Министерство образования и науки Российской Федерации
Ярославский государственный университет им. П.Г. Демидова
Кафедра дискретного анализа

А. В. Николаев

Геометрический подход к задаче о разрезе

Методические указания

Рекомендовано научно-методическим советом
университета для студентов, обучающихся по направлению 010400.68
«Прикладная математика и информатика»

Ярославль 2014

УДК 519.16 + 514.17
ББК В 151я73 + В 176я73

Рекомендовано
Редакционно-издательским советом университета
В качестве научного издания
План 2013-2014 года.

Рецензент:
Кафедра дискретного анализа
Ярославского государственного университета им. П.Г. Демидова

Николаев, А.В. Геометрический подход к задаче о разрезе: метод. указания / А.В. Николаев; Яросл. гос. ун-т им. П. Г. Демидова. – Ярославль: ЯрГУ, 2014. – 76 с.

В настоящем издании приводятся различные подходы и алгоритмы решения, а также построения оценок сложности, задач комбинаторной оптимизации геометрическими методами. На примере задачи о разрезе в графе и ряда других комбинаторных задач, таких как задача коммивояжера и задача о минимальном остовном дереве, излагаются основные идеи геометрической интерпретации задач и алгоритмов.

Предназначено для студентов, обучающихся по направлению 010400.68 «Прикладная математика и информатика» (дисциплина «Геометрические вопросы комбинаторной оптимизации», цикл М1) очной формы обучения.

УДК 519.16 + 514.17
ББК В 151я73 + В 176я73

Содержание

1. Задача о разрезе 4
2. Сложность задач 9
3. Геометрическая интерпретация задач 12
4. Выпуклые многогранники 15
5. Многогранники задач 21
6. Расширенная формулировка 30
7. Конусные разбиения 35
8. Линейные разделяющие деревья 39
9. Алгоритмы прямого типа 45
10. Задача о разрезе с неотрицательными весами 51
11. Релаксационные многогранники 56
12. Полуопределенное программирование 63
Список литературы 72

1. Задача о разрезе

Одной из основных задач прикладной математики является задача комбинаторной оптимизации, которая заключается в поиске оптимального объекта в некотором конечном множестве подобных объектов.

Общую постановку задачи комбинаторной оптимизации можно сформулировать следующим образом: задано конечное множество X и функция $f: X \to \mathbb{R}$, определенная на этом множестве и принимающая действительные значения. Требуется найти такой элемент $x^* \in X$, для которого $f(x^*) \geq f(x)$ (задача максимизации) или $f(x^*) \leq f(x)$ (задача минимизации) для любого $x \in X$.

На первый взгляд решение задачи комбинаторной оптимизации не представляет никакой сложности. В силу конечности множества X достаточно проверить все элементы X (перебрать все возможные варианты) и выбрать оптимальный. Такой подход носит название полного перебора и вполне допустим для некоторых простых задач, таких как поиск наибольшего простого трехзначного числа. Интересно, что в английском языке данная техника имеет менее нейтральное название: brute-force search («поиск методом грубой силы») или exhaustive search (дословно «исчерпывающий, истощающий поиск»). Дело в том, что, как правило, элементы множества X (варианты решения) не задаются напрямую, а возникают опосредованно и имеют комбинаторную природу. При этом происходит, так называемый, «комбинаторный взрыв», когда число возможных вариантов растет невероятно быстро с экспоненциальной или сверхэкспоненциальной скоростью.

Одной из фундаментальных задач комбинаторной оптимизации является ***задача поиска минимального (максимального) разреза в неориентированном взвешенном графе*** [17].

Заданы: неориентированный граф $G = (V, E)$ и некоторая функция $C: E \to \mathbb{R}_{\geq 0}$, каждому ребру $e \in E$ сопоставляющая неотрицательное действительное число $C(e)$, называемое весом ребра.

Требуется найти такое разбиение множества вершин V на два непересекающихся подмножества P и Q (разрез), чтобы сумма весов ребер из

E, соединяющих вершины из различных подмножеств, была наименьшей (минимальный разрез) или наибольшей (максимальный разрез).

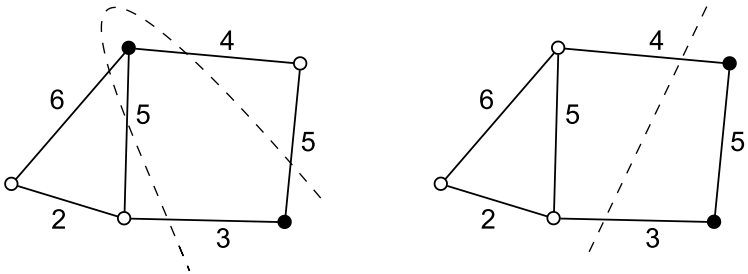

Рис. 1. Максимальный и минимальный разрезы в графе

Отметим, что в классической постановке задачи речь идет о неотрицательных весах ребер. Задача с произвольными весами также имеет право на существование, однако следует отметить, что в такой постановке задачи на минимум и на максимум принимают принципиально другой характер и фактически совпадают. Действительно, для преобразования задачи о минимальном разрезе с произвольными весами в задачу о максимальном разрезе достаточно поменять знаки у всех ребер графа.

Общее количество возможных разрезов в графе на n вершинах, учитывая что подмножества P и Q не упорядочены, составляет 2^{n-1}. Таким образом, решение задачи о разрезе полным перебором всех вариантов в графе на 100 вершинах потребует проверки $2^{99} \approx 6.34 \cdot 10^{29}$ различных разрезов. Причем, для проверки каждого разреза необходимо просмотреть все ребра графа, максимальное число которых равно $n(n-1)/2$.

Производительность современных суперкомпьютеров измеряется в петафлопсах, т.е. 10^{15} операций с плавающей запятой в секунду. Так теоретический пик производительности суперкомпьютера Cray Titan на 18688 шестнадцатиядерных процессорах AMD Opteron 6274 и 18688 ускорителей (GPU) Nvidia Tesla K20x составляет 27 петафлопс ($2.7 \cdot 10^{16}$ операций в секунду). Соответственно, для решения задачи о разрезе в графе на 100 вершинах полным перебором ему потребуется приблизительно 3.5 миллиарда лет. Причем, при добавлении в граф каждой новой вершины сложность задачи и, соответственно, время перебора будет возрастать в 2 раза.

В силу эффекта комбинаторного взрыва любой алгоритм, в том числе полный перебор, трудоемкость которого нельзя оценить сверху полиномом от длины входа считается неэффективным.

Напомним, что трудоемкость алгоритмов как правило оценивается не точным значением, а асимптотически, указывая порядок роста функции с увеличением размера задачи и объема входных данных. Говорят, что

$$f(n) = O\big(g(n)\big) \Leftrightarrow \exists (C > 0), n_0 \colon \forall (n > n_0)\, f(n) \leq C \cdot g(n),$$

функция $f(n)$ асимптотически ограничена сверху функцией $g(n)$ с точностью до множителя. Алгоритм считается полиномиальным, если для функции его трудоемкости $f(n)$ найдется такое $k \in \mathbb{N}$, что

$$f(n) = O(n^k).$$

Задача о минимальном разрезе играет особую роль в планировании коммуникационных сетей и определении их надежности, выявляя наиболее уязвимое место в сети. Действительно, если по некоторым причинам часть связей в сети будет нарушена, грамотно разработанная сеть по-прежнему сможет передавать информацию между любой парой узлов, практически без потери скорости или объема передачи. Однако если нарушить связи, входящие в минимальный разрез графа, это может привести к значительному падению в качестве работы сети или даже полному ее коллапсу.

Ввиду особой практической важности задача о минимальном разрезе часто являлась объектом пристального внимания. Как правило, при ее решении опираются на другую важную задачу из теории графов: ***поиск максимального потока в транспортной сети***.

Заданы: транспортная сеть – ориентированный граф $G = (V, E)$, в котором каждое ребро $(u, v) \in E$ имеет неотрицательную пропускную способность $c(u, v) \geq 0$.

Выделяются две вершины: источник s и сток t, такие, что любая другая вершина сети лежит на пути из s в t.

Потоком называется функция $f \colon E \to \mathbb{R}$, каждому ребру сопоставляющая действительное число, со следующими свойствами:

1) *поток не может превышать пропускную способность:*

$$f(u, v) \leq c(u, v);$$

2) *поток из u в v должен быть противоположным потоку из v в u (антисимметричность):*

$$f(u,v) = -f(v,u);$$

3) *сохранение потока:*

$$\sum_{w \in V} f(u,w) = 0,$$

для всех $u \in V$, кроме источника и стока.

Требуется найти такой поток по транспортной сети, чтобы сумма потоков выходящих из истока, или, что тоже самое, сумма потоков входящих в сток была максимальной.

Связь задач о минимальном разрезе в графе и максимальном потоке в транспортной сети была установлена Л. Фордом и Р. Фалкерсоном в их классическом результате [15].

Теорема Форда – Фалкерсона. *Величина максимального потока в транспортной сети равна пропускной способности минимального разреза.*

Наиболее известная модификация алгоритма Форда-Фалкерсона, разработанная Е. Диницом, Дж. Эдмондсом и Р. Карпом на основе поиска кратчайшего дополняющего пути, решает задачу о максимальном потоке (и соответственно о минимальном разрезе) в транспортной сети за время $O(|V|^2|E|)$ [12,13].

Решение задачи о минимальном разрезе для транспортной сети легко обобщить для любого неориентированного графа: достаточно произвольным образом выбрать вершину-источник s, а в качестве стока t последовательно перебрать все остальные вершины графа, после чего из полученных данным способом разрезов выбрать минимальный. Таким образом, решение задачи о минимальном разрезе потоковыми методами потребует $|V| - 1$ последовательных применений алгоритма Форда-Фалкерсона, и в модификации Диница-Эдмондса-Карпа будет иметь трудоемкость $O(|V|^3|E|)$. Дж. Хао и Дж. Орлин показали, что в силу специфики алгоритма для неориентированного графа все $|V| - 1$ задач о максимальном потоке можно решить за время асимптотически равное однократному применению алгоритма Форда-Фалкерсона [22]. Наконец, М. Стоер и Ф. Вагнер продемонстрировали эффективный алгоритм решения задачи о минимальном разрезе, не использующий потоковые методы и имеющий трудоемкость $O(|V||E| + V^2 \log |V|)$ [32].

В свою очередь, задача о нахождении максимального разреза графа имеет важное прикладное значение в кластерном анализе: задаче разбиения некоторой выборки данных на подмножества, называемые кластерами, так, чтобы каждый кластер состоял из схожих объектов, а объекты разных кластеров существенно отличались. Каждому объекту данных можно сопоставить вершину графа данных, а каждой паре объектов – ребро с весом, соответствующим их «схожести». Так для числовых данных в качестве такого критерия может быть использовано евклидово расстояние. Кроме того, Ф. Барахона, М. Гретчел, М. Юнгер и Г. Рейнелт показали, что задача о максимальном разрезе графа напрямую возникает при проектировании интегральных схем [1].

Несмотря на практическую важность задачи о максимальном разрезе, не удалось построить ни одного алгоритма ее решения отличного от полного перебора. Исключениями являются лишь некоторые частные случаи задачи, например, поиск максимального разреза в планарном графе, для которого были найдены полиномиальные алгоритмы [21]. Более того, было установлено, что задача о поиске максимального разреза в графе принадлежит классу NP-трудных задач и является одной из фундаментальных труднорешаемых задач. Отметим, что такая ситуация не является уникальной. В качестве аналога можно привести другую пару хорошо известных задач из теории графов: поиск кратчайшего пути (полиномиальна) и поиск самого длинного пути в графе (NP-трудна) [17].

Упражнение. Проверить решение задачи о минимальном разрезе, приведенное на Рис. 1, с помощью алгоритма Форда-Фалкерсона.

2. Сложность задач

Напомним, что в теории сложности алгоритмов выделяют, как правило, три базовых класса задач (общая классификация содержит более 60 классов сложности): P (polynomial), NP (non-deterministic polynomial) и NPC (non-deterministic polynomial complete, или NP-полные задачи).

В данной терминологии, как правило, рассматриваются задачи распознавания (задачи, предполагающие ответ «да» или «нет»), однако задачи распознавания и оптимизации эквивалентны с точки зрения принципиальной трудоемкости: если существует полиномиальный алгоритм для первой задачи, то и вторая полиномиально разрешима, и наоборот. ***В форме задачи распознавания задача о разрезе*** примет вид:

Существует ли во взвешенном графе $G = (V, E)$ с функцией весов $C: E \to \mathbb{R}_{\geq 0}$ такой разрез, что сумма весов входящих в него ребер не больше некоторой фиксированной величины S (минимальный разрез) или не меньше некоторой фиксированной величины T (максимальный разрез)?

В класс P входят задачи комбинаторной оптимизации, для которых существуют алгоритмы решения, время работы которых полиномиально зависит от размера входных данных. Известный тезис Кобхэма-Эдмондса гласит: вычислительная задача может быть решена на некотором вычислительном устройстве за разумное время только в том случае, если ее трудоемкость оценивается сверху полиномиальной функцией [6]. Поэтому алгоритмы из класса P носят название эффективных. Следует отметить, что данное положение не совсем бесспорно: алгоритм с трудоемкостью $10^{100}n$ считается эффективным, так как это линейная функция, а алгоритм с экспоненциальной трудоемкостью $2^{0.00001n}$ – неэффективным. При этом первый не представляется возможным реализовать на практике даже при $n = 1$, в то время как трудоемкость второго и при $n = 10^6$ составляет всего лишь 1024 операции. Тем не менее, в общем случае, за исключением искусственно сконструированных примеров, отождествление полиномиальных алгоритмов с эффективными, а неполиномиальных с неэффективными (в силу эффекта комбинаторного взрыва) представляется верным, и такая классификация закрепилась.

В класс NP входят задачи, решение которых можно проверить за полиномиальное время на машине Тьюринга. Так задача о разрезе в форме

распознавания, очевидно, принадлежит классу NP, так как если известен некоторый разрез, то для его проверки достаточно сложить веса всех входящих в разрез ребер, число которых не превосходит $n(n-1)/2$, и сравнить с заданной фиксированной величиной S или T. Проверка произвольного разреза потребует не более $O(n^2)$ операций, где n – число вершин графа G.

Вложенность $P \subseteq NP$ прямо следует из определения классов. Вопрос равенства или неравенства P и NP остается нерешенным, является одним из основных вопросов современной математики и входит в список семи задач тысячелетия по версии Математического института Клэя [5].

Фундаментальным прорывом в теории сложности стала теорема Кука-Левина об NP-полноте задачи выполнимости булевых функций [7,40]. Задача называется NP-полной (NP-complete, NPC), если к ней можно свести любую другую задачу из класса NP за полиномиальное время. В дальнейшем класс NP-полных задач был значительно расширен [17], включив в себя, в частности, задачу о максимальном разрезе в форме распознавания, и значительно превзошел по размерам класс P – полиномиально разрешимых задач. В настоящий момент большинство известных труднорешаемых задач (задач для которых не известно полиномиальных алгоритмов решения) отнесены в класс NPC, за исключением таких задач как изоморфизм графов, факторизация целых чисел и дискретное логорифмирование, для которых было доказано, что в случае $P \neq NP$ они не принадлежат ни P ни NP и образуют промежуточный класс NP-intermediate.

Задача о максимальном разрезе в оптимизационной форме входит в еще один класс сложности, так называемых NP-трудных задач (NP-hard), состоящий из задач не уступающих по сложности NP-полным задачам, но не входящих в класс NP (класс задач распознавания).

Гениальность идеи NP-полных задач заключается в том, что построив эффективный алгоритм решения произвольной NP-полной задачи (например, задачи о разрезе) можно перекинуть мостик на любую другую задачу распознавания из класса NP и решить ее за достаточно быстрое (полиномиальное) время. Подобный вариант равенства классов сложности P и NP перевернул бы картину современной математики. Например, было бы опровергнуто существование односторонних функций, на которых построены все алгоритмы шифрования, что положило бы конец всей современной

криптографии. Более того, Стивен Кук [8] показал, что при выполнении $P = NP$ компьютер смог бы построить формальное доказательство абсолютно любой теоремы (разумной длины), так как формальное доказательство легко проверить за полиномиальное время. Однако вопрос так и остается открытым.

Упражнение. Доказать NP-полноту задачи о максимальном разрезе, сведя к ней другую известную NP-полную задачу, например, максимальную 2-выполнимость.

3. Геометрическая интерпретация задач

Напомним несколько базовых определений из теории комбинаторной оптимизации.

Определение. *Массовая задача (задача) – общий вопрос, на который следует дать ответ. Задается списком параметров и критериями, которым должен удовлетворять ответ.*

Определение. *Индивидуальная задача – задача, полученная из массовой присвоением конкретных значений всем параметрам.*

Кроме того, можно рассмотреть массовую задачу с фиксированием одного из параметров.

Массовой задачей о разрезе будет общая постановка задачи без указания размера графа и функции весов ребер.

Рассмотрим массовую задачу о разрезе с фиксированным числом вершин графа $G = (V, E)$, $|V| = n$. Без ограничения общности положим, что граф G является полным. В противном случае в индивидуальной задаче для отсутствующих ребер можно задать нулевые веса. Число ребер в полном графе обозначим через $d = |E| = n(n-1)/2$. Упорядочим ребра в графе и присвоим каждому уникальный номер от 1 до d.

