallel computing, propaedeutics, TRIZformatics, TRIZformashka.

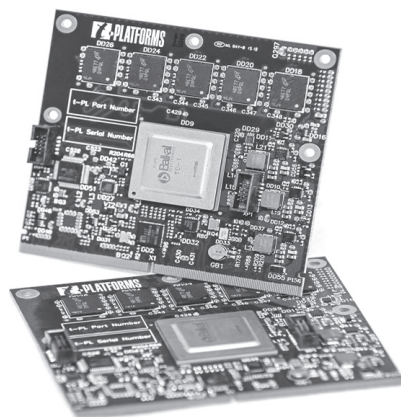


Фото 1. Процессорный модуль стандарта SMARC с российским двухъядерным процессором «Байкал-Т1»
(источник фото: <http://www.t-platforms.ru/products/dev/sf-bt1.html>)



Фото 2. Первый вычислительный кластер ВЦ МГУ в 1999 году объединял 18 процессоров Intel Pentium III.
На фотографии — расширенный 24-процессорный вариант
(источник фото: http://parallel.ru/cluster/photo_history.html)



Фото 3. Суперкомпьютер «Ломоносов»
(источник фото: <http://www.t-platforms.ru/solutions/completed/lomonosov-supercomputer>)

Проиллюстрируем это утверждение. Наверно, самый известный российский суперкомпьютер — это «Ломоносов», запущенный в ноябре 2009 года в Московском государственном университете. Строительство и модификация этого гиганта обошлись более чем в два с половиной миллиарда рублей. Вес аппаратуры составляет около полутора сотен тонн. Площадь собственно машины — около двухсот пятидесяти квадратных метров. Но к этому надо прибавить еще пятьсот квадратных метров под вспомогательное оборудование. Оперативная память — 72 терабайта, дисковая — 162 терабайта. Реальная производительность — 900 терафлопс. Так вот, через каких-нибудь десять лет эта громадина будет размещаться на столе. А еще через десять ее будут носить в кармане.

И все эти мощности хотелось бы использовать с возможно большим КПД.

Массовое распространение параллелизма влечет серьезные последствия, которые еще предстоит выявить и проанализировать. Перечислим некоторые из них.

Необходимость изменения теоретической базы информатики

Сложности возникают в базовых понятиях, на которых строилось преподавание информатики, по

крайней мере, последние двадцать лет: алгоритм, исполнитель, система команд исполнителя, команда.

Современная теория алгоритмов создавалась в расчете на понятие последовательного алгоритма. Каким образом отразится на понятии алгоритма отказ от требования последовательности выполнения шагов?

По крайней мере, последние двадцать лет понятие «алгоритм» вводилось в школе в неразрывной связке с понятием «исполнитель». Для последовательного алгоритма это естественно. Как быть с алгоритмом параллельным? Его выполняет один исполнитель или группа исполнителей? Или нужно вводить специальное понятие «параллельный исполнитель»? Или точнее будет «групповой исполнитель», «коллективный исполнитель»? Ведь различные «одиночные исполнители», совместно выполняющие некоторую работу, могут действовать не одновременно.

Для конкретности в качестве примера рассмотрим компьютерную обучающую программу «Танковый экипаж» [15]. В этой программе от учащегося требуется запрограммировать действия экипажа танка, состоящего из трех человек: наводчика, водителя и заряжающего. Каждый из них имеет свою систему команд. Для того чтобы выполнить боевую задачу (поразить все цели), все члены экипажа должны действовать согласованно. Пример игрового поля программы «Танковый экипаж» см. на рисунке 1.

