Theoretical Computer Science: взгляд математика

Александр Разборов

19 августа 2013 г.

> *It is very easy to use this thing. Just imagine you have a Turing machine with random access.*
> (из объяснения экскурсоводом Филадельфийского музея изящных искусств правил пользования аудио-гидом).

1 Часть историческая

Научная дисциплина с неоднозначно переводимым на русский язык названием Theoretical Computer Science (теоретическая компьютерная наука? или всё-таки информатика? я в любом случае буду для краткости использовать аббревиатуру TCS) существует в виде, более-менее напоминающем современный около 40 лет. Если взглянуть на журнал с одноимённым названием
(`http://www.elsevier.nl/inca/publications/store/5/0/5/6/2/5/`), являющийся органом Европейской Ассоциации TCS, или на "Справочную книгу по TCS" (`http://www.elsevier.nl/inca/publications/store/5/0/1/4/4/2/`), довольно быстро выяснится, что в настоящий момент под этой аббревиатурой понимается объединение двух по существу независимых направлений, со своими собственными историей, внутренней логикой, методами и предметом исследований. Во избежание недоразумений хочу сразу подчеркнуть, что у нас речь пойдёт лишь об одном из них, которое можно условно назвать "сложностные аспекты вычислительных процессов"; именно это направление (за неимением лучшего термина, и только для целей данного эссе) и будет называться TCS. Другое направление,

включающее формальную семантику языков программирования и математические методы анализа программ (название также условно) нами не рассматривается.

Развитие TCS по существу началось с появлением первых ЭВМ, и это, разумеется, не случайно. Именно тогда и именно по этой причине интерес многих специалистов по абстрактной теории алгоритмов (и в их числе нескольких выдающихся математических логиков того времени) начал постепенно смещаться от вопросов *существования* алгоритмов для тех или иных алгоритмических задач к изучению их *эффективности* и пригодности с практической точки зрения. Этот переход во-многом стимулировался пониманием того, что даже для наиболее базисных задач эти две вещи далеко не одно и то же.

Например, имеются универсальные алгоритмы, позволяющие проверять истинность любого утверждения про вещественные числа (которое можно записать в стандартном языке математической логики первого порядка – отметим, что путем введения декартовых координат к такому виду можно привести, например, любую задачу элементарной геометрии). Первый такой алгоритм был придуман американским логиком А. Тарским ещё в 1948 г. Однако алгоритм Тарского оказался неэффективен со всех мыслимых точек зрения, а в 1974 г. было строго доказано, что время работы любого алгоритма для этой задачи по крайней мере экспоненциально от длины исходного утверждения.

Этот пример относится к области применения ЭВМ, считавшейся в 60-е годы чуть ли не единственной, и которую я буду условно называть *классические вычисления*, в наиболее широком смысле этого термина. Отличительными чертами классического вычисления является традиционная схема

$$\text{Программа } P + \text{ входные данные } x \Rightarrow \text{ вычислительное устройство } \Rightarrow \text{ ответ } P(x)$$

и отсутствие интерактивности. Короче, классическое вычисление – это всё, для чего вы можете использовать такие почтенные языки как АЛГОЛ или ФОРТРАН и не испытывать при этом существенных неудобств.

Строгие математические модели, позволяющие изучать внутреннюю сложность алгоритмических задач (и доказывать про неё нетривиальные результаты!) были также созданы в 60-е годы, и первоначально они относились исключительно к классическим вычислениям. Общеизвестно, что изучение любого явления математическими методами становится

возможным лишь после некоторых допущений, позволяющих абстрагироваться от ненужных деталей и при этом по возможности не утратить суть изучаемого явления. К числу наиболее важных допущений, принятых в теории сложности вычислений в 60-е годы и надолго определивших её последующее развитие, относятся, в частности, следующие принципы:

ПЕРВЫЙ ПРИНЦИП. При фиксированном размере входных данных n сложность алгоритма (в этой роли чаще всего выступают время работы, используемая память и.т.д.) измеряется "в наихудшем случае т.е. берётся максимум по всем входным данным длины n. Постулируется, что для большинства разумных алгоритмов качество их работы не должно сильно зависеть от структуры исходных данных (коль скоро известна их битовая длина), а если это оказывается не так, то скорее всего плоха изначальная постановка задачи (не учтена какая-либо важная специфика). Наиболее известное исключение: симплекс-метод, который хорошо работает для большинства задач линейного программирования, но требует экспоненциального по n числа итераций в отдельных патологических случаях.

