Н. М. Новикова

ОСНОВЫ ОПТИМИЗАЦИИ
(курс лекций)

МОСКВА 1998

Ответственный редактор
академик РАН П. С. Краснощеков

В сжатой форме дается изложение основ теории сложности, линейного программирования (ЛП) — с описанием полиномиальных алгоритмов, целочисленного ЛП, математического программирования (необходимые условия экстремума при ограничениях-неравенствах, локальные методы безусловной оптимизации, метод штрафов, идеи глобальной оптимизации), схем методов динамического программирования и ветвей и границ.

Работа написана на базе семестрового курса лекций, читаемого автором студентам 4-го курса программистского потока факультета ВМиК МГУ, с учетом дополнений и замечаний, указанных студентами. Автор благодарит всех студентов, содействовавших изданию этого курса и предложивших исправления, способствующие его улучшению, в том числе, Ласкавого Сергея, Санникова Андрея и Свахина Николая. Замеченные опечатки и неточности просьба сообщать автору по адресу nnovik@ccas.ru

Работа частично поддержана грантом РФФИ No.96-01-00786.

Рецензенты: С. К. Завриев,
А. В. Лотов

Часть 1. ВВЕДЕНИЕ В ТЕОРИЮ СЛОЖНОСТИ

Литература:

1. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.

2. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. М.: Мир, 1985.

§1. Понятие о сложности решения задач

Основные определения: индивидуальная и массовая задачи, кодировка, алгоритм решения массовой задачи, временна́я сложность алгоритма. Классы **P** *и* **NP** *(формальные определения, примеры).*

1. На вопрос, для чего надо иметь представление о сложности решаемых задач, наиболее наглядный ответ дан во введении к книге [1]. В этой книге также приводится более 500 задач (из самых разных областей, включая теорию графов и сетей, теорию расписаний, теорию автоматов и языков, оптимизацию программ, базы данных, игры и головоломки и т.п.), для которых в настоящее время нет оснований надеяться построить эффективные алгоритмы их решения. Что это значит формально, будет рассказано в данном разделе (§§1–4), а соответствующие практические выводы каждый человек, так или иначе связанный с разработкой алгоритмов и программ, делает для себя сам. Кроме того, теория сложности — новая, модная, интенсивно развивающаяся область математики и кибернетики, ее терминология широко распространена в современной научной литературе и требует определенного с ней знакомства.

Появление вычислительной техники привело к тому, что все реже приходится решать отдельную конкретную задачу, а все больше писать программы, рассчитанные на целый класс задач, получающихся одна из другой заменой ряда исходных данных. Поэтому имеет смысл говорить о сложности не для одной *индивидуальной* задачи **I**, а для *массовой задачи, или проблемы* **П**, соответствующей множеству индивидуальных задач.

Формально массовая задача **П** определяется

1^0 общим списком всех параметров задачи (свободных параметров, значения которых не заданы),

2^0 формулировкой свойств, которым должен удовлетворять ответ (решение задачи).

Индивидуальная задача $\mathbf{I} \in \mathbf{\Pi}$ получается из $\mathbf{\Pi}$, если всем параметрам присвоить конкретные значения.

Для примера рассмотрим задачу коммивояжера: найти минимальный маршрут обхода группы объектов (условно говоря, городов) с возвратом в начальную точку. Для $\mathbf{\Pi}$ коммивояжера введем

1^0 входные параметры: число городов m или множество городов $C = \{c_1, \ldots, c_m\}$ и набор расстояний между городами

$$\{d(c_i, c_j) > 0 : \quad c_i, c_j \in C, \quad i \neq j\};$$

2^0 требования к решению: $[c_{\pi(1)}, \ldots, c_{\pi(m)}]$ реализует

$$\min_{\pi} \left[\sum_{i=1}^{m-1} d(c_{\pi(i)}, c_{\pi(i+1)}) + d(c_{\pi(m)}, c_{\pi(1)}) \right],$$

где минимум берется по всем возможным перестановкам π на множестве индексов городов. Конкретизируем параметры 1^0, чтобы получить индивидуальную задачу $\mathbf{I}$ коммивояжера: $m = 4$, $d(c_1, c_2) = 10$, $d(c_1, c_3) = 5$, $d(c_1, c_4) = 9$, $d(c_2, c_4) = 9$, $d(c_3, c_4) = 3$, $d(c_3, c_2) = 6$, $d(c_i, c_j) = d(c_j, c_i)$. Тогда в задаче $\mathbf{I}$ оптимальным оказывается маршрут $[c_1, c_2, c_4, c_3]$, реализующий путь минимальной длины 27.

Кроме первичных понятий массовой и индивидуальной задачи ($\mathbf{\Pi}$ и $\mathbf{I}$) мы будем использовать термин алгоритм и обозначение $\mathbf{A}$ для пошаговой процедуры (решения задачи), в частности машины Тьюринга или программы для ЭВМ. Будем говорить, что *алгоритм* $\mathbf{A}$ *решает массовую задачу* $\mathbf{\Pi}$, если для любой индивидуальной задачи $\mathbf{I} \in \mathbf{\Pi}$ алгоритм $\mathbf{A}$ применим к $\mathbf{I}$ (т.е. останавливается за конечное число шагов) и $\forall \mathbf{I} \in \mathbf{\Pi}$ алгоритм $\mathbf{A}$ дает решение задачи $\mathbf{I}$. Например, для $\mathbf{\Pi}$ коммивояжера существует алгоритм, который решает ее на основе полного перебора всех маршрутов (перестановок π).

Большинство дискретных и комбинаторных задач допускает решение с помощью некоторого процесса перебора вариантов, однако число возможных вариантов растет экспоненциально в зависимости от размеров задачи (так, в задаче коммивояжера $m!$ маршрутов).

Поэтому переборные алгоритмы решения массовых задач считаются неэффективными (могут решать лишь небольшие индивидуальные задачи). В отличие от них эффективными называются *полиномиальные алгоритмы* решения массовой задачи, т.е. такие, которые решают произвольную $\mathbf{I} \in \mathbf{\Pi}$ за время, ограниченное полиномом от "размера" $\mathbf{I}$. Несмотря на определенную условность этого разделения с точки зрения практического счета, оно объясняется прежде всего тем, что центральным для дискретной оптимизации является вопрос, можно ли построить алгоритм решения массовой задачи (т.е. любой индивидуальной), не перебирающий всех или почти всех вариантов ее решения. Если для массовой задачи $\mathbf{\Pi}$ существует полиномиальный алгоритм, ее решающий, значит, ее можно решить не путем перебора — эффективно. Указанные задачи $\mathbf{\Pi}$ называются полиномиальными. Перейдем к их формальному определению.

