ЧТО МЫ МОЖЕМ СДЕЛАТЬ С ЗАДАЧАМИ ПОЛНОГО ПЕРЕБОРА?

Г.М.Адельсон-Вельский
Москва, СССР

А.О.Слисенко
Ленинград, СССР

Мы считаем вычислительную задачу, т.е. задачу вычисления значений некоторой функции, трудной, если нам не известно, удастся ли ее фактически решить на реальных вычислительных машинах. С точки зрения крассической теории алгоритмов или теоретико-множественной математики (но вряд ли с точки зрения Аль-Хорезми) такая задача может быть разрешима: имеется тривиальный способ ее решения, состоящий (для данного аргумента) в более или менее полном переборе конечного множества кандидатов на звание ответа задачи и несложной проверки, является ли этот кандидат ее решением. (По теореме Клини о нормальной форме [I], § I.IO, всякая вычислимая функция может быть вычислена таким способом в области ее определения.) Однако фактически такой способ вычисления реализовать не удастся: солнце погаснет, а машина еще будет находиться в начале счета по программе, задающей это вычисление.

Временной сложностью программы Π мы считаем функцию, ставящую в соответствие аргументу, т.е. входным данным, число машинных операций, выполненных программой на этом аргументе. Значение этой функции на аргументе X будем обозначать $t^*_\Pi(X)$. Получить детальное описание поведения t^*_Π довольно трудно, а информативность этого описания подчас не оправдывает затрат на его получение. Более удобно и достаточно информативно оценивать сверху величину $t^*_\Pi(X)$ на данном классе аргументов. Так мы приходим к понятию "сложности в худшем случае": временной сложностью в худшем случае называется функция, ставящая в соответствие натуральному числу n максимум значений $t^*_\Pi(X)$ для X с длиной не больше n. Временная сложность в худшем случае будет обозначаться $t_\Pi(n)$ и называться просто сложностью.

Кроме времени работы, есть и другие сложностные характе-

ристики вычисления, которые хотелось бы оценить - это размер памяти, длина ячейки и т.п. Однако, без оценки времени, оценки других характеристик сложности, как правило, малосодержательны. С другой стороны, при разумно выбранной вычислительной модели - см., например, [2] - верхняя оценка времени дает достаточно нетривиальные, иногда даже хорошие, оценки для упоминавшихся выше других сложностных характеристик.

Сложность задачи определяется двумя оценками: верхней и нижней. Одноместная функция φ является верхней оценкой сложности данной задачи f, если можно построить программу Π, решающую данную задачу (т.е. вычисляющую f), и такую, что $t_{\Pi}(n) \le \varphi(n)$ для всех n. Двухместная функция ϕ является нижней оценкой сложности f, если для всех Π, вычисляющих f, при всех n выполнено $t_{\Pi}(n) \ge \phi(|\Pi|, n)$, где $|\Pi|$ - длина Π. Из приведенных определений видно, что мы можем говорить о совпадении верхних и нижних оценок лишь с некоторой степенью приближенности. "Идеальной" близостью этих оценок является ситуация, когда $\phi(k,n) = \phi_1(k)\varphi(n)$, а ϕ_1 "не очень быстро" убывает. Но такой близости не всегда можно добиться даже теоретически.

О некоторых задачах мы можем доказать, что для них нет практически эффективных алгоритмов, которые бы их решали при всех исходных данных. Такие задачи мы будем называть истинно трудными. Если же мы не можем доказать такого, но и не знаем эффективного алгоритма для решения данной задачи, то мы будем называть ее неэффективизированной, или задачей с неясной трудностью.

Истинно трудными являются задачи вычисления функций, принимающих очень большие значения, например, задача перечисления элементов конечного множества, если число их может оказаться очень велико. Однажды конструкторы машин предложили первому из авторов создать программу, перечисляющую все ориентированные циклы ориентированного графа. Число их может быть и не очень большим, но может достичь величины $\Sigma_{k=1}^{n-1} k!$, где n - число вершин графа. Нам не известно, что стали бы делать заказчики, если бы количество циклов оказалось только в 10^6 раз меньше этой оценки, скажем, при $n=90$.

Истинно трудные задачи можно получить, используя различные

диагональные конструкции. В частности, по данной оценке φ можно построить двузначную функцию (т.е. задачу распознавания множества), которая будет иметь обе оценки сложности, мало отличающиеся от φ (при этом предполагается, что сама φ вычисляется достаточно просто, например, φ есть полином, экспонента и т.п.). Хотя построенные таким образом примеры истинно сложных задач являются искусственными, их иногда удается весьма просто свести к естественным задачам и тем самым получить высокую оценку сложности для последних. Такого рода рассуждения являются довольно точным аналогом доказательств алгоритмической неразрешимости конкретных задач. Результаты этого рода носят отрицательный характер; как правило, они означают, что исходная задача практически неразрешима, и нам надо менять постановку задачи. Доказательство истинной трудности конкретной задачи может иметь определенное позитивное значение для поиска более реалистической постановки соответствующей ей "физической" задачи. Эти соображения можно проиллюстрировать на примере проблемы разрешимости для алгебры Тарского. Мы исходим из полиномиальных неравенств вида $P(x_1,\ldots,x_n)\geq 0$, где P - полином с рациональными коэффициентами, а $x_1,\ldots,x_n$ - переменные для вещественных чисел. Далее с помощью обычных логических связок из таких неравенств строятся суждения. Одним из простейших примеров такого суждения является суждение, утверждающее разрешимость над полем вещественных чисел системы конкретных полиномиальных неравенств от одной переменной. Проблема разрешимости для алгебры Тарского состоит в построении алгоритма, устанавливающего по произвольному суждению описанного выше вида истинно оно или нет. Хорошо известно, что такой алгоритм есть, однако сложность этой проблемы разрешения не меньше экспоненты (см., например, [3]); причем нижняя оценка достаточно убедительно указывает на то, что эта задача истинно трудная. Даже при беглом взгляде на доказательство видно, что сложно распознаваемые суждения алгебры Тарского имеют очень длинные, в совокупности неограниченные цепочки вложенных кванторов: $\forall x_1 \ldots \exists x_2 \ldots \forall x_3 \ldots$. Обычно же нас интересуют формулы с весьма небольшим числом кванторов, например, при всех ли значениях параметров разрешима система полиномиальных неравенств данного вида. Такого рода анализ

доказательства истинной сложности задачи можно продолжить и извлечь из него ряд ограничений, которые позволяют сделать рассматриваемую задачу более податливой для эффективизации и при этом сохранить ее "физический" смысл. Однако, обычно анализ этих доказательств не дает возможности так сузить вид входных данных, чтобы получить практически эффективный алгоритм, - трудность задачи остается неясной.

Неэффективизированные задачи, в целом, остаются малоисследованной областью. Правда, за последние годы их удалось до какой-то степени расклассифицировать, что внесло определенную ясность в вопрос об их относительной сложности и в вопрос о целесообразности сведения одних задач к другим.

Очень многие интересные неэффективизированные задачи, включая ряд задач дискретной оптимизации, принадлежат классу $\mathcal{NP}$- полных задач [4]. Напомним определение этого класса.

Сначала введем класс $\mathcal{P}$ - это класс задач распознавания (т.е. функций, принимающих 2 значения), решаемых с полиномиальной сложностью. Например, распознавание свойства "список ребер y является гамильтоновым циклом в графе x" лежит в $\mathcal{P}$, если, разумеется, x и y задаются обычным образом (см., например, [5], гл. 10). Иными словами, существует алгоритм α и полином p, такие что α распознает упомянутое свойство и $t^*_\alpha(x,y) \le t_\alpha(|x|+|y|) \le p(|x|+|y|)$; здесь $|z|$ обозначает длину z как слова в соответствующем алфавите (все подразумеваемые алфавиты содержат не менее двух букв).

Теперь рассмотрим условия вида

$$|y| \le q(|x|) \;\&\; Q(x, y), \qquad (I)$$

где q - полином и $Q \in \mathcal{P}$. Будем трактовать x как переменную для кода исходных данных, а y как "параметр перебора", принимающий значения из множества, содержащего искомое решение (например, y - переменная для списков различных ребер графа x). Неравенство $|y| \le q(|x|)$ определяет область перебора, число элементов в этой области есть $C^{q(|x|)} \ge 2^{|x|}$.