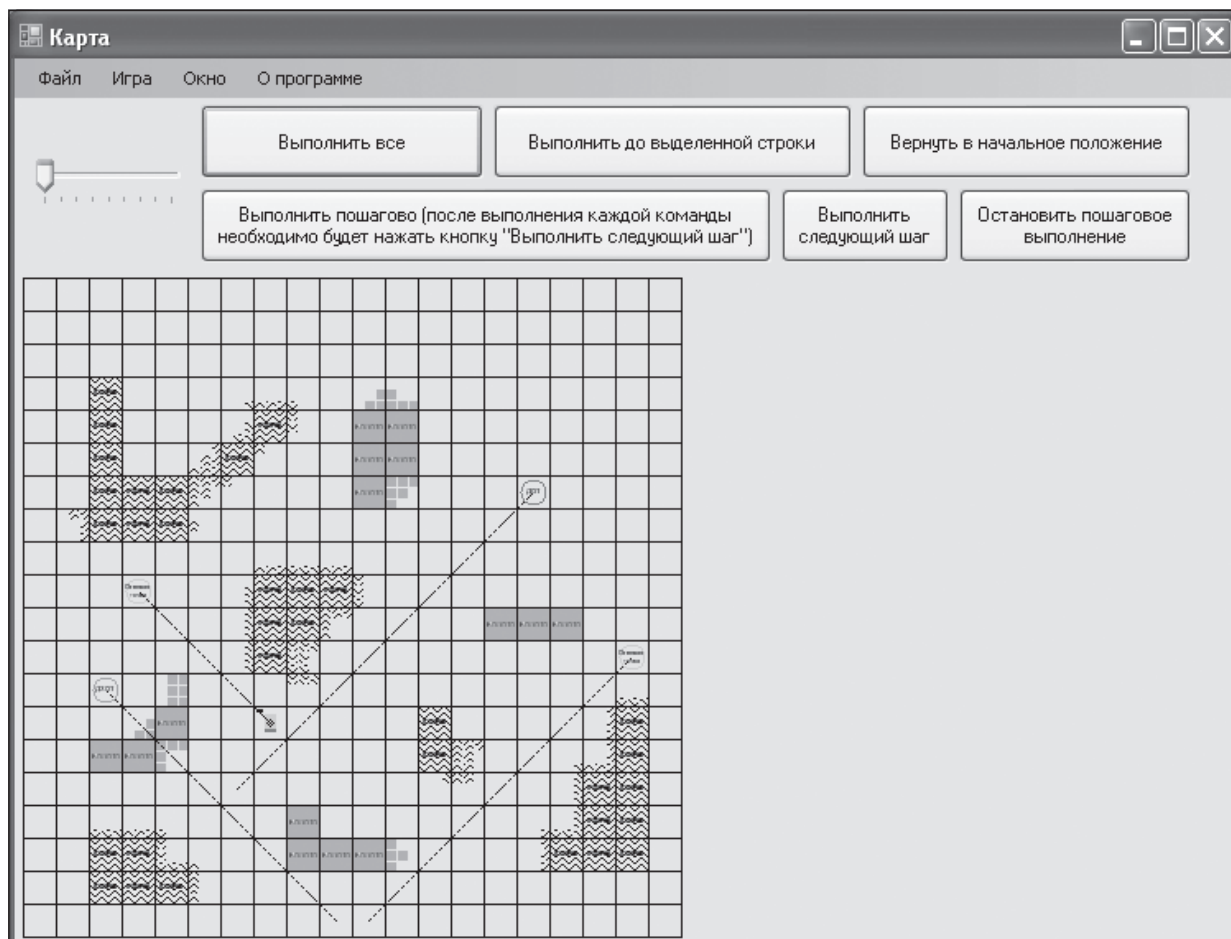


Рис. 1. Фрагмент игрового поля программы «Танковый экипаж»

Вопрос: надо рассматривать этих трех действующих лиц как независимых исполнителей или как три составные части (компоненты? устройства?) одного сложного исполнителя? Для экипажа танка более естественным представляется второй вариант, поскольку ни один персонаж сам по себе выполнить задание не в состоянии. Но как быть, если игра будет усложнена и боевая задача будет поставлена сразу для двух танков? Для трех танков? Трех членов одного экипажа вполне можно рассматривать как три части одного исполнителя. Но каждый экипаж, очевидно, является самостоятельным исполнителем. Значит, параллельный алгоритм для нескольких танков будет выполняться сразу группой исполнителей. Получается, что для параллельного алгоритма рассматривать надо обе возможности: выполнение параллельных действий одним «составным» исполнителем и группой исполнителей. В случае с танковым экипажем границу провести просто. Исполнитель — это тот, кто в состоянии решить поставленную задачу. Этот исполнитель может состоять из нескольких компонент, каждая из которых выполняет некую часть задания, но не может самостоятельно, без помощи других компонент выполнить задание целиком. Но всегда ли разделение «целых исполнителей» и частей сложного исполнителя будет так же просто — сейчас сказать нельзя.

Выделение частей исполнителя, способных к самостоятельным действиям, требует как-то эти части назвать. Причем название должно допускать рекурсию, поскольку действующие части исполнителя сами могут иметь сложную структуру.

Нужно договориться о термине для обозначения группы совместно действующих исполнителей. Термин «команда» не годится, ассоциируется с «системой команд исполнителя» и с «командами центрального процессора». «Коллектив исполни-

телей»? «Бригада исполнителей»? Автор данной статьи в настоящее время склоняется к «группе исполнителей».

Требует доработки традиционное понятие системы команд исполнителя (СКИ) и само понятие команды. Если мы считаем, что три члена танкового экипажа образуют единого исполнителя, то что считать СКИ этого исполнителя? И что считать командой? Или оставить понятие СКИ для каждого персонажа? То есть это уже не система команд *исполнителя*, а система команд *одной из компонент исполнителя* (для которой еще нет названия)?

Понятие команды удобно расширить до «линейки команд». Пример линейек команд танкового экипажа см. на рисунке 2. Однако понятие «линейка команд» хорошо работает только для линейных алгоритмов. В остальных случаях линейки формируются динамически. Изобразить их в виде наглядной статической таблицы невозможно.

При описании команды «коллективного исполнителя» возникают ассоциации с микропрограммированием. Одну команду «танкового экипажа» можно рассматривать как совокупность трех «микрокоманд» — своей «микрокоманды» для каждого члена экипажа. Тогда алгоритм состоит из таких троек.

Получается, что система команд исполнителя — это совокупность всех возможных троек — комбинаций команд членов экипажа. И описывать ее надо принципиально иначе, чем мы описывали до сих пор. А именно:

- 1) структурировать по членам экипажа;
- 2) описать команды каждого члена экипажа;
- 3) описать правила согласования.