2. Формализация проводится для *задач распознавания свойств*. Это — массовые задачи, предполагающие ответ "да" или "нет" в качестве решения. Таким образом, в п.2^0 определения $\mathbf{\Pi}$ распознавания свойств стоит некоторый альтернативный вопрос и решением каждой индивидуальной задачи $\mathbf{I} \in \mathbf{\Pi}$ является правильное распознавание, принадлежит ли она к задачам с ответом "да". Последнее подмножество множества индивидуальных задач будем обозначать $\mathbf{Y}$. Теперь введем обозначение $\mathbf{D}$ для множества всех возможных значений параметров, заданных в п.1^0 определения $\mathbf{\Pi}$. Очевидно, что набор $[\mathbf{D(\Pi)},\mathbf{Y(\Pi)}]$ полностью характеризует соответствующую массовую задачу $\mathbf{\Pi}$ распознавания свойств. Несмотря на специфичность постановки, класс задач распознавания свойств является достаточно широким: по крайней мере, для любой задачи дискретной оптимизации можно указать аналогичную $\mathbf{\Pi}$ распознавания свойств. В частности, для $\mathbf{\Pi}$ коммивояжера, если ввести в п.1^0 еще один параметр B — длину маршрута, то вопрос в п.2^0 "существует ли маршрут длины, не превышающей B?" дает ее переформулировку как задачи распознавания свойств. Полученная $\mathbf{\Pi}$ коммивояжера имеет в литературе обозначение $\mathbf{KM}$ (или ЗК [2]), для нее

$$\mathbf{D(KM)} = \{C,\ \ \{d(c_i, c_j) \in \mathbf{Z}_+ |\ c_i, c_j \in C,\ i < j\},\ \ B \in \mathbf{Z}_+\}.$$

Здесь и далее $\mathbf{Z}_+$ — множество натуральных чисел, $\mathbf{Z}$ — целых.

Для формализации "размера" индивидуальной задачи свяжем с

каждой проблемой $\mathbf{\Pi}$ определенную *схему кодирования (кодировку)*. Введем конечное множество — *алфавит* $\Sigma = \{\sigma_i\}$, например $\Sigma = \{0,\ 1\}$, а также множество Σ^* *слов над алфавитом* Σ — произвольных конечных последовательностей, составленных из символов алфавита, возможно повторяющихся, $\sigma = \sigma_{i_1}\sigma_{i_2}\ldots\sigma_{i_n},\quad \sigma_{i_j} \in \Sigma\ \ \forall i_j$; например, пустое множество $\emptyset$ или 011000. Число n называется *длиной слова* σ и обозначается знаком модуля, $n = |\sigma|$. *Кодировкой задачи* $\mathbf{\Pi}$ назовем такое отображение e: $\mathbf{\Pi} \to \Sigma^*$, ставящее в соответствие любой индивидуальной задаче $\mathbf{I} \in \mathbf{\Pi}$ ее код $e(\mathbf{I}) = \sigma \in \Sigma^*$ (слово из алфавита Σ^*), что

1^* возможно однозначное декодирование: $\forall \mathbf{I}_1 \neq \mathbf{I}_2\ \ e(\mathbf{I}_1) \neq e(\mathbf{I}_2)$;

2^* $e,\ e^{-1}$ полиномиально вычислимы: существует алгоритм, реализующий $e,\ e^{-1}$, и полином $p(\cdot)$, для которого $\forall \mathbf{I} \in \mathbf{\Pi}$ время определения $e(\mathbf{I})$ и $e^{-1}(e(\mathbf{I}))$ не превосходит $p(|e(\mathbf{I})|)$;

3^* кодировка неизбыточна: для любой другой кодировки e', удовлетворяющей условиям 1^*,2^*, найдется полином $p'(\cdot)$ такой, что $\forall \mathbf{I} \in \mathbf{\Pi}\ \ |e(\mathbf{I})| < p'(|e'(\mathbf{I})|)$. Например, для записи целых чисел неизбыточной является любая k-ичная система счисления с $k > 1$, кодировка чисел тем же количеством палочек (случай $k = 1$) избыточна.

УПРАЖНЕНИЕ 1. Предложить неизбыточную кодировку и оценить по порядку длину входа задачи коммивояжера, сравнить полученную оценку с указанной в [1] на с. 35:

$$m + \lceil \log_2 B \rceil + \max\{\lceil \log_2 d(c_i, c_j)\rceil |\ c_i, c_j \in C\}.$$

Здесь и далее знаком $\lceil\cdot\rceil$ обозначается ближайшее целое сверху к числу в скобках, а $\lfloor\cdot\rfloor$ — целая часть числа.

Начиная с этого момента, в §§ 1–3 мы будем, как правило, рассматривать $\mathbf{\Pi}$ распознавания свойств, оговаривая другие случаи особо.

После того как для массовой задачи $\mathbf{\Pi}$ введена кодировка, с любой индивидуальной задачей $\mathbf{I} \in \mathbf{\Pi}$ будет связано некоторое слово σ в алфавите Σ этой кодировки. Слова, которые соответствуют индивидуальным задачам распознавания свойств, имеющим ответ “да”, условимся считать “правильными” и множество правильных слов в Σ^* назовем *языком*. Формально, язык $L(\mathbf{\Pi},e) \doteq e(\mathbf{Y}(\mathbf{\Pi}))\doteq$

$$\doteq \{\sigma \in \Sigma^* |\ \Sigma \text{ — алфавит } e,\quad \sigma = e(\mathbf{I}),\quad \mathbf{I} \in \mathbf{Y}(\mathbf{\Pi})\}.$$

С алгоритмом $\mathbf{A}$ решения задачи $\mathbf{\Pi}$ распознавания свойств будем ассоциировать машину Тьюринга (программу для детермини-

рованной машины Тьюринга) с входным алфавитом Σ и конечными состояниями q_Y ("да") и q_N ("нет") и аналогично назовем языком алгоритма $\mathbf{A}$ множество слов, *принимаемых* $\mathbf{A}$ (с которыми на входе $\mathbf{A}$ останавливается в состоянии q_Y — "да"),

$$L(\mathbf{A}) \doteq \{\sigma \in \Sigma^* |\ \Sigma \text{ — алфавит } \mathbf{A}, \text{ и } \mathbf{A}(\sigma) = q_Y\}.$$

Определение 1. Алгоритм $\mathbf{A}$ *решает* массовую задачу $\mathbf{\Pi}$ с кодировкой e, если $L(\mathbf{A}) = L(\mathbf{\Pi},e)$ и $\forall\sigma \in \Sigma^*$ $\mathbf{A}$ останавливается. Обозначим $t_{\mathbf{A}}(\sigma)$ время работы над словом $\sigma \in \Sigma^*$ (число шагов) алгоритма $\mathbf{A}$ до остановки. *Временно́й сложностью* алгоритма $\mathbf{A}$ решения массовой задачи $\mathbf{\Pi}$ назовем функцию $T_{\mathbf{A}}(\cdot)$, определяемую как

$$T_{\mathbf{A}}(n) = \max_{\sigma\in\Sigma^*:\ |\sigma|<n} t_{\mathbf{A}}(\sigma) \quad \forall n \in \mathbf{Z}_+.$$

Таким образом, при оценке временно́й сложности алгоритмов мы рассчитываем на "худшую" из возможных индивидуальных задач (данного размера), поскольку заранее не известно, с какой конкретной задачей придется работать.