ВТОРОЙ ПРИНЦИП. Алгоритмы изучаются с точки зрения их асимптотической сложности, т.е. когда n стремится к бесконечности. Постулируется, что асимптотическое поведение в подавляющем большинстве случаев адекватно отражает поведение алгоритма для тех малых значений n, которые на самом деле представляют интерес. Например, ожидается, что разумный алгоритм, работающий за время $O(n \log n)$ окажется и на практике лучше $O(n^2)$-алгоритма. Наиболее известное исключение: быстрые алгоритмы перемножения матриц. Именно, $n \times n$ матрицы можно перемножать, используя лишь $O(n^{2.376})$ умножений (текущий рекорд), однако подразумеваемая под O мультипликативная константа настолько велика, что алгоритм имеет смысл лишь для астрономических значений n.

ТРЕТИЙ ПРИНЦИП. Особое внимание уделяется алгоритмам, время работы которых ограничено некоторым полиномом от n. Постулируется, что в подавляющем большинстве случаев требование полиномиальности достаточно адекватно отражает "практическую осуществимость"алгоритма. Наиболее известное исключение: метод эллипсоидов Л. Хачияна для линейного программирования, алгоритм полиномиальный, но не слишком практичный.

Как мы знаем, виртуальная реальность оказалась намного сложнее прогнозов, сделанных на заре компьютерной эры, и классические вычис-

ления оказались далеко не единственной сферой применения компьютеров. Всё большее место завоёвывала идея использования вычислительной техники в качестве удобного инструмента для реализации разнообразных *интерактивных* процессов: сетевых и криптографических протоколов, баз данных и.т.д. Развитие TCS отражало эту эволюцию как в зеркале, а во многих случаях предвосхищало и определяло её.

Точно так же, как в реальном мире в настоящий момент наблюдается огромное стилевое разнообразие приложений, для которых используются компьютеры, TCS является весьма протяжённой и в некотором смысле буферной дисциплиной, непрерывно заполняющей целый спектр между чистой логикой/математикой с одной стороны и, скажем, реальными задачами, решаемыми системщиками – с другой. Наиболее существенная разница между настоящими вычислительными процессами и их теоретическими моделями, в какой бы части спектра это ни происходило, состояла (и состоит) в том, что все достижения TCS по определению имеют форму математических теорем, удовлетворяющих всем стандартам строгости классической математики.

Принципы, сформулированные выше для классических вычислений, в общем и целом оказались вполне адекватными для изучения интерактивных процессов вычисления напоминающие уже отдалённо (именно поэтому мы уделили этим принципам столько внимания). Справедливости ради, впрочем, следует отметить, что чем менее математичными являются потребители идей и результатов данной области TCS, тем более низкий уровень абстрагирования характерен для самой области. Поэтому по мере продвижения от математического конца спектра к прикладному происходит определённая ревизия сформулированных выше основных принципов. Например, сделав всего один шаг в сторону от теории сложности (классических) вычислений, мы оказываемся в области построения алгоритмов для конкретных задач дискретной оптимизации. Здесь наш третий принцип (полиномиальные алгоритмы тождественны практичным) уже не имеет такого решающего значения, и, соответственно, огромное внимание уделяется улучшению уже имеющихся полиномиальных алгоритмов.

2 Часть полемическая

Такое протяжённое и по меньшей мере двойственное положение TCS, помимо очевидных преимуществ, предоставляемых междисциплинарностью, навлекает на её голову поток критических замечаний, местами даже яростных. Любопытно (хотя и предсказуемо) при этом то, что сыплющиеся с двух сторон замечания имеют строго противоположную направленность и, соответственно, предлагают прямо противоположные действия по исправлению ситуации.