Класс переборных задач распознавания (класс $\mathcal{NP}$) - это класс всех задач вида «существует y, такой что (I)», где q и Q фиксированы для данной задачи. Например, задача о существовании гамильтонова цикла в графе принадлежит $\mathcal{NP}$- область перебора определяется неравенством $|y| \le |x|$.

Класс переборных задач минимизации (класс $\mathcal{NP}_{min}$) - это класс всех задач f вида

$$f(x) = \min\{\nu(y):(1)\}, \qquad (2)$$

где ν - рациональнозначная функция, вычисляемая за полиномиальное время. Например, если x - граф с целыми весами на ребрах, а $\nu(y)$, где y - список ребер, определяется как сумма весов на этих ребрах, то формула (2) определяет задачу о коммивояжера, если Q(x, y) есть условие «список y является гамильтоновым циклом в x» и $q(|x|) = |x|$.

В рассматриваемой ниже классификации переборных задач класс $\mathcal{P}$ трактуется как класс "простых" задач и все рассмотрение ведется "по модулю преобразований с полиномиальной сложностью". Это, конечно, связано не с тем, что вычисления с полиномиальной сложностью практически эффективны, а с тем, что рассматриваемые неэффективизированные задачи с точки зрения имеющихся алгоритмов очень далеки от вычислений с полиномиальной сложностью - известные верхние оценки сложности этих задач не меньше экспоненты. И мы ставим вопрос о грубой оценке сложности: полином или нет; или, можно ли уменьшить полный экспоненциальный перебор до полиномиального перебора.

Классу $\mathcal{NP}$ принадлежат многие известные задачи, как-то - распознавание простоты числа, изоморфности двух графов и ряд других. О наиболее интересных задачах этого типа пока не доказано, лежат ли они в $\mathcal{P}$ или нет. Но для большинства таких естественных задач доказано следующее свойство универсальности: «если данная задача лежит в $\mathcal{P}$, то $\mathcal{P}=\mathcal{NP}$». Задачи из $\mathcal{NP}$, обладающие таким свойством универсальности, называются $\mathcal{NP}$-полными. Аналогично определяется $\mathcal{NP}$-полнота задач из $\mathcal{NP}_{min}$ - надо заменить $\mathcal{NP}$ на $\mathcal{NP}_{min}$, а $\mathcal{P}$ - на класс функций, вычислимых за полиномиальное время.

Огромное число давно известных задач дискретной оптимизации и соответствующих им задач распознавания оказались $\mathcal{NP}$-полными - см., например, [5], гл. IO. В частности, упомянутые выше задачи о коммивояжере и о гамильтоновом цикле (даже в плоском графе [6]) являются таковыми.

$\mathcal{NP}$-полные задачи обычно трудно решать. Однако далеко не всегда, точнее, не для всех исходных данных. Есть различные методы сокращения перебора, например, метод ветвей и границ[7].

Ни для одной "серьезной" задачи не удалось доказать, что его применение дает возможность гарантированно решить задачу за полиномиальное время, но многие конкретные случаи различных массовых задач удалось решить этим методом. (Говоря о решении данной задачи для конкретного аргумента, мы будем использовать такие слова как «решение конкретизации данной задачи», «решение частного случая» и т.п.)

НУЖНО ЛИ РЕШАТЬ ТРУДНЫЕ ЗАДАЧИ?

Известные истинно трудные задачи в наибольшей степени интересны, пожалуй, самим фактом своей сложности, а те их частные случаи, которые действительно решаются на машинах, требуют не так уж много времени. Трудно обосновать, но, по-видимому, еще труднее опровергнуть утверждение, что так и должно быть. До какой-то степени это подтверждается следующим обстоятельством. По вычислимой нетривиальной нижней оценке сложности ϕ, особенно "хорошего" вида, упоминавшегося выше, мы не умеем строить даже искусственную задачу распознавания, которая бы имела сложность не меньше ϕ на достаточно плотном множестве аргументов. А на всех аргументах такое вообще не возможно (ввиду возможного равенства $\phi(|\Pi|,x) = 0$ при маленьких x, фактически, здесь речь идет о всех аргументах, начиная с некоторого места, вычислимого по нижней оценке ϕ и Π). К сожалению, плотностные соображения в данном вопросе слишком уязвимы и неубедительны.

Отметим, что известная аргументация о том, что в действительности нас всегда интересует лишь небольшое конечное множество конкретизаций данной задачи и, следовательно, в принципе существует алгоритм решающий их быстро, на самом деле несостоятельна. Мы просто забываем о том, что вычислительную сложность можно "перекачать" в сложность обоснования корректности алгоритма; если же ввести в рассмотрение и эту характеристику сложности, то в конечных областях мы будем иметь те же "алгоритмические неразрешимости", "высокие нижние оценки сложности" и т.п., как и в бесконечных областях - см.[8].

I. Утверждать, что задачи без достаточно низкой оценки сложности сверху интересны, труднее, чем говорить то же самое об истинно трудных задачах. Многие естественным образом фор-

мулируемые задачи дискретной математики, исследования операций, эконометрики, распознавания образов и искусственного интеллекта являются $\mathcal{NP}$-полными или, уж во всяком случае, задачами с неясной трудностью. Выше мы привели два примера $\mathcal{NP}$-полных задач - задачу о гамильтоновом цикле и задачу о коммивояжере. Нетрудно видеть, что задача о коммивояжере с помощью дихотомии сводится к решению нескольких ($\mathcal{NP}$-полных) задач о существовании гамильтонова цикла данного веса, причем число последних задач не превосходит размера (длины записи) задачи о коммивояжере. $\mathcal{NP}$-полными являются задачи технологической или экономической оптимизации, например, задача Джонсона для линии из трех и более станков, оптимизация технологических моделей, требующая решения целочисленных и частично целочисленных задач линейного программирования, минимизации срока окончания работ в сетевой модели, их выполнения с учетом ограниченных нескладируемых ресурсов и т.д.

Мы приведем одну постановку последней задачи. Структура ее исходных данных может показаться сложной, однако детальность этой структуры и ясный физический смысл отдельных параметров позволяют легко увидеть, какие содержательные соображения могли бы помочь при ее решении. Формулируемая ниже задача является $\mathcal{NP}$-полной [5].

Исходные данные рассматриваемой задачи планирования состоят из: множества ресурсов $\{R_1,\dots,R_r\}$ с границей B_i для каждого R_i, B_i - общее количество ресурсов i-го вида; множества работ $\{T_1,\dots,T_m\}$ с частичным порядком $\prec$ на них; каждой работе T_j соответствует время τ_j ее выполнения и требуемые ресурсы $R_i(T_j)$ каждого вида. Пусть D - натуральное число (общее время выполнения всех работ). Функция $f:\{T_j\} \to \{0,1,\dots,D-1\}$ называется планом, если

$$f(T_i) + \tau_i \le D;$$

$$T_i \prec T_j \Rightarrow f(T_i) + \tau_i \le f(T_j);$$

$$\sum_{\{T_i : f(T_i) \le t < f(T_i)+\tau_i\}} R_j(T_i) \le B_j \quad \text{при} \quad 0 \le t < D .$$

Задача состоит в том, чтобы минимизировать D, при котором существует план, и для минимального D найти какой-нибудь план.

Мы завершим список примеров $\mathcal{NP}$-полных задач задачей, с которой обычно начинается обоснование $\mathcal{NP}$-полноты конкретных задач и которая в каком-то смысле наиболее рельефно демонстрирует трудности решения этих задач в общем случае, - имеется в виду задача о пропозициональной выполнимости. Исходными данными этой задачи являются таблицы высоты три и произвольной длины l ; в клетках этой таблицы стоят переменные или их отрицания:

$x_{1,1}$	$x_{2,1}$	...	$x_{l,1}$
$x_{1,2}$	$x_{2,2}$	...	$x_{l,2}$
$x_{1,3}$	$x_{2,3}$	...	$x_{l,3}$

(Ø)

Можно считать, что x_{ij} - это либо двоичные натуральные числа (собственно переменные), либо выражения вида $\neg x$, где x - натуральное число, а $\neg$ - знак отрицания. Путь в таблице Ø - это список вида $x_{1,i_1},x_{2,i_2},\ldots,x_{l,i_l}$. Путь закрыт, если он содержит контрарную пару, т.е. переменную и ее отрицание; путь открыт, если он не закрыт. Задача о пропорциональной выполнимости ставится так: по таблице узнать, есть ли в ней открытый путь.