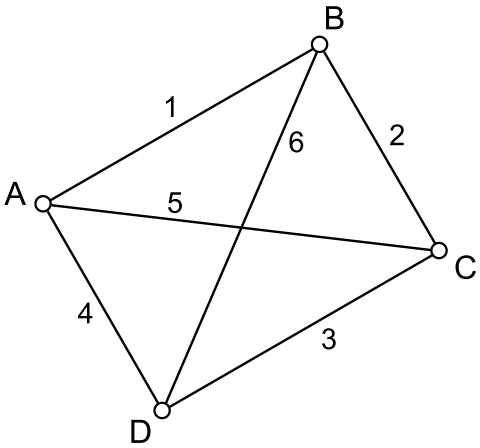

Рис. 2. Пример нумерации ребер в графе K_4

Каждому разрезу в графе G сопоставим характеристический $0/1$-вектор (для задачи о разрезе он также называется разрезным вектором) из пространства $\mathbb{R}^d$ по следующему правилу: если ребро попало в разрез (соединяет вершины из различных подмножеств), то соответствующая координата приравнивается к 1, в противном случае – к 0. Так, для полного графа на четырех вершинах (Рис. 2) существует 8 различных разрезов, которые можно закодировать следующим образом:

P	Q	Разрезной вектор
$\emptyset$	A,B,C,D	(0,0,0,0,0,0)
A	B,C,D	(1,0,0,1,1,0)
B	A,C,D	(1,1,0,0,0,1)
C	A,B,D	(0,1,1,0,1,0)
D	A,D,C	(0,0,1,1,1,0)
A,B	C,D	(0,1,0,1,1,1)
A,C	B,D	(1,1,1,1,0,0)
A,D	B,C	(1,0,1,0,1,1)

Таблица 1. Разрезные вектора для графа K_4.

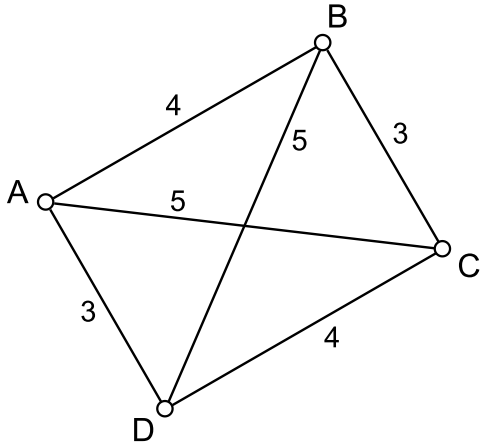

Рис. 3. Пример индивидуальной задачи о разрезе

Обозначим множество всех разрезных векторов массовой задачи о разрезе с фиксированным числом вершин графа n через X_n. Тогда любая индивидуальная задача о разрезе (Рис. 3) может быть представлена в форме оптимизации линейной целевой функции на множестве точек в евклидовом векторном пространстве $\mathbb{R}^d$

$$c^T x = (c, x) \to min, \;\; x \in X_n \subset \mathbb{R}^d \text{ (минимальный разрез)},$$

$$c^T x = (c, x) \to max, \;\; x \in X_n \subset \mathbb{R}^d \text{ (максимальный разрез)},$$

где вектор $c \in \mathbb{R}^d$ – это вектор весов ребер (в примере из Рис. 3, вектор $c = (4,3,4,3,5,5)$).

Обратим внимание, что к указанной форме оптимизации линейной функции на множестве точек в $\mathbb{R}^d$ может быть приведена любая задача по выбору оптимального подграфа: задача коммивояжера, задача о кратчайшем пути, задача о минимальном остовном дереве, задача о максимальном взвешенном паросочетании и другие. Достаточно занумеровать все ребра в графе и закодировать все возможные решения задачи характеристическими векторами, координаты которых равны 1, если соответствующее ребро попадает в подграф, взяв в качестве целевого вектора c вектор весов ребер.

Аналогично можно рассмотреть и другие задачи комбинаторной оптимизации, например задачу о назначениях (оптимальное назначение работников на должности), для которой множество X_n состоит из $n!$ перестановочных матриц назначений, а вектор c – матрица эффективности (соответствия) работника должности [35].

Таким образом, к указанной форме можно привести не только задачу о разрезе, но и большинство задач комбинаторной оптимизации при условии линейности целевой функции, которое, как правило, выполняется.

Упражнение. Описать геометрическую интерпретацию задачи сортировки n действительных чисел по возрастанию в форме оптимизации линейной функции на множестве точек в евклидовом пространстве.

4. Выпуклые многогранники

Определение. *Отрезком, соединяющим точки x и y из $\mathbb{R}^d$, называется множество точек вида $\lambda x + (1-\lambda)y$, где $\lambda \in \mathbb{R}$ и $0 \le \lambda \le 1$ (обозначение $[x, y]$).*

Определение. *Множество S называется выпуклым (convex set), если вместе с любыми своими двумя точками оно содержит соединяющий их отрезок*

$$\forall x, y \in S\colon\ [x, y] \subset S.$$

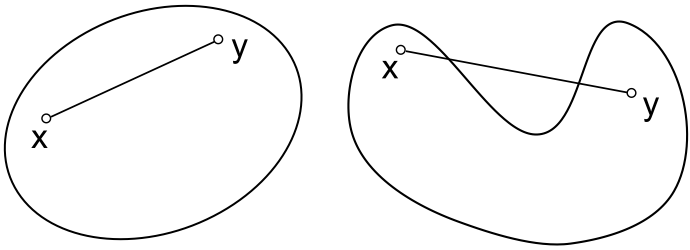

Рис. 4. Примеры выпуклого и невыпуклого множеств

Особый интерес в контексте геометрического подхода к задачам комбинаторной оптимизации представляют следующие примеры выпуклых множеств:

Определение. *Гиперплоскостью $H \subset \mathbb{R}^d$ называется подпространство размерности $d-1$, представляющее собой множество решений одного линейного уравнения*

$$H = \{x \in \mathbb{R}^d\colon\ (a, x) = b\}.$$

Каждая гиперплоскость разбивает пространство $\mathbb{R}^d$ на два полупространства H^+ и H^-, которые называются замкнутыми, если содержат гиперплоскость H (множества решений нестрогих неравенств $(a, x) \ge b$ и $(a, x) \le b$), или открытыми, если не содержат H (множества решений строгих неравенств $(a, x) > b$ и $(a, x) < b$ соответственно).

Нетрудно убедиться, что гиперплоскость и полупространство являются выпуклыми множествами.

Определение. *Многогранным множеством (полиэдром, polyhedron) называется пересечение конечного числа замкнутых полупространств, т.е. множество решений конечной системы линейных неравенств.*

Теорема. *Пересечение произвольного числа выпуклых множеств есть выпуклое множество.*

Таким образом, полиэдр – это выпуклое множество.

Определение. *Ограниченное многогранное множество называется выпуклым многогранником (convex polytope).*

Напомним, что множество ограничено, если оно содержится внутри некоторого шара фиксированного радиуса.

Так, выпуклый многогранник P на Рис. 5 представляет собой множество решений системы

$$\begin{cases} x \geq 0, \\ y \geq 0, \\ x \leq 2, \\ y \leq 2, \\ x + y \leq 3. \end{cases}$$

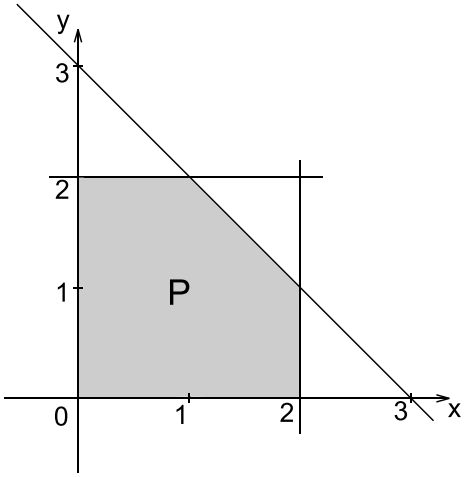

Рис. 5. Пример выпуклого многогранника.

Определение. *Гранью выпуклого многогранника*

$$P = \{x: Ax \leq b\}$$

называется выпуклый многогранник, полученный заменой части неравенств системы $Ax \le b$ на равенства

$$F = \{x\colon\ Ix \le c, Qx = d\}, \qquad \text{где } A = \begin{pmatrix} I \\ Q \end{pmatrix}, b = \begin{pmatrix} c \\ d \end{pmatrix}.$$

Теорема. *Размерность грани F равна*

$$\dim P - \operatorname{rg} Q.$$

Определение. *Грань размерности 0 называется вершиной многогранника, грань размерности 1 – ребро многогранника, грань размерности k – это k-грань многогранника, грань размерности $\dim P - 1$ – фасета или гиперграни многогранника.*

Определение. *Линейная комбинация точек $x_1, x_2, \dots, x_k \in \mathbb{R}^d$*

$$\lambda_1 x_1 + \lambda_2 x_2 + \cdots + \lambda_k x_k$$

называется выпуклой, если

$$\forall i\colon\ \lambda_i \ge 0, \qquad \text{и} \ \sum_{i=1}^{k} \lambda_i = 1.$$

Определение. *Выпуклой оболочкой* $\operatorname{conv}(S)$ *(convex hull) множества S называется пересечение всех выпуклых множеств, содержащих S (наименьшее выпуклое множество, содержащее S).*

Выпуклая оболочка конечного множества точек совпадает с множеством всех возможных выпуклых комбинаций этих точек

$$\operatorname{conv}(\{x_1, \dots, x_k\}) = \left\{ y\colon\ y = \lambda_1 x_1 + \cdots + \lambda_k x_k,\ \ \lambda_i \ge 0, \sum_{i=1}^{k} \lambda_i = 1 \right\}.$$

Основной теоремой в теории выпуклых многогранников является

Теорема Вейля – Минковского. *Множество P является ограниченным пересечением конечного числа замкнутых полупространств*

$$P = \{x\colon Ax \le b\} \text{ и } P - \text{ограничено},$$

тогда и только тогда, когда P – выпуклая оболочка конечного множества точек (своих вершин)

$$P = \operatorname{conv}\{x_1, x_2, \dots, x_k\}.$$

Теорема Вейля-Минковского представляет два независимых описания для выпуклых многогранников:

1) внешнее описание (H-polytope) – многогранник как ограниченное множество решений системы линейных неравенств;
2) внутреннее описание (V-polytope) – многогранник как выпуклая оболочка своих вершин.

Так, многогранник P на Рис. 5 может быть описан как

$$P = \text{conv}\{(0,0), (2,0), (2,1), (1,2), (0,2)\}.$$

Теорема названа в честь сразу двух математиков, так как представляет собой объединение двух различных теорем: теоремы Минковского (H-polytope – это V-polytope) и теоремы Вейля (V-polytope – это H-polytope).

Переход между двумя независимыми описаниями представляет собой две задачи вычислительной геометрии:

1) задача перечисления вершин выпуклого многогранника (vertex enumeration problem);
2) задача построения выпуклой оболочки конечного множества точек (convex hull problem).

Классический метод решения задачи перечисления вершин – алгоритм Фурье-Моцкина (Fourier-Motzkin elimination), который является аналогом метода Гаусса для систем линейных уравнений и заключается в последовательном исключении неизвестных из системы неравенств [33].

Отметим, что выполнение исключения одной переменной в системе из m неравенств алгоритмом Фурье-Моцкина может привести к появлению $m^2/4$ новых неравенств. Таким образом, после d последовательных шагов алгоритма система может содержать до

$$4\left(\frac{m}{4}\right)^{2^d}$$

неравенств, что порождает сверхэкспоненциальную трудоемкость в худшем случае. Это происходит вследствие порождения значительного числа избыточных линейно зависимых неравенств. Однако и число линейно независимых неравенств будет расти с экспоненциальной скоростью [26]. Для алгоритма Фурье-Моцкина построено значительное число различных модификаций, самой известной из которых является метод двойного описания

(double description method), которые, однако, также имеют экспоненциальную в худшем случае трудоёмкость [3].

Для обратной задачи построения выпуклой оболочки известно значительное число эффективных алгоритмов, решающих задачу на плоскости, где она фактически сводится к задаче сортировки. Так алгоритм Грэхема (Graham scan) решает поставленную задачу за время $O(n \log n)$. Однако в пространствах произвольной размерности сложность построения выпуклой оболочки многократно возрастает. Дело в том, что для построения выпуклой оболочки множества точек необходимо найти все линейные неравенства, описывающие фасеты соответствующего выпуклого многогранника, число которых может быть очень велико.

Определение. *Выпуклый многогранник называется k-смежностным, если любые k его вершин образуют грань многогранника.*

Наиболее известным смежностным многогранником является циклический многогранник, впервые описанный Константином Каратеодори в 1907 [4] году и независимо переоткрытый Дэвидом Гейлом в 1956 году [16].

Определение. *Кривая $x(t) = (t, t^2, t^3, \dots, t^d)$ в $\mathbb{R}^d$ называется моментной кривой (moment curve).*

Определение. *Циклическим многогранником $C_d(n)$ называется выпуклая оболочка n различных точек на моментной кривой в $\mathbb{R}^d$*

$$C_d(t_1, t_2, \dots, t_n) = conv\{x(t_1), x(t_2), \dots, x(t_n)\}.$$

Теорема (Каратеодори, Гейл). *Циклический многогранник $C_d(n)$ является $\lfloor d/2 \rfloor$-смежностным многогранником.*

Определение. *Многогранник называется смежностным, если он $\lfloor d/2 \rfloor$-смежностной.*

Теорема МакМаллена о верхней границе. *Среди всех d-мерных многогранников на n-вершинах смежностные многогранники (в частности циклический многогранник) имеют наибольшее количество k-мерных граней для любого $k = 1,2,3, \dots, d-1$.*

Теорема. *Число фасет циклического многогранника $C_d(n)$ равно*

$$\sum_{i=0}^{d/2} 2 \binom{n-d-1+i}{i}$$

Таким образом, если размерность d фиксирована, то число фасет циклического многогранника растет как полином степени $\lfloor d/2 \rfloor$ от числа n вершин многогранника. Соответственно, ни один алгоритм построения выпуклой оболочки множества из n точек в $\mathbb{R}^d$ не может иметь трудоемкость менее $\Omega(n^{\lfloor d/2 \rfloor})$ в худшем случае. В частности, такую трудоемкость имеют алгоритмы gift wrapping (алгоритм «заворачивания подарков» или алгоритм Джарвиса) и QuickHull (алгоритм быстрой оболочки).

Напомним, что через Ω обозначают асимптотическую оценку снизу на значение функции с точностью до множителя

$$f(n) = \Omega\big(g(n)\big) \Leftrightarrow \exists (C > 0), n_0 \colon \forall (n > n_0)\, f(n) \geq C \cdot g(n).$$

Упражнения. 1) Доказать, что пересечение произвольного числа выпуклых множеств есть выпуклое множество.

2) Доказать, что алгоритм построения выпуклой оболочки множества точек на плоскости не может быть выполнен быстрее, чем за $\Omega(n \log n)$ шагов, т.е. имеет ту же трудоемкость, что и задача сортировки чисел.

5. Многогранники задач

Пусть $X = X_n$ некоторая массовая задача комбинаторной оптимизации с фиксированным параметром n. Например, задача о максимальном разрезе в графе на n вершинах. Множество возможных решений задачи кодируется характеристическими векторами и представляет собой множество точек, которое также обозначается X_n, в евклидовом пространстве $\mathbb{R}^d$, где d – некоторая функция от n. Так для задачи о разрезе d равно числу ребер в полном графе на n вершинах: $n(n-1)/2$. Пусть целевая функция задачи линейна, и задача в геометрической интерпретации принимает вид

$$(c, x) \to \min \text{ или } \max, \quad x \in X_n \subset \mathbb{R}^d.$$

Обозначим через $P(X) = \operatorname{conv} X_n$. По теореме Вейля-Минковского выпуклая оболочка конечного множества точек – это выпуклый многогранник, который назовем многогранником задачи X. Множество вершин многогранника $\operatorname{ext} P(X)$ будет подмножеством множества точек X_n. Без ограничения общности, если X_n не содержит заведомо невыполнимых решений задачи, можно положить

$$\operatorname{ext} P(X) = X_n.$$

Приведем классический вариант построения многогранника задачи о разрезе [11]. Каждому подмножеству $S \subseteq \mathbb{N}_n = \{1,2,3,\dots,n\}$ (каждому разрезу), сопоставляется характеристический $0/1$-вектор, который также называется *разрезным вектором*, по следующему правилу

$$x(S) \in \{0,1\}^d, \qquad \text{где } d = \frac{n(n-1)}{2},$$

$$x(S)_{i,j} = \begin{cases} 1, \text{если } |S \cap \{i,j\} = 1|, \\ 0, \text{в противном случае;} \end{cases}$$

$$1 \le i < j \le n.$$

Тогда разрезной многогранник $CUT(n)$ представляет собой выпуклую оболочку всех возможных разрезных векторов

$$CUT(n) = conv\,\{x(S) : S \subseteq \mathbb{N}_n\} \subset \mathbb{R}^d.$$

Отметим, что под разрезным многогранником (и в общем случае под многогранником задачи) понимается не один уникальный многогранник, а последовательность многогранников массовых задач о разрезе с

произвольным числом n вершин графа, уникальные представители которых получаются при фиксировании параметра n. Это делается по аналогии с определениями гиперкуба или симплекса, которые включают целые семейства, соответственно, d-мерных гиперкубов и симплексов.