С точки зрения "чистых"математиков, математические проблемы, решаемые TCS (а зачастую сюда включается любая дискретная математика, имеющая дело лишь с конечными объектами) слишком сильно заземлены на практику и, как следствие, им недостаёт глубины и элегантности. По этой причине TCS мало связана с другими частями современной математики и не является её органичной частью.

Так как мнение этой стороны читателям Компьютерры, по-видимому, менее интересно, я ограничусь лишь следующим кратким замечанием. Эта критика, безусловно, конструктивна и оправдана, но не следует забывать о младенческом возрасте TCS по сравнению с возрастом самой математики или любой другой естественной науки. Сейчас ситуация в TCS уже другая чем, скажем, 10 лет назад. Накоплен и накапливается достаточно большой запас задач, которые представляют общематематический интерес и не решаются через месяц или год после их формулировки (наиболее известным примером такого рода, конечно, является знаменитая $P = NP$ проблема, о которой мы поговорим в следующей части). Для решения своих внутренних задач TCS привлекает всё более серьёзную математику – в качестве примера можно упомянуть теорию сложности на вещественных числах, созданную и активно развиваемую лауреатом премии Филдса американским топологом С. Смейлом со своими сотрудниками. Поэтому уже сейчас математики в-среднем относятся к TCS гораздо благосклоннее, чем 10 лет назад и готовы признать за её теоретическим ядром (т.е. теорией сложности классических вычислений) право на существование в качестве раздела математики, правда, с ярко выраженной спецификой.

С точки зрения многих прикладников TCS, увы, является нагромождением математических абстракций, всё более отрывающихся от реальной жизни. В частности, критике подвергаются все три сформулированных выше исходных принципа, а часто используемый при этом полеми-

ческий приём состоит в том, что известным и общепризнаваемым исключениям уделяется больше внимания, чем той основной массе примеров, в которых эти принципы подтверждаются рутинным образом. Упомянутые нами выше исключения используются для этой цели примерно в половине случаев.

Как мне представляется, любая честная дискуссия по этому поводу должна начинаться с ответа на следующий основной вопрос: является ли тот комплекс явлений в реальном мире, на описание которого претендует TCS, достаточно *фундаментальным*, чтобы оправдать существование отдельной *фундаментальной* науки для изучения этих явлений. А осмысленной (опять же с моей точки зрения) эта дискуссия может стать только после признания всеми её участниками того обескураживающего факта, что любая фундаментальная наука развивается прежде всего по законам своей внутренней логики, существенно перерабатывает приходящие из реального мира запросы прежде, чем они станут её органичной частью и обладает ужасающе низким коэффициентом полезного действия. Таким образом, вопрос не в том, хорошо или плохо иметь *фундаментальную* науку, которая ставила бы *приоритетной* целью учёт всех специфических и разнородных деталей возникающих на практике задач, а в том насколько такой гибрид возможен. Лично мне такие предложения представляются примерно равносильными предложениям учитывать достижения и перспективы карандашной промышленности или, скажем, астрономии при построении эвклидовой геометрии (в практических приложениях прямые имеют ненулевую ширину и ограниченную размерами известной части Вселенной длину, поэтому... знакомо, не правда ли?)

Итак, если читатель не готов принять предложенные правила того, что с моей точки зрения было бы честной и осмысленной дискуссией по данному вопросу, я рекомендую ему не тратить время и перейти непосредственно к заключительной, третьей части. В рамках же этих правил я готов предложить его вниманию следующие три тезиса в пользу существования TCS именно как фундаментальной науки, изучающей вычислительные процессы.

1. *Коэффициент полезного действия TCS не ниже, чем у других фундаментальных наук*, а если учесть, что все её достижения относятся к последним 40 годам, оказывается выше, чем у многих из них. Многие из моих более центристски настроенных коллег (в спектре предоставляемых TCS возможностей автор данного эссе занимает наиболее крайнюю,

математическую позицию) смогли бы проиллюстрировать этот тезис гораздо лучше меня. Всё же рискну предложить вниманию читателя пару банальных примеров.