2. Фактически сложными являются многие задачи искусственного интеллекта. Например, задача выбора лучшего или хотя бы хорошего хода в данной шахматной позиции. Доказать $\mathcal{NP}$-полноту этой задачи вряд ли возможно в настоящее время , даже если не считать формального соображения о конечности задачи, проверить по данному ходу, является ли он выигрывающим, столь же трудно, как и найти его. Но задача очевидно трудна - попробуйте создать хорошую шахматную программу.

Задачи установления доказуемости суждений в тех или иных формальных системах, как правило, истинно трудны. Уже простейшая из этого класса задач - установление выполнимости формул в классическом исчислении высказываний-является $\mathcal{NP}$-полной; а это всего лишь модельная задача. Как только мы переходим к задаче установления выводимости в более богатых формальных системах, мы сталкиваемся с экспоненциальными нижними оценками [3], [9] или с алгоритмической неразрешимостью. Тем не менее, интерес к задачам автоматического доказательства теорем

не ослабевает. Сейчас основные усилия сосредоточены на доказательстве теорем из узких разделов математики с максимальным использованием специфики этих узких разделов. Даже если оставить в стороне столь выдающееся достижение как решение проблемы четырех красок, можно указать разделы математики, где в области машинного доказательства теорем получены яркие результаты - см. [IO], [II].

3. Несмотря на приведенные выше соображения о важности решения трудных задач, некоторые считают, что заниматься их исследованием не нужно. Для обоснования этого приводятся различные аргументы, начиная от "высоких" философских и эстетических принципов и кончая крайне прагматическими рассуждениями. К числу аргументов первого типа относятся суждения о том, что настоящая наука делается "голыми мозгами", и поэтому научные открытия, полученные с помощью вычислительной машины (например, решение проблемы четырех красок), человек не способен понять, а значит, они не представляют теоретического интереса. С такой точки зрения многие задачи искусственного интеллекта представляются некоторой блажью. Да и истинная гармония мира должна быть простой, а сложности (которые конечно же порождены дьяволом) лишь уводят нас от ее познания.

Другая крайняя точка зрения выглядит примерно так. Математическая формулировка практической задачи, например, задачи оптимизации производства, всегда получается в результате некоторых упрощающих допущений. Мы принимаем их потому, что верим "честному слову наследного принца" - человека, изучавшего проблему не с математической точки зрения, а по существу. Так поговорим с ним, и нам удастся сделать задачу совсем простой, решение которой будет почти очевидно. Полемизировать с такими мнениями, основываясь только на общих соображениях, вряд ли имеет смысл. Уже полученные результаты позволяют придерживаться более оптимистических взглядов на ценность исследования сложных задач. Нашу точку зрения можно сформулировать следующим образом:

I) классификация математических задач по трудоемкости их решения, в том числе, и относительной, имеет принципиальное теоретическое значение. Еще более важны оценки сложности конкретных алгоритмов и особенно оценки сложности задач,

независимые от того, каким образом можно их решить. Без ответа на многие вопросы, возникающие при исследовании сложности задач, картина мира будет иметь серьезные изъяны. В частности, очень важно было бы выяснить, совпадают ли классы $\mathcal{P}$ и $\mathcal{NP}$. Любое решение этой задачи оказалось бы прочной моральной опорой для решателей $\mathcal{NP}$-полных задач. Альтернатива $\mathcal{P}=\mathcal{NP}$ стимулировала бы поиск лучших методов решения для конкретных классов задач, решаемых с полиномиальной сложностью. Противоположная альтернатива заставила бы обратить внимание на вопросы самой постановки задачи.

2) кроме методов построения простых поделей, основанных на "содержательном" исследовании ситуаций, есть общематематические, которыми следует уметь пользоваться. (Именно им и посвящен наш доклад; однако мы не претендуем на исчерпывающую полноту обзора таких методов.). Уместно вспомнить, что алгоритмика исследует мир, в значительной степени создаваемый человеком и в этом смысле искусственный, на формирование этого мира она влияет, в частности, конструируемыми ею алгоритмами. Поэтому алгоритм, эффективный в некоторой несуществовавшей ситуации, может вызвать к жизни эту ситуацию.

3) математическая постановка задачи часто оказывается более или менее корректной и в тех случаях, когда существуют обоснованные возражения против некоторых допущений. Это означает, что решения небезупречно сформулированных задач и упрощенных при помощи общематематических соображений окажутся неплохими для практических приложений.

4) проблемы искусственного интеллекта уже сейчас актуальны, и их актуальность будет только возрастать. Хотя сейчас нельзя говорить об очень большом успехе, но и в программировании игр, и в автоматическом доказательстве теорем, и в распознавании образов есть определенные достижения. Работа над этими задачами привела также к некоторым результатам, имеющим теоретическое значение.

Однажды поспорили два волшебника - Ф.С.Киврин и К.Хунта.

- Г-голубчики,-сказал Федор Симеонович озадаченно, разобравшись в почерках. - Это же п-проблема Бен Б-бецалеля. К-Калиостро же доказал, что она н-не имеет р-решения.

- Мы сами знаем, что она не имеет решения, - сказал Хун-

та, немедленно ощетиниваясь. - Мы хотим знать, как ее решать.

- К-как-то ты странно рассуждаешь, К-кристо... К-как же искать решение, к-когда его нет? Б-бессмыслица какая-то...

- Извини, Теодор, но это ты очень странно рассуждаешь. Бессмыслица - искать решение, если оно и так есть. Речь идет о том, как поступать с задачей, которая решения не имеет. Это глубоко принципиальный вопрос. ([I2], стр. 385)

Из сказанного в этом разделе должно быть ясно, что мы на стороне Кристобаля Хунты.

КАК РЕШАЮТ ЗАДАЧИ ПЕРЕБОРНОГО ТИПА?

Задачи переборного типа приходится решать, и очень часто их конкретные случаи успешно решаются. Подчас трудная на первый взгляд задача оказывается в каких-то отношениях простой и эта ее простота дает ключ к ее эффективному решению. Частных приемов решения, работающих в узких классах задач, слишком много, чтобы пытаться дать их обзор. Мы остановимся лишь на некоторых общих принципах, которые могут обеспечить эффективизацию некоторых переборных задач.

Все те подходы к решению трудных задач, о которых пойдет речь ниже, по сути дела, сводятся к некоторой модификации исходной постановки задачи. Разумеется, в таких модификациях, по возможности, сохраняется "физическая" сущность задачи. Различные соображения показывают, что такого рода подходы к трактовке трудных задач в настоящее время и, по-видимому, в ближайшем будущем окажутся наиболее полезными практически и наиболее плодотворными теоретически. Грубо говоря, эти подходы разбиваются на два типа. При подходе первого типа мы сужаем область исходных данных, а при подходе второго типа мы жертвуем точностью решения. К подходам первого типа относится "выделение подклассов полиномиальной сложности". Это выделение означает, что мы ограничиваемся некоторым подмножеством исходных данных, специфика которых позволяет построить алгоритм для решения исходной задачи, который на аргументах из этого ограниченного множества имеет небольшую (полиномиальную верхнюю оценку сложности. В определенном смысле к такому же подходу относятся эвристические алгоритмы. К подходам второго типа относятся приближенные и вероятностные решения.