Введение новой конструкции многогранника задачи позволяет задействовать всю мощь теории линейного программирования, заложенной Леонидом Канторовичем в 1939 году [39], и удостоенной нобелевской премии по экономике в 1975 году.

Основная теорема линейного программирование. *Целевая функция задачи линейного программирования на выпуклом многограннике P*

$$(c, x) \to \min(\max),$$

$$x \in P = \{x \in \mathbb{R}^d : Ax \leq b\},$$

$$P - \text{ограниченое множество},$$

достигает своего экстремума (максимума и минимума) в некоторой вершине многогранника P.

Как правило, основная теорема линейного программирования применяется для сведения задачи оптимизации линейной целевой функции на всем многограннике (континуальном множестве) к оптимизации на множестве его вершин (дискретном конечном множестве) с последующим применением симплекс-метода Данцига для грамотного перебора вершин. Однако в нашем случае имеет смысл выполнить обратный переход от оптимизации на дискретном множестве решений X_n (множестве вершин многогранника) к оптимизации на всем многограннике, что позволит свести задачу комбинаторной оптимизации в геометрической интерпретации к задаче линейного программирования

$$(c, x) \to \min(\max),$$

$$x \in P(X) = \{x \in \mathbb{R}^d : Ax \leq b\} = \operatorname{conv} X_n.$$

Идея задействовать алгоритмы линейного программирования для решения сложных оптимизационных задач появилась еще на заре развития данной теории. Так Джордж Данциг, Рэй Фалкерсон и Селмер Джонсон опубликовали в 1954 году свою знаменитую работу [9] по геометрическому решению ***задачи коммивояжера***:

Заданы: неориентированный граф $G = (V, E)$ *и некоторая функция* $C: E \to \mathbb{R}_{\geq 0}$*, каждому ребру* $e \in E$ *сопоставляющая неотрицательное действительное число* $C(e)$*, называемое весом ребра.*

Требуется найти гамильтонов цикл (путь, содержащий каждую вершину графа ровно один раз и возвращающийся в исходную вершину) с минимальной суммой весов входящих в него ребер.

В задаче коммивояжера вершины графа обычно называются городами, а сама задача заключается в поиске кратчайшего маршрута, по которому коммивояжер смог бы посетить все города. Общее число возможных маршрутов (гамильтоновых циклов в неориентированном графе) составляет $\frac{(n-1)!}{2}$.

Семью годами ранее упомянутой выше статьи Данциг разработал симплекс-метод [10], который быстро стал основным аппаратом решения задач линейного программирования и считается одним из 10 основных алгоритмов 20-го века [28]. На волне воодушевления от бесконечных приложений симплекс-метода Данциг, Фалкерсон и Джонсон преобразовали задачу коммивояжера к задаче целочисленного линейного программирования и, применив симплекс-метод, решили задачу для карты автомобильных дорог и 49 городов США (по одному городу на каждый из 48 материковых штатов за исключением Гавайев и Аляски, и города Вашингтон, который территориально принадлежит федеральному округу Колумбия), что на тот момент было феноменальным результатом (Рис. 6). Общее число возможных маршрутов для 49 городов составляет $\frac{48!}{2} \approx 6.2 \cdot 10^{60}$, что находится за гранью возможностей перебора современных суперкомпьютеров.

Получаем унифицированный аппарат сведения практически любой задачи комбинаторной оптимизации, таких как задача о разрезе или задача коммивояжера, к задаче линейного программирования с последующим ее решением, например, симплекс-методом.

Следует отметить, что, строго говоря, симплекс-метод не может считаться эффективным алгоритмом, так как в 1972 году Виктор Кли и Джордж Минти показали, что на незначительно скошенном единичном кубе (куб Кли-Минти) симплекс-метод работает по наихудшему сценарию и последовательно перебирает все вершины многогранника, что требует экспоненциального числа шагов [25].

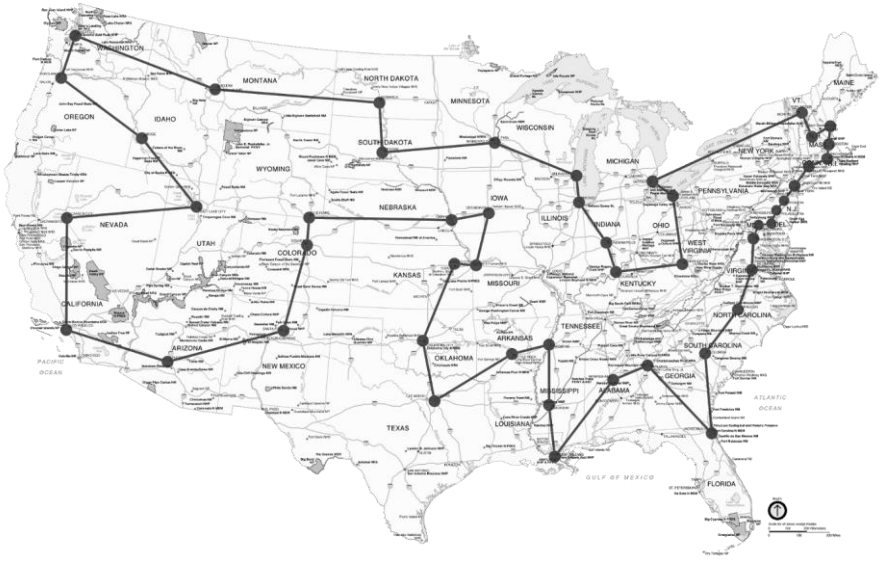

Рис. 6. Решение задачи коммивояжера для 49 городов США

Тем не менее, уже в 1978 году Леонид Хачиян модифицировал метод эллипсоидов Шора для решения задачи линейного программирования [41], доказав, что время вычисления будет гарантированно меньше $O(n^6L)$, где n – число переменных, а L – длина входа в битах, то есть полиномиальной функции от входных данных. Однако степень полинома здесь слишком высока для применения алгоритма Хачияна при решении задач. В действительности, на практических задачах «неэффективный» симплекс-метод обгоняет полиномиальный алгоритм эллипсоидов в 99.9% случае, кроме специально сконструированных примеров (куб Кли-Минти). И все же результат Хачияна имел фундаментальный характер, так как доказал принципиальную возможность построения полиномиальных алгоритмов для решения задачи линейного программирования и, соответственно, принадлежности задачи классу P полиномиально разрешимых задач. А еще шестью годами позже в 1984 году Нарендра Кармаркар [23] предложил первый по настоящему эффективный полиномиальный алгоритм для задачи линейного программирования с трудоемкостью всего $O(n^{3.5}L)$. Современные алгоритмы из класса методов внутренней точки, которые в отличие от симплекс-метода

достигают оптимального решения путем обхода внутренней допустимой области, не уступают симплекс-методу не только при худшем сценарии, но и на произвольной задаче.

Таким образом, практически любую задачу комбинаторной оптимизации, в том числе задачу о разрезе или задачу коммивояжера, можно представить в форме задачи оптимизации линейной целевой функции на множестве точек в евклидовом пространстве, что в свою очередь по основной теореме линейного программирования сводится к задаче оптимизации линейной функции на выпуклом многограннике. Задача линейного программирования полиномиально разрешима и мы получаем сценарий $P = NP$. Однако не все так просто.

Проблема № 1: выпуклая оболочка. Алгоритмы линейного программирования предполагают на входе описание выпуклого многогранника как области допустимых решений системы линейных неравенств (внешнее описание, H-polytope). Задачи комбинаторной оптимизации сводятся к оптимизации линейной функции на множестве вершин многогранника (внутреннее описание, V-polytope). По теореме Вейля-Минковского теоретически эти два описания эквивалентны. Однако они отнюдь не эквивалентны с вычислительной точки зрения. Переход от внутреннего описания к внешнему предполагает решение задачи построения выпуклой оболочки (convex hull problem), трудоемкость которой не может быть меньше числа фасет (линейных неравенств во внешнем описании) результирующего многогранника. При худшем сценарии число фасет оценивается снизу как $\Omega(m^{\lfloor d/2 \rfloor})$, где m-число точек по которым строится выпуклая оболочка, а d-размерность пространства. Однако в нашем случае для геометрической интерпретации задачи m и d являются функциями от фиксированного параметра массовой задачи n. Так, для задачи о разрезе число возможных решений $m = 2^{n-1}$, а размерность пространства $d = n(n-1)/2$, что дает

$$\Omega\left(2^{\left\lfloor \frac{n(n-1)^2}{2} \right\rfloor}\right),$$

сверхэкспоненциальную трудоемкость для задачи построения внешнего описания разрезного многогранника $CUT(n)$.

Тем не менее, задача построения выпуклой оболочки сама по себе пока не ставит крест на всей конструкции. Во-первых, трудоемкость задачи была оценена в худшем случае для циклических многогранников. В некоторых случаях, например для n-мерного гиперкуба, число граней которого равно $2d$, где d-размерность пространства, построение выпуклой оболочки не представляет никакой сложности. Во-вторых, следует отметить, что мы рассматриваем геометрическую интерпретацию не индивидуальной, а массовой задачи комбинаторной оптимизации. Это означает, что выпуклый многогранник нужно построить всего один раз для каждого фиксированного параметра n массовой задачи (для каждого уникального числа вершин графа в задаче о разрезе), что можно выполнить, например, на суперкомпьютере. Далее же для решения индивидуальной задачи будет достаточно менять лишь вектор целевой функции. Фактически, имеет место подход программиста к задаче: решить один раз более сложную задачу вместо многократного решения более простых задач.

Проблема № 2: число фасет. К сожалению, интересующие нас многогранники: разрезной многогранник $CUT(n)$ и многогранник коммивояжера $TSP(n)$ (как в прочем и подавляющее большинство многогранников задач комбинаторной оптимизации) имеют по настоящему много фасет (Табл. 2 и Табл. 3) [31].

Многогранник (число вершин графа)	Число фасет (линейных неравенств во внешнем описании)
$TSP(6)$	100 фасет
$TSP(7)$	3 437 фасет
$TSP(8)$	194 187 фасет
$TSP(9)$	42 104 442 фасет
$TSP(10)$	51 043 900 866 фасет
$TSP(11)$	полное описание многогранника не найдено

Таблица 2. Число известных фасет для многогранника коммивояжера

Многогранник (число вершин графа)	Число фасет (линейных неравенств во внешнем описании)
$CUT(5)$	56 фасет
$CUT(6)$	368 фасет

$CUT(7)$	116 764 фасет
$CUT(8)$	217 093 472 фасет
$CUT(9)$	12 246 651 158 320 фасет
$CUT(10)$	полное описание многогранника не найдено

Таблица 3. Число известных фасет для разрезного многогранника

Число фасет и, соответственно, линейных ограничений во внешнем описании многогранников задач о разрезе и коммивояжера растет, судя по всему, со сверхэкспоненциальной скоростью. Во-первых, это непосредственно препятствует решению задачи построения выпуклой оболочки и нахождению внешнего описания соответствующих многогранников. Так, уже для задачи коммивояжера в графе на 11 вершинах и для задачи о разрезе в графе на 10 вершинах полного описания всех фасет до сих пор не было найдено. А, во-вторых, даже если предположить, что внешнее описание соответствующего многогранника будет каким-либо образом построено, то наша задача (например задача о разрезе) сведется к задаче линейного программирования со сверхэкпоненциальным числом линейных неравенств во входных данных. Трудоемкость алгоритма Кармаркара для линейного программирования составляет $O(n^{3.5}L)$, где L длина входа, и потенциальная эффективность алгоритма будет полностью утеряна за счет чрезмерного объема входных данных.

Проблема № 3: размер коэффициентов. Гретчел, Ловас и Шрайвер [20], в своем исследовании метода эллипсоидов Хачияна и его фундаментального значения в комбинаторной оптимизации показали, что алгоритм не только полиномиален, но и, в силу своей специфики, крайне нечувствителен к числу ограничений во внешнем описании многогранника P: для решения задачи линейного программирования методом эллипсоидов не требуется напрямую задавать все фасеты P, достаточно описать процедуру для любого вектора $x \in \mathbb{R}^d$ проверяющую принадлежность $x \in P$ и, в случае $x \notin P$, строящую гиперплоскость, отделяющую x от P. Поиск такой гиперплоскости является другой задачей комбинаторной оптимизации, которая, однако, как правило намного проще исходной. По данной методике Гретчел, Ловас и Шрайвер смогли построить полиномиальные алгоритмы для таких комбинаторных задач как вершинная упаковка в совершенном графе, задача о пересечении матроидов и ряда других.

Таким образом, метод эллипсоидов может обмануть проблему со сверхэкспоненциальным числом фасет и, соответственно, линейных ограничений в формулировке задачи комбинаторной оптимизации как задачи линейного программирования. Тем не менее, алгоритму по-прежнему придется работать с отдельными фасетами многогранника задачи. На первый взгляд в этом нет ничего страшного, однако даже одна единственная фасета может стать причиной значительных неприятностей.

По построению разрезной многогранник (а также многогранник коммивояжера и другие многогранники задач выбора оптимального подграфа) принадлежит классу $0/1$-многогранников: его вершины – это подмножество вершин единичного $0/1$-гиперкуба. Ответ на вопрос: «насколько велики могут быть целые (рациональные) коэффициенты в линейных неравенствах, описывающих $0/1$-многогранник?» дает [33]

Теорема Алона-Ву. *Наибольший целый коэффициент* $coeff(d)$ *в описании фасет всех полноразмерных* $0/1$*-многогранников в* $\mathbb{R}^d$ *удовлетворяет неравенствам*

$$\frac{(d-1)^{\left\lfloor\frac{d-1}{2}\right\rfloor}}{2^{2d+o(d)}} \leq coeff(d) \leq \frac{d^{\left\lfloor\frac{d}{2}\right\rfloor}}{2^{d-1}}.$$

Например для задачи о разрезе в графе на 15 вершинах размерность разрезного многогранника

$$\dim\big(CUT(15)\big) = \frac{15(15-1)}{2} = 105.$$

Наибольший целый коэффициент в описании фасет полноразмерных 105-мерных 0/1-многогранников удовлетворяет нижней оценке

$$coeff\ (105) \geq \frac{104^{52}}{2^{209}} \approx 9.34 \times 10^{41}.$$

Учитывая, что фасеты разрезного многогранника $CUT(15)$ еще не описаны, то эта оценка вполне может достигаться и на нем.

Сверхэкспоненциальный размер коэффициентов в описании фасет многогранника не позволит эффективно решить задачу даже если удастся обмануть проблему №2 с числом фасет. Однако вторую и третью проблемы можно обойти, если несколько изменить точку зрения.

Упражнение. Построить многогранник задачи сортировки n чисел. Найти для него количество фасет и величину наибольшего коэффициента во внешнем описании.

6. Расширенная формулировка

В некоторых случаях может показаться, что некоторая задача (или в нашем случае многогранник задачи) чрезмерно сложна. Однако возможно речь идет лишь о неудачной точки зрения.

Определение. *Пусть P и Q выпуклые многогранники, тогда Q называется расширенной формулировкой P (extended formulation):* $Q = EF(P)$*, если найдется такое линейное отображение проектирования* $proj_x$*, что*

$$P = \text{proj}_x Q = \left\{ x \in \mathbb{R}^d : \exists z \in \mathbb{R}^k \ \begin{pmatrix} x \\ z \end{pmatrix} \in \mathbb{R}^{d+k} \right\},$$

$$P \subset \mathbb{R}^d, \qquad Q \subset \mathbb{R}^{d+k}.$$

На Рис. 7 приведен пример $Q = EF(P)$, относительно линейного отображения ортогонального проектирования $\mathbb{R}^3$ на плоскость Oxy.

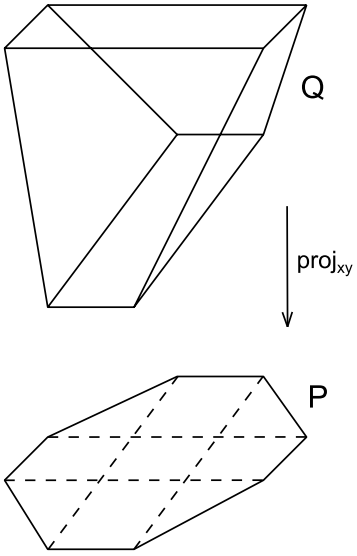

Рис. 7. P – проекция многогранника Q на плоскость Oxy

Задачу линейного программирования на многограннике P нетрудно преобразовать к задаче линейного программирования на многограннике Q – расширенной формулировке P:

$$\max\{(c,x): x \in P\} = max\left\{(c,x) + (0,z): \binom{x}{z} \in P\right\}.$$

При этом, многогранник Q расширенной формулировки задачи может иметь принципиально более простое внешнее описание, чем исходный многогранник задачи P. Так в примере на Рис. 7 многогранник P на плоскости имеет 8 фасет и задается системой из 8 линейных неравенств с двумя неизвестными x и y, в то время как его расширенная формулировка Q в трехмерном пространстве имеет всего 6 фасет и, соответственно, определяется системой из 6 неравенств с тремя неизвестными x, y, z. Добавлением новой координаты удалось уменьшить число фасет многогранника и упростить его внешнее описание.