Концепция криптографии с публичным (открытым) ключом возникла в TCS в рамках всё той же "асимптотической"идеологии, на которой основана теория сложности вычислений. В частности, криптосистема RSA впервые появилась в 1978 г. в строго математической статье, написанной тремя теоретиками Р. Ривестом, А. Шамиром и Л. Адлеманом. Хотя современная криптография давно является самостоятельной областью, в ней не существует никакого ярко выраженного деления на "теоретическую"и "практическую"части, и значительная часть практически ценных результатов и по сей день инициируется теоретическими исследованиями в рамках сформулированных нами общих принципов. В частности, среди статей, написанных всеми тремя создателями схемы RSA в 1999-2000 г. добрая половина является строго математическими.

Создателем одной из наиболее известных и удачливых интернет-компаний Akamai является профессор Массачусетского Технологического Института Том Лейтон, работающий в TCS- группе, а сама компания основана на запатентованной им математической теореме (см. http://www.akamai.com/html/en/ia/our_roots.html). Лейтон продолжает публиковать теоретические статьи и вообще оставаться активным членом TCS-сообщества, а добрая половина штатных сотрудников и внештатных консультантов Akamai привлечены из числа теоретиков (от полных профессоров до студентов), работающих в лучших американских университетах. Замечу в скобках, что по поводу того хорошо это или плохо существуют разные мнения, но к тезису о полезности TCS это прямого отношения не имеет.

2. Наш второй тезис состоит в том, что *TCS является одной из самых дешевых фундаментальных наук*, не требующей больших затрат на дорогостоящее оборудование, и в этом отношении она вполне сравнима с классической математикой. Данный тезис самоочевиден.

3. В той мере, в которой это допускается внутренней логикой её развития, *TCS адекватно реагировала и реагирует на запросы практики.* Осмысленная (в том же смысле, что и выше) критика этого тезиса должна в каждом конкретном случае подразумевать факт наличия лучших моделей, достаточно математичных, общих и абстрактных, чтобы их можно было изучать в рамках фундаментальной науки, которыми, однако, теоретики не могут или не хотят заниматься. В противном случае

честный (хотя и неприятный для TCS) вывод может состоять лишь в том, что, с точки зрения автора критики, расхождение между теорией и практикой неизбежно настолько велико, что дальнейшее изучение данного комплекса вычислительных явлений методами фундаментальной науки нецелесообразно.

Любая серьёзная аргументация за или против последнего вывода, коль скоро он сделан честным и осмысленным образом, немедленно выведет нас далеко за рамки данного эссе, а всё, что я могу сказать по этому поводу кратко уже было сказано выше. Сейчас моя задача более скромна – попытаться ответить на честный и осмысленный вопрос, достаточное ли внимание уделяется теоретиками поискам более адекватных, но всё ещё приемлемых с их точки зрения моделей. Ещё проще, достаточный ли вклад они вносят в поиски того разумного компромисса, без которого невозможно взаимодействие фундаментальной и прикладной науки?

Мне, насколько я помню, ни разу не приходилось слышать критические замечания, в которых не замалчивалось бы важное и тонкое различие между тем, что фундаментальная наука может и чего она не может в принципе. Так что, в завершение полемической части я просто приведу несколько характерных примеров в непосредственной близости от теории сложности классических вычислений.

Пожалуй, наибольшее неприятие вызывает наш третий принцип об отождествлении полиномиальных алгоритмов с практичными. Класс всех полиномов замкнут относительно взятия суперпозиции, и без этого базового свойства любая математическая теория сложности моментально рассыпается в пыль (можете себе представить язык программирования, в котором не предусматривается разбиение программ на модули?). Насколько мне известно, никто из критиков TCS не пытался разъяснить как именно строить осмысленную теорию без свойства замкнутости относительно суперпозиции.

Единственный другой известный разумный класс с таким свойством – это класс *квазилинейных функций*, т.е. функций вида $n \cdot (\log n)^{O(1)}$. Когда (неожиданно) начало выясняться, что многие важные алгоритмы в самом деле работают квазилинейное время (на RAM – машинах со случайным доступом), теоретики всерьёз заинтересовались этим классом, и степень их интереса пропорциональна числу обнаруженных в нём важных и красивых алгоритмов.