Общие принципы выделения содержательных подклассов полиномиальной сложности не известны, и, по-видимому, их выявление является очень серьезной задачей. На первый взгляд может показаться, что задача об описании "просто" разрешимых подклассов данного класса сложных задач не является математической, а требует лишь анализа содержательных истоков рассматриваемых задач. Однако фактические попытки выделить полиномиальные подклассы показывают, что сложностные, алгоритмические соображения играют более активную роль в этом процессе. Это удобно проиллюстрировать на примере задачи об изоморфизме графов. При попытках построить эффективные алгоритмы для распознавания изоморфизма произвольных графов было замечено, что естественные локальные соображения не эффективны лишь для сильно регулярных графов. Поэтому графы без сильно регулярных нетривиальных подграфов составляют класс графов, для которого проблема изоморфизма решается алгоритмом с небольшой сложностью [I3]. Но в практических ситуациях, например, при классификации структурных формул химических соединений, мы имеем дело именно с такими графами, весьма далекими от сильно регулярных. По-видимому, этот прием описания достаточно содержательных и просто разрешимых подклассов трудных задач, а именно, запрещение в рассматриваемых структурах "очень симметричных" подструктур, является достаточно универсальным. Трудность в его применении связана с тем, что мы далеко не всегда понимаем "симметричность" чего в структуре входных данных мешает эффективизации алгоритмов. Заметим, что для некоторых простых задач [I4], которые не являются истинно переборными в нашем смысле, все "симметричности", мешающие эффективизации естественных алгоритмов, удается расклассифицировать и на этой основе построить очень быстрые алгоритмы.

Обсуждавшееся выше свойство графов ("не имеет сильно регулярных подграфов") не является наглядным или просто проверяемым, хотя мы можем декларировать его из физических соображений и даже, возможно, доказать его выполнение для некоторого класса задач, пользуясь этими соображениями. Свойства такого же типа часто используются при выработке различных эвристических приемов решения задач. С математической точки зрения, более привлекательны свойства с более ясной синтак-

сической структурой. Разумеется, если просто ограничить размерность входов или какие-либо параметры, характеризующие число "степеней свободы", то можно получить просто решаемую задачу. Но на таком пути редко выявляются действительно интересные задачи, такие как задача о назначениях, однопродуктовая потоковая задача и т.п. [I5], [I6].

По-видимому, до какой-то степени совместить синтаксическую простоту и физическую содержательность можно, если использовать для описания области определения задачи какое-либо исчисление достаточно простого типа, например, бесконтекстную грамматику (или что-то вроде грамматики со "слабой" контекстностью). Мы проиллюстрируем эту идею (она высказывалась еще в [2]) на не очень строгом уровне - она, кажется, заслуживает более тщательной проработки.

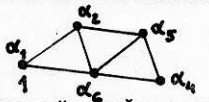
Рассмотрим задачу о существовании гамильтонова цикла в графе. В качестве физической основы будем иметь в виду классическую задачу о коммивояжере и попытаемся описать класс сетей дорог, которые эволюционируют некоторым несложным путем. Попытаемся описать эту эволюцию бесконтекстной грамматикой для графов (разумеется, мы не имеем в виду, что каждое применение правила грамматики соответствует какому-то шагу эволюции). Узлы и ребра графа могут иметь некоторые вспомогательные отметки, в частности, узлы могут быть отмечены как терминальные (которые уже нельзя преобразовывать) и нетерминальные. Аксиома и правила вывода имеют вид:

аксиома: s•

(узел с приписанным нетерминальным символом);

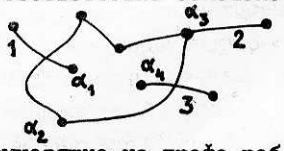
правила вывода:

из 

выводимо 

(к узлу добавляется конкретный граф; соответствие отмечено цифрой 1);

из 

выводимо 

(узел заменяется на граф; некоторые выходящие из графа ребра - их число равно степени заменяемого узла - прикрепляются к тем же узлам, что и соответствующие ребра заменяемого узла);

- в этом описании S,A- нетерминальные символы;

$\alpha_1, \alpha_2, \ldots$ - произвольные (например, пустые).

В процессе пророждения графа по правилам описанного вида мы можем легко поддерживать знание следующего свойства: для данной пары ребер, инцидентных данному узлу, имеется гамильтонов цикл во всем текущем графе, проходящий через данный узел по данной паре ребер. Таким образом, задача о существовании гамильтонова цикла в данном графе свелась к задаче построения вывода (т.е. способа порождения) этого графа в нашей грамматике. Поскольку при естественных предположениях последняя задача решается за полиномиальное время, то и исходная задача тоже решится за полиномиальное время.

Так что для множества графов, порождаемых достаточно простой грамматикой, задача о существовании гамильтонова цикла решается за полиномиальное время.

ПЕРЕБОР С ОТСЕЧЕНИЯМИ

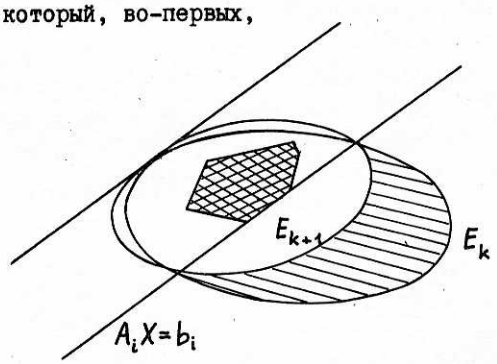
Перед чисто дискретной задачей в общей формулировке мы почти беспомощны. Вспомним общую задачу линейного программирования:

$$\sum_{j=1}^{n} c_j x_j \to \min,$$
$$\sum_{j=1}^{n} a_{ij} x_j = A_i X \le b_i, \quad i = 1,2,\ldots,m, \qquad (3)$$

где a_{ij}, b_i, c_j - целые числа. Вопрос о ее разрешимости для $x_j \in \{0,1\}$ - это $\mathcal{NP}$-полная задача. Вопрос о ее разрешимости в целых числах тоже. Казалось бы вопрос о ее разрешимости в рациональных числах не намного проще, хотя и делает задачу недискретной в очевидном смысле. Однако экспериментально давно было установлено, что в фактически встречающихся ситуациях недискретная задача практически проста. Недавно Хачиян [I7] с помощью изящной конструкции показал, что недискретная задача линейного программирования лежит в $\mathcal{P}$. Так что непрерывность является некоторым общим соображением, резко сокращающим перебор. Сформулируем основные идеи конструкции Хачияна. Для простоты будем считать, что все неравенства в (3) строгие, и рассмотрим лишь вопрос о совместимости этой системы неравенств.

Пусть L - битовая длина входа. До какой-то степени эта

задача, как и всякая другая вычислительная задача является дискретной. Если система (3) совместна, то в шаре E_0 радиуса 2^L с центром в нуле имеется шар радиуса 2^{-Ln}, целиком лежащий в многограннике (3). По индукции строим последовательность эллипсоидов. Пусть построены эллипсоиды $E_0, E_1, \ldots, E_k$ с центрами $X_0, X_1, \ldots, X_k$. Проверяем, удовлетворяет ли X_k системе (3). Если да, то процесс кончается. Если нет, то $A_i X > b_i$ для некоторого $1 \le i \le m$. Построим эллипсоид E_{k+1} (см. рис. ниже), который, во-первых,

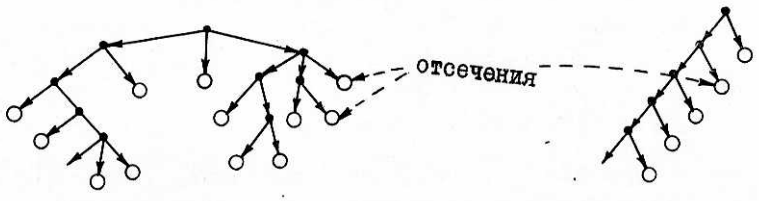


содержит пересечение E_k с гиперплоскостью $A_i X = b_i$, во-вторых, касается эллипсоида E_k в точке касания E_k гиперплоскостью, параллельной $A_i X = b_i$ и лежащей в полупространстве $A_i X \le b_i$, и минимальный объем среди таких эллипсоидов. Оказывается, что объем E_{k+1} в c раз, где c - константа, большая I, меньше объема E_k, и, кроме того, E_{k+1} содержит пересечение E_0 с исходным многогранником (3). Очевидно, что за время, полиномиально зависящее от $n+L$, объем E_k станет меньше шара радиуса 2^{-Ln}. Если к этому времени центры эллипсоидов $E_k, k \le 1$ не попадут в многогранник (3), то он пуст.