Классическим примером успешного применения расширенной формулировки является задача ***нахождения минимального остовного дерева*** (minimum spanning tree) в связанном взвешенном графе:

Заданы: неориентированный связанный граф $G = (V, E)$ и некоторая функция $C: E \to \mathbb{R}_{\geq 0}$, каждому ребру $e \in E$ сопоставляющая неотрицательное действительное число $C(e)$, называемое весом ребра.

Требуется найти остовное дерево (ациклический связный подграф, в который входят все вершины графа G) с минимальной суммой весов входящих в него ребер.

Задача о минимальном остовном дереве напрямую применяется при проектировании сетей, включая компьютерные сети, телекоммуникационные сети, транспортные сети, сети электро- и водоснабжения.

Геометрическая интерпретация задачи поиска минимального остовного дерева вводится классическим образом, как и для других задач выбора оптимального подграфа: достаточно занумеровать все ребра в графе и закодировать возможные решения (остовные деревья) в виде характеристических $0/1$-векторов по включению соответствующих ребер в остовное дерево. Общее число остовных деревьев (и потенциальных решений задачи) в полном графе на n вершинах равно n^{n-2}. Выпуклая оболочка характеристических векторов всех n^{n-2} возможных остовных деревьев называется многогранником задачи об остовном дереве или spanning tree polytope, $SPT(n)$, где n – число вершин графа задачи.

Хорошей новостью является тот факт, что, в отличии от многогранников задач о разрезе и коммивояжера, для многогранника задачи об остовном дереве построено полное внешнее описание (полный список фасет) для любой размерности задачи и любого числа вершин n исходного графа G.

$$SPT(n) = \left\{ \begin{array}{c} x \in \mathbb{R}^{\frac{n(n-1)}{2}}: \\ \sum_{e \in E} x_e = n-1, \\ \sum_{e \in E(S)} x_e \leq |s| - 1, \forall S \subseteq V, \\ x_e \geq 0, \forall e \in E \end{array} \right\}.$$

Природу данных ограничений нетрудно понять. Размерность пространства $\mathbb{R}^d$ равна числу ребер в полном графе $n(n-1)/2$. За x_e обозначается координата характеристического вектора x, соответствующая ребру $e \in E$. Первое ограничение $\sum_{e \in E} x_e = n - 1$ означает, что каждое остовное дерево содержит ровно $n-1$ ребро, а второе ограничение $\sum_{e \in E(S)} x_e \leq |s| - 1, \forall S \subseteq V$ гарантирует, что никакой подграф остовного дерева не содержит циклов.

Отметим, что проблема № 1 из предыдущего параграфа о сложности построения выпуклой оболочки здесь полностью решена теоретическими методами. Проблема № 3 о сверхэкспоненциальных коэффициентах в описании фасет многогранника задачи здесь также не существенна: наибольший коэффициент присутствует в правой части первого уравнения и равен $n-1$. Однако остается проблема № 2 о числе фасет многогранника. Ограничение $\sum_{e \in E(S)} x_e \leq |s| - 1, \forall S \subseteq V$ предполагает рассмотрение всех возможных подмножеств S множества вершин графа V, число которых, исключая пустое подмножество, составляет $2^n - 1$. Многогранник $SPT(n)$ имеет экспоненциально много фасет, что препятствует решению задачи о минимальном остовном дереве напрямую методами линейного программирования (однако следует отметить, что число фасет все же принципиально меньше числа n^{n-2} характеристических векторов в геометрической интерпретации задачи до построения выпуклой оболочки).

Введем новые переменные и увеличим размерность пространства. Пусть $\lambda_{k,i,j} = 1$, если j – родитель (находится на один шаг ближе к корню) узла i в дереве с корнем k, в противном случае $\lambda_{i,j,k} = 0$, и рассмотрим следующий многогранник:

$$EF\big(SPT(n)\big) = \left\{ \begin{array}{c} \binom{x}{\lambda} \in \mathbb{R}^{\frac{n(n-1)}{2}+n^3}, x \in \mathbb{R}^{\frac{n(n-1)}{2}}, \lambda \in \mathbb{R}^{n^3}: \\ \sum\limits_{i,j \in V} x_{i,j} = n-1, \\ x_{i,j} \geq 0, \qquad i,j \in V, \\ x_{i,j} = \lambda_{k,i,j} + \lambda_{k,j,i}, 1 \leq i,j,k \leq n, \\ \sum\limits_{j \in V} \lambda_{k,i,j} = 1,\ 1 \leq i,k \leq n, i \neq k, \\ \lambda_{k,i,j} \geq 0, \lambda_{k,k,j} = 0, \lambda_{k,i,i} = 0, 1 \leq i,j,k \leq n \end{array} \right\}.$$

Многогранник $EF\big(SPT(n)\big)$ является расширенной формулировкой многогранника $SPT(n)$, который получается ортогональным проектирование на $\mathbb{R}^{\frac{n(n-1)}{2}}$ (приравниванием к 0 всех «новых» координат $\lambda_{k,i,j}$).

Нетрудно заметить, что число линейных ограничений в описании многогранника $EF\big(SPT(n)\big)$ составляет $O(n^3)$. Таким образом, введением новых переменных $\lambda_{k,i,j}$ и полиномиальным увеличением размерности пространства на n^3 число фасет многогранника задачи об остовном дереве удалось сократить с экспоненциального $O(2^n)$ до полиномиального $O(n^3)$, после чего задача эффективно решается алгоритмами линейного программирования.

Безусловно, задача о минимальном остовном дереве может быть решена напрямую, без геометрической интерпретации, жадными алгоритмами Прима и Краскала с трудоемкостью $O(n^2 \log n)$, которые превосходят по эффективности алгоритмы линейного программирования. Однако нам важна принципиальная возможность подобного эффективного геометрического решения задачи, которую хотелось бы перенести на задачу о максимальном разрезе или задачу коммивояжера, для которых не известны эффективные негеометрические алгоритмы.

К сожалению, последние исследования в данной области [14] показывают, что это невозможно.

Определение. *Сложность расширенной формулировки задачи – минимальное число линейных ограничений во внешнем описании всех возможных расширенных формулировок многогранника задачи.*

Теорема (Фиорини и др.) *Сложность расширенной формулировки многогранника задачи о разрезе* $CUT(n)$ *равна* $2^{\Omega(n)}$.

Теорема (Фиорини и др.) *Сложность расширенной формулировки многогранника задачи коммивояжера* $TSP(n)$ *равна* $2^{\Omega(\sqrt{n})}$.

Таким образом, задачи о максимальном разрезе и коммивояжер при любой записи их в виде задач линейного программирования будут содержать экспоненциально большое число линейных ограничений в системе.

Упражнение. Найти точное значение числа фасет многогранника расширенной формулировки задачи об остовном дереве $EF(SPT(n))$ и проверить, что оно полиномиально по числу n вершин исходного графа задачи.

7. Конусные разбиения

Рассмотрим еще одну интерпретацию задачи комбинаторной оптимизации в форме выбора оптимального выпуклого конуса решения [35].

Определение. *Конусом (cone) называется такое подмножество* $K \subseteq \mathbb{R}^d$*, что*

$$\forall x \in K, \forall \lambda \in \mathbb{R}\ (\lambda \geq 0): \lambda x \in K.$$

Таким образом, конус K вместе с любой своей точкой $x \in K$ содержит луч, выходящий из нуля и проходящей через x.

Определение. *Выпуклый конус (convex cone) – это конус, который является выпуклым множеством.*

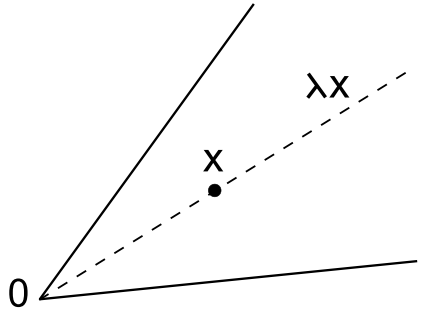

Рис. 8. Пример выпуклого конуса

Определение. *Линейная комбинация точек* $x_1, x_2, \dots, x_k \in \mathbb{R}^d$

$$\lambda_1 x_1 + \lambda_2 x_2 + \cdots + \lambda_k x_k$$

называется конической, если $\forall i:\ \lambda_i \geq 0$.

Определение. *Конической оболочкой* cone (S) *(conical hull) множества S называется пересечение всех выпуклых конусов, содержащих S (наименьший выпуклый конус, содержащий S).*

Коническая оболочка конечного множества точек совпадает с множеством всех возможных конических комбинаций этих точек

$$\text{cone}(\{x_1, \dots, x_k\}) = \{y:\ y = \lambda_1 x_1 + \cdots + \lambda_k x_k,\ \forall i: \lambda_i \geq 0\}.$$

Для выпуклых конусов по аналогии с выпуклыми многогранниками справедлива теорема Вейля-Минковского [34].

Теорема Вейля – Минковского для выпуклых конусов. *Конус K – это коническая оболочка конечного множества точек (лучей)*

$$K = \text{cone}\{x_1, x_2, \dots, x_k\}.$$

тогда и только тогда, когда K является множеством решений системы однородных линейных неравенств

$$K = \{x: Ax \leq 0\}.$$

Пусть $X = X_n$ некоторая массовая задача комбинаторной оптимизации с фиксированным параметром n.

Как и ранее через $P(X) = \text{conv}\, X$ определим многогранник задачи X, и пусть $\text{ext}\, P(X) = X$.

Определение. *Для произвольного решения $x \in X$ обозначим через*

$$K(x) = \{c \in \mathbb{R}^d: (c, x) \geq (c, y), \forall y \in X\}$$

множество всех целевых векторов $c \in \mathbb{R}^d$, для которых линейная целевая функция (c, x) достигает своего максимума в точности в точке $x \in X$.

Для задачи на минимум достаточно поменять в определении знак неравенства.

$K(x)$ – это множество решений конечной системы однородных линейных неравенств, т.е. выпуклый конус, который назовем конусом решения x.

Определение. *Учитывая, что*

$$\bigcup_{x \in X} K(x) = \mathbb{R}^d,$$

совокупность всех конусов $K(x), x \in X$ называется конусным разбиением пространства $\mathbb{R}^d$ по множеству X.

Конусное разбиение является аналогом диаграммы Вороного, в точности совпадая с ней, если евклидовы нормы всех точек множества X равны между собой.

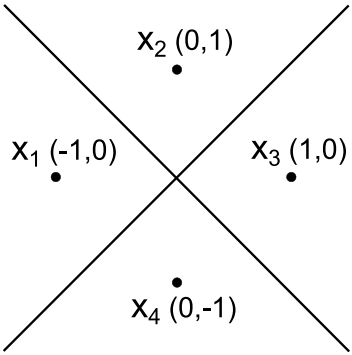

Рис. 9. Пример конусного разбиения пространства

Индивидуальная задача комбинаторной оптимизации в геометрической интерпретации

$$(c, x) \to max, \;\; x \in X,$$

сводится к поиску такого $x_0 \in X$, что $c \in K(x_0)$, т.е. к определению того конуса в конусном разбиении пространства $\mathbb{R}^d$ по множеству X, которому принадлежит целевой вектор c.

Так на Рис. 9 и 10 приведен пример конусного разбиения плоскости по множеству точек $X = \{x_1(-1,0), x_2(0,1), x_3(1,0), x_4(0,-1)\}$. На этом множестве целевая функция (c, x) индивидуальной задачи с целевым вектором $c(2,-0.5)$ достигает максимума в точке $x_3(1,0)$, поэтому вектор c принадлежит конусу $K(x_3)$.

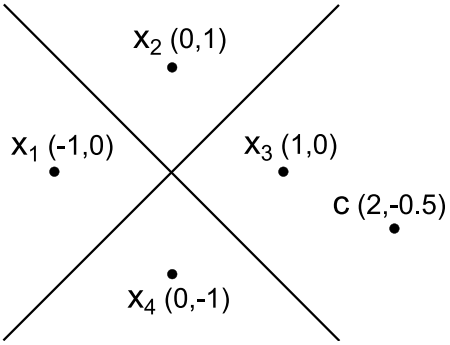

Рис. 10. Пример конусного разбиения с целевым вектором $c(2,-0.5)$

Связь между структурой конусного разбиения пространства по множеству X и строением граничного комплекса многогранника $P(X)$ устанавливает следующая теорема [35]

Теорема (Бондаренко). *Пусть* $x_1, x_2 \in X \subset \mathbb{R}^d$ *и* r *– наименьшая размерность граней многогранника* $P(X)$*, содержащих одновременно* x_1 *и* x_2*, тогда*

$$\dim\big(K(x_1) \cap K(x_2)\big) = d - r.$$

Следствие. *Вершины* x_1 *и* x_2 *многогранника* $P(X)$ *смежны в том и только том случае, когда конусы* $K(x_1)$ *и* $K(x_2)$ *имеют общую гиперграни*

$$\dim\big(K(x_1) \cap K(x_2)\big) = d - 1.$$

Упражнения. 1) Показать, что любой полиэдр можно представить в виде суммы выпуклого многогранника и конуса.

2) Доказать, что конусное разбиение совпадает с диаграммой Вороного, если евклидовы нормы всех точек множества X равны (точки лежат на сфере с центром в начале координат).

8. Линейные разделяющие деревья

Одной из классических моделей представления алгоритмов в теории исследования операций является модель деревьев принятия решений (decision tree). Интерпретация алгоритмов комбинаторной оптимизации в виде деревьев принятия решений получила широкое распространение в силу своей простоты и наглядности. Так на Рис. 11 приведен пример дерева для задачи сортировки трех чисел a, b, c.

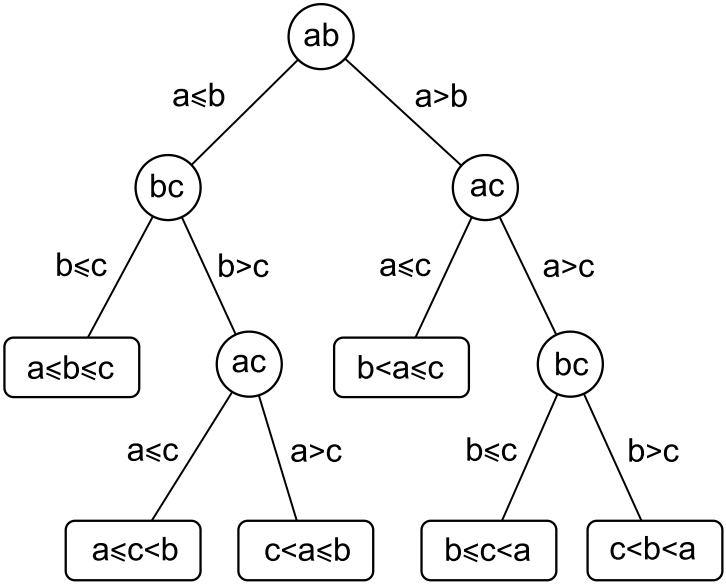

Рис. 11. Пример дерева сортировки трех чисел

Подобные деревья, где каждый узел имеет не более двух потомков называются бинарными или двоичными.

Определение. *Высота (глубина) дерева* $h(T)$ *– длина максимального пути от корня до листа.*

Количество всех возможных перестановок n чисел в задаче сортировки равно $n!$. Общее число листьев полного бинарного дерева глубины h составляет 2^h. Таким образом, получаем известную нижнюю оценку на

глубину бинарного дерева задачи сортировки и, соответственно, на все сортировочные алгоритмы

$$h(T) \geq \log_2(n!) = \Omega(n \log_2 n).$$

Рассмотрим важный класс деревьев принятия решений – линейные разделяющие деревья [35].

Пусть $X = X_n$ некоторая массовая задача комбинаторной оптимизации с фиксированным параметром n.

Определение. *Линейным разделяющим деревом задачи* X ($X \subset \mathbb{R}^d$) *называется ориентированное дерево, обладающее следующими свойствами:*

а) *в каждый узел, за исключением одного, называемого корнем, входит одна дуга;*

б) *из каждого узла выходит две дуги (внутренний узел) или не выходит ничего (лист);*

в) *каждому внутреннему узлу соответствует линейный функционал (вектор)* $f \in \mathbb{R}^d$*;*

г) *каждому листу соответствует элемент множества* X*;*

д) *каждой дуге* d *соответствует число* $\operatorname{sgn} d$*, равное* 1 *или* -1*, дуги, выходящие из одного узла, имеют разный знак;*

е) *для каждой цепи* $W = f_1 d_1 f_2 d_2 \dots f_k d_k x$*, соединяющей корень и лист, для любого вектора* $c \in \mathbb{R}^d$ *из неравенств*

$$(f_i, c) \operatorname{sgn} d_i \geq 0, \qquad i = 1, \dots, k,$$

следует включение $c \in K(x)$.