Ещё один хороший пример связан с нашим первым принципом, и хорош он своей конструктивностью. Часто утверждается, что в возни-

кающих на практике задачах входные данные поступают в соответствии с некоторым вероятностным распределением, и поэтому сложность "в наихудшем случае"в таких случаях неадекватна.

Построение общей теории, отдельно описывающей алгоритмы для (предположительно) правильного распределения в каждом конкретном случае – дело совершенно невозможное и ненужное. Однако задача построения такой теории для достаточно общего и разумного *класса* вероятностных распределений, которые достаточно адекватно описывали бы многие практические ситуации, вполне осмысленна и является прекрасным модельным компромиссом между интересами теоретиков и практиков. Могу заверить читателей Компьютерры, что работа в этом направлении ведётся, и наибольший прогресс пока был достигнут в работах американских теоретиков Л. Левина и Ю. Гуревича (оба, впрочем, российского происхождения).

Заключительный пример относится к алгоритмам для параллельных вычислений, которым в наблюдаемым мною части TCS-спектра достаётся, пожалуй, больше всего. В TCS такие алгоритмы, так же как и последовательные, в-основном строятся для Parallel RAM, что является некоторой "идеальной"формой параллелизма. Основная претензия, насколько я понимаю, состоит в том, что теоретические алгоритмы плохо соответствуют реально существующей архитектуре распараллеливания.

В данном случае, более тесное следование "запросам практики"мне представляется делом не только малоосуществимым, но и попросту вредным. Задачей фундаментальной науки является демонстрация вещей, которые можно сделать в идеальном мире, соответствующем внутренней природе параллелизма и не зависящим от конкретной архитектуры. Дело практики – оценить, насколько эти возможности привлекательны и учитывать эту оценку при работе над дальнейшим улучшением архитектуры параллельных машин... Разумеется, если выяснится, что такое улучшение упирается в трудности принципиального характера, опять наступает очередь теории сказать слово, однако не вполне очевидно что в случае параллелизма ситуация достигла этой печальной стадии.

3 Перспективы

Как я уже писал выше, развитие TCS происходит параллельно изменениям в реальном компьютерном мире, и темпы этих изменений примерно

одни и те же. Вряд ли кто-нибудь сейчас возьмётся всерьёз предсказывать, скажем, состояние интернета через 5 лет; точно так же и и я не буду даже пытаться делать какие-либо долгосрочные прогнозы относительно развития TCS. В этом заключительном разделе читатель найдёт всего лишь описание нескольких "точек роста в которых в настоящий момент происходят наиболее интересные вещи. Нет нужды повторять, что выбор тем сильно обусловлен личными вкусами и пределами компетенции автора и в-основном концентрируется вокруг математического ядра TCS, т.е. теории сложности (классических) вычислений.

Первое, что необходимо отметить даже в нашем кратком обзоре, это то, что проблема #1 в TCS переходит на следующий век. Читатель, поводимому, уже догадался, что я говорю про $P = NP$-проблему, которая, между прочим, в известном списке центральных открытых задач *всей* математики [1] занимает почётное третье место, сразу вслед за гипотезами Римана и Пуанкаре. Эта проблема удивительно многолика, и одна из её популярных формулировок состоит в следующем: можно ли в общем случае *устранить трудоемкий* (т.е. экспоненциальный по n) *перебор вариантов* из произвольного алгоритма? Более того, в настоящее время известны тысячи так называемых *NP-полных задач* (возникающих в самых разных областях человеческой деятельности), которые решаются очевидным "переборным"алгоритмом и обладающих тем свойством, что полиномиальный алгоритм для *любой* из них влечёт за собой полиномиальный алгоритм для *всех остальных*.

Значительную часть современной TCS можно довольно успешно классифицировать в соответствии с её отношением к феномену перебора, квинтэссенцией которого является $P = NP$-проблема. Наиболее теоретические разделы этот феномен изучают, другие области основаны на предположении, что $P \neq NP$, т.е. устранение перебора в общем случае невозможно. Наконец, в строгом соответствии со знаменитым местом из Стругацких (о решении задач, не имеющих решения), теоретики пытаются понять, что же делать, когда $P \neq NP$, а решать те или иные алгоритмические задачи всё равно надо.