Предложенный взгляд на программу срабатывает, если все «микрокоманды» синхронны. А если нет? Синхронность требует, чтобы за одно и то же время выполнялись:

Алгоритм			
Файл			
#	Наводчик	Водитель	Заряжающий
1	Повернуть орудие по часовой стрелке на 45 гра...	Стоп	Зарядить 1
2	Пауза	Стоп	Зарядить 2
3	Огонь!	Повернуться против часовой стрелки на 90 градусов	Зарядить 1
4	Пауза	Вперед	Зарядить 2
5	Огонь!	Стоп	Зарядить 1
6	Пауза	Стоп	Зарядить 2
7	Огонь!	Стоп	Зарядить 1
8	Пауза	Повернуться по часовой стрелке на 45 градусов	Зарядить 2
9	Пауза	Вперед	Пауза
10	Пауза	Вперед	Пауза
11	Пауза	Вперед	Пауза
12	Пауза	Повернуться по часовой стрелке на 45 градусов	Пауза
13	Пауза	Вперед	Пауза
14	Огонь!	Повернуться по часовой стрелке на 45 градусов	Зарядить 1

Рис. 2. Фрагмент программы для «Танкового экипажа» (пример линейек команд)

- 1) все «частные» команды («микрокоманды») членов экипажа;
- 2) переходы каждого члена экипажа от одной «микрокоманды» к другой.

Но и выполнение этих требований обеспечивает синхронность только при линейном алгоритме. Циклы и ветвления ее нарушают. Рассмотрим этот вопрос подробнее.

В случае параллельности требует корректировки понятие «время выполнения алгоритма». При анализе последовательных алгоритмов время понимается как определитель последовательности шагов. Следующий шаг выполняется *после* предыдущего*. Время «физическое» («астрономическое», «реальное»), нужное для выполнения каждого шага, в последовательном программировании роли не играет. Поэтому, например, при построении трассировочной таблицы мы учитываем результаты команд присваивания, но пренебрегаем сравнениями и переходами. При оценке временной сложности алгоритма команды делятся на три группы: присваивания, сравнения и все остальные. Мы вычисляем количество присваиваний и количество сравнений, но пренебрегаем затратами времени на выполнение остальных команд (передач управления). При этом затраты на присваивание и затраты на сравнение оцениваются как постоянные внутри каждой группы, но различающиеся между группами. И поэтому затраты по каждой группе считаются отдельно.

В реальности команды перехода тоже требуют времени. Для последовательного алгоритма пренебрежение этим фактом всего лишь снижает точность оценки временных затрат. Но как только мы переходим к рассмотрению параллельных процессов, пренебрежение временем, потраченным на передачу управления, ставит под сомнение синхронность выполнения параллельных процессов. Время требуется для любых выполняемых исполнителем действий. Синхронизация нескольких процессов требует учета «общего» («физического», «астрономического») времени.

В случае параллельности среди свойств алгоритма выделяется новая практически значимая характеристика: *способность к распараллеливанию*. При этом возникает вопрос о цели распараллеливания. В качестве таковой, естественно, воспринимается ускорение работы. При делении алгоритма на параллельно выполняемые части время выполнения всего алгоритма будет определяться временем выполнения самой длинной части. Значит, делить алгоритм на части надо так, чтобы все они выполнялись за примерно одинаковое время.

Для сравнения времени выполнения разных частей программы требуется некая единица. Она должна, с одной стороны, достаточно точно представлять время выполнения алгоритма, с другой — не требовать чрезмерных затрат по учету. Например, в случае представления алгоритма в виде блок-схемы легко посчитать количество выполнения блоков-дей-

ствий (прямоугольников). Но ведь при продвижении по блок-схеме приходится проходить и через другие блоки: ветвления, заголовки циклов. Их учет возможен, но требует дополнительных усилий.

Отдельного внимания требует следующий факт. Как правило, каждому исполнителю для того, чтобы приступить к выполнению своей части алгоритма, требуются некоторые подготовительные действия. А в конце могут потребоваться некоторые завершающие. И выполнение этих действий также требует времени. И это надо учесть при оценке трудоемкости работы, выделенной каждому исполнителю.

Продолжение и уточнение вопроса о способности к распараллеливанию — вопрос о возможной степени распараллеливания. До какой степени имеет смысл увеличивать количество процессоров при выполнении данного алгоритма?

Ответы на эти вопросы требуют перехода от представления алгоритма в виде традиционной последовательной блок-схемы к ярусно-параллельной форме [3, 4].

На сегодня человечество накопило колоссальный фонд алгоритмов и программ. В подавляющем большинстве они являются последовательными. Что лучше: переписать эти алгоритмы заново уже как параллельные или попробовать распараллелить уже имеющиеся? И какими методами можно решить задачу распараллеливания последовательных алгоритмов?

Вопрос психологам

Тридцать лет назад начинающаяся массовая компьютеризация потребовала увеличения уровня компьютерной грамотности населения. Это привело к введению в школьную программу в 1985 году курса информатики. Но курс информатики в советском (затем в российском) исполнении никогда не сводился к «кнопочной информатике» — к освоению технологии работы с пакетами прикладных программ и компьютерными играми. Он начал изменять стиль мышления подрастающего поколения. В первую очередь это касалось алгоритмичности, точности, строгости. Затем курс информатики вобрал в себя элементы логики и системного анализа. Впоследствии все это значительно упростило распространение так необходимого в XXI веке проектного подхода. Сейчас речь идет о том, что в течение следующего десятилетия параллельные алгоритмы должны стать элементом общей культуры мышления. Вопрос: *каким образом скажется на мышлении следующего поколения освоение понятия параллельного алгоритма, к чему приведет перестройка сознания «на параллельный лад»?*

Необходимость методики массового обучения параллельному программированию

До недавнего времени параллельное программирование было уделом небольшого числа высококвалифицированных системных программистов. Сегодня оно становится частью профессиональной компетен-

* Достаточно сравнить вызов $f(x, y)$ в математике и в программировании. Для математики вопрос о том, какой из параметров вычисляется раньше, какой — позже, просто не имеет смысла. А для программирования это может иметь решающее значение.