Отметим, что линейное разделяющее дерево не является алгоритмом в классическом смысле из-за недостаточного описания. Оно лишь отражает макроструктуру вычислительного процесса, направленного на решение задачи X. Процесс решения индивидуальной задачи X с целевым вектором c представляет собой прохождение некоторой цепи W в линейном разделяющем дереве задачи, где на i-том шаге к вектору c применяется линейный функционал (линейный тест) f_i, выбор которого осуществляется исходя из знака линейной формы (f_{i-1}, c). Пример линейного разделяющего дерева для задачи сортировки трех чисел с целевым вектором (a, b, c) приведен на Рис. 12.

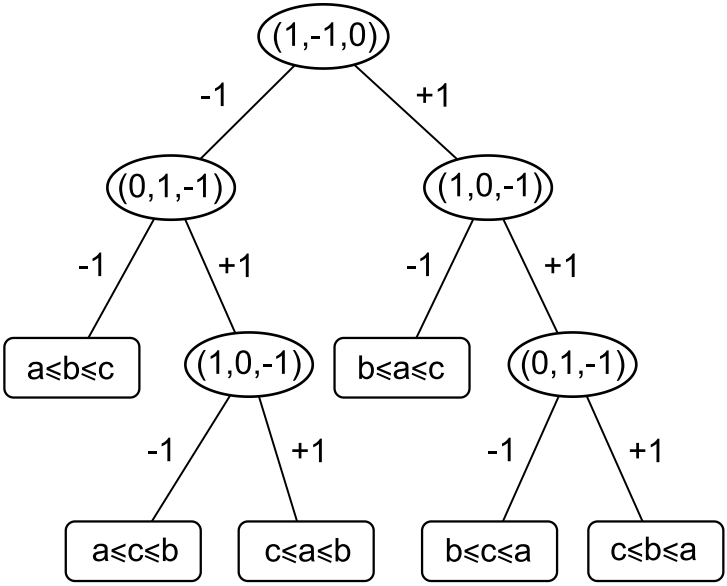

Рис. 12. Линейное разделяющее дерево для задачи сортировки

Линейные разделяющие деревья напрямую связаны с геометрической интерпретацией задач комбинаторной оптимизации и конусными разбиениями пространства.

Определение. *Разбиением D пространства $\mathbb{R}^d$ назовем конечную совокупность d-мерных выпуклых многогранных множеств (полиэдров) $Q_1, Q_2, \dots, Q_t$, которые попарно не имеют общих внутренних точек, и объединение которых совпадает с $\mathbb{R}^d$.*

В качестве примеров можно рассмотреть конусное разбиение пространства $\mathbb{R}^d$ по множеству X (которое обозначим через $D(X)$) или диаграмму Вороного.

Определение. *Разбиение D_b называется бинарным, если оно является результатом последовательных операций следующего вида: на первом шаге пространство $\mathbb{R}^d$ разрезается гиперплоскостью на 2 полупространства; на последующих шагах некоторые (возможно все) полученные на предыдущем шаге многогранные множества разрезаются на пару многогранных множеств размерности d.*

Определение. *Будем говорить, что разбиение D_2 вложено в разбиение D_1 (обозначение $D_2 \le D_1$), если каждое многогранное множество, входящее в D_2, содержится в некотором многогранном множестве из D_1.*

Теорема (Бондаренко). *Любое линейное разделяющее дерево решений задачи X ($X \subset \mathbb{R}^d$) определяет бинарное разбиение пространства $\mathbb{R}^d$, вложенное в конусное разбиение $D(X)$. При этом полиэдрами из разбиения D_b служат конусы*

$$K(W) = \{c \in \mathbb{R}^d : (f_i, c)\ \mathrm{sgn}\ d_i \ge 0, i = 1, \dots, k\},$$

соответствующие всевозможным цепям $W = f_1 d_1 f_2 d_2 \dots f_k d_k x$, ведущим из корня в лист x. Вложение $D_b \le D(x)$ обеспечивается равенствами

$$K(x) = \bigcup_{W:\, x \in W} K(W), \qquad x \in X.$$

Верно и обратное утверждение: каждое бинарное разбиение D_b, вложенное в конусное разбиение $D(X)$, определяет линейное разделяющее дерево задачи X.

Геометрическая интерпретация линейных разделяющих деревьев позволяет описать множество линейных функционалов, фигурирующих в определении дерева, которые обязательно будут реализованы алгоритмом, решающим задачу.

Пусть $T = T(X)$ – линейное разделяющее дерево задачи X ($X \subset \mathbb{R}^d$). Пусть f – линейный функционал ($f \in \mathbb{R}^d$). Будем писать $f \in T$, если f пропорционален некоторому внутреннему узлу дерева T.

Теорема о необходимых сравнениях. *Для любых смежных вершин x и y многогранника задачи $P(X)$ выполняется включение*

$$x - y \in T(X),$$

где $T(X)$ произвольное линейное разделяющее дерево задачи X.

Применительно к линейным разделяющим деревьям особо следует отметить результаты М.Ю. Мошкова [27] относительно оценок сложности рассматриваемых алгоритмов.

Обозначим через $S(X) = \min h(T)$, где минимум берется по всем линейным разделяющим деревьям T задачи X, тогда имеет место

Теорема Мошкова. *Для любого d и любой задачи $X \subset \mathbb{R}^d$: выполняется неравенство*

$$S(X) \leq 7d^3 \log_2 |X|.$$

Так для задачи о разрезе в графе на n вершинах число возможных решений $|X| = 2^{n-1}$, размерность пространства в геометрической интерпретации $d = n(n-1)/2$, что дает нам

$$S(X) \leq 7\left(\frac{n(n-1)}{2}\right)^3 \log_2 2^{n-1} = O(n^7),$$

т.е полиномиальную верхнюю оценку на высоту линейного разделяющего дерева задачи о разрезе.

На первый взгляд тот факт, что для любой задачи комбинаторной оптимизации существует линейное разделяющее дерево полиномиальной высоты означает выполнение равенства $P = NP$, однако, как уже отмечалось ранее, линейное разделяющее дерево – это еще не алгоритм. Рассмотрим два варианта реализации дерева в виде алгоритма.

Вариант № 1: хранение полной структуры дерева. Безусловно в том случае, если полная структура линейного разделяющего дерева задачи с полиномиальной высотой построена и находится в памяти компьютера, дальнейшее решение задачи не представляет собой алгоритмической сложности, и полученный алгоритм для индивидуальной задачи будет эффективным относительно временной трудоемкости. Однако нетрудно заметить, что это потребует хранения сверхэкспоненциального, в общем случае, объема данных для полной структуры дерева, что, в силу эффекта комбинаторного взрыва, но уже для памяти, представляется настолько же нереализуемым на практике, как и первоначальный алгоритм полного перебора вариантов. Трудоемкость задач по использованию памяти имеет свою классификацию, так мы попали в класс сложности $EXPSPACE \supset NP$.

Определение. *Класс сложности $EXPSPACE$ образуют задачи распознавания, разрешимые на детерминированной машине Тьюринга с $O(2^{p(n)})$ памятью, где $p(n)$ – полином по n.*

Вариант № 2: генерация узлов дерева по ходу алгоритма. В силу невозможности хранения полной структуры дерева остается вариант генерации линейных функционалов узлов на каждом шаге алгоритма, но не до

конца понятно по какому принципу это делать. Теорема о необходимых сравнениях подсказывает, что функционалы должны соответствовать парам смежных вершин многогранника задачи. Однако вопрос смежности вершин многогранников задач комбинаторной оптимизации сам зачастую является достаточно сложным, так имеет место [30]

Теорема (Пападимитриу). *Задача распознавания смежности двух вершин x и y многогранника задачи коммивояжера TSP является NP-полной.*

Таким образом, ответ на вопрос: «смежны ли два обхода графа в многограннике задачи коммивояжера» является не менее трудным чем сама задача коммивояжера.

Не для всех задач комбинаторной оптимизации вопрос со смежностью вершин является столь же сложным как для задачи коммивояжера, однако существуют и другие препятствия на пути построения линейных разделяющих деревьев с полиномиальной высотой, о чем более подробно будет рассказано в следующем разделе.

Упражнение. Построить линейное разделяющее дерево для задачи о максимальном разрезе в графе, приведенном на Рис. 3.

9. Алгоритмы прямого типа

Дальнейшие построения будут основаны на следующем геометрическом факте [35]:

Теорема. *Пусть* $S = \{Q_1, Q_2, \dots, Q_p\}$ *– семейство* d*-мерных полиэдров в* $\mathbb{R}^d$*, любые два из которых имеют общую гиперграни и не имеют общих внутренних точек. Тогда для любой гипеплоскости* H *в* $\mathbb{R}^d$ *хотя бы одно из открытых полупространств* H^+ *и* H^- *пересекается по меньшей мере с* $p-1$ *полиэдрами из* S.

Для линейных разделяющих деревьев и конусных разбиений это утверждение означает, что любое линейное сравнение на каждом шаге позволяет отбросить хотя бы при одном исходе из двух возможных не более одного конуса из попарно смежных.

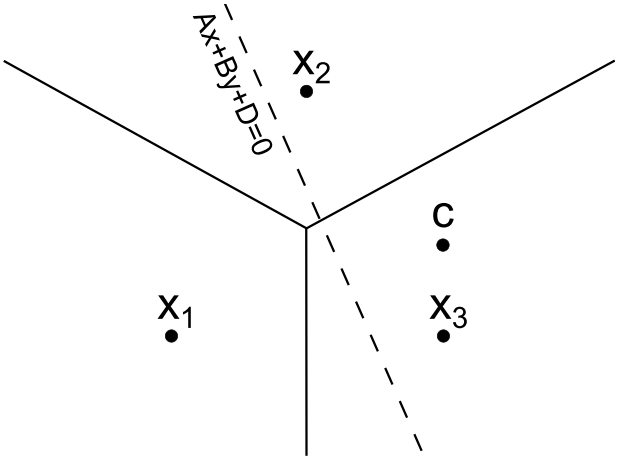

Рис. 13. Любая гиперплоскость отбрасывает не более одного конуса

Отметим, что это не исключает принципиальной возможности отбрасывания за несколько сравнений большего числа конусов, чем число выполненных сравнений, о чем свидетельствуют результаты Мошкова, однако это может привести к работе с системами с большим числом неравенств и соответствующим увеличением трудоемкости алгоритма.

Рассмотрим подмножество линейных разделяющих деревьев особого вида. Пусть T – линейное разделяющее дерево задачи X и f – его внутренний

узел. Обозначим через X_f – множество листьев дерева T, которым предшествует узел f, а через X_f^+ и X_f^- – подмножества X_f, соответствующие выходящим из f дугам.

Определение. *Линейное разделяющее дерево T задачи X является деревом прямого типа, если для любого внутреннего узла f и для любого подмножества Y попарно смежных точек из X выполняется неравенство*

$$\max\{|X_f^+ \cap Y|, |X_f^- \cap Y|\} \geq |X_f \cap Y| - 1.$$

Таким образом, для любого подмножества попарно смежных решений Y после выполнения линейного теста f хотя бы в одной из веток выходящей из f окажутся все попарно смежные решения Y кроме возможно одного.

Линейное разделяющее дерево прямого типа обладает упрощенной для реализации структурой. Соответствующий ему алгоритм прямого типа выполняет исключительно линейные сравнения: на каждом шаге он проводит гиперлоскость H и определяет какому из полупространств H^+ или H^- принадлежит целевой вектор c, отбрасывая все нереализуемые решения, конусы которых целиком лежат в полупространстве не содержащем вектора c. К классу алгоритмов прямого типа относятся все известные алгоритмы сортировки, использующие попарные сравнения чисел, жадный алгоритм, динамическое программирование, венгерский алгоритм для задачи о назначениях, метод ветвей и границ и многие другие классические алгоритмы [35].

Оценим трудоемкость алгоритмов прямого типа.

Определение. *Пусть $\omega(X)$ – кликовое число (число вершин в наибольшем полном подграфе) графа многогранника $P(X)$ задачи X.*

Теорема (Бондаренко). *Пусть T – дерево прямого типа задачи X, тогда*

$$h(T) \geq \omega(X) - 1,$$

где $h(T)$ – высота дерева T.

Действительно, так как T – дерево прямого типа, то на каждом шаге f нашего алгоритма хотя бы в одной ветке, выходящей из f, окажутся все попарно смежные решения из множества X_f, кроме возможно одного. Алгоритм прямого типа за один линейный тест не способен отбрасывать более одного попарно смежного решения, соответственно высота T не может быть меньше значения $\omega(X) - 1$.

Таким образом, кликовое число $\omega(X)$ графа многогранника $P(X)$ задачи X является нижней оценкой трудоемкости алгоритмов в худшем случае для достаточно широкого класса алгоритмов прямого типа, основанных на линейных тестах, в который попадают большинство известных классических алгоритмов.

Что это дает применительно к рассматриваемым задачам? Как уже было отмечено выше, задача распознавания смежности вершин многогранника задачи коммивояжера является NP-полной. Однако можно получить нижнюю оценку для кликового числа графа многогранника [35]

Теорема (Бондаренко). *Кликовое число $\omega(X)$ графа многогранника $TSP(n)$ задачи коммивояжера для n городов удовлетворяет неравенству*

$$\omega(X) \geq 2^{\left(\sqrt{\left[\frac{n}{2}\right]-9}\right)/2}.$$

Относительно задачи о разрезе, оценка кликового числа строится несколько проще [2]

Теорема (Барахона, Маджуб). *Граф многогранника $CUT(n)$ задачи о разрезе является полным.*

Все возможные разрезы графа являются попарно смежными и $\omega(X) = 2^{n-1}$.

Таким образом, ни один алгоритм прямого типа в принципе не сможет решить задачи о разрезе или коммивояжера за полиномиальное время, так как нижняя оценка на трудоемкость является сверхполиномиальной для коммивояжера и экспоненциальной для разреза. Интересно, что граф разрезного многогранника является полным, а значит любой алгоритм прямого типа должен в худшем случае перебрать все возможны решения задачи, т.е. свестись к полному перебору. Действительно, для задачи о разрезе с произвольными весами и для задачи о максимальном разрезе с неотрицательными весами, о чем будет более подробно рассказано в следующем разделе не известно ни одного алгоритма решения отличного от полного перебора.

Отметим, что данное утверждение не означает напрямую выполнения $P \neq NP$. Потенциально могут существовать эффективные алгоритмы, решающие такие NP-полные задачи как разрез и коммивояжер, использующие не только или не столько линейные сравнения. Так, для задачи линейного

программирования основанный на линейных тестах симплекс-метод не является эффективным в худшем случае, в то время как полиномиальные алгоритмы внутренней точки являются нелинейными. Тем не менее, используя идею деревьев прямого типа, можно разом отбросить огромный пласт алгоритмов, которые в принципе не смогут решить задачу. Например, для задачи о разрезе с произвольными весами можно даже не пытаться построить алгоритм, использующий только линейные тесты и отличный от полного перебора.

Пример. Рассмотрим наиболее простую задачу дискретной оптимизации: выбор наибольшего числа в массиве $a = (a_1, a_2, \dots a_n)$. Стандартный алгоритм, основанный на попарных сравнениях чисел должен будет один раз пройти весь массив и выполнить $n - 1$ сравнение.

Предположим, что у нас есть алгоритм, который вместо попарных сравнений вычисляет знак линейной функции $(c, a) \geq 0$ или $(c, a) \leq 0$, где $c \in \mathbb{R}^n$. Например, если

$$(1, -1, -1, 0, \dots, 0) \cdot (a_1, a_2, \dots a_n)^T \geq 0,$$

$$a_1 - a_2 - a_3 \geq 0,$$

$$a_1 \geq a_2 + a_3,$$

$$a_1 \geq a_2, \qquad a_1 \geq a_3,$$

что позволяет за одно линейное сравнение отбросить сразу два числа.

Вопрос: можно ли придумать подобный алгоритм, трудоемкость которого в худшем случае составит менее $n - 1$ линейных сравнения.

Геометрическая интерпретация задачи поиска наибольшего числа имеет вид

$$(a, x) \to max, \qquad x \in X = \begin{Bmatrix} (1,0, \dots, 0), \\ (0,1, \dots, 0), \\ \dots \\ (0,0, \dots, 1) \end{Bmatrix}.$$

Вершины многогранника задачи $P(X)$ аффинно независимы (не лежат в одной гиперплоскости), выпуклая оболочка n аффинно независимых точек – это $(n - 1)$-мерный симплекс (треугольник для трех точек, не лежащих на одной прямой, тетраэдр для четырех точек, не лежащих в одной плоскости и т.д.). Любое подмножество вершин симплекса – это подмножество аффинно независимого множества, т.е. аффинно независимое множество, выпуклая

оболочка которого, соответственно, тоже симплекс. В частности, все вершины симплекса попарно смежны и граф $P(X)$ является полным.

Рассмотренный алгоритм, по построению, является алгоритмом прямого типа, а значит его трудоемкость не может быть меньше значения

$$h(T) \geq \omega(X) = n - 1,$$

т.е. трудоемкости полного перебора с попарными сравнениями.