Область, основанная на предположении $P \neq NP$ – это, прежде всего, современная криптография с открытым ключом. Если противник имеет доступ к экспоненциальному перебору вариантов (т.е., в частности, умеет разлагать числа на множители, брать дискретный логарифм в конечном поле и.т.д.), все известные протоколы моментально теряют смысл. К сожалению, рассказ об этой бурно развивающейся и увлекательной

дисциплине выходит далеко за рамки нашего эссе, и я лишь могу порекомендовать книгу [2] для более подробного знакомства с предметом.

Наиболее бурно развивающаяся область, которая пытается понять, что и как можно вычислить несмотря на $P \neq NP$ – это, конечно, квантовые вычисления. Компьютерра уже подробно писала об этом предмете несколько лет назад (47(224), 1997), поэтому я ограничусь лишь парой кратких замечаний. По-видимому, центральная открытая проблема здесь состоит в получении внутреннего (т.е. не зависящего от принципов квантовой механики) описания всех тех *классических* задач, которые можно эффективно решать на квантовом компьютере. Кроме того, квантовые вычисления – пример благополучной области, в которой теория и практика (которую пока, впрочем, в-основном представляют физики) гармонично взаимодействуют с взаимным уважением к интересам и особенностям партнёра. Результат, как говорится, налицо.

Ещё одна крайне интересная область, в которой теоретики также пытаются смоделировать перебор с помощью природных явлений – это генетические алгоритмы. К сожалению, моя компетенция здесь оставляет желать лучшего; кроме того, насколько мне известно, Компьютерра планирует посвятить этой теме специальную подборку.

Совершенно фантастические результаты получаются в бурно развивающейся области *вероятностно проверяемых доказательств* (PCP, от Probabilistically Checkable Proofs) и их приложений к сложности *аппроксимации оптимизационных задач*. Всё началось с концепции *интерактивного доказательства* (1987 г.), которая на самом деле очень близка многим жизненным процессам. Представьте себе ответ студента на устном экзамене преподавателю, которому лень проверять знание студентом доказательства целиком; вместо этого он подбрасывает монетку и, в соответствии с результатами бросания, задаёт выборочные вопросы, пытаясь поймать студента на противоречии. С помощью интерактивных доказательств можно эффективно доказывать очень многие вещи, пока не поддающиеся доказательствам обычным (скажем, то, что два данных графа не изоморфны – один из самых первых примеров). Это привело к целой революции в наших представлениях о том, что такое доказательство вообще (кстати, довольно точно отражающей возврат к изначальному древнегреческому определению "рассуждение, которое убеждает") и, после довольно изрядной петли в своём развитии, к так называемой *основной PCP-теореме*.

Не могу отказать себе в удовольствии привести её в слегка популяр-

ной форме. Сложности с неизоморфизмом графов возникают из-за того, что математике неизвестен полный и легко вычислимый набор инвариантов, определяющих граф с точностью до изоморфизма, поэтому полиномиального классического доказательства неизоморфизма может и не существовать. Рассмотрим теперь *любой* факт, обладающий таким доказательством, например, факт изоморфизма двух графов. Оказывается, что (ещё более ленивый) преподаватель может потребовать от студента записать доказательство в таком виде, что ему достаточно проверить наугад фиксированное (т.е. никак не зависящее от длины доказательства) число бит, и при этом выявить возможную или намеренную ошибку с вероятностью 99%.

Всё это может показаться (и казалось до 1990 г.) забавным, но далёким от жизни занятием, пока не обнаружилось следующее. Большое число переборных задач на самом деле являются задачами *оптимизации* какой-либо целевой функции на множестве всех допустимых решений. Коль скоро задача выбора наиболее оптимального решения оказывается сложной (т.е. NP-полной), можно пытаться ограничиться поисками "примерно оптимального"(скажем, с точностью до мультипликативной константы) решения. Так вот, за редкими исключениями такие задачи вообще не поддавались никакой классификации пока не выяснилось, что PCP-техника замечательно приспособлена для этой цели. За прошедшие 10 лет на её основе фактически с нуля была создана стройная схема классификации оптимизационных задач (с точки зрения их приближённого решения), неизмеримо более богатая, чем обычная теория NP-полноты и пока изобилующая многочисленными белыми пятнами. В процессе этой работы открываются новые важные алгоритмы; я могу отметить в этой связи недавний алгоритм С. Ароры для приближённого решения задачи коммивояжёра на плоскости. По-видимому, от этой области в ближайшие годы можно ждать новых сюрпризов.