ции. Но технология параллельного программирования существенно отличается от технологии традиционного последовательного. В подтверждение этого вслед за Л. Л. Босовой [1] процитируем крупнейшего российского специалиста в области параллельных вычислений В. В. Воеводина [3, с. 150, 155]:

«...Освоение вычислительной техники параллельной архитектуры <...> молодыми специалистами идет с большими трудностями. На наш взгляд, это связано с тем, что знакомство с параллельными вычислениями, как и образование в этой области в целом, начинается не с того, с чего надо бы начинать. К тому же то, с чего надо начинать, не рассказывается ни в каких курсах вообще. Возможность быстрого решения задач на вычислительной технике параллельной архитектуры вынуждает пользователей изменять весь привычный стиль взаимодействия с компьютерами. По сравнению, например, с персональными компьютерами и рабочими станциями меняется практически все: применяются другие языки программирования, видоизменяется большинство алгоритмов, от пользователей требуется предоставление многочисленных нестандартных и трудно добываемых характеристик решаемых задач, интерфейс перестает быть дружественным, и т. п. Важным является то обстоятельство, что неполнота учета новых условий работы может в значительной мере снизить эффективность использования новой и к тому же достаточно дорогой техники».

«Важно лишь, чтобы обучающийся как можно раньше узнал, что существуют другие способы организации вычислительных процессов, а не только последовательное выполнение “операция за операцией”, что на этих других способах строится самая мощная современная вычислительная техника, что только на такой технике удастся решать крупные промышленные и научные задачи и т. д. Важно, в первую очередь, для того, чтобы как можно раньше обратить внимание обучающихся на необходимость критического отношения к философии последовательных вычислений. Ведь именно с этой философией им приходится сталкиваться на протяжении всего образования как в школе, так и в вузе. И именно эта философия мешает пониманию особенностей работы на вычислительной технике параллельной архитектуры».

Сегодня нам нужны методики для массового обучения технологии параллельного программирования. Автор данной статьи считает, что в процессе обучения настало время для переворота в отношениях последовательного и параллельного программирования. До сих пор мы сначала учили последовательному программированию, а потом — распараллеливанию последовательных алгоритмов. Сейчас надо ставить вопрос о том, чтобы *сразу учить параллельному программированию. А последовательный алгоритм рассматривать как некую часть параллельного алгоритма, которая не требует связи с другими его частями*. В настоящее время этот подход положен в основу алгоритмической линии «Пермской версии» пропедевтического курса информатики (авторский коллектив: М. А. Плаксин, Н. Г. Иванова, О. Л. Русакова; рабочее название курса — «ТРИЗформатика») [23, 24].

Массовое распространение параллельной обработки информации делает актуальным перемещение соответствующих понятий в разряд общедоступных и общекультурных. Знакомство с параллельными алгоритмами должно стать частью грамотности так, как это за последнюю четверть века произошло с базовыми понятиями теории алгоритмов. Сделать это можно только одним путем — включением соответствующих тем в школьный курс информатики. Значит, нужна методика начального знакомства с параллельным программированием на уровне средней школы.

Исторически первая попытка включения тематики параллельных вычислений в школьный курс информатики была сделана еще двадцать лет назад. Двадцать лет назад в курсе под названием «Алгоритмика» [1, 9] был описан исполнитель «Директор строительства», который командовал параллельными действиями нескольких бригад, строящих сооружение из блоков прямоугольной и треугольной формы. Более того, для этого исполнителя была создана программная реализация. Увы! Эта замечательная методическая разработка в середине 90-х оказалась не востребована. Она почти на двадцать лет опередила свое время!

Сегодня положение таково, что тематика параллельных вычислений в средней школе в первую очередь оказалась связана с темой суперкомпьютеров. Достаточно сказать, что соответствующий раздел в журнале «Информатика в школе» носит название «Суперкомпьютерное образование в школе». Именно здесь публикуются новые методические разработки [5, 7, 8, 13, 14, 16, 19]. Их авторы зачастую акцентируют внимание учащихся на суперкомпьютерах даже тогда, когда в этом нет необходимости. Такая ситуация имеет как положительные, так и отрицательные стороны.

Среди положительных сторон надо назвать:

- интерес, который вызывает в обществе, в том числе в среде учащихся, тема суперкомпьютеров. Этот интерес повторяет на современном уровне интерес, который полвека назад вызывали большие машины — суперкомпьютеры своего времени;
- организационную поддержку со стороны суперкомпьютерного сообщества. Каждое лето на факультете вычислительной математики и кибернетики МГУ проводится Летняя Суперкомпьютерная Академия [17]. И каждое лето в рамках этой Академии организуется школьный трек для учителей информатики. Обучение проводится бесплатно. Иногородние слушатели обеспечиваются жильем на весьма льготных условиях. На конференциях Russian Supercomputing Days (Москва, сентябрь 2015 года, сентябрь 2016 года) и «Параллельные вычислительные технологии» (Архангельск, март-апрель 2016 года) были организованы школьные секции и мастер-классы для учителей информатики. Последовательная организационная работа привела к выявлению и формированию группы учителей, заинтересованных в продвижении данной тематики;

- наличие яркого харизматичного лидера, каковым является Владимир Валентинович Воеводин — доктор физико-математических наук, профессор, член-корреспондент РАН, заведующий кафедрой суперкомпьютеров и квантовой информатики факультета вычислительной математики и кибернетики МГУ имени М. В. Ломоносова, заместитель директора Научно-исследовательского вычислительного центра МГУ.