Итак, в случае если граф многогранника является полным, все алгоритмы прямого типа вырождаются в полный перебор. Интересным представляется вопрос: а насколько ожидаема такая ситуация. В пространствах $\mathbb{R}^2$ и $\mathbb{R}^3$ только симплексы (треугольник и тетраэдр) обладают полным графом. Однако уже в пространстве размерности четыре ситуация принимает принципиально иной характер. Как уже отмечалось ранее, циклический многогранник (выпуклая оболочка произвольных точек на моментной кривой $x(t) = (t, t^2, t^3, \dots, t^d)$ в $\mathbb{R}^d$) является смежностным (любые $\lfloor d/2 \rfloor$ его вершин образуют грань). Соответственно, уже в четырехмерном пространстве можно построить циклический многогранник с произвольным числом вершин, причем все они будут попарно смежны, и граф многогранника окажется полным. При увеличении размерности подобная ситуация становится общей. Так имеет место [35]

Теорема (Бондаренко). *Выберем случайным образом с равномерным распределение вероятности k вершин d-мерного куба $\{0,1\}^d$. Обозначим через $P_{k,d}$ вероятность того, что все вершины многогранника – выпуклой оболочки выбранных точек окажутся попарно смежными. Тогда $\forall d > 3$ и $k \leq 2^d$ справедливо неравенство*

$$P_{k,d} > 1 - \frac{k^4}{4}\left(\frac{5}{8}\right)^d.$$

Так, если выбрать случайным образом 25000 вершин 100-мерного куба, вероятность $P_{25000,100}$ получить многогранник с полным графом составит более $0{,}99$.

Упражнение. 1) Показать, что в теореме о попарно смежных полиэдрах условие выпуклости является ключевым, и в случае его исключения теорема перестает работать: построить контрпример произвольных попарно смежных многоугольников на плоскости.

2) Доказать, что любой алгоритм сортировки, использующий попарные сравнения чисел будет алгоритмом прямого типа.

10. Задача о разрезе с неотрицательными весами

Граф разрезного многогранника $CUT(n)$ является полным и ни один алгоритм прямого типа не сможет решить задачу о разрезе быстрее чем $\omega(X) - 1 = 2^{n-1} - 1$ шагов, т.е. не выполнив полный перебор всех возможных разрезов в графе. Действительно [17], задача о разрезе в оптимизационной форме принадлежит классу NP-трудных задач, и для нее не известно ни одного алгоритма решения отличного от полного перебора. Однако все вышесказанное относится к задаче о разрезе в графе с произвольными весами ребер, в то время как классическая постановка задачи дополнена условием, что ребра могут иметь только неотрицательные веса. При этом задачи о минимальном и максимальном разрезе с неотрицательными весами принципиально различаются. В то время как задача о максимальном разрезе остается в такой формулировке NP-трудной, задача о минимальном разрезе полиномиально разрешима, в частности потоковыми методами Форда-Фалкерсона (более подробно смотри главу 1 «Задача о разрезе»). Для пояснения сложившейся ситуации необходимо переопределить конусные разбиения пространства.

Пусть X ($X \subset \mathbb{R}^d$) массовая задача.

Определение. *Для произвольного решения $x \in X$ введем следующие множества*

$$K^+_{min}(x) = \{c \in \mathbb{R}^d, c \geq 0 : (c,x) \geq (c,y), \forall y \in X\},$$

$$K^+_{max}(x) = \{c \in \mathbb{R}^d, c \geq 0 : (c,x) \leq (c,y), \forall y \in X\},$$

которые назовем неотрицательными конусами решения x для задач на минимум и на максимум.

Определение. *Учитывая, что*

$$\bigcup_{x \in X} K^+_{min}(x) = \bigcup_{x \in X} K^+_{max}(x) = \mathbb{R}^d_{\geq 0},$$

совокупности всех конусов $K^+_{min}(x)$ и $K^+_{max}(x)$ $(x \in X)$ назовем неотрицательными конусными разбиением пространства $\mathbb{R}^d$ по множеству X для задач на минимум и на максимум, соответственно.

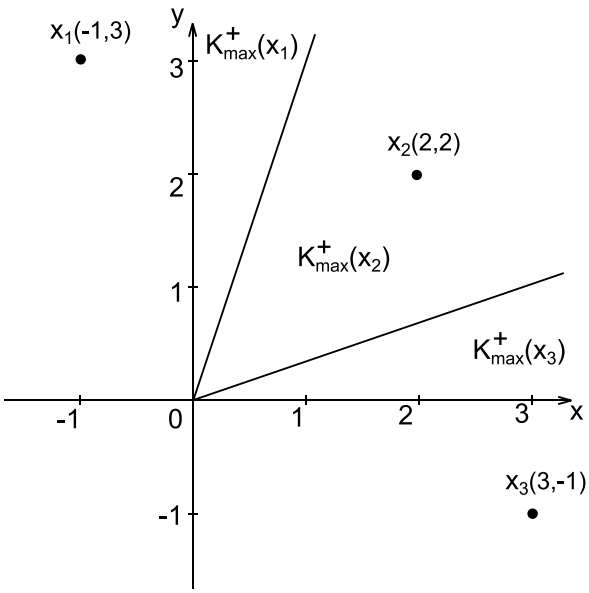

Рис.14. Пример неотрицательного конусного разбиения

Введем граф конусного разбиения пространства $\mathbb{R}^d$ ($\mathbb{R}^d_{\geq 0}$) по множеству X, вершинами которого будут конусы, и два конуса $K(x)$ и $K(y)$ смежны тогда и только тогда, когда они имеют общую гиперграни

$$\dim\big(K(x_1) \cap K(x_2)\big) = d - 1.$$

По построению, граф конусного разбиения пространства $\mathbb{R}^d$ совпадет с графом многогранника задачи $P(X)$, что обеспечит перенос всех результатов относительно кликового числа графа многогранника и трудоемкости алгоритмов прямого типа на граф конусного разбиения. Однако если рассматривается задача с дополнительным условием неотрицательности исходных данных, то граф конусного разбиения пространства $\mathbb{R}^d_{\geq 0}$ уже не будет совпадать с графом $P(X)$. Так, в примере на Рис. 14 неотрицательные конусы $K^+_{max}(x_1)$ и $K^+_{max}(x_3)$ не имеют общей гиперграни и, соответственно, не смежны. В то же время вершины x_1 и x_3 многогранника $P(X)$ соединены ребром, так как $P(X)$ – это выпуклая оболочка трех аффинно независимых точек (треугольник).

Таким образом, для задач с неотрицательными исходными данными, например, для классической постановки задач о максимальном и минимальном разрезах с неотрицательными весами, для получения нижней оценки на трудоемкость алгоритмов прямого типа необходимо исследовать не граф многогранника, а граф неотрицательного конусного разбиения. Соответственно, для задачи на максимум высота дерева прямого типа удовлетворяет неравенству

$$h_{max}(T) \geq \omega\left(K^{+}_{max}(X)\right) - 1,$$

где $\omega\left(K^{+}_{max}(X)\right)$ – кликовое число графа неотрицательного конусного разбиения. Аналогично для задачи на минимум имеем

$$h_{min}(T) \geq \omega\left(K^{+}_{min}(X)\right) - 1.$$

Тогда полученные в работе [36] результаты совпадают с известной трудоемкостью алгоритмов для задач о минимальном и максимальном разрезе.

Теорема (Бондаренко, Николаев). *Кликовое число графа конусного разбиения задачи о минимальном разрезе с неотрицательными весами линейно по числу n вершин исходного графа задачи и равно*

$$\omega\left(K^{+}_{min}(X)\right) = 2n - 3.$$

В качестве примера $2n-3$ разрезов (P, Q) в графе на n вершинах, конусы которых образуют клику можно рассмотреть следующие разрезы

- n разрезов вида $P_i = \{i\}, Q_i = \mathbb{N}_n \backslash \{i\}, \forall i \in \mathbb{N}_n$;
- $n-3$ разреза вида $P_k = \{1,2,\dots,k\}$, $Q_k = \{k+1, k+2, \dots, n\}$, где $2 \leq k \leq n-2$.

Теорема (Бондаренко, Николаев). *Кликовое число графа конусного разбиения задачи о максимальном разрезе с неотрицательными весами сверхполиномиально по числу n вершин исходного графа задачи*

$$\omega\left(K^{+}_{max}(X)\right) \geq C_n^{\frac{n}{2}-1} + n \geq \frac{2^n}{\sqrt{2n}}\left(1 - \frac{2}{n+2}\right) + n \text{ (для четных } n),$$

$$\omega\left(K^{+}_{max}(X)\right) \geq C_n^{\frac{n-1}{2}} + n \geq \frac{2^n}{\sqrt{2(n-1)}}\left(1 - \frac{1}{n+1}\right) + n \text{ (для нечетных } n).$$

Клика в графе неотрицательного конусного разбиения достигается, как и для задачи на минимум, на разрезах, отделяющих одну вершину от остальных,

а также на разрезах, разбивающих множества вершин на равные или почти равные (если n нечетно) подмножества.

Таким образом, получаем линейную нижнюю оценку $\Omega(n)$ трудоемкости алгоритмов прямого типа для задачи о минимальном разрезе. Лучший алгоритм решения задачи на сегодняшний момент обладает трудоемкостью $O(n^3)$ для полного графа. А также сверхполиномиальную нижнюю оценку $\Omega(\frac{2^n}{\sqrt{n}})$ для задачи о максимальном разрезе, единственным известным алгоритмом для которой является полный перебор $O(2^n)$.

Сложившаяся ситуация объясняется характерными особенностями самой задачи о разрезе. Для других задач ситуация может быть принципиально иной [36].

Теорема (Бондаренко, Николаев). *Если все точки множества* $X \subset \{0,1\}^d$ *лежат на сфере с центром в нуле:*

$$\forall a, b \in X: \quad \|a\| = \|b\|,$$

и для пары точек $x, y \in X$ *конусы* $K_{max}(x)$ *и* $K_{max}(y)$ *смежны, то и конусы* $K^+_{max}(x)$ *и* $K^+_{max}(y)$ *также смежны.*

Аналогичное утверждение верно и для задачи на минимум.

В результате, для задач, множества решений которых представляют собой вершины единичного d-мерного куба и евклидова норма векторов константа, графы простых конусных разбиений совпадут с графами неотрицательных конусных разбиений и, соответственно, с графом многогранника задачи $P(X)$.

Это верно, например, для задачи коммивояжера (все решения – гамильтоновы циклы содержат ровно n ребер и столько же единиц в кодирующих их характеристических векторах) и для задачи о минимальном остовном дереве (каждое остовное дерево содержит в точности $n-1$ ребро). Действительно, все версии задачи коммивояжера NP-трудны, в то время как все постановки задачи об остовном дереве полиномиально разрешимы. В то же время, для задачи о пути в графе маршрут может содержать произвольное число ребер и возможные решения не принадлежат сфере. Имеем, если ребра графа неотрицательны: задача о кратчайшем пути полиномиально разрешима, например, известным алгоритмов Дейкстры, а задача о самом длинном пути в графе NP-трудна.

Упражнение. Проверить сверхполиномиальные нижние оценки

$$C_n^{\frac{n}{2}-1} + n \geq \frac{2^n}{\sqrt{2n}}\left(1 - \frac{2}{n+2}\right) + n \text{ (для четных } n\text{)},$$

$$C_n^{\frac{n-1}{2}} + n \geq \frac{2^n}{\sqrt{2(n-1)}}\left(1 - \frac{1}{n+1}\right) + n \text{ (для нечетных } n\text{)}$$

для кликового числа графа неотрицательного конусного разбиения задачи о максимальном разрезе, используя свойства центрального биномиального коэффициента.

11. Релаксационные многогранники

Вернемся к геометрическим способам решения задач комбинаторной оптимизации. Прямой подход со сведением к задаче линейного программирования не прошел в силу сверхэкспоненциальной сложности, полученных в результате построения выпуклой оболочки множества решений многогранников. Однако проблему можно обойти с неожиданной стороны: если исключить ограничение, что решения могут быть только целыми.

Рассмотрим многогранник $RMET_n$ из пространства $\mathbb{R}^{4n^2}$, удовлетворяющий системе ограничений

$$x_{i,j} + y_{i,j} + z_{i,j} + t_{i,j} = 1,$$

$$x_{i,j} + y_{i,j} = x_{k,j} + y_{k,j},$$

$$x_{i,j} + z_{i,j} = x_{i,k} + z_{i,k},$$

$$x_{i,j} = x_{j,i}, t_{i,j} = t_{j,i}, y_{i,j} = z_{j,i},$$

$$y_{i,i} = z_{i,i} = 0,$$

$$x_{i,j} \geq 0, y_{i,j} \geq 0, z_{i,j} \geq 0, t_{i,j} \geq 0,$$

где i, j, k пробегают независимо значения $1, \dots, n$.

Впервые многогранник $RMET_n$ был описан В.А. Бондаренко в работе [37], позднее независимо описан М. Падбергом в [29], в связи с задачей булева квадратичного программирования. Подробно свойства многогранника исследованы в монографии М. Деза и М. Лоран «Геометрия разрезов и метрик» [11], где он получил название *корневого полуметрического многогранника* (rooted semimetric polytope, $RMET$).

Первое, что следует отметить применительно к $RMET_n$, – это тот факт, что число линейных ограничений во внешнем описании многогранника полиномиально по n и не превосходит $9n^2$ (таким образом по размерности пространства $4n^2$ оно линейно). Это выгодно отличает корневой полуметрический многогранник от рассматриваемых ранее многогранников задач о разрезе и коммивояжера, обладающих сверхэкспоненциальным числом фасет, и, соответственно, линейных ограничений во внешнем описании.

Учитывая структуру ограничений $RMET_n$, наиболее удобно координаты его точек представлять в виде блочной матрицы (Табл. 4). Причем, принимая

во внимание симметриченость матрицы координат, как правило можно ограничиться лишь верхней половиной.

$x_{i,i}$	0	$x_{i,j}$	$y_{i,j}$	$x_{i,k}$	$y_{i,k}$
0	$t_{i,i}$	$z_{i,j}$	$t_{i,j}$	$z_{i,k}$	$t_{i,k}$
$x_{i,j}$	$z_{i,j}$	$x_{j,j}$	0	$x_{j,k}$	$y_{j,k}$
$y_{i,j}$	$t_{i,j}$	0	$t_{j,j}$	$z_{i,k}$	$t_{j,k}$
$x_{i,k}$	$z_{i,k}$	$x_{j,k}$	$z_{j,k}$	$x_{k,k}$	0
$y_{i,k}$	$t_{i,k}$	$y_{i,k}$	$t_{j,k}$	0	$t_{k,k}$

Таблица 4. Блочная матрица координат $RMET_n$

Так как описывающая корневой полуметрический многогранник система содержит не только неравенства, но и уравнения, $RMET_n$ не является полноразмерным многогранником: его реальная размерность не совпадает с размерностью пространства, в котором он описан. Например, выпуклая оболочка трех точек в $\mathbb{R}^3$, не лежащих на одной прямой, будет треугольником, т.е. двумерным многогранником, лежащем в трехмерном пространстве. Действительная размерность корневого полуметрического многогранника составляет $n(n+1)/2$, а число фасет равно $2n^2$.

Принципиальным отличием корневого полуметрического многогранника от ранее рассмотренных является тот факт, что не все его вершины целочисленные. На самом деле, как было установлено в работе [29], они являются полуцелыми: координаты вершин принимают значения из множества $\left\{0, \frac{1}{2}, 1\right\}$. Все вершины $RMET_n$ полностью описаны [35], в частности целочисленные вершины определяются в соответствии со следующим утверждением:

Теорема. *Множество целочисленных вершин корневого полуметрического многогранника $RMET_n$ состоит из 2^n точек, каждая из которых имеет координаты*

$$x_{i,j} = x_{i,i} \cdot x_{j,j}, \qquad y_{i,j} = \left(1 - x_{i,i}\right) \cdot x_{j,j},$$

$$z_{i,j} = x_{i,i} \cdot (1 - x_{j,j}), \qquad t_{i,j} = (1 - x_{i,i}) \cdot (1 - x_{j,j}).$$

Таким образом, координаты целых вершин $RMET_n$ напрямую определяются через значения диагональных координат $x_{i,i}$.

А теперь перейдем к непосредственному применению корневого полуметрического многогранника. Установим связь между $RMET_n$ и задачей о разрезе.

Каждому подмножеству $S \subseteq \mathbb{N}_n$ (т.е. каждому разрезу в полном графе) сопоставим целую вершину многогранника $RMET_n$ по следующему правилу:

$$x_{i,i} = \begin{cases} 1, & \text{если } i \in S, \\ 0, & \text{если } i \in \mathbb{N}_n \backslash S. \end{cases}$$

Произвольному взвешенному неориентированному графу $G = (V, E)$ на n вершинах ($|V| = n, C: E \to \mathbb{R}_{\geq 0}$) сопоставим целевой вектор $\hat{c} \in \mathbb{R}^{4n^2}$, все координаты которого равны 0, кроме

$$\forall i, j \ (1 \leq i < j \leq n) \quad \hat{y}_{i,j} = \hat{z}_{i,j} = \begin{cases} C_{i,j}, & \text{если ребро } (i,j) \in E, \\ 0, & \text{в противном случае.} \end{cases}$$

Рассмотрим целевую функцию $\hat{f} = (\hat{c}, x)$. Нетрудно убедиться, что значение построенной данным образом функции $\hat{f}(z)$ в некоторой целой вершине z многогранника $RMET_n$ в точности равно величине разреза соответствующего этой вершине. Таким образом, задача о разрезе сводится к задаче оптимизации линейной целевой функции на множестве целых вершин корневого полуметрического многогранника, т.е. к задаче целочисленного линейного программирования на $RMET_n$.