УПРАЖНЕНИЕ 2. Дать алгоритм распознавания простоты числа, оценить временну́ю сложность алгоритма.

Определение 2. *Класс полиномиально разрешимых задач* $\mathbf{P} \doteq \{L(\mathbf{\Pi},e)|\ \exists\mathbf{A}$, решающий $\mathbf{\Pi}$ с кодировкой e, $\exists p(\cdot)$ — полином: $T_{\mathbf{A}}(n) < p(n) \quad \forall n \in \mathbf{Z}_+\}$.

Если для задачи $\mathbf{\Pi}$ существует такая кодировка e, что $L(\mathbf{\Pi},e) \in \mathbf{P}$, то будем называть задачу $\mathbf{\Pi}$ *полиномиально разрешимой* или просто *полиномиальной* и пользоваться обозначением $\mathbf{\Pi}\in\mathbf{P}$, отождествляя массовую задачу и язык. (С учетом условия неизбыточности кода указанная процедура корректна, ибо для полиномиальной $\mathbf{\Pi}$ получаем $L(\mathbf{\Pi},e) \in \mathbf{P}$ $\forall e$.)

Примером полиномиальной задачи является распознавание четности целого числа. (С еще одной полиномиальной задачей мы встретимся в разд.2.) Для задачи распознавания простоты числа (**ПЧ**) вопрос о ее полиномиальности пока открыт. Для ряда других задач удается доказать их неполиномиальность. Так, известны

1) алгоритмически неразрешимые задачи, когда не существует алгоритма, решающего любую индивидуальную задачу, т.е. $\forall\mathbf{A}$ $\exists\mathbf{I} \in \mathbf{\Pi}$: $\mathbf{A}$ не применим к $\mathbf{I}$, в частности, $t_A(e(\mathbf{I})) = \infty$; например, 10-я проблема Гильберта: по данному многочлену g с целыми

коэффициентами выяснить, имеет ли уравнение $g = 0$ целочисленное решение (неразрешимость доказал Ю. М. Матиясевич в 1970 г.);

2) задачи (не являющиеся задачами распознавания свойств), для которых длина записи решения превосходит любой наперед заданный полином от длины входа, например в задаче коммивояжера, если требуется найти все маршруты (их экспоненциальное число);

3) в остальных случаях формально имеем $\forall \mathbf{A}$, решающего $\mathbf{\Pi}$ с кодировкой e, $\forall p(\cdot)\ \exists \mathbf{I} \in \mathbf{\Pi}$: $t_{\mathbf{A}}(e(\mathbf{I})) > p(|e(\mathbf{I})|)$. Здесь и далее $p(\cdot)$, возможно с индексами, служит для обозначения полиномов.

В настоящее время для любой массовой задачи $\mathbf{\Pi}$, для которой доказано последнее условие, получен и более сильный результат: отсутствие полиномиального алгоритма, использующего произвольное (пусть бесконечное) число параллельных процессоров. Вопрос, существуют ли неполиномиальные задачи $\mathbf{\Pi}$ распознавания свойств, которые оказываются полиномиально разрешимыми при возможности распараллеливания вычислений, является *основной* методологической *проблемой* теории сложности (обусловившей ее формирование как самостоятельной научной дисциплины). Ответ, по-видимому, должен быть положительным, и уже указан большой класс массовых задач в качестве кандидатов (см. класс $\mathbf{NPC}$ в §2), но доказать или опровергнуть эту гипотезу в данный момент представляется нереальным. Для ее формализации вводится объемлющий $\mathbf{P}$ класс *недетерминированно полиномиальных задач* — $\mathbf{NP}$.

3. Определим *недетерминированную машину Тьюринга* (НДМТ) $\hat{\mathbf{A}}$ как набор обычных — детерминированных — машин Тьюринга (ДМТ) $\mathbf{A}(S)$ с алфавитом Σ, где S пробегает все множество слов из Σ^*:

$$\hat{\mathbf{A}} \doteq \{\mathbf{A}(S)\}_{S \in \Sigma^*}.$$

НДМТ $\hat{\mathbf{A}}$ останавливается, когда останавливается первая из ДМТ $\mathbf{A}(S)$, принимающая входное слово. Соответствующим конечным состоянием будет q_Y. *Язык НДМТ* — множество слов, принимаемых хотя бы одной ДМТ $\mathbf{A}(S)$ из $\hat{\mathbf{A}}$:

$$\hat{L}(\hat{\mathbf{A}}) \doteq \{\sigma \in \Sigma^* |\ \exists S \in \Sigma^* :\ \ \sigma \in L(\mathbf{A}(S))\}.$$

Слова S в определении НДМТ можно проинтерпретировать как подсказки к решению (догадки), тогда ДМТ $\mathbf{A}(S)$ проверяет для

входного слова σ подсказку S и в случае правильности останавливается в состоянии q_Y. НДМТ $\hat{\mathbf{A}}$ проверяет для входного слова σ все возможные подсказки, и если хоть одна правильная догадка существует, то НДМТ останавливается с ответом "да". (В силу бесконечности числа догадок, в состоянии q_N НДМТ остановиться не может.)

Определение 3. НДМТ $\hat{\mathbf{A}}$ *решает* массовую задачу $\mathbf{\Pi}$ с кодировкой e, если $L(\mathbf{\Pi},e) = \hat{L}(\hat{\mathbf{A}})$, т.е. языки НДМТ и задачи совпадают: $\forall\sigma \in L(\mathbf{\Pi},e)\ \exists S \in \Sigma^*$: ДМТ $\mathbf{A}(S)$ останавливается в состоянии q_Y, и $\forall\sigma \in \Sigma^* \setminus L(\mathbf{\Pi},e)$, $\forall S \in \Sigma^*$ ДМТ $\mathbf{A}(S)$ не принимает σ (не останавливается или останавливается в состоянии q_N).