Недостаток «суперкомпьютерного» подхода заключается в зауживании тематики параллельных вычислений. Сами суперкомпьютеры школьникам, как правило, недоступны (разве что в крупных городах на них можно поглядеть на экскурсии). Задачи, на решение которых они нацелены, для школьников слишком сложны и зачастую не представляют непосредственного интереса. С точки зрения занятий параллельным программированием суперкомпьютеры являются чем-то запредельным. В то же время практически всем доступны такие параллельные устройства, как многоядерные процессоры и видеокарты с графическими ускорителями. Именно на них, по мнению автора, и стоит делать упор. Для школьника гораздо интереснее (и престижнее в глазах окружающих) заставить работать синхронно пару ядер в процессоре своего мобильного телефона (и продемонстрировать это товарищам и родителям), чем запараллелить несколько сотен или даже тысяч ядер в суперкомпьютере, который расположен в другом городе и который очень мало кто видел воочию.

Из методических разработок, связанных с параллельным программированием, отметим кандидатскую диссертацию М. А. Соколовской по методике обучения будущих учителей информатики основам параллельного программирования [26] и опыт Е. Ю. Киселевой по освоению школьниками технологии CUDA [13].

Предложения по пропедевтике параллельных вычислений в школьной информатике

По мнению автора данной статьи, сосредоточение внимания на суперЭВМ и параллельном программировании существенно обедняет и усложняет тему параллельных вычислений, отвлекает учащихся от множества важных и доступных вопросов. Целью темы «Параллельные вычисления» в средней школе является не обучение «реальному» параллельному программированию (изучение соответствующих языковых конструкций, языков программирования и технологий), а ознакомление учащихся с соответствующим набором понятий и понимание особенностей параллельной работы. Мир вокруг и внутри нас представляет собой сложную параллельную систему. И эта система сама по себе дает массу материала для освоения понятий и механизмов параллелизма. Никакие сложные искусственные конструкции типа технологий MPI и OpenMP для этого не нужны. Школьная информатика должна воспитать мышление, настроенное на «параллельный

лад». А дальше университет пусть закладывает в это мышление профессиональные знания, умения, навыки. В школе акцентировать внимание имеет смысл не на знакомстве с суперкомпьютерами и изучении параллельного программирования, а на освоении механизмов «совместной деятельности», постоянно и широко используемых в жизни.

В курсе предлагается отразить следующие вопросы:

1. Совместная работа нескольких исполнителей (копание канавы несколькими землекопами) и распараллеливание «внутри» одного исполнителя при наличии нескольких обрабатывающих устройств (читаю и ем яблоко). В computer science это будут многомашинный комплекс и многоядерный процессор.
2. Виды параллелизма: параллелизм истинный и псевдопараллелизм (один процессор выполняет частями несколько программ).
3. Исполнители однотипные (землекопы) и разнотипные (экипаж танка).
4. Работы однотипные и разнотипные.
5. Соотношение «исполнители — работы»: 1 исполнитель — 1 работа, 1 исполнитель — N работ (псевдопараллельное выполнение или истинный параллелизм при наличии нескольких обрабатывающих устройств для разных работ), N исполнителей — 1 работа, N исполнителей — N работ.
6. Согласование деятельности исполнителей. Виды согласования: по частям работы, по времени, по результатам деятельности, по ресурсам.
7. Ресурсы. Ресурсы разделяемые и неразделяемые, расходуемые и повторно используемые. Утилизация потребленных ресурсов («сборка мусора» в широком смысле).
8. Выполнение одной и той же работы одним исполнителем и группой исполнителей. Зависимость скорости работы от количества исполнителей. Зависимость стоимости работы от количества исполнителей. Нелинейный рост скорости работы при росте количества исполнителей. Критический путь. Оптимальное количество исполнителей. Оптимальная загрузка исполнителей. Оптимальный порядок действий. Балансировка нагрузки.
9. Конкуренция исполнителей за ресурсы. Блокировка. Клинч (тупик).
10. Механизмы согласования действий исполнителей.
11. Псевдопараллельное выполнение процессов на компьютере (разделение между исполнителями-процессами одного ресурса — процессора).
12. Пригодность алгоритмов к распараллеливанию. Возможная степень распараллеливания. Существование алгоритмов, не поддающихся распараллеливанию.

Отметим, что приведенный список представляет собой частное мнение автора статьи и открыт для обсуждения, дополнения и корректировки. Более того, по мнению автора, было бы очень полезно, чтобы «суперкомпьютерное сообщество» сформу-

лировало «социальный заказ» для школы: какие именно знания-умения-навыки оно хочет видеть в выпускниках школы. Чем выпускник школы «суперкомпьютерного мира» должен отличаться от выпускника сегодняшнего? Будет заказ — будет и результат. Пример. В первый день Russian Supercomputing Days — 2015 в двух докладах прозвучала мысль, что быстродействие современных суперЭВМ определяется не мощностью процессоров (которая находится в центре внимания публики), а быстродействием оперативной памяти. Именно она становится бутылочным горлышком, пропускная способность которого определяет продуктивность всей системы. В результате на второй день конференции участники учительского мастер-класса обкатывали придуманную автором данной статьи игру, демонстрирующую взаимодействие центрального процессора, оперативной памяти и кэш-памяти.