В описанном выше алгоритме применяется некоторый тип отсечений, который делает весьма целенаправленным поиск решения или перебор кандидатов на решение. Различные соображения по структурированию перебора весьма широко используются в практических алгоритмах. К сожалению математические идеи, лежащие в основе таких алгоритмов, не часто отличаются глубиной и обычно не приводят к хорошим теоретическим оценкам сложности; алгоритм Хачияна, если на него смотреть с этой точки зрения,

является довольно редким исключением. В целом же, основные идеи структурированного перебора дают пищу для серьезного исследования.

При структурированном переборе конечное (и, обычно, очень большое) множество перебираемых элементов разбивается на подмножества, или общее, накрывается подмножествами меньшего размера. Эти подмножества анализируются, часть из них отбрасывается как заведомо не содержащее решений, а оставшиеся снова редуцируются к меньшим подмножествам и т.д. Если отбрасываний (т.е. отсечений) мало, то мы имеем почти полный перебор. В лучшем случае, когда отсекаются все множества, кроме одного, мы имеем детерминированный (некоторые говорят «направленный») перебор (например, как в алгоритме Хачияна) и получаем полиномиальную верхнюю оценку сложности.



Перебор с отсечениями Детерминированный перебор

Если смотреть на алгоритм Хачияна как на алгоритм детерминированного перебора, то его отличительной чертой являются различного рода "немонотонности": E_{k+1} не содержится в E_k (т.е. происходит не разбиение E_k, а его накрытие), невязка (в терминах (3) - это $\max_i(b_i - A_i X_k)$) меняется немонотонно при переходе от E_k к E_{k+1}.

Структурированный перебор часто можно представить как применение метода ветвей и границ [7], хотя это и не всегда естественно. Общая схема этого метода выглядит примерно так. Пусть на конечном множестве A определена функция f, принимающая рациональные значения; нужно найти то $a_o \in A$, для которого $f(a_o)=\min\{f(a): a\in A\}$, и само $f(a_o)$. Пусть $\{B_\alpha\}$ семейство подмножеств множества A, по которому идет структурированный перебор, причем на этих B_α можно задать некоторую функцию μ

(нижнюю грань), удовлетворяющую условиям

$$\mu(B_\alpha) \le \min\{f(a): a \in B_\alpha\}.$$

Метод анализирует текущее B_α и определяет некоторые элементы $a_1, \ldots, a_m \in A$. Если $f(a_i) < \mu(B_\alpha)$ для некоторого $i, 1 \le i \le m$, то множество B_α следует отсечь. Иногда удается ввести еще и верхнюю грань M : $\min\{f(a): a \in B_\alpha\} \le M(B_\alpha)$. Тогда условием отсечения B_α является и неравенство $\mu(B_\alpha) > M(B_\beta)$.

По схеме метода ветвей и границ описано очень много алгоритмов. Хотя число публикаций, описывающих конкретные алгоритмы этого типа, за последнее время уменьшилось, на практике метод ветвей и границ по-прежнему используется довольно часто. К сожалению, теоретическое исследование эффективности метода оставляет желать лучшего. Между тем метод ветвей и границ заслуживает серьезного теоретического изучения. Особенно важными представляются следующие две проблемы.

I) Имеются более или менее общие соображения о том, как выбирать подмножества B_α и задавать грани μ и M . Однако они не систематизированы и не проведен их достаточно полный сравнительный анализ. В частности, не ясно, как ведет себя метод ветвей и границ при решении "канонической" $\mathcal{NP}$-полной задачи установления пропозициональной выполнимости по сравнению с известными методами поиска вывода - теоретическое изучение последних продвинуто несколько дальше, чем методы решения других $\mathcal{NP}$-полных задач.

2) Не проведен сложностный анализ метода ветвей и границ. Есть пессимистическая гипотеза о том, что в худшем случае он дает алгоритмы экспоненциальной сложности для $\mathcal{NP}$-полных задач.

Не всегда эвристические соображения по сокращению перебора удается довести до строгих формулировок, например, в виде граней. Тогда, реализуя в алгоритме некоторые представления "здравого смысла", мы не в состоянии обосновать их корректность или хотя бы достаточно удовлетворительно оценить качество алгоритма - это касается как времени его работы, так и точности получаемого решения. Именно такого сорта алгоритмы чаще всего и называют эвристическими.

Например, в упоминающейся выше задаче минимизации общей продолжительности работ для задачи сетевого планирования с ограниченными ресурсами эвристические соображения могут фор-

мулироваться в виде назначения приоритетов работ в конфликтных ситуациях. При построении календарного плана, т.е. назначении сроков начал работ, возникают конфликтные ситуации: для начала всех работ T_j, которые можно начать в соответствии с частичным порядком работ, не хватает ресурсов. Тогда мы вычисляем приоритеты работ, т.е. переводим на язык чисел соображения о том, какие работы сильнее всего тормозят дело. Когда приоритеты вычислены, конфликтная ситуация решается в пользу работ с большими приоритетами [I8].

Проще всего считать, что эвристические алгоритмы не имеют к математике никакого отношения. Однако исследование эвристик - это, кроме всего прочего, источник новых математических постановок и, возможно, новых идей. Время для математического исследования нестрогих выводов из "содержательных" соображений уже пришло, и такие исследования ведутся. В заключительной части нашего обзора будет приведен пример такого исследования - оценка качества выбора хода в игре двух противников с полной информацией при помощи игровой модели Шеннона.

А сейчас мы остановимся на некоторых общих соображениях, касающихся описания эвристических алгоритмов и некоторых их свойств.

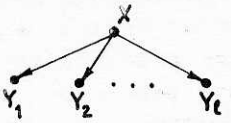
СТРАТЕГИЯ УВЕЛИЧЕНИЯ СВОБОДЫ ВЫБОРА

С.Ю.Маслов [I9] предложил в качестве модели для изучения эвристических алгоритмов использовать представление такого алгоритма в виде алгоритма поиска вывода в некотором исчислении. Такое исчисление строится по данной задаче и по возможности отражает наши соображения о способах ее решения. Исчисление состоит из конечного или бесконечного числа правил вывода. Разумеется, интересны достаточно простые правила вывода - например, типа правил вывода логических исчислений. В последнем случае мы имеем конечное число схем правил вывода, а число конкретизаций этих схем, т.е. собственно правил, бесконечно. Вывод в таком исчислении, назовем его F, начинается с конкретного исходного аргумента данной задачи, иными словами, мы рассматриваем решение частного случая нашей задачи. На каждом шаге вывода мы можем применить одно из конечного числа правил вывода F, пока не получим аксиому. Мы

считаем все правила вывода однопосылочными, ибо это не умаляет общности. Грубо говоря, правило вывода редуцирует нашу задачу, точнее, конкретизацию задачи, к другой задаче, возможно, несколько иного типа - последнее существенно, ибо для сохранения однопосылочности нашего исчисления мы записываем, например, редукцию к нескольким задачам того же типа как редукцию к списку задач.

Таким образом, конкретному аргументу задачи соответствует дерево всевозможных выводов в F этого аргумента. В терминах свойств этого дерева можно пытаться изучать влияние тех или иных эвристических идей на поиск вывода - см. [I9]. Важной характеристикой этого дерева, особенно если мы считаем все правила вывода равными или близкими по силе заложенных в них эвристических идей, является "кустистость" дерева. При упомянутых предположениях ряд соображений показывает, что наибольшие шансы на успех имеет стратегия движения в сторону "наибольшей кустистости", т.е. в сторону, где в наибольшей степени сохраняется свобода выбора.

В [I9] эти идеи формализованы следующим образом. Исчислению F ставится в соответствие алгоритм α, который каждому слову X, из которого можно вывести за один шаг слова $Y_1, Y_2, \ldots, Y_l$



, (4)

приписывает вероятности $p_1, \ldots, p_l$ (здесь p_i - вероятность перехода из X в Y_i, $p = 1 - \sum_i p_i \geq 0$; число p интерпретируется как "специальная" вероятность обрыва - вероятность того, что процесс поиска вывода прекращается на слове X). Алгоритм α определяет вероятностную меру на множестве всех выводов; для любого Q определяется вероятность p_Q обрыва поиска на слове Q. Тогда паре F, α соответствует информация

$$И(X) = И_{F,\alpha}(X) = \sum_{\substack{Q \text{ выводимо} \\ \text{в } F \text{ из } X}} p_Q \cdot |\log_2 p_Q|,$$

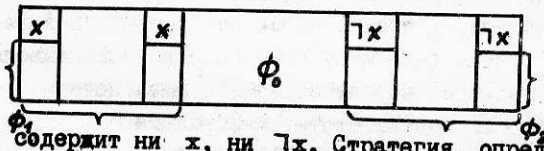
которая является мерой неопределенности на множестве выводимых слов.