Ещё одна важная “точка роста” – это дерандомизация алгоритмов. Многие алгоритмы (в частности, основанные на методе Монте-Карло) существенно используют генераторы случайных чисел. Так как наличие таких генераторов в природе – вещь далеко не саморазумеющаяся, возникает естественный вопрос, в каких случаях их можно из алгоритма устранить (полная дерандомизация) или хотя бы заменить на генератор *псевдослучайных* чисел, к которым предъявляются требования менее строгие, чем полная случайность. Удивительным образом оказывается, что при наиболее естественном определении генераторов псевдослучай-

ных чисел их существование эквивалентно существованию *односторонних функций*, на которых основана современная криптография с открытым ключом. Эти результаты уже стали классическими, а вот пример другой удивительной конструкции, *экстракторов* (от английского слова "extract"), которую теоретики активно изучают в настоящее время.

Пусть у вас есть устройство, генерирующее n псевдослучайных битов, про которые известно лишь одно – никакое двоичное слово длины n не наблюдается со слишком большой вероятностью. Это крайне слабое ограничение называется оценкой снизу на слабую энтропию. Оказывается, что используя в качестве "катализатора"всего $O(\log n)$ по-настоящему случайных битов, можно "выжать"практически всю содержащуюся в любом устройстве с достаточно большой слабой энтропией случайность (например, можно построить $\sqrt{n}$ битов, отличие которых от полностью случайных экспоненциально мало). Такие конструкции и называются экстракторами.

Наконец, к настоящему моменту стало понятно, что решить $P = NP$ проблему одним наскоком, по-видимому, не удастся, и TCS постепенно переходит к тому, что в таких случаях делают все нормальные математики, т.е. планомерной осаде и попыткам проникнуть в суть трудностей, с которыми мы сталкиваемся. Перспективной областью с этой точки зрения выглядит *теория сложности доказательств*, которая, грубо говоря, изучает какие факты обладают эффективным (полиномиальной длины) и классическим (без привлечения интерактивности) доказательством.

Попытки установить, что $P \neq NP$ не обладает по крайней мере эффективным доказательством (пока в этом направлении есть лишь частные результаты) привели к более глубокому пониманию природы самой проблемы. Примечательно, что всё большую роль в этом анализе играют генераторы псевдослучайных чисел, и связь здесь примерно следующая. Если у вас есть генератор псевдослучайных чисел определённого вида, то вы можете построить ансамбль алгоритмических задач, каждая из которых разрешима за полиномиальное время, а сам ансамбль в некотором смысле выглядит как псевдослучайный. Теперь, если я хочу доказать, что $P \neq NP$, мне вначале надо научиться доказывать, что фиксированная NP-полная задача не принадлежит этому ансамблю. Ввиду его псевдослучайности, уже эта ограниченная цель весьма трудна, а для целого класса возможных методов было строго показано, что она просто недостижима.

Цитированная литература

[1] S. Smale, "Mathematical problems for the next century *Mathematical Intelligencer*, 1998, vol. 20, number 2, pages 7-15.

[2] А. Саломаа, "Криптография с открытым ключом М.: Мир, 1996.

Литература, использованная при подготовке эссе

[1] С. Кук, "Вычислительная сложность функций высшего типа *Международный конгресс математиков в Киото*, М.: Мир, 1996, стр. 78-100.

[2] O. Goldreich and A. Wigderson, "Theory of Computation: A Scientific Perspective http://theory.lcs.mit.edu/~oded/toc-sp.html.

[3] А. Разборов, "О сложности вычислений *Математическое Просвещение*, сер. 3, том 3, стр. 127-141, http://www.mi.ras.ru/~razborov/lecture.ps.

[4] А. Разборов, "$P \stackrel{?}{=} NP$ или проблема перебора: взгляд из 90-х http://www.mi.ras.ru/~razborov/phasis.ps.