Определим $\forall\sigma \in \hat{L}(\hat{\mathbf{A}})$ *время работы НДМТ* $\hat{\mathbf{A}}$ над словом σ как минимальное из времен работы над входом σ ДМТ $\mathbf{A}(S)$, принимающих σ, с учетом времени прочтения слова S (т.е. его длины):

$$\hat{t}_{\hat{\mathbf{A}}}(\sigma) \doteq \min_{\{S|\ \sigma \in L_{\mathbf{A}(S)}\}} \{|S|\ + t_{\mathbf{A}(S)}(\sigma)\}.$$

Временно́й сложностью НДМТ $\hat{\mathbf{A}}$ решения массовой задачи $\mathbf{\Pi}$ назовем функцию $\hat{T}_{\hat{\mathbf{A}}}(\cdot)$: $\quad \forall n \in \mathbf{Z}_+$

$$\hat{T}_{\hat{\mathbf{A}}}(n) = \max_{\sigma \in \hat{L}(\hat{\mathbf{A}}):\ |\sigma|<n} \hat{t}_{\hat{\mathbf{A}}}(\sigma) \doteq \max_{\sigma \in \hat{L}(\hat{\mathbf{A}}):\ |\sigma|<n}\ \min_{\{S|\sigma \in L_{\mathbf{A}(S)}\}} \{|S| + t_{\mathbf{A}(S)}(\sigma)\}.$$

Подчеркнем разницу с определением временно́й сложности ДМТ: для НДМТ рассматриваются лишь слова из языка (соответствующие индивидуальным задачам с ответом "да").

Определение 4. *Класс недетерминированно полиномиальных задач* $\mathbf{NP} \doteq \{L(\mathbf{\Pi},e)|\ \exists\hat{\mathbf{A}}$ — НДМТ, решающая $\mathbf{\Pi}$ с кодировкой e, $\exists p(\cdot)$ — полином: $\hat{T}_{\hat{\mathbf{A}}}(n) < p(n) \quad \forall n \in \mathbf{Z}_+\}$. Если для задачи $\mathbf{\Pi}$ существует такая кодировка e, что $L(\mathbf{\Pi},e) \in \mathbf{NP}$, то будем называть задачу $\mathbf{\Pi}$ *недетерминированно полиномиальной* и пользоваться обозначением $\mathbf{\Pi}\in\mathbf{NP}$ (как и для класса $\mathbf{P}$, корректным).

Примером недетерминированно полиномиальной задачи является $\mathbf{KM}$, ибо в качестве догадки можно использовать маршрут и проверка его допустимости полиномиальна.

Отметим, что полиномиальность проверки гарантируется только для индивидуальных задач с ответом "да" (и возможно, лишь при

единственной подсказке), а для $\mathbf{I} \in \mathbf{D}(\mathbf{\Pi}) \setminus \mathbf{Y}(\mathbf{\Pi})$ НДМТ просто не остановится. В этом — существенное отличие классов **P** и **NP**. Непосредственно из определений следует

Утверждение 1. $\mathbf{P} \subseteq \mathbf{NP}$.

Вопрос о наличии строгого включения и является формализацией основной проблемы теории сложности.

§2. NP-полные (универсальные) задачи

Теорема об экспоненциальной оценке временно́й сложности для задач из класса **NP**. *Класс* **co-NP**. *Задачи, имеющие хорошую характеризацию. Определение полиномиальной сводимости. Класс* **NPC**. *Теорема Кука (без доказательства). Критерий* **NP***-полноты. Доказательство* **NP***-полноты задачи* **БЛН** *(булевы линейные неравенства).*

1. Рассмотрим подробнее класс **NP**.

Теорема 1. Для любой недетерминированно полиномиальной задачи существует ДМТ, решающая ее с экспоненциальной временной сложностью, т.е. $\forall \mathbf{\Pi} \in \mathbf{NP}$ $\exists p(\cdot)$ — полином — и ДМТ **A**:

$$\mathbf{A} \text{ решает } \mathbf{\Pi} \text{ и } T_{\mathbf{A}}(n) < 2^{p(n)} \quad \forall n \in \mathbf{Z}_+.$$

Доказательство. Так как $\mathbf{\Pi} \in \mathbf{NP}$, то для любого слова σ из языка задачи $\mathbf{\Pi}$ существует правильная догадка S полиномиальной длины: $|S| < p_1(|\sigma|)$, $p_1(\cdot)$ — полином, и существует ДМТ $\mathbf{A}(S)$: $t_{\mathbf{A}(S)}(\sigma) < p_2(|\sigma|)$, $p_2(\cdot)$ — полином. Построим ДМТ **A**, которая работает над любым входным словом $\sigma \in \Sigma^*$ (с любой индивидуальной задачей $\mathbf{I} \in \mathbf{\Pi}$) следующим образом: рассматриваются все слова S из Σ^* длины меньше $p_1(|\sigma|)$ и делается не более $p_2(|\sigma|)$ шагов с каждой ДМТ $\mathbf{A}(S)$. Если очередная ДМТ останавливается в состоянии q_Y (т.е. соответствующая догадка оказалась правильной), считаем слово σ принятым и работу ДМТ **A** законченной; если ни одна из ДМТ $\mathbf{A}(S)$ не остановилась за отведенное время или остановилась в состоянии q_N, то заканчиваем работу ДМТ **A** и приписываем ей конечное состояние q_N. В последнем случае ДМТ **A** делает наибольшее число шагов, и это число меньше $p_2(|\sigma|) \cdot |\Sigma|^{p_1(|\sigma|)}$ (второй сомножитель равен числу проверяемых догадок, $|\Sigma|$ — число символов в алфавите Σ). Отсюда уже нетрудно получить утверждение теоремы.

Для того чтобы лучше почувствовать различие классов **P** и **NP**, введем понятие *дополнительной к* **П** массовой задачи $\overline{\mathbf{П}}$, получающейся из **П** распознавания свойств заменой альтернативного вопроса, определяющего ответ в задаче (см. п.2^0 определения **П** в §1) его отрицанием, например вопросом в $\overline{\mathbf{КМ}}$ "правда ли, что не существует маршрута длины, не превосходящей B?". Формально $\mathbf{D}(\overline{\mathbf{П}}) = \mathbf{D}(\mathbf{П})$, $\mathbf{Y}(\overline{\mathbf{П}}) = \mathbf{D}(\mathbf{П}) \setminus \mathbf{Y}(\mathbf{П})$.