В апреле 2016 года вопрос о социальном заказе был озвучен автором на конференции «Параллельные вычислительные технологии» в Архангельске. После чего автор в личной беседе получил от Вл. В. Воеводина формулировку социального заказа, выраженную в двух фразах: «Давайте жить дружно! Давайте работать быстро!» То есть в пропедевтическом курсе акцент предлагается сделать на двух моментах: организации совместной работы и увеличении производительности за счет распараллеливания алгоритмов.

Порядок и форма изложения материала, предлагаемого к включению в пропедевтический курс параллельных вычислений, — вопрос открытый.

Материал должен быть продемонстрирован на примерах, не связанных с работой ЭВМ. Исполнители должны манипулировать материальными объектами.

Как можно большая часть обучения должна носить характер деловых (организационно-деятельностных) игр.

Выполнение этих требований упростит понимание изучаемого материала. Это будет полезно как при использовании данной методики на уроках информатики в школе (в том числе начальной!), так и при обучении взрослых — учителей информатики и студентов. Школьник, школьный учитель, студент непрофильной специальности смогут остановиться на уровне ознакомления и понимания. Студент-профессионал должен будет сделать следующий шаг и от знакомства перейти к изучению этих механизмов на профессиональном уровне. Но это уже шаг за пределы методики начального ознакомления с темой.

«Пермская версия» пропедевтического курса параллельных вычислений в школьной информатике

Работу над подготовкой методики изучения параллельных вычислений автор данной статьи начал в 2013 году в ходе подготовки конкурса «ТРИЗ-формашка-2013» [6, 10, 11, 12, 15, 21, 22, 25]. («ТРИЗформашка» — межрегиональный интернет-конкурс по информатике, системному анализу и ТРИЗ. Проводится ежегодно во второй половине марта. Возраст участников — с I класса до IV курса.

География — от Владивостока до Риги. Среднее число участников — 100 команд (300 чел.), максимальное — 202 команды (более 600 чел.). Сайт конкурса: <http://www.trizformashka.ru>

Тогда, в 2013 году цели работы были сформулированы следующим образом:

1. В течение двух-трех лет подготовить описание исполнителей, набор игр и задач, связанных с параллельными вычислениями.
2. Предложить их (по частям, ежегодно) участникам конкурса.
3. Проанализировать реакцию участников (оценить количество решавших, их возраст, успешность решения, типичные ошибки, обнаруженные неточности в формулировке задач и т. д.).

Итоги работы за три года выглядят так:

1. Сформулирован подход к изучению темы «Параллельные вычисления»: идти не от проблем computer science, а «от жизни», делать акцент на «совместную деятельность».
2. Сформулирован перечень вопросов, которые предлагается отразить в пропедевтическом курсе параллельных вычислений.
3. Подготовлены элементы методики «Учить сразу параллельному программированию; последовательное — частный случай». Материалы прошли апробацию в четвертом и пятом классах лицея № 10 Перми.
4. Подготовлена глава про параллелизм для новой версии «пермского» учебника информатики для четвертого класса [23, 24]. Материал прошел апробацию в третьих и четвертых классах лицея № 10 Перми. В 2017 году в издательстве «БИНОМ. Лаборатория знаний» запланирован выход новой версии учебника, в которую входит эта глава.
5. Нарботан список типов задач, пригодных для включения в пропедевтический курс параллельных вычислений. Для каждого типа придумано и обкатано несколько задач. Задачи прошли апробацию в конкурсах «ТРИЗформашка» за 2013–2016 годы и/или в начальной школе (на занятиях с учениками третьих-четвертых классов лицея № 10 Перми).
6. Разработана компьютерная игра «Танковый экипаж» [15]. Игра прошла апробацию в трех конкурсах «ТРИЗформашка» (2014, 2015 и 2016 годы).
7. Разработан ряд деловых (организационно-деятельностных) игр [18, 19, 21, 22], в которых отражены следующие вопросы:
 - Необходимость согласования деятельности исполнителей. Связь между исполнителями (по частям работы, по времени выполнения, по результатам деятельности, по требуемым ресурсам).
 - Выполнение одной и той же работы одним исполнителем и группой исполнителей. Зависимость скорости выполнения работы от количества исполнителей. Нелинейное возрастание скорости выполнения работы при увеличении количества исполнителей. Критический путь.

- Оптимальное количество исполнителей. Оптимальная загрузка исполнителей. Оптимальный порядок действий.
- Ресурсы. Конкуренция исполнителей за ресурсы. Блокировка. Клинч (тупик).
- Различные способы управления совместной деятельностью: квантование времени, управление с помощью приоритетов, управление по принципу «мастер — работники».

Игры прошли апробацию в начальной школе (в лицее № 10 Перми) и на ряде мероприятий для учителей. В частности, были представлены на школьном треке Летней Суперкомпьютерной Академии ВМК МГУ в 2014 году, на мастер-классе для учителей на конференциях Russian Supercomputing Days — 2015 и «Параллельные вычислительные технологии — 2016» [21], на нескольких других конференциях (в том числе на конференции «ИТ-Образование — 2015» ассоциации АПКИТ) и других мероприятиях для учителей информатики.

8. Написана и один раз исполнена (пятиклассниками лицея № 10 Перми) пьеса «Многозадачность и многопоточность».
9. Задания на параллелизм начиная с 2013 года вошли в конкурс «ТРИЗформашка» (с 2013 года конкурс имеет подзаголовок «Параллельные вычисления»). Конкурс «ТРИЗформашка» оказался удобной апробационной площадкой, поскольку позволял получить реакцию учащих всех возрастов (от первого класса школы до четвертого курса вуза), из разных регионов, из различных учебных заведений.