Собственно вероятностные соображения могут при этом не

играть существенной роли. В связи с этим далее рассматривается "равновероятностный" алгоритм α_p, т.е. такой, что $p_1=\ldots=p_l$, а p одно и то же для всех x. При этом $И(x)$ превращается в функцию $И_p(x)$ от одного p. Существенная характеристика исчисления доставляется асимптотикой $И_p(x)$ при $p\to 0$. (Например, [I9], для конечных исчислений эта функция ограничена, для детерминированных она ведет себя как $\log\frac{1}{p}$, для исчисления с равномерно ограниченным ветвлением - как $\frac{1}{p}$, для произвольного канонического исчисления Поста - как $\frac{1}{p^2}$ и т.д.).

Стратегия увеличения свободы выбора состоит в том, что в разветвлении (4) движение идет по ветви, где достигается $\max\{И(Y_i): 1\le i\le l\}$. Такая стратегия естественна для многих лабиринтных проблем и позиционных игр с полной информацией. Чисто теоретически, соответствующие задачи решаются этой стратегией в том смысле, что при достаточно малом p перебор превращается в детерминированный поиск. Разумеется, для нетривиальных случаев p должно быть столь мало, что вычисление становится практически невыполнимым. Так что для реализации описанной стратегии нужны достаточно хорошие и просто вычислимые аппроксимации к $И$. Отчасти эта идея используется, например, в шахматных программах путем включения в оценочную функцию таких параметров как число возможных ходов.

Рассмотрим в качестве примера задачу установления пропозициональной выполнимости, речь о которой шла выше. Пусть у нас есть таблица ϕ, в которой надо найти открытый путь. Простейшее приближение исходит из анализа одной переменной. Пусть мы выбрали x. Тогда ϕ можно представить в виде



где ϕ_0 не содержит ни x, ни $\neg x$. Стратегия, определяемая по соответствующему приближению к $И$, предписывает добавление в конструируемый открытый путь переменной x, если ϕ_1 длиннее ϕ_2 (это является простейшей аппроксимацией меры свободы выбора, поскольку продолжение $\phi_0\phi_2$ с большей вероятностью содержит открытый путь, чем $\phi_0\phi_1$). В противоположном случае

выбирается продолжение конструируемого пути с помощью $\neg x$. Точнее, речь каждый раз идет о выборе такой переменной x (или $\neg x$), на которой реализуется максимальный перепад в длине соответствующих ϕ_1 и ϕ_2.

С так сформулированной стратегией распознавания выполнимости проводился эксперимент на случайных формулах (при равномерном распределении) - почти всегда открытый путь находился с первой попытки (программа Ю.Н.Курьерова).

Теперь мы остановимся на подходах, в которых в жертву приносится точность решения.

ПРИБЛИЖЕННЫЕ РЕШЕНИЯ

Вместо точных решений задач минимизации вида

$$f(x) = \min\{\nu(y) : |y| \le q(|x|) \ \&\ Q(x, y)\}$$

можно ставить вопрос о нахождении приближенных значений $f(x)$ (причем "приближенных" в разных смыслах), о поиске приближенного значения к y_0, такого что $f(x)=\nu(y_0)$. Мы уделим основное внимание задаче поиска приближенного значения $f(x)$. Начнем с приближений, описываемых в терминах относительной погрешности.

Будем говорить, что z является ε-приближением к $f(x)$, если

$$|z - f(x)| \le \varepsilon \cdot z.$$

Сложность нахождения такого z естественно оценивать как функцию от $|x|$ и $1/\varepsilon$.

В [20] было обнаружено, что по отношению к нахождению ε-приближений $\mathcal{NP}$-полные задачи ведут себя по-разному - одни остаются $\mathcal{NP}$-полными, другие попадают в класс $\mathcal{P}$. Пожалуй, наиболее известными примерами являются задачи о коммивояжере и задача с рюкзаке [5]. Нахождение ε-приближения к решению первой задачи является снова $\mathcal{NP}$-полной задачей [20]. С другой стороны, имеется алгоритм с верхней оценкой сложности полиномиальной от длины входа и $1/\varepsilon$, который находит ε-приближение к решению задачи о рюкзаке [21].

Интересен следующий факт. При наложении дополнительных "физических" ограничений на граф в задаче о коммивояжере она делается более "податливой" быстрому поиску приближенных решений. Так в [22]показано, что если веса на ребрах удовлетво-

ряют неравенству треугольника, то ε-приближенное решение можно найти за полиномиальное время.

Ситуация в целом не меняется, если мы будем рассматривать абсолютную (а не относительную) погрешность при определении приближенного решения. Примером задачи, сложность которой сохраняется в рассматриваемой ситуации, очевидно, является та же задача о коммивояжере. Отметим, еще один интересный пример, правда, касающийся задачи $\mathcal{NP}$-полнота которой неизвестна, - задача о построении минимальной дизъюнктивной нормальной формы полиномиально сводима к задаче нахождения такой формы, минимальной с аддитивной ошибкой h, где $h \geq 1$ - произвольная константа [23].

В качестве задачи, когда даже "небольшая" абсолютная ошибка дает возможность построить алгоритм для ее решения с полиномиальной сложностью, рассмотрим задачу о банкнотах из [24]:

$$\sum_{i=1}^{n} x_i \to \min;$$

$$\sum_{i=1}^{n} a_{ji} x_i \geq A_j, \quad j = 1,2,\ldots,m,$$

$$x_i \in \{0,1\}, \; i = 1,2,\ldots,n,$$

при $a_{ji} \geq 0$. Ограничимся случаем $m = 2$. Тогда эта задача может иметь следующие наглядные интерпретации:

а) имеется n банкнот двойного достоинства, i-я банкнота стоит a_i долларов и b_i франков. Требуется выбрать наименьшее количество банкнот так, чтобы их суммарная стоимость по долларам была не менее A и по франкам не менее B (эта первоначальная постановка задачи принадлежит А.С.Кронроду);

б) пусть требуется разыскать по телефону двух людей, причем известны вероятности a_i и b_i их нахождения по каждому (i-му) из n телефонных номеров ($\Sigma a_i = \Sigma b_i = 1$). Каков минимальный список телефонных номеров, гарантирующий вероятность $1 - \varepsilon_1$ отыскания первого человека и $1 - \varepsilon_2$ - второго?

Обсуждаемая задача является $\mathcal{NP}$-полной. Но если допустить ошибку (неоптимальность), равную единице, то соответствующее решение можно найти за время $O(n^2)$. В [24] также рассмотрен случай произвольного m.

Возможны и другие типы приближений, но в настоящее время они мало изучены (недавно М.Ш.Левин рассмотрел задачу о рас-

пределении блоков программы во внешней памяти при использовании, кроме обычных приближений к оптимальному значению, также и приближений к ограничениям и построил для нее алгоритм полиномиальной сложности).

ВЕРОЯТНОСТНЫЕ РЕШЕНИЯ

"Вероятность" при решении вычислительных задач может иметь различное происхождение. Мы можем иметь некоторое распределение вероятностей на входах и оценивать алгоритм с точки зрения вероятности получения точного решения или вероятности получения решения за определенное время. Мы можем использовать вероятностный алгоритм, т.е. алгоритм, который выбирает некоторые свои шаги из фиксированного числа возможных после обращения к датчику случайных чисел (то есть алгоритм, использующий метод Монте-Карло, в определенном смысле). Разумеется, можно рассматривать различные комбинации этих подходов, да и принципы использования вероятностных подходов не исчерпываются двумя идеями, упомянутыми выше. Интересно было бы выяснить связь между различными подходами - сравни [25].