Обозначим через Z множество целых вершин корневого полуметрического многогранника и построим его выпуклую оболочку $RMET_n^Z = \text{conv}\,(Z)$. Вершины $RMET_n^Z$ однозначно соответствуют множеству всех разрезных векторов полного графа и, соответственно, множеству вершин разрезного многогранника. Если быть точнее, существует линейная биекция между $RMET_n^Z$ и CUT_{n+1}, учитывая, что число разрезов в полном графе на n вершинах равно 2^{n-1}[11]. А если быть еще точнее, $RMET_n^Z$ совпадает с многогранником BQP_n другой известной оптимизационной задачи: задачи булева квадратичного программирования, который, соответственно, находится в биективном соотношении с CUT_{n+1}.

Если считать, что за вершины многогранника мы принимаем множество возможных решений задачи, то корневой полуметрический многогранник получится в том случае, когда мы допускаем в задаче о разрезе существование нецелочисленных решений: возможность «разрезать» некоторую вершину графа пополам и включить ее в оба подмножества P и Q. Многогранники, которые получаются из задач исключением ограничения о целочисленности возможных решений называются релаксационными многогранниками. Так корневой полуметрический многогранник является релаксационным многогранником задачи о разрезе (и задачи булева квадратичного программирования в силу вышесказанного).

Безусловно, подобная конструкция не позволит нам напрямую решить задачу о разрезе, так как задача целочисленного программирования и в общем случае и на корневом полуметрическом многограннике является NP-полной. Однако она дает нам новый инструментарий, который может быть применен в неожиданных направлениях. Так, рассмотрим ***задачу об ориентированном максимальном разрезе*** в следующей формулировке [35]

Заданы: ориентированный граф $G = (V, A)$ *и некоторая функция* $C: A \to \mathbb{R}_{\geq 0}$, *каждой дуге из* A *(упорядоченной паре вершин* $\{i, j\}$, *таких что* $i, j \in V, i \neq j$*) сопоставляющая неотрицательное действительное число* $C_{i,j}$, *называемое весом дуги.*

Требуется найти такое разбиение множества узлов V *на два непересекающихся подмножества* P *и* Q *(ориентированный разрез), чтобы величина*

$$\sum_{i \in P, j \in Q} C_{i,j} - \sum_{i \in Q, j \in P} C_{i,j}$$

была максимальной.

Определим целевую функцию $\check{f} = (\check{c}, x)$ с целевым вектором $\check{c} = (\check{x}_{i,j}, \check{y}_{i,j}, \check{z}_{i,j}, \check{t}_{i,j}, i, j \in \mathbb{N}_n)$, положив

$$z_{i,j} = C_{i,j}, y_{i,j} = -C_{i,j}, i, j \in \mathbb{N}_n \ (i \neq j),$$

а остальные координаты равными нулю.

Нетрудно убедиться, что значение $\check{f} = (\check{c}, x)$ в целых вершинах $RMET_n$ в точности совпадает с значением величины ориентированного максимального разреза.

Для корневого полуметрического многогранника известно, что

Теорема (Бондаренко). *Если для некоторого целевого вектора* $\check{c}$ *выполняется условие*

$$\forall i,j \in \mathbb{N}_n \ (i \neq j) \quad \check{x}_{i,j} + \check{t}_{i,j} = \check{y}_{i,j} + \check{z}_{i,j},$$

то на корневом полуметрическом многограннике $RMET_n$ *целевая функция* $\check{f} = (\check{c}, x)$ *достигает своего максимума строго в целой вершине.*

Таким образом, задача о максимальном разрезе в ориентированном графе сводится к задаче линейного программирования на корневом полуметрическом многограннике $RMET_n$, а значит полиномиально разрешима.

Другое интересное свойство корневого полуметрического многогранника связано с еще одной задачей на целых вершинах: ***задачей распознавания целочисленности***.

Пусть $\bar{P}$ *– некоторый класс выпуклых многогранников. Пусть* $P \in \bar{P}$ *и* Z *– множество всех целых точек из* P. *Пусть* $f(x) = (c, x)$ *некоторая линейная целевая функция. Требуется выяснить: выполняется ли равенство*

$$\max_{x \in M} f(x) = \max_{z \in Z} f(z).$$

Для решения задачи распознавания целочисленности необходимо ответить на вопрос: «есть ли среди вершин, на которых заданная линейная целевая функция достигает своего максимума, хотя бы одна целая?»

Построим новый многогранник MET_n, полученный из $RMET_n$ наложением дополнительных ограничений

$$x_{i,j} + t_{i,j} + x_{i,k} + t_{i,k} + y_{j,k} + z_{j,k} \leq 2,$$
$$x_{i,j} + t_{i,j} + y_{i,k} + z_{i,k} + x_{j,k} + t_{j,k} \leq 2,$$
$$y_{i,j} + z_{i,j} + x_{i,k} + t_{i,k} + x_{j,k} + t_{j,k} \leq 2,$$
$$y_{i,j} + z_{i,j} + y_{i,k} + z_{i,k} + y_{j,k} + t_{j,k} \leq 2,$$

для каждой тройки индексов $1 \leq i < j < k \leq n$.

Нетрудно проверить, что все целые вершины $RMET_n$ удовлетворяют новым ограничениям и, соответственно, являются вершинами MET_n. Таким образом, многогранник MET_n также является релаксационным многогранником задачи о разрезе и называется метрическим многогранником. В свою очередь, дополнительные ограничения, переводящие корневой полуметрический многогранник в метрический, носят название «неравенств треугольника».

$$CUT_{n+1} \sim BQP_n = RMET_n^Z \subseteq MET_n \subseteq RMET_n.$$

Метрический многогранник обладает рядом интересных свойств, в частности, в работе [38] была установлена

Теорема (Бондаренко, Урываев). *Каждая точка MET_n является выпуклой комбинацией вершин $RMET_n$ среди которых есть хотя бы одна целая вершина.*

Для некоторой линейной целевой функции $f(x) = (c, x)$, в силу вложенности многогранников $RMET_n^Z \subseteq MET_n \subseteq RMET_n$, выполняется

$$\max_{x \in RMET_n^Z} f(x) \leq \max_{x \in MET_n} f(x) \leq \max_{x \in RMET_n} f(x).$$

Рассмотрим задачу распознавания целочисленности на корневом полуметрическом многограннике. Дважды решим задачу линейного программирования и найдем $\max f(x)$ на MET_n и $RMET_n$. Если

$$\max_{x \in RMET_n^Z} f(x) \leq \max_{x \in MET_n} f(x) < \max_{x \in RMET_n} f(x),$$

то, очевидным образом, максимум $f(x)$ не достигается в целой вершине $RMET_n$, и задача распознавания целочисленности предполагает ответ «нет». Если же

$$\max_{x \in RMET_n^Z} f(x) \leq \max_{x \in MET_n} f(x) = \max_{x \in RMET_n} f(x),$$

тогда вершина MET_n, на которой $f(x)$ достигает максимума, является внутренней точкой грани $RMET_n$, хотя бы одна из вершин которой целочисленная (в силу свойств MET_n). Так как оптимальные значения функций равны, то эта грань максимизирует $f(x)$ на всем многограннике $RMET_n$, а значит ответ в задаче распознавания целочисленности будет «да»: среди вершин $RMET_n$, на которых $f(x)$ достигает максимума есть хотя бы одна целая. Задача распознавания целочисленности на корневом полуметрическом многограннике сводится к двойному применению алгоритмов линейного программирования, следовательно, она полиномиально разрешима.

Таким образом, полиномиально разрешимой будет любая задача комбинаторной оптимизации, которую можно свести к распознаванию целочисленности на $RMET_n$.

Данный результат также позволяет, в том случае, когда для заданной целевой функции задача распознавания целочисленности на корневом

полуметрическом многограннике предполагает ответ «да», найти решение задачи о разрезе за полиномиальное время. Получаем процедуру, которая или решит задачу о разрезе, или скажет, что не может найти решение. В строгом смысле, это не алгоритм, так как не гарантирует верного результата для всех допустимых исходных данных. Но все же это значительно лучше, чем совсем ничего (т.е. полный перебор).

Упражнение. Свести полиномиально разрешимую задачу комбинаторной оптимизации в форме распознавания к задаче распознавания целочисленности на корневом полуметрическом многограннике.

12. Полуопределенное программирование

Задача о максимальном разрезе принадлежит классу NP-трудных задач и для нее не известно ни одного алгоритма точного решения, кроме полного перебора. Как правило, в подобных случаях, если не возможно за разумное время найти точного решения задачи, прибегают к так называемым аппроксимационным или приближенным (approximation algorithms) алгоритмам, которые за полиномиальное время могут решить задачу с некоторой гарантированной точностью. Приведем строгое определение [18]

Определение. *Пусть $\mathcal{P}$ – это некоторая массовая задача максимизации, а $\mathcal{I}$ - ее множество возможных индивидуальных задач. Пусть A – это алгоритм, который для каждой индивидуальной задачи $I \in \mathcal{I}$ возвращает допустимое решение $A(I)$ со значением $\omega(A(I))$. Пусть $\delta: \mathcal{I} \to \mathbb{R}_{\geq 0}$ некоторая функция.*

Алгоритм A называется δ-аппроксимационным алгоритмом для задачи $\mathcal{P}$, если выполняются следующие условия:

а) найдется такой полином p, что для любой индивидуальной задачи $I \in \mathcal{I}$ время работы алгоритма A ограничено сверху функцией $p(|I|)$, где $|I|$ – длина входа индивидуальной задачи I;

б) для любой индивидуальной задачи $I \in \mathcal{I}$

$$\omega\big(A(I)\big) \geq \delta(I)\mathrm{opt}(I),$$

где $\mathrm{opt}(I)$ – это оптимальное решение для индивидуальной задачи максимизации I.

Таким образом, величина δ обозначает гарантированную точность приближенного алгоритма. Интересным частным случаем является постоянное значение δ ($\delta = const \in [0,1]$). Очевидно, что чем ближе значение δ к единице, тем выше качество аппроксимации.

Крайне полезным является обобщение определения для вероятностных алгоритмов (алгоритмов, предусматривающий обращение на определённых этапах своей работы к таблице случайных чисел для принятия решений, randomized algorithms).

Вероятностный δ-аппроксимационный алгоритм должен иметь ожидаемое полиномиальное время работы и всегда возвращать такое решение, что

$$E\left(\omega\big(A(I)\big)\right) \geq \delta(I)\mathrm{opt}(I).$$

Для вероятностных алгоритмов значение $\omega(A(I))$ – это случайная величина, и мы требуем, чтобы ее математическое ожидание было хорошим приближением точного решения задачи.

Если рассматривать задачу минимизации, то в предложенном определении нужно потребовать выполнение

$$\omega\big(A(I)\big) \leq \delta(I)\mathrm{opt}(I), \qquad \forall I \in \mathcal{I},$$

что приведет к δ-аппроксимационным алгоритмам, где величина $\delta \geq 1$.

Построим вероятностный аппроксимационный алгоритм для задачи о максимальном разрезе в взвешенном графе $G = (V, E)$ с функцией весов $C: E \to \mathbb{R}_{\geq 0}$. Рассмотрим случайную величину ξ с дискретным равномерным распределением, принимающую с вероятностью $1/2$ значения 0 или 1 (подбрасывание монетки). Для каждой вершины графа $v \in V$ найдем значение ξ. Если $\xi(v) = 1$, то $v \in P$, в противном случае $v \in Q$. Назовем этот алгоритм *FlipCoinMaxCut.*

Может показаться, что построенный подобным образом алгоритм никуда не годится, ведь он даже не учитывает веса ребер. Однако имеет место

Теорема. *Алгоритм FlipCoinMaxCut является вероятностным* $1/2$*-аппроксимационным алгоритмом для задачи о максимальном разрезе.*

Действительно, математическое ожидание величины полученного данным образом разреза составляет

$$E\big(\omega(ApproximateMaxCut)\big) = \sum_{e \in E}\Big(C(e) \cdot \text{вероятность}\big(e \in E(P, Q)\big)\Big).$$

Для каждого ребра $(i, j) \in E$ имеем дискретное равномерное распределение с четырьмя значениями: $i, j \in P$, или $i \in P, j \in Q$, или $j \in P, i \in Q$, или $i, j \in Q$. Таким образом,

$$E\big(\omega(ApproximateMaxCut)\big) = \sum_{e \in E}\left(C(e) \cdot \frac{1}{2}\right) = \frac{1}{2}\sum_{e \in E} C(e) \geq \frac{1}{2}\mathrm{opt}(G).$$

Половина суммы всех весов ребер в графе G гарантированно не может быть меньше значения величины максимального разреза. Таким образом, в худшем случае решение, полученное алгоритмом *FlipCoinMaxCut*, будет отличаться от точного не более чем в два раза.

Известны некоторые модификации алгоритма, которые позволяют незначительно улучшить точность его работы, например, построить $0.5(1 + 1/m)$-аппроксимационный алгоритм, где $m = |E|$. Однако до 1994 года не было описано ни одного δ-аппроксимационного алгоритма с постоянным значением $\delta > 0.5$.

Принципиальный прорыв произошел в знаменитой работе М. Гоманса и Д. Уильямсона [19], за которую авторы получили престижную премию Фалкерсона. Гоманс и Уильямсон смогли построить вероятностный 0.878-аппроксимационный алгоритм для задачи о максимальном разрезе с помощью геометрической интерпретации задачи и теории полуопределенного программирования.

Чтобы пояснить идею полуопределенного программирования (semidefinite programming, SDP) мы сначала вернемся к привычной концепции линейного программирования:

$$(c, x) = c^T x \to max,$$
$$x \in \mathbb{R}^n:$$
$$Ax = b,$$
$$x \geq 0.$$

Чтобы перейти к задаче полуопределенного программирования мы заменим векторное пространство $\mathbb{R}^n$ на векторное пространство

$$S_n = \left\{X \in \mathbb{R}^{n \times n} : \forall\, x_{i,j} = x_{j,i}\right\}$$

симметрических $n \times n$ матриц. Матрицу A заменим на линейный оператор $A: S_n \to \mathbb{R}^m$, а стандартное скалярное произведение $(x, y) = x^T y$ над $\mathbb{R}^n$ на стандартное скалярное произведение для матриц

$$(X, Y) = \mathrm{Tr}(X^T Y) = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j} y_{i,j}$$

над пространством S_n. Напомним, что след квадратной матрицы X (trace, $\mathrm{Tr}(X)$) – это сумма диагональных элементов X.

Наконец, ограничение $x \geq 0$ мы заменим на $X \succcurlyeq 0$, что означает, что матрица X положительно полуопределена.

Определение. *Симметрическая матрица называется положительно полуопределенной, если все ее собственные значения неотрицательны.*

Так как положительно полуопределенная матрица является симметрической, то по спектральной теореме все ее собственные значения – действительные числа, и, соответственно, могут быть сравнимы с нулем. Различные эквивалентные формулировки положительно полуопределенных матриц можно обобщить следующим теоремой:

Теорема. *Пусть матрица* $M \in S_n$, *тогда следующие утверждения эквивалентны*

а) M *– это положительно полуопределенная матрица, т.е. все собственные значения* M *неотрицательны;*

б) $x^T M x \geq 0$, *для всех* $x \in \mathbb{R}^n$;

в) найдется такая матрица $U \in \mathbb{R}^{n \times n}$, *что* $M = U^T U$ *(разложение Холецкого).*

Таким образом, постановка задачи полуопределенного программирования примет вид

$$\mathrm{Tr}(C^T X) \to max,$$
$$X \in S_n:$$
$$A(X) = b,$$
$$X \succcurlyeq 0,$$

где $A: S_n \to \mathbb{R}^m$ – это линейный оператор.

Известно [18], что методы внутренней точки для задачи полуопределенного программирования могут решить задачу с любой как угодно малой аддитивной ошибкой ϵ (соответственно с любой требуемой точностью) за время, полиномиально зависящее от длины входа и функции $\log\frac{1}{\epsilon}$. Подобная оценка сложности несколько непривычна, особенно после задачи линейного программирования, для которой за полиномиальное время можно найти точное решение. Но применение методов полуопределенного программирования крайне эффективно для построения аппроксимационных алгоритмов решения сложных задач.