Были предложены и опробованы следующие типы задач [20, 21, 22]:

1. Задачи на виды согласования. (Какие виды согласования существуют в школьной столовой?)
2. Игра «Танковый экипаж». Задание на построение параллельного алгоритма.
3. «Параллельный исполнитель» «Стройка» [6]. Одновременно работающие бригады строят сооружение из горизонтальных и вертикальных балок. Задачи включают в себя задания на исполнение указанного алгоритма, на разработку нового алгоритма, на поиск ошибок в заданном алгоритме, на исследование алгоритмов (сравнение сроков строительства по разным алгоритмам, сравнение стоимости строительства, оценка возможности сэкономить за счет перераспределения рабочей силы и др.).
4. Конкуренция за ресурсы. Три поросенка готовят каждый сам себе обед. Для каждого поросенка указано, какие блюда он готовит, какие ресурсы (оборудование, посуда и т. д.) ему для этого нужны и в течение какого времени эти ресурсы должны использоваться. Требуется составить график работы каждого поросенка, если он готовит на кухне один; если они готовят парами; если готовят все трое сразу. Время приготовления пищи должно быть минимизировано.
5. Ярусно-параллельные формы. Их построение и оптимизация. Планирование работ по разным критериям. Даны задание на работу, про-

изводительность работников, правила оплаты. Требуется определить количество работников, нужных, чтобы выполнить работу в заданное время, определить срок работы при заданном количестве работников, определить количество работников, нужное для минимизации стоимости работ.

6. Сетевой график. Дан сетевой график. Требуется изобразить (схематически) сооружение, которое будет построено, определить, сколько дней потребуется для строительства при том или ином числе бригад, какая часть работы будет выполнена к определенному времени.
7. Диаграммы Ганта. Описан текстом план работ по реконструкции цеха: продолжительность и взаимная последовательность действий, требуемые работники. Требуется определить срок сдачи объекта, изменение срока при тех или иных изменениях в рабочей силе, список работников, задействованных на конкретную дату.
8. Согласование повторяющихся работ. Пусть дано задание: в минимальный срок изготовить партию приборов при условии, что каждый прибор должен пройти обработку на разном оборудовании, имеется разное количество оборудования с разной производительностью. Требуется спланировать время начала работы каждого оборудования, минимизировать простои.

В настоящее время в журнале «Информатика в школе» публикуется серия статей с подробным изложением перечисленных материалов.

Литературные и интернет-источники

1. Босова Л. Л. Параллельные алгоритмы в начальной и основной школе // Информатика в школе. 2015. № 2.
2. Внутри суперкомпьютера. Как устроен и зачем нужен самый мощный компьютер в России // Русский Интерес. <http://russianinterest.ru/content/vnutri-superkompyutera-kak-ustroen-i-zachem-nuzhen-samy-moshchnyy-kompyuter-v-rossii>
3. Воеводин В. В. Вычислительная математика и структура алгоритмов: 10 лекций о том, почему трудно решать задачи на вычислительных системах параллельной архитектуры и что надо знать дополнительно, чтобы успешно преодолевать эти трудности: учебник. М.: Изд-во МГУ, 2010.
4. Воеводин В. В., Воеводин Вл. В. Параллельные вычисления. СПб.: БХВ-Петербург, 2002.
5. Гаврилова И. В. Первое путешествие в «параллельный мир» // Информатика в школе. 2015. № 6.
6. Дитер М. Л., Плаксин М. А. Параллельные вычисления в школьной информатике. Игра «Стройка» // Информатика в школе: прошлое, настоящее и будущее: материалы Всеросс. науч.-метод. конф. по вопросам применения ИКТ в образовании, 6–7 февраля 2014 года, Пермский государственный национальный исследовательский университет. Пермь, 2014.
7. Еремин Е. А. Изучаем компьютерную архитектуру: от однопроцессорных машин к многопроцессорным системам // Информатика в школе. 2016. № 2.
8. Еремин Е. А. Как наглядно продемонстрировать школьникам преимущества многопроцессорных систем // Информатика в школе. 2015. № 9.
9. Звонкин А. К., Кулаков А. Г., Ландо С. К., Семенов А. Л., Шень А. Х. Алгоритмика: 5–7 классы: учебник

и задачник для общеобразоват. учебных заведений. М.: Дрофа, 1996.

10. *Иванова Н. Г., Плаксин М. А., Русакова О. Л.* Задачи на параллельное программирование в конкурсе «ТРИЗформашка-2013» // Информационные технологии в образовании. XXIII Международная конференция-выставка: сборник трудов. Ч. II. М.: Издательский отдел факультета ВМК МГУ имени М. В. Ломоносова, 2013.

11. *Иванова Н. Г., Плаксин М. А., Русакова О. Л.* Конкурс «ТРИЗформашка» как площадка для апробации заданий на параллельное программирование // Информатика в школе: прошлое, настоящее и будущее. Материалы Всеросс. науч.-метод. конф. по вопросам применения ИКТ в образовании, 6–7 февраля 2014 года, Пермский государственный национальный исследовательский университет. Пермь, 2014.

12. *Иванова Н. Г., Плаксин М. А., Русакова О. Л.* ТРИЗформашка // Информатика. 2010. № 5.

13. *Киселева Е. Ю.* Потенциал суперкомпьютерной тематики в проектно-исследовательской деятельности учащихся // Информатика в школе. 2015. № 2.

14. *Киселева Е. Ю.* Реконструкция параллельной модели вычислений на примере задачи суммирования чисел // Информатика в школе. 2013. № 10.

15. *Кучев А. Д., Плаксин М. А.* Параллельные вычисления в школьной информатике. Игра «Танковый экипаж» // Информатика в школе: прошлое, настоящее и будущее. Материалы Всеросс. науч.-метод. конф. по вопросам применения ИКТ в образовании, 6–7 февраля 2014 года, Пермский государственный национальный исследовательский университет. Пермь, 2014.

16. *Логинов А. В.* Исторические предпосылки и перспективы суперкомпьютерного образования в школьном курсе информатики // Информатика в школе. 2015. № 2.

17. Материалы Летней Суперкомпьютерной Академии. <http://academy.hpc-russia.ru/>

18. *Плаксин М. А.* Комплекс деловых игр для знакомства с параллельными вычислениями в начальной школе // Преподавание информационных технологий в Российской Федерации. Материалы Тринадцатой открытой всероссийской конференции «ИТ-Образование-2015» (г. Пермь,

14–15 мая 2015 года), Пермский государственный национальный исследовательский университет. Пермь, 2015.

19. *Плаксин М. А.* Комплект деловых игр для начального знакомства с параллельными вычислениями // Информатика в школе. 2016. № 5.

20. *Плаксин М. А.* О методике начального знакомства с параллельными вычислениями в средней школе // Информатика в школе: прошлое, настоящее и будущее. Материалы Всеросс. науч.-метод. конф. по вопросам применения ИКТ в образовании, 6–7 февраля 2014 года, Пермский государственный национальный исследовательский университет. Пермь, 2014.

21. *Плаксин М. А.* Подготовка методики преподавания темы «Параллельные вычисления» в средней школе // Параллельные вычислительные технологии (ПаВТ'2016). Труды международной научной конференции (г. Архангельск, 28 марта — 1 апреля 2016 года). Челябинск: Издательский центр ЮУрГУ, 2016.

22. *Плаксин М. А.* «Суперкомпьютеры» vs «параллельное программирование». «Параллельное программирование» vs «совместная деятельность». Как изучать тему «параллельные вычисления» в средней школе? // Современные информационные технологии и ИТ-образование. 2015. Т. 1 (№ 11).

23. *Плаксин М. А., Иванова Н. Г., Русакова О. Л.* Информатика: учебник для 3 класса: В 2 ч. М.: БИНОМ. Лаборатория знаний, 2013.

24. *Плаксин М. А., Иванова Н. Г., Русакова О. Л.* Информатика: учебник для 4 класса: В 2 ч. М.: БИНОМ. Лаборатория знаний, 2012.

25. *Плаксин М. А., Иванова Н. Г., Русакова О. Л.* Набор заданий для знакомства с параллельными вычислениями в конкурсе «ТРИЗформашка» // Преподавание информационных технологий в Российской Федерации. Материалы Тринадцатой открытой всероссийской конференции «ИТ-Образование-2015» (г. Пермь, 14–15 мая 2015 года), Пермский государственный национальный исследовательский университет. Пермь, 2015.

26. *Соколовская М. А.* Методическая система обучения основам параллельного программирования будущих учителей информатики: автореф. дис. ... канд. пед. наук. Красноярск, 2012.

НОВОСТИ

Итоги Национального суперкомпьютерного форума — 2016

С 29 ноября по 2 декабря 2016 года в Переславле-Залесском проходил V Национальный суперкомпьютерный форум.

В докладах на конференции форума были изложены стратегическое видение рынка высокопроизводительных вычислений и практика применения суперкомпьютерных технологий в мире с точки зрения ведущих мировых производителей суперкомпьютерной техники: NVidia, AMD, Intel, Hewlett-Packard Enterprise, Lenovo, Mellanox, Xilinx. Достижения отечественных разработчиков суперкомпьютерных технологий представили российские компании: МЦСТ, «Т-Платформы», «Байкал», НИЦЭВТ, «Квант», РФЯЦ-ВНИИЭФ, Immers, ТЕСИС, Fidesys, концерн «Вега» и др. На НСКФ-2016 работали две специализированные секции: «Развитие академических суперкомпьютерных центров коллективного пользования» и «Компьютерное моделирование в материаловедении». Intel Software Group впервые на площадке форума провел «Intel Software 2016: HPC Code Modernization Workshop for Intel Xeon & Xeon Phi».

Одним из событий форума стало вручение премии НСКФ. Она присуждается за заслуги перед суперкомпьютерной отраслью России. Каждый год выбираются три лауреата, при этом все решения (выдвижение номинантов, принятие окончательного решения о лауреатах) принимаются всем отраслевым сообществом — всеми участниками форума.

Лауреатами премии НСКФ-2016 стали:

- **Воеводин Владимир Валентинович**, доктор физ.-мат. наук, чл.-корр. РАН (Научно-исследовательский вычислительный центр МГУ, г. Москва);
- **Каляев Игорь Анатольевич**, доктор тех. наук, академик РАН (Научно-исследовательский институт многопроцессорных вычислительных систем имени академика А. В. Каляева Южного федерального университета, г. Таганрог);
- **Лацис Алексей Оттович**, доктор физ.-мат. наук (Институт прикладной математики имени М. В. Келдыша РАН, г. Москва).

Редакция журнала «Информатика и образование» поздравляет лауреатов премии!