Из распределений на входах, в основном, изучалось равномерное распределение (нормальное распределение рассмотрено в [26]). Для этого распределения показано, что весьма простые (иногда тривиальные) алгоритмы за небольшое время, примерно n^2 или меньше, с большой вероятностью решают задачу о коммивояжере, раскраске графа и т.д. см. [27], [28]. Уже тривиальность используемых алгоритмов заставляет немедленно поставить вопрос об адекватности рассматриваемого распределения физическому распределению входов. На примере задачи о коммивояжере видно, что практически встречающиеся графы весьма далеки от среднестатистического графа равномерного распределения. Еще более очевидна неадекватность этого распределения, когда входными данными являются искусственно создаваемые управляющие системы, например, схемы из функциональных элементов.

По-видимому, вопрос о построении адекватного распределения весьма сложен, он сродни вопросу об описании практических подклассов полиномиальной сложности. Одним из возможных подходов к решению этой задачи является анализ способов порождения практически встречающихся исходных данных; здесь можно

пытаться использовать те же соображения, о которых шла речь при обсуждении подклассов полиномиальной сложности.

Второй подход, а именно, использование алгоритмов с датчиком случайных чисел, несколько привлекательнее, по крайней мере, с точки зрения используемых алгоритмических конструкций. Наиболее известными результатами, полученными на базе этого подхода, являются "быстро" работающие алгоритмы для проверки простоты числа с вероятностью сколь угодно близкой к единице [29], - алгоритм из [29] делает $6m|x|$ шагов для проверки простоты x с вероятностью $1 - 2^{-m}$. При этом алгоритм использует простейший бернуллиевский датчик случайных чисел (проще говоря, для проверки простоты числа x он случайным образом выбирает натуральные числа из сегмента $[1, x-1]$ и вычисляет некоторые простые функции на них).

При анализе этого подхода возникает вопрос о реализуемости датчика случайных чисел. Вопрос о возможности использования датчика псевдослучайных чисел в алгоритмах обсуждаемого типа не изучен, а вопрос о существовании подходящих физических датчиков остается открытым и, более того, возможность его положительного решения вызывает сомнения. В связи с последним предположением имеет смысл упомянуть неоконченные эксперименты покойного М.М.Бонгарда по алгоритмическому прогнозированию поведения счетчика радиоактивного распада. Он построил несложную программу, которая по некоторому начальному куску показаний счетчика либо отказывалась "играть" с ним (т.е. предсказывать его поведение), либо "играла" с ним. В целом небольшой выигрыш был на стороне алгоритма. Бонгард собирался проверить, не объясняется ли этот выигрыш усталостью воспринимающего датчика. Он не успел сделать этого. Очевидно, что подобные вопросы необычайно интересны и заслуживают того, чтобы их изучение было продолжено. Особенно интересно изучать физические датчики с высокой плотностью выхода (например, 10^4 бит/с).

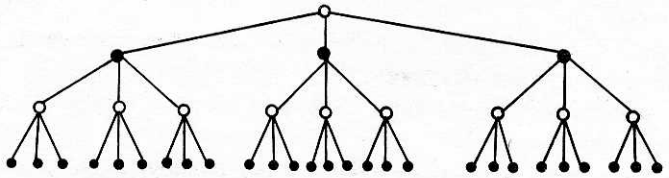
Вероятностные идеи привлекательны как средство анализа эвристических алгоритмов. Эта возможность упоминалась в предыдущем разделе, но там они играли второстепенную роль. Мы опишем подход, предложенный Г.М.Адельсоном-Вельским и В.Л.Арлазаровым [30], в котором вероятностные идеи используются по существу.

ИСПОЛЬЗОВАНИЕ ВЕРОЯТНОСТНОГО ОРАКУЛА

Подход из [30] , который назван использованием вероятностного оракула, применяется для анализа алгоритмов (прежде всего, игры в шахматы), не всегда дающих правильный ответ. В ситуации, отличной от рассмотренных выше, оценивается вероятность правильного ответа и показано, что в некоторых случаях ее можно сделать достаточно большой.

Представим себе вычисление, использующее оракул (т.е. возможность за один шаг получать решение некоторой задачи для аргумента, вырабатываемого этим вычислением), но такой оракул, который не всегда дает правильный ответ - например, дает правильный ответ только с некоторой вероятностью. Соответствующим образом ставится задача об оценке корректности и сложности алгоритма - они будут оцениваться при заданной вероятности получить правильный ответ оракула. Оракул [30] дает ответ для входов массовой задачи, но для данного входа он всегда дает один и тот же ответ, вероятности определены на множестве всех интересующих нас входов задачи.

Рассмотрим игру двух противников с полной информацией, например, шахматы. Как показал Цермело [31], каждая позиция этой игры имеет истинную оценку, т.е. результат продолжения партии из этой позиции, если оба противника будут играть лучшим образом. Рассмотрим конечное дерево игры из данной позиции



Тогда оценки заключительных позиций известны (они равны результату закончившейся партии), а оценки остальных позиций рекуррентно определяются по формуле Цермело, которую в интересующем нас случае (имеется в виду, что в рассматриваемой игре, как и в шахматах, сумма игры равна I и противники ходят по очереди):

$$Z(A) = \max_{(A,B)} (1 - Z(B)),$$

где $Z(A)$ - выигрыш той стороны, которой принадлежит ход в позиции A, (A, B) - допустимый ход из позиции A в позицию B и max берется по всем таким ходам.

Однако, когда дерево игры необозримо велико (а данное рассмотрение ориентировано как раз на такую игру - шахматы) вычисление оценки по формуле Цермело фактически невозможно. Шеннон [32], [33] предложил рассмотреть не все дерево, а лишь часть его, начиная от корня до глубины n, и считать позиции глубины n заключительными, а оценки в них вычислять, формализовав общие соображения шахматной теории о сильных и слабых сторонах позиций. Таким образом, для каждой позиции определено значение некоторой функции $f(A)$, которую мы будем называть оценочной, а рассматриваемая часть дерева исходной игры и значения оценочности функции в позициях глубины n определяют модельную игру, причем оценки ее позиций машина может вычислить.

Почему со времен Шеннона все надеются, что вычисленные таким образом модельные оценки позиций будут близки к их истинным оценкам и при том эта близость будет возрастать с ростом n —глубины перебора?

В [30] исследовано это гипотетическое соотношение модельных оценок в предположении, что связь между f и Z имеет вероятностный характер. Там рассмотрена идеализированная игра, для которой все нужные вероятности легко считаются и показано, что качественные результаты сохраняются и при более реалистических допущениях.

Пусть G - игра двух противников, для позиций которых выполняются следующие условия:

1) возможные исходы игры - только выигрыш $(Z(A)=1)$ или проигрыш $(Z(A) = 0)$;

2) из каждой незаключительной позиции можно сделать одно и то же число ходов m, а все заключительные позиции имеют одну и ту же достаточно большую глубину N;

3) противники ходят по очереди;

4) в каждой выигранной позиции A (т.е. такой, что $Z(A)=1$; лучше было бы называть такую позицию выигрывающей) есть ровно s выигрывающих ходов (A, B), где $Z(B) = 0$; остальные $m-s$ ходов - проигрывающие (в проигранной позиции все ходы проигрывающие);

5) оценочная функция f(A) является случайной величиной, независимой для разных позиций, причем для всех выигранных позиций одинаковы вероятности

$$P(f(A) = 1 \mid Z(A) = 1) = p,$$

а для всех проигранных позиций

$$P(f(A) = 1 \mid Z(A) = 0) = q.$$

Таким образом, значения f - это ответы оракула, правильные с некоторыми вероятностями. Можно обратиться к этому оракулу с вопросом об оценке данной позиции; но мы никак не сможем повысить вероятность правильности оценки путем повторного обращения к оракулу: вторичное обращение с вопросом об оценке той же позиции дает старый ответ.