Сведем задачу о максимальном разрезе к задаче полуопределенного программирования за несколько шагов. Прежде всего сформулируем задачу о разрезе в форме задачи условной оптимизации. Пусть множество вершин V графа G равно $V = \mathbb{N}_n$. Введем новые переменные

$$x_1, x_2, \dots, x_n \in \{-1,1\}.$$

Каждое назначение этих переменных кодирует разрез (P, Q), где

$$P = \{i \in V : x_i = 1\}, \qquad Q = \{j \in V : x_j = -1\}.$$

Пусть $C_{i,j}$ равно весу ребра (i,j), если это ребро есть в графе G, и нулю в противном случае. Тогда величина

$$C_{i,j}\left(\frac{1 - x_i x_j}{2}\right)$$

в точности соответствует вкладу ребра (i,j) в соответствующий разрез. Действительно, если пара (i,j) не является ребром, или не входит в разрез, то $C_{i,j} = 0$ или $x_i x_j = 1$, и вклад ребра в величину разреза равен нулю. Если же ребро (i,j) попало в разрез, то $x_i x_j = -1$ и вклад ребра составляет $C_{i,j}$.

Таким образом задача о максимальном разрезе примет вид

$$\sum_{i<j} C_{i,j}\left(\frac{1 - x_i x_j}{2}\right) \to max,$$

$$x_1, x_2, \dots, x_n \in \{-1,1\}.$$

Обозначим эту задачу как $CO(G)$ (от constraint optimization).

А теперь ключевой шаг: мы составим задачу полуопределенного программирования, решение которой будет верхней оценкой на величину максимального разреза в графе $\mathrm{opt}(G)$. Для этого мы заменим каждую переменную x_i на вектор u_i, принадлежащий единичной $(n-1)$-мерной сфере

$$u_i \in S^{n-1} = \{u \in \mathbb{R}^n : \|u\| = 1\}.$$

Тогда задача примет вид

$$\sum_{i<j} C_{i,j}\left(\frac{1 - u_i^T u_j}{2}\right) \to max,$$

$$u_1, u_2, \dots, u_n \in S^{n-1}.$$

Эту задачу обозначим как $RCO(G)$ (от relaxed constraint optimization).

Учитывая тот факт, что сферу $S^0 = \{-1,1\}$ можно вложить в S^{n-1} с помощью отображения $\tau : x \to (0,0,\dots,0,x)$, мы получаем следующее важное свойство: если $(x_1, x_2, \dots, x_n)$ – это допустимое решение $CO(G)$ с значением целевой функции

$$z = \sum_{i<j} C_{i,j} \left(\frac{1 - x_i x_j}{2}\right),$$

то $(\tau(x_1), \tau(x_2), \dots, \tau(x_n))$ – это допустимое решение $RCO(G)$ с значением целевой функции

$$\sum_{i<j} C_{i,j} \left(\frac{1 - \tau(x_i)^T \tau(x_j)}{2}\right) = \sum_{i<j} C_{i,j} \left(\frac{1 - x_i x_j}{2}\right) = z.$$

Таким образом, задача $RCO(G)$ является релаксацией задачи $CO(G)$, так как имеет больше допустимых решений, следовательно, ее решение не может быть меньше чем величина максимального разреза $\mathrm{opt}(G)$. Отметим, что значение целевой функции $RCO(G)$ по прежнему ограничено, так как $u_i^T u_j \geq -1$ для всех i, j.

Остается выполнить еще всего одну замену переменных $x_{i,j} = u_i^T u_j$, и мы придем к задаче полуопределенного программирования $SDP(G)$:

$$\sum_{i<j} C_{i,j} \left(\frac{1 - x_{i,j}}{2}\right) \to max,$$

$$x_{i,i} = 1, \qquad i = 1,2,..,n,$$

$$X \succcurlyeq 0.$$

Действительно, так как $x_{i,j} = u_i^T u_j$, мы имеем

$$X = U^T U,$$

где матрица U получена из столбцов $u_1, u_2, \dots, u_n$. Это разложение Холецкого и, соответственно, матрица X положительно полуопределена. Уравнения $x_{i,i} = 1$ вытекают из условия $u_i \in S^{n-1}$.

Так как задача $SDP(G)$ имеет допустимые решения с тем же значением целевой функции $\gamma \geq \mathrm{opt}(G)$, что и $RCO(G)$, мы можем за полиномиальное время найти такую матрицу $X^* \succcurlyeq 0$, что $x_{i,i} = 1$ для всех i, и значение целевой функции $SDP(G)$ составляет не менее $\gamma - \epsilon$ для любого $\epsilon > 0$. Также за полиномиальное время $O(n^3)$ можно построить разложение Холецкого для матрицы $X^* = (U^*)^T U^*$ [18]. Тогда столбцы $u_1^*, u_2^*, \dots, u_n^*$ – это единичные вектора, которые дают приближенное решение

$$\sum_{i<j} C_{i,j}\left(\frac{1-(u_i^*)^T u_j^*}{2}\right) \geq \gamma - \epsilon \geq \mathrm{opt}(G) - \epsilon.$$

Напомним, что мы хотим решить задачу $CO(G)$ с неизвестными x_i из множества $S^0 = \{-1,1\}$, набор значений которых определяет разрез (P, Q), где $P = \{i \in V: x_i = 1\}$.

У нас есть приближенное решение релаксационной задачи $RCO(G)$ с векторами решений $u_i \in S^{n-1}$. Теперь нам требуется отобразить сферу S^{n-1} обратно на S^0 и при этом не сильно испортить значение целевой функции. Выберем $p \in S^{n-1}$ и определим отображение

$$\lambda_p: S^{n-1} \to S^0,$$

$$\lambda_p(u) = \begin{cases} 1, & \text{если } p^T u \geq 0, \\ -1, & \text{в противном случае.} \end{cases}$$

Данная процедура называется вероятностным округлением.

С геометрической точки зрения ситуация следующая: вектор p разбивает сферу S^{n-1} на замкнутую полусферу $H = \{u \in S^{n-1}: p^T u \geq 0\}$ и ее дополнение. Вектора из H отображаются в 1, а вектора из дополнения к H в -1 (Рис. 15).

Остается только выбрать вектор p. Мы будем это делать случайным образом. Если быть точнее, выберем случайное $p \in S^{n-1}$ с непрерывным равномерным распределение вероятности. Тогда имеет место [19]

Лемма (Гоманс, Уильямсон). *Пусть* $u_i^*, u_j^* \in S^{n-1}$, *тогда*

$$\text{вероятность } \left(\lambda_p(u_i^*) \neq \lambda_p\left(u_j^*\right)\right) = \frac{1}{\pi}\arccos\left((u_i^*)^T u_j^*\right).$$

Лемма дает нам значение вероятности того, что при отображении λ_p вершины i и j попадут в разные подмножества, а значит ребро (i, j) войдет в разрез. Тогда математическое ожидание величины полученного разреза составит

$$\sum_{i<j} C_{i,j}\left(\frac{\arccos\left((u_i^*)^T u_j^*\right)}{\pi}\right).$$

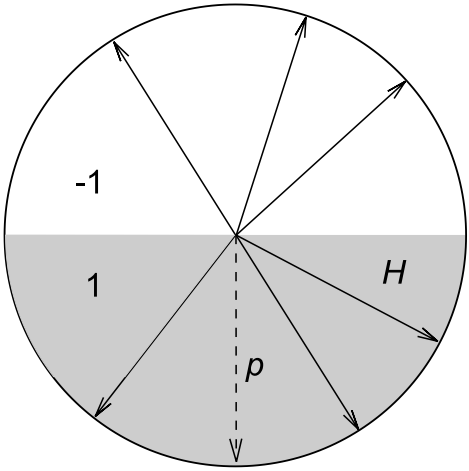

Рис. 15. Пример вероятностного округления

Остается техническая часть, чтобы оценить полученную сумму. Назовем описанный алгоритм решения задачи о разрезе *SDPMaxCut*, тогда выполнена [19]

Теорема (Гоманс, Уильямсон). *Математическое ожидание величины максимального разреза, полученного по алгоритму SDPMaxCut составляет*

$$\sum_{i<j} C_{i,j}\left(\frac{\arccos\left((u_i^*)^T u_j^*\right)}{\pi}\right) \geq 0.87856723 \sum_{i<j} C_{i,j}\left(\frac{1-(u_i^*)^T u_j^*}{2}\right) \geq$$

$$\geq 0.87856723(\mathrm{opt}(G) - \epsilon) \geq 0.878\ \mathrm{opt}(G),$$

если выбрать значение $\epsilon \leq 5 \times 10^{-4}$.

Получаем вероятностный 0.878-аппроксимационный алгоритм решения задачи о максимальном разрезе, который значительно превосходит по точности все негеометрические алгоритмы.

Вопрос о возможности построения более эффективных аппроксимационных алгоритмов для задачи о разрезе остается открытым. Известно, что задача нахождения приближенного решения для задачи о максимальном разрезе с точностью выше $\frac{16}{17} \approx 0.941$ является NP-трудной задачей. Если же верна так называемая «unique games conjecture» [24], то

полученное Гомансом и Уильямсоном приближение является наилучшим из тех, что можно найти за полиномиальное время.

Прорывной результат Гоманса и Уильямсона стимулировал значительный интерес к разработке аппроксимационных алгоритмов для задач комбинаторной оптимизации, основанных на идеях полуопределенных релаксаций. В частности, приближенные алгоритмы аналогичные *SDPMaxCut* были построены для таких задач комбинаторной оптимизации как раскраска 3-хроматического графа, максимизация квадратичной формы на графе и многих других. Более подробно с этой тематикой можно ознакомиться в [18].

Список литературы

1. *Barahona F., Grotschel M., Junger M. and Reinelt G.* An Application of Combinatorial Optimization to Statistical Physics and Circuit Layout Design // Operations Research. Vol. 3. 1988. pp. 493–513.
2. *Barahona F., Mahjoub A.R.* On the CUT polytope // Mathematical Programming. Vol. 36. 1986. pp. 157–173.
3. *Bremner D.* Incremental convex hull algorithms are not output sensitive. Discrete Comput. Geom. № 21. 1999. pp. 57–68.
4. *Caratheodory C.* Uber den Variabilit atsbereich der Fourier'schen Konstanten von positiven harmonischen Funktionen. Rendiconto del Circolo Matematico di Palermo, Vol. 32. 1911. 193–217. Reprinted in Constantin Carethedory, Gesammelte Mathematische Schriften, H. Tietze (ed.), C.H. Beck'sche Verlagsbuchhandlung. Munichen. 1955. pp. 78–110.
5. *Carlson J., Jaffe A., MA, and Wiles A.* The Millennium Prize Problems. AMS and Clay Mathematics Institute. 2006. 165 p.
6. *Cobham A.* The intrinsic computational difficulty of functions // Proc. Logic, Methodology, and Philosophy of Science II. North Holland. 1965. pp. 24–30.
7. *Cook S.* The Complexity of Theorem Proving Procedures // Proceedings of the third annual ACM symposium on Theory of computing. 1971. pp. 151–158.
8. *Cook S.* The P versus NP problem // Clay Mathematical Institute; The Millennium Prize Problem. 2000.
9. *Dantzig G.B., Fulkerson D.R., Johnson S.M.* Solution of a large-scale traveling salesman problem // Operations Research. Vol. 2, № 4. 1954. pp. 393–410.
10. *Dantzig G.B.* Programming in a linear structure // Econometrica. Vol. 17. 1949. pp. 73–74.
11. *Deza M.M., Laurent M.* Geometry of Cuts and Metrics (Algorithms and Combinatorics). Springer. 1997. 599 p.
    Имеется русский перевод:
    Деза М.М., Лоран М. Геометрия разрезов и метрик. М.: МЦНМО. 2001. 736 с.
12. *Dinic, E.A.* Algorithm for solution of a problem of maximum flow in a network with power estimation // Soviet Math. Doklady (Doklady) 11. 1970. pp. 1277–1280.

13. *Edmonds J., Karp R.M.* Theoretical improvements in algorithmic efficiency for network flow problems // Journal of the ACM (Association for Computing Machinery) 19 (2). 1972. pp. 248–264.
14. *Fiorini S., Massar S., Pokutta S., Tiwary H.R., Wolf R.* Exponential Lower Bounds for Polytopes in Combinatorial Optimization. Accepted for publication in Journal of ACM. arXiv:1111.0837.
15. *Ford, L.R., Jr., Fulkerson, D.R.* Maximal Flow through a Network // Canadian Journal of Mathematics. 1956. pp. 399-404.
16. *Gale D.* Neighboring vertices on a convex polyhedron // Linear inequalities and related system. Annals of Mathematics Studies. № 38. Princeton. 1956. pp. 255–263.
    Имеется русский перевод:
    Гейл Д. Соседние вершины на выпуклом многограннике // Линейные неравенства и смежные вопросы. М.: ИЛ, 1959. С. 355–362.
17. *Garey M.R., Johnson D.S.* Computers and Intractability: A Guide to the Theory of NP-Completeness. A Series of Books in the Mathematical Sciences. W. H. Freeman and Co. 1979. 338 p.
    Имеется русский перевод:
    Гэри М.Р., Джонсон Д.С. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982. 416 с.
18. *Gärtner B., Matousek J.* Approximation Algorithms and Semidefinite Programming. Springer. 2012. 251 p.
19. *Goemans M.X., Williamson D.P.* Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, Journal of the ACM (JACM). № 42. 1995. pp. 1115–1145.
20. *Groetschel M., Lovasz L., Schrijver A.* The ellipsoid method and its consequences in combinatorial optimization // Combinatorica. Vol. 1, № 2. Springer Berlin. 1984. pp. 169–197.
21. *Hadlock F.* Finding a Maximum Cut of a Planar Graph in Polynomial Time // SIAM Journal on Computing. Vol. 4, Issue 3. 1975. pp. 221–225.
22. *Hao J., Orlin J.B.* A Faster Algorithm for Finding the Minimum Cut in a Directed Graph // Journal of Algorithms, Vol. 17. 1994. pp. 424–446.
23. *Karmarkar N.A.* New Polynomial Time Algorithm for Linear Programming // Combinatorica. Vol. 4, № 4. Springer Berlin. 1984. pp. 373–395.
24. *Khot S.* On the power of unique 2-prover 1-round games // Proceedings of the thirty-fourth annual ACM symposium on Theory of computing. pp. 767–775.

25. *Klee V., Minty G.J.* How good is the simplex algorithm? // Inequalities. Vol. 3. Academic Press Inc. 1972. pp. 159–175.
26. *Monniaux D.* Quantifier elimination by lazy model enumeration // Computer aided verification (CAV). Lecture Notes in Computer Science. Vol. 6174. 2010. pp 585-599.
27. *Moshkov. M. Ju.* Time complexity of decision trees. In Transactions on Rough Sets III, James F. Peters and Andrzej Skowron (Eds.). Springer-Verlag, Berlin, Heidelberg. 2005. pp. 244–459.
28. *Nash J.C.* The (Dantzig) Simplex Method for Linear Programming // Computing in Science and Engineering. Vol. 2, No. 1. ACM New York, NY, USA. 2000. pp. 29–31.
29. *Padberg M.V.* The Boolean quadratic polytope: some characteristics, facets and relatives // Mathematical Program. V. 45. 1989. pp. 139–172.
30. *Papadimitriou C.H.* The adjacency relation on the traveling salesman polytope is NP-Complete // Mathematical Programming. 1978, Vol. 14, № 1, pp 312–324.
31. SMAPO – library of linear descriptions of low-dimensional 0/1-polytopes connected with small instances of combinatorial optimization problems. Universität Heidelberg. http://www.iwr.uni-heidelberg.de/groups/comopt/software/SMAPO/
32. *Stoer M., Wagner F.* A Simple Min Cut Algorithm // Algorithms–ESA '94, LNCS 855. 1994. pp. 141–147.
33. *Ziegler G.M.* Lectures on 0-1 polytopes // Polytopes - Combinatorics and Computation (G. Kalai and G.M. Ziegler, eds.), DMV Seminars Series. Birkhauser, Basel. 2000. 45 p.
34. *Ziegler G.M.* Lectures on Polytopes. Springer. 1995. 370 p.
35. *Бондаренко В.А., Максименко А.Н.* Геометрические конструкции и сложность в комбинаторной оптимизации. М.: ЛКИ, 2008. 184 с.
36. *Бондаренко В.А., Николаев А.В.* Комбинаторно-геометрические свойства задачи о разрезе. Доклады академии наук. Т.452, №2, 2013, С. 127–129. *Bondarenko V.A., Nikolaev A.V.* Combinatorial and Geometric Properties of the Max-Cut and Min-Cut Problems. Doklady Mathematics. 2013. Vol. 88, № 2, pp. 516–517.
37. *Бондаренко В.А.* Об одном комбинаторном многограннике // Моделирование и анализ вычислительных систем. Сб. науч. тр. Ярославль: Яросл. гос. ун-т., 1987. С. 133–134.

38. *Бондаренко В.А., Урываев Б.В.* Об одной задаче целочисленной оптимизации // Автоматика и телемеханика. 2007 №6. С. 18–23.
Bondarenko V.A., Uryvaev B.V. On one problem of integer optimization // Automation and Remote Control. 2007. Vol. 68. № 6. pp 948–953.
39. *Канторович Л.В.* Математические методы организации и планирования производства. ЛГУ. 1939. 56 с.
40. *Левин Л.* Универсальные задачи перебора // Проблемы передачи информации. Т. 9, № 3. 1973. С. 115–116.
41. *Хачиян Л.Г.* Полиномиальные алгоритмы в линейном программировании // ЖВМ и МФ. Т. 20, №1. 1980. С. 51–69.