Определим классы дополнительных задач $\mathbf{co\text{-}P} \doteq \{\overline{\mathbf{П}} |\ \mathbf{П} \in \mathbf{P}\}$ и $\mathbf{co\text{-}NP} \doteq \{\overline{\mathbf{П}} |\ \mathbf{П} \in \mathbf{NP}\}$. Из определений очевидно, что, если ДМТ **A** решает **П**, то ДМТ $\overline{\mathbf{A}}$ решает $\overline{\mathbf{П}}$, где программа ДМТ $\overline{\mathbf{A}}$ получена из программы ДМТ **A** простой заменой конечных состояний q_Y и q_N друг на друга. Таким образом, справедливо

Утверждение 2. $\mathbf{co\text{-}P} = \mathbf{P}$.

Аналогичное утверждение для класса **NP** до сих пор не удается ни доказать ни опровергнуть: приведенное выше для ДМТ рассуждение нельзя обобщить на НДМТ, ибо для индивидуальных задач **I** с ответом "нет" (т.е. $\mathbf{I} \notin \mathbf{Y}(\mathbf{П})$, или $\mathbf{I} \in \mathbf{Y}(\overline{\mathbf{П}})$) НДМТ не останавливается за время, ограниченное полиномом от длины входа **I**. В частности, не известна НДМТ, решающая $\overline{\mathbf{КМ}}$ за полиномиальное время, так как для нее не придумано подсказки полиномиальной длины (естественный вариант — показать все маршруты — не полиномиален); включение $\overline{\mathbf{КМ}} \in \mathbf{NP}$ не доказано и не опровергнуто.

УПРАЖНЕНИЕ 3. Доказать, что задача распознавания простоты числа принадлежит классу co-NP, т.е. $\overline{\mathbf{ПЧ}} \in \mathbf{NP}$.

Определение 5. Массовая задача распознавания свойств называется *имеющей хорошую характеризацию*, если для нее выполнено $\mathbf{П} \in \mathbf{NP} \cap \mathbf{co\text{-}NP}$.

Из утверждения 2 следует, что $\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$. Современная гипотеза состоит в равенстве этих классов. Отсюда и термин "хорошая характеризация", так как для подобных задач есть основания надеяться на возможность построения полиномиальных алгоритмов (см. задачу **ЛН** — линейные неравенства — в разд.2). Однако для задачи **ПЧ**, обладающей хорошей характеризацией (для доказательства того, что $\mathbf{ПЧ} \in \mathbf{NP}$, см. [2, с. 414]), детерминированного полиномиального алгоритма пока не найдено, несмотря на ее непосредственную практическую значимость.

2. Задач распознавания свойств — большое разнообразие, и для теории представляет интерес не только возможность их классификации, но и способы определения класса сложности одних задач на основе известного класса сложности других. Поэтому вводится базовое для теории сложности понятие *полиномиальной сводимости.*

Определение 6. Будем говорить, что массовая задача распознавания свойств $\mathbf{\Pi}'$ с кодировкой e' *полиномиально сводится* к задаче $\mathbf{\Pi}$ с кодировкой e, если любая индивидуальная задача $\mathbf{I}' \in \mathbf{\Pi}'$ может быть сведена за полиномиальное от ее длины время к некоторой $\mathbf{I} \in \mathbf{\Pi}$ с сохранением ответа. Формально

существует функция сводимости $f:\ e'(\mathbf{D}(\mathbf{\Pi}')) \to e(\mathbf{D}(\mathbf{\Pi}))$, такая что $f(e'(\mathbf{Y}(\mathbf{\Pi}'))) = e(\mathbf{Y}(\mathbf{\Pi}))$, т.е. $\forall \sigma' \in e'(\mathbf{Y}(\mathbf{\Pi}'))\ f(\sigma') \in e(\mathbf{Y}(\mathbf{\Pi}))$ и $\forall \sigma'' \in e'(\mathbf{D}(\mathbf{\Pi}') \setminus \mathbf{Y}(\mathbf{\Pi}))\ \ f(\sigma'') \in e(\mathbf{D}(\mathbf{\Pi}) \setminus \mathbf{Y}(\mathbf{\Pi}))$,

и существует ДМТ $\mathbf{A_f}$, реализующая f за полиномиальное время, т.е. $\exists p_f(\cdot)$ — полином: $\forall \sigma \in e'(\mathbf{D}(\mathbf{\Pi}'))\ \ T_{\mathbf{A_f}}(|\sigma|) < p_f(|\sigma|)$.
В случае, когда соответствующие кодировки не избыточны, будем использовать термин полиномиальной сводимости по отношению к самим задачам (без указания кодировок) и применять обозначение

$$\mathbf{\Pi}' \propto \mathbf{\Pi}.$$

(Корректность упрощения вытекает из полиномиальной сводимости задачи к самой себе, но с другой неизбыточной кодировкой, и следующего очевидного утверждения — транзитивности отношения $\propto$.)

Утверждение 3. Если $\mathbf{\Pi}_1 \propto \mathbf{\Pi}_2$ и $\mathbf{\Pi}_2 \propto \mathbf{\Pi}_3$, то $\mathbf{\Pi}_1 \propto \mathbf{\Pi}_3$.

Существенным для теории сложности является

Утверждение 4. Если $\mathbf{\Pi}' \propto \mathbf{\Pi}$ и $\mathbf{\Pi} \in \mathbf{P}$, то и $\mathbf{\Pi}' \in \mathbf{P}$.

Доказательство. Обозначим $\mathbf{A}$ ДМТ, решающую $\mathbf{\Pi}$ с полиномиальной временной сложностью, и построим ДМТ $\mathbf{A}'$, решающую $\mathbf{\Pi}'$ с полиномиальной временной сложностью, как суперпозцию ДМТ $\mathbf{A}$ и $\mathbf{A_f}$: $\mathbf{A}' = \mathbf{A} \circ \mathbf{A_f}$, т.е. сначала к любому входному слову $\sigma' \in e'(\mathbf{D}(\mathbf{\Pi}'))$ применяется $\mathbf{A_f}$, а потом к получившемуся слову $\sigma = f(\sigma')$ (длиной не более $p_f(|\sigma'|)$) применяется $\mathbf{A}$. Временная сложность $\mathbf{A}'$ — $T_{\mathbf{A}'}(\cdot) \le T_{\mathbf{A_f}}(\cdot) + T_{\mathbf{A}}(p_f(\cdot))$ — полином.

Аналогично доказывается (при замене слова ДМТ на НДМТ)

Утверждение 5. Если $\mathbf{\Pi}' \propto \mathbf{\Pi}$ и $\mathbf{\Pi} \in \mathbf{NP}$, то и $\mathbf{\Pi}' \in \mathbf{NP}$.

Определение 7. Массовая задача $\mathbf{\Pi}$ называется $\mathbf{NP}$-*полной или универсальной*, если $\mathbf{\Pi} \in \mathbf{NP}$ и $\forall \mathbf{\Pi}' \in \mathbf{NP} \quad \mathbf{\Pi}' \propto \mathbf{\Pi}$ (т.е. любая недетерминированно полиномиальная задача полиномиально сводится к $\mathbf{\Pi}$). Класс всех $\mathbf{NP}$-полных задач (распознавания свойств) обозначается $\mathbf{NPC}$ ($\mathbf{NP}$-complete).

Непустоту класса $\mathbf{NPC}$ доказал С. А. Кук в 1971 г. Им была рассмотрена задача о выполнимости (**ВЫП**): выяснить выполнимость конъюнктивной нормальной формы (КНФ) — конъюнкции конечного числа дизъюнктивных функций, т.е. дизъюнкций булевых переменных z_i или их отрицаний $\overline{z_i}$. А именно, в задаче **ВЫП** требуется распознать для КНФ на входе, существует ли выполняющий набор z^0 (для которого значение КНФ равно 1).

Теорема 2 (S. A. Cook). $\mathbf{ВЫП} \in \mathbf{NPC}$.

Доказательство полиномиальной сводимости к **ВЫП** любой недетерминированно полиномиальной задачи основано на формальной записи условия принадлежности слова σ языку из класса $\mathbf{NP}$ (того, что σ принимается некоторой НДМТ, а значит, и какой-то ДМТ) в виде набора дизъюнктивных функций от специально вводимых булевых переменных, связанных с состояниями ДМТ в различные моменты времени, и является недостаточно простым для вводного курса (см. [1,2]). Поэтому мы лишь убедимся в том, что $\mathbf{ВЫП} \in \mathbf{NP}$. Действительно, входное слово (параметры, определяющие индивидуальную задачу выполнимости) содержит число дизъюнктивных функций в КНФ и указание для каждой из них, какие переменные входят с отрицанием, а какие не входят вообще. Длину такого слова можно ограничить снизу суммой длин дизъюнктивных функций, понимая под длиной функции число ее переменных (или число знаков дизъюнкции + 1). Если теперь в качестве подсказки для определяемой входным словом КНФ взять z^0 — выполняющий ее набор, то вычисление на нем значения КНФ (проверка выполнимости) потребует такого же по порядку числа шагов.

Из определения $\mathbf{NP}$-полноты непосредственно следует

Утверждение 6. Если $\mathbf{P} \cap \mathbf{NPC} \neq \emptyset$, то $\mathbf{P} = \mathbf{NP}$. А если $\mathbf{NPC} \cap (\mathbf{NP} \setminus \mathbf{P}) \neq \emptyset$, то $\mathbf{NPC} \subseteq \mathbf{NP} \setminus \mathbf{P}$.

Таким образом, если бы удалось найти полиномиальный алгоритм решения хоть одной $\mathbf{NP}$-полной задачи, то были бы построены

полиномиальные алгоритмы решения всех **NP**-полных задач и всех задач из класса **NP**, а если для какой-либо **NP**-полной задачи доказать отсутствие полиномиального алгоритма ее решения, то это не только дает строгое включение $\mathbf{P} \subset \mathbf{NP}$ (т.е. ответ к основной проблеме теории сложности), но и влечет за собой доказательство невозможности построения полиномиального алгоритма решения любой задачи из класса **NPC**. Поскольку ни того, ни другого пока не сделано, считается, что задачи из **NPC** отвечают житейскому представлению о трудной задаче и вряд ли допускают эффективное решение. Поэтому, если встречается задача, для которой на практике не удается придумать неперебoрный алгоритм, то имеет смысл попытаться доказать ее **NP**-полноту, чтобы оправдать применение к ней тех или иных переборных схем.

3. После того как была установлена непустота класса **NPC** (теоремой Кука), появилась возможность доказательства **NP**-полноты массовой задачи **П** путем полиномиального сведения к **П** одной из известных **NP**-полных задач (соответствующий список см. в [1]). Действительно, из утверждения 3 следует

Теорема 3 (*критерий* **NP**-*полноты*). Массовая задача **П** распознавания свойств **NP**-полна тогда и только тогда, когда она принадлежит классу **NP** и к ней полиномиально сводится какая-либо **NP**-полная задача:

$$\{\mathbf{П} \in \mathbf{NPC}\} \iff \{\mathbf{П} \in \mathbf{NP} \text{ и } \exists \mathbf{П}' \in \mathbf{NPC}: \ \mathbf{П}' \propto \mathbf{П}\}.$$

Пользуясь теоремой 3, можно показать **NP**-полноту задачи о существовании целочисленного решения системы линейных неравенств с целыми коэффициентами (**ЦЛН**).

Утверждение 7. $\mathbf{ЦЛН} \in \mathbf{NPC}$.

Доказательство. 1) $\mathbf{ЦЛН} \in \mathbf{NP}$, так как подсказкой может служить решение системы, а его проверка сводится к умножению на заданные коэффициенты и сложению, что не превосходит полинома от длины записи всех коэффициентов (доказательство полиномиальности длины записи решения см. в [2, с. 330]).

2) $\mathbf{ВЫП} \propto \mathbf{ЦЛН}$. Общий вид системы линейных неравенств

$$a_{j1}z_1 + a_{j2}z_2 + \ldots + a_{jn}z_n \le b_j, \quad j = 1, \ldots, m.$$

Нетрудно представить в подобной форме условие истинности дизъюнктивной функции. Для этого заменим в каждой j-й функции знаки

дизъюнкции знаками суммы, а отрицания переменных z_i — на $(1-z_i)$ и напишем для получившейся линейной функции условие ≥ 1, добавив ограничения $z_i \geq 0$ и $z_i \leq 1$ на все переменные. Целочисленное решение $z^0 = \{z_i^0\}$ системы всех построенных неравенств является выполняющим набором для исходной КНФ (так как истинность КНФ эквивалентна истинности всех образующих ее дизъюнктивных функций). Таким способом решение любой индивидуальной задачи о выполнимости сводится к решению некоторой индивидуальной задачи $\mathbf{I} \in \mathbf{ЦЛН}$. Полиномиальность сведения очевидна.

Заметим, что фактически в п.2 данного доказательства доказан более сильный результат о сведении **ВЫП** к подзадаче **ЦЛН** — задаче о существовании булева решения системы линейных неравенств с целыми коэффициентами (**БЛН**). Доказательство принадлежности **БЛН** классу недетерминированно полиномиальных задач повторяет п.1 данного доказательства без ссылки на [2] (так как полиномиальность длины булева решения очевидна), тем самым получено и

Утверждение 8. $\mathbf{БЛН} \in \mathbf{NPC}$.

§3. Классы сложности. Сильная NP-полнота и псевдополиномиальность

Доказательство **NP***-полноты задачи о 3-выполнимости. Взаимоотношение классов* **P, NP** *и* **NPC, NP** *и* **co-NP**. **NP***-трудные задачи. Класс* **PSPACE**. *Псевдополиномиальные алгоритмы. Пример для задачи о рюкзаке. Сильная* **NP***-полнота. Теорема о связи сильной* **NP***-полноты задачи с существованием псевдополиномиального алгоритма ее решения.*

1. Кроме задачи о выполнимости, **NP**-полнота всех остальных известных задач из класса **NPC** (в том числе и **КМ**) была доказана на основе теоремы 3 с помощью полиномиального сведения. Общие рецепты доказательства полиномиальной сводимости (см. в [1]) легко использовать лишь в простейших случаях. Чтобы научиться их применять, надо разобрать большое число примеров (в частности, имеющиеся в [1,2]), на что у нас в рамках данной работы нет возможности. Однако, еще один пример будет далее приведен с целью показать, что не только любая подзадача сводится к соответствующей задаче (автоматически), но возможно и обратное сведение.

Рассмотрим частный случай задачи о выполнимости, когда в КНФ могут входить лишь дизъюнктивные функции трех переменных (**3-ВЫП**). Поскольку **D(3-ВЫП)⊂D(ВЫП)**, то по определению **3-ВЫП∝ВЫП**. Так что **3-ВЫП∈NP** (по утверждению 5). Но ее **NP**-полнота требует специального доказательства, ибо частные массовые задачи содержат меньше индивидуальных задач и могут оказаться проще; например, аналогичная задача **2-ВЫП** полиномиальна. Для получения результата **3-ВЫП** $\in$ **NPC** докажем, что **NP**-полная задача о выполнимости сводится к своей подзадаче (частному случаю) **3-ВЫП**.

Утверждение 9. **ВЫП** $\propto$ **3-ВЫП**.

Доказательство. Покажем, что произвольную дизъюнктивную функцию $f^j(z^j)$ k переменных можно представить в виде конъюнкции дизъюнктивных функций от трех переменных (за счет введения дополнительных переменных u^j). Обозначим через y_i переменную z_i^j или $\overline{z}_i^j$ в зависимости от того, как i-я компонента z^j входит в рассматриваемую дизъюнктивную функцию; тогда последнюю можно записать как $y_1 \vee y_2 \vee \ldots \vee y_k$ и при $k > 3$ заменить на КНФ:

$$(y_1 \vee y_2 \vee u_1^j)\&(y_3 \vee \overline{u_1^j} \vee u_2^j)\&(y_4 \vee \overline{u_2^j} \vee u_3^j)\& \ldots$$
$$\ldots \&(y_{k-2} \vee \overline{u_{k-4}^j} \vee u_{k-3}^j)\&(y_{k-1} \vee y_k \vee \overline{u_{k-3}^j}).$$

Отметим, что данная замена не является эквивалентной. Действительно, если исходная дизъюнктивная функция равнялась нулю, то построенная КНФ равна нулю при всех значениях u, но если исходная дизъюнктивная функция равнялась 1, то найдется такое значение u, чтобы КНФ равнялась 1. Этого, однако, достаточно для сохранения ответа на вопрос о существовании выполняющего набора.

УПРАЖНЕНИЕ 4. Завершить доказательство NP-полноты задачи 3-ВЫП (рассмотреть случаи $k < 3$).

2. Универсальность задач из класса **NPC** (**NP**-полных задач) состоит в том, что основные нерешенные вопросы для класса **NP** (недетерминированно полиномиальных задач) достаточно разрешить хотя бы для одной **NP**-полной задачи, чтобы получить ответ для всего класса **NP**. Кроме утверждения 6 здесь также важно

Утверждение 10. Если для некоторой **NP**-полной задачи **П** дополнительная к ней $\overline{\mathbf{П}}$ принадлежит классу **NP**, то **NP = co-NP**.

Доказательство. Так как $\mathbf{\Pi} \in \mathbf{NPC}$, то $\forall \mathbf{\Pi}' \in \mathbf{NP}\ \ \mathbf{\Pi}' \propto \mathbf{\Pi}$, отсюда и $\overline{\mathbf{\Pi}}' \propto \overline{\mathbf{\Pi}}$ (полиномиальное сведение осуществляется той же функцией — см. определение 6). Но $\overline{\mathbf{\Pi}} \in \mathbf{NP}$, значит, $\overline{\mathbf{\Pi}}' \in \mathbf{NP}$ по утверждению 5. С учетом произвольности $\mathbf{\Pi}' \in \mathbf{NP}$ получили, что $\mathbf{co\text{-}NP} \subseteq \mathbf{NP}$. Обратное включение доказывается на основании очевидного равенства $\mathbf{\Pi} = \overline{\overline{\mathbf{\Pi}}}$.

Напомним, что гипотеза $\mathbf{NP} = \mathbf{co\text{-}NP}$ в настоящее время кажется нереальной (реальная гипотеза $\mathbf{P} = \mathbf{NP} \cap \mathbf{co\text{-}NP}$), и вряд ли для какой-либо **NP**-полной задачи удастся доказать принадлежность классу **co-NP**. Поэтому для конкретной недерминированно полиномиальной массовой задачи $\mathbf{\Pi} \in \mathbf{NP}$, если ее решение представляет интерес, имеет смысл попытаться доказать включение $\overline{\mathbf{\Pi}} \in \mathbf{NP}$ (т.е. ее хорошую характеризацию) и затем построить полиномиальный алгоритм решения, либо, когда указанное включение не доказывается, надо попробовать полиномиально свести к $\mathbf{\Pi}$ одну из известных **NP**-полных задач (т.е. показать **NP**-полноту $\mathbf{\Pi}$) и в случае успеха подыскивать переборную схему решения (учитывая ограничения на размерность практически решаемых индивидуальных задач).

Доказательство универсальности $\mathbf{\Pi}$, т.е. включения $\mathbf{\Pi} \in \mathbf{NPC}$, удается не всегда, и в теории сложности был введен объемлющий **NPC** класс **NP**-*трудных* задач, содержащий

1) $\mathbf{\Pi}$ распознавания свойств, для которых доказано, что $\mathbf{\Pi}' \propto \mathbf{\Pi}$ для $\mathbf{\Pi}' \in \mathbf{NPC}$, но не показано, что $\mathbf{\Pi} \in \mathbf{NP}$ (в частности, это задачи $\mathbf{\Pi} \in \mathbf{co\text{-}NPC}$);

2) $\mathbf{\Pi}$ оптимизации, для которых соответствующие $\mathbf{\Pi}$ распознавания свойств **NP**-полны (например, задача коммивояжера из п.1 §1);

3) и остальные массовые задачи (не обязательно распознавания свойств), к которым *сводятся по Тьюрингу* какие-либо **NP**-полные задачи $\mathbf{\Pi}' \in \mathbf{NPC}$, где сводимость по Тьюрингу — $\mathbf{\Pi}' \propto_T \mathbf{\Pi}$ — означает, что из существования полиномиального алгоритма решения $\mathbf{\Pi}$ следует существование полиномиального алгоритма и для $\mathbf{\Pi}'$.

Задачи $\mathbf{\Pi}$ (не обязательно распознавания свойств), для которых $\exists \mathbf{\Pi}' \in \mathbf{NPC}$: $\mathbf{\Pi}' \propto_T \mathbf{\Pi}$ и $\exists \mathbf{\Pi}'' \in \mathbf{NP}$: $\mathbf{\Pi} \propto_T \mathbf{\Pi}''$, называются задачами из класса **NP**-*эквивалентных*.

Все рассмотренные нами классы задач **П** — *классы сложности* включаются в общий класс **PSPACE** массовых задач (не обязательно распознавания свойств), для решения которых существуют алгоритмы, требующие не более чем полиномиальной памяти. Вопрос о равенстве **P** и **PSPACE** является открытым. Рабочая гипотеза состоит в наличии строгого включения **P**$\subset$**PSPACE**, при этом **NP**-полные, **NP**-трудные и **NP**-эквивалентные задачи должны включаться в **PSPACE** $\setminus$ **P**. (Соответствующую диаграмму взаимосвязи классов сложности см. в [2, с. 412].)

Заметим, что теоретически рассмотренную схему построения классов сложности можно применять и для других схем классификации; в частности, вводят также класс **PSPACE**-полных задач (к которым полиномиально сводится любая задача из класса **PSPACE**). Кроме полиномиальной сводимости можно аналогично говорить о логарифмической сводимости, о задачах, требующих логарифмической памяти и т.п. В настоящее время наиболее интенсивно изучаемым здесь оказывается класс **NC** (Nick Class, автор N. Pippenger) задач, решаемых за время, ограниченное полиномом от логарифма длины входа, и требующих полиномиальной памяти (логарифмическое время при возможности полиномиального числа процессоров). К сожалению, изложение полученных для **NC** результатов выходит за рамки введения в теорию сложности.

3. Ранее уже отмечалось, что с практической точки зрения разница между полиномиальным алгоритмом (для полиномов высоких степеней) и экспоненциальным весьма условна, а все дело в возможности или невозможности избежать полного перебора. Вопрос, все ли **NP**-полные и **NP**-трудные задачи трудны для практического счета, мы обсудим ниже в этом параграфе.

Рассмотрим самую простую (по формулировке) **NP**-трудную задачу — задачу о рюкзаке (**ЗР**), заключающуюся в том, чтобы из имеющегося набора полезных вещей собрать рюкзак ограниченного объема с наибольшей пользой. Формально требуется найти

$$\max_{z:\ z_j \in \{0,1\}\ \forall j=1,\dots,n} \{\sum_{j=1}^{n} c_j z_j \mid \sum_{j=1}^{n} w_j z_j \le K\},$$

где $c_j \in \mathbf{Z}_+$ — полезность (ценность), $w_j \in \mathbf{Z}_+$ — объем (вес) j-й вещи, а переменная z_j определяет класть или не класть ее в рюкзак; максимально возможный объем (вес) рюкзака задается параметром $K \in \mathbf{Z}_+$. Соответствующая задача распознавания свойств — существует ли булево решение системы двух линейных неравенств

$$\sum_{j=1}^{n} c_j z_j \geq B \quad \text{и} \quad \sum_{j=1}^{n} w_j z_j \leq K$$

с натуральными коэффициентами — **NP**-полна (доказательство не следует из утверждения 8, так как рассматривается частный случай **БЛН**, поэтому см. [1, с. 87 или 2, с. 386]). Для решения **ЗР** известен следующий метод (*динамического программирования*).

Произвольная индивидуальная задача **I**$\in$**ЗР** погружается в семейство задач поиска

$$F_i(E) \doteq \max_{z:\ z_j \in \{0,1\}\ \forall j=i,\ldots,n} \{\sum_{j=i}^{n} c_j z_j \mid \sum_{j=i}^{n} w_j z_j \leq K - E\},$$

$F_1(0)$ — значение **ЗР**. И решаются все задачи данного семейства по рекуррентным формулам, где i убывает с n до 1. А именно, положим $F_i(E) \doteq 0 \ \ \forall E \geq K, \ \ \forall i$. Имеем $\forall E = \overline{0, K-1}$:

$$F_n(E) = \begin{cases} 0, & E > K - w_n, \\ c_n & \text{иначе}, \end{cases}$$

и $\forall i = n-1, \ldots, 2$: $F_i(E) = \max\{F_{i+1}(E),\ c_i + F_{i+1}(E + w_i)\} \doteq$

$$\doteq \max_{z_i \in \{0,1\}} \{c_i z_i + F_{i+1}(E + w_i z_i)\}; \quad F_1(0) = \max\{F_2(0),\ c_1 + F_2(w_1)\}.$$

Число итераций предложенного алгоритма равно nK и того же порядка будет его временна́я сложность. Отметим, что алгоритм не является полиномиальным, ибо для записи числа K требуется порядка $\log_2 K$ символов; он также оказывается переборным — перебирает все варианты заполненности рюкзака. Однако при не очень больших объемах рюкзака можно довольно быстро получить решение. Для обобщения указанного свойства дадим

Определение 8. Обозначим через $\mathrm{num}(\mathbf{I})$ максимальное по модулю целое число (или 0), фигурирующее при задании числовых параметров для индивидуальной задачи $\mathbf{I}$, и через $|\mathbf{I}| \doteq |e(\mathbf{I})|$ — длину записи $