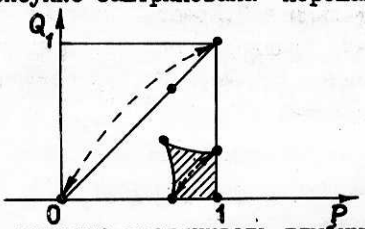
Обозначим оценку позиции A , являющейся корнем модели Шеннона для нашей игры через $S_n(A)$, где n - глубина модели, причем n много меньше N . Определим вероятности

$$P_n = P(S_n(A) = 1 \mid Z(A) = 1),$$
$$Q_n = P(S_n(A) = 1 \mid Z(A) = 0).$$

Они вычисляются по рекуррентным формулам

$$P_0 = p, \quad Q_0 = q,$$
$$P_{n+1} = 1 - P_n^{m-s} Q_n^s,$$
$$Q_{n+1} = 1 - P_n^m.$$

В зависимости от значений p и q , либо $P_n \to 1$ и $Q_n \to 0$ (т.е. вероятность получить правильную оценку растет с увеличением глубины), либо обе вероятности поочередно приближаются к I и 0, что означает бесполезность увеличения глубины перебора. На приведенном ниже рисунке заштрихована "хорошая" область



значений, для которых полезно увеличивать глубину (и отмечены стационарные точки и пары точек).

Проведенные исследования показали, что

I) в рассмотренной задаче использование вероятностного оракула, описанного выше типа, дает возможность построить ал-

горитм, который дает правильный ответ с вероятностью не меньше $1 - e^{-\theta t}$, где t - время счета, а θ - положительная константа;

2) для реализации такого алгоритма оракул должен быть достаточно хорошим, т.е. вероятности его правильных ответов должны быть больше некоторых пороговых значений; кроме того, выигрыш в игре должен достигаться не единственным образом (то есть число выигрывающих ходов должно быть больше единицы);

3) вычисления модельных оценок по формуле Цермело можно заменить вычислениями по более сложным формулам, отражающим вероятностные предположения об игре, и улучшить значения порогов (этот результат принадлежит Н.Э.Косачевой).

Качественный характер результатов сохраняется для моделей, более близких к реальным играм. А именно:

1) можно отказаться от независимости значений оценочных функций и ввести более реалистичное предположение о положительной корреляции значений оценочных функций до и после хода;

2) предположение о равномерном характере игры (одно и то же число ходов во всех позициях и т.п.) можно заменить предположением о том, что задана вероятность числа ходов и т.п.;

3) можно рассмотреть случай, когда f и Z принимают не только значения 0 и 1, но и все промежуточные;

4) можно ввести содержательные характеристики типа эндшпильности, остроты позиции и т.п., от которых будут зависеть рассматриваемые вероятности.

Мы признательны А.П.Ершову, который стимулировал написание этой статьи, пригласив нас на Ургенчский симпозиум, посвященный памяти Аль-Хорезми. Мы благодарим С.Ю.Маслова за полезные обсуждения.

Литература

1. Роджерс Х. Теория рекурсивных функций и эффективная вычислимость. - М.: Мир, 1972.
2. Слисенко А.О. Сложностные задачи теории вычислений: Препринт, - М., 1979. - В надзаг.: Научн. совет по

компл. проблеме "Кибернетика" АН СССР.
3. Fischer M.J., Rabin M.O. Super-exponential complexity of Presburger arithmetic. - "Complexity Comput. (SIAM-AMS Proc., v.7)" Providence, R.I., 1974, p. 27-41.
4. Кук С. Сложность процедур вывода теорем. - Кибернет. сб., М.: Мир, 1975, новая сер., вып. 12, с. 5-15.
5. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. - М.: Мир, 1979.
6. Garey M.R., Johnson D.S., Tarjan R.E. The planar Hamiltonian circuit problem is NP-complete. - SIAM J. Comput., 1976, v.5, no.4, p. 704-714.
7. Land A.H., Doig A.G. An automatic method of solving discrete programming problems. - Econometrica, 1960, v.28, no.3, p. 497-520.
8. Слисенко А.О. Финитный подход к задаче оптимизации алгорифмов установления выводимости. - Записки научн. семинаров Ленинград. отд. Матем. ин-та АН СССР, 1975, т. 49, с. 123-130.
9. Lewis H.R. Complexity of solvable cases of decision problem for predicate calculus. - IEEE 19th Annu. Symp. Found.Comput.Sci., Ann Arbor, Michigan, 1978, p. 35-47.
10. Матросов В.М., Васильев С.Н., Каратуев В.Г., Новиков М.А., Суменков Е.А., Ядыкин С.А. Машинный вывод теорем о динамических свойствах с вектор-функциями Ляпунова. - Кибернетика, 1979, № 2, с. 27-36.
11. Калужнин Л.А., Стогний А.А. Теория и практика применения ЭВМ в алгебраических исследованиях. - В кн.: Вычисления в алгебре и комбинаторике. Киев, 1978, с. 3-40.
12. Стругацкий А.Н., Стругацкий Б.Н. Понедельник начинается в субботу. - В кн.: Библиотека современной фантастики, М., 1966, т. 7.
13. Corneil D.G., Gotlieb C.C. An efficient algorithm for graph isomorphism. - J. Assoc. Comput. Mach., 1970, v.17, no.1, p. 51-64.
14. Slisenko A.O. Computational complexity of string and graph identification. - Lecture Notes Comput. Sci., 1979, v.74, p. 182-190.

I5. Адельсон-Вельский Г.М., Диниц Е.А., Карзанов А.В. Потоковые алгоритмы. - М.: Наука, I975.
I6. Galil Z., Naamad A. Network flow and generalized path compression. - Proc. 11th Annu. ACM Symp. Theory Comput., Atlanta, Georgia, 1979, p. 13-26.
I7. Хачиян Л.Г. Полиномиальный алгоритм в линейном программировании. - В кн.: Докл. АН СССР, I979, т. 244, № 5, с. I063-I066.
I8. Келлен Дж. Календарное планирование. - В кн.: Экономические модели в управлении. - М.: ИЛ, I967.
I9. Маслов С.Ю. Информация в исчислении и рационализация переборов. - Кибернетика, I979, № 2, с. 20-26.
20. Sahni S., Gonzales T. P-complete problems and approximate solutions. - IEEE 15th Annu. Symp. Switch. and Automata Theory, New Orleans, 1974, p. 28-31.
2I. Ibarra O.H., Kim C.E. Fast approximation algorithms for the knapsack and sum of subset problems. - J. Assoc. Comput. Mach., 1975, v.22, no.4, p. 463-468.
22. Christofides N. Worst-case analysis of a new heuristic for the traveling salesman problem. - In: Algorithms and Complexity. New Directions and Recent Results, ed. Traub. Academic Press, Inc., 1976, p. 441.
23. Нигматуллин Р.Г. Сложность приближенного решения комбинаторных задач. - В кн.: Докл. АН СССР, I975, т. 224, № 2, с. 289-292.
24. Диниц Е.А., Карзанов А.В. Булева задача оптимизации при ограничениях одного знака: Препринт. - М., I978 - 42 с. В надзаг.: ВНИИ Системных Исследований.
25. Yao A.C.-C. Probabilistic computations: toward a unified measure of complexity. - IEEE 18th Annu. Symp. Found. Comput. Sci., Providence, R.I., 1977, p. 222-227.
26. Lueker G.S. Maximization problems on graphs with edge weights chosen from a normal distribution. - Proc. 10th Annu. ACM Symp. Theory Comput., San Diego,1978, p. 13-19.
27. Гимади Э.Х., Глебов Н.И., Перепелица В.А. Алгоритмы с оценками для задач дискретной оптимизации. - Пробл. кибернет. - М.: Наука, I976, вып. 3I, с. 35-42.

28. Angluin D., Valiant L.G. Fast probabilistic algorithms for Hamiltonian circuits and matching. - CSR-17-77, Dept. Comput. Sci., Univ. of Edinburgh, 1977, p. 59.
29. Solovey R., Strassen V. A fast Monte-Carlo test for primality. SIAM J. Comput., 1977, v.6, no.1, p. 84-85.
30. Адельсон-Вельский Г.М., Арлазаров В.Л., Донской М.В. Программирование игр. - М.: Наука, 1978.
31. Цермело Э. О применении теории множеств к теории шахматной игры. - В кн.: Матричные игры. М., Физматгиз, 1961, с. 167-172.
32. Шеннон К.Э. Машина для игры в шахматы. - В кн.: Работы по кибернетике и теории информации. М., Физматгиз, 1956, с. 180-191.
33. Шеннон К.Э. Играющие машины. - В кн.: Работы по кибернетике и теории информации. М., Физматгиз, 1956, с. 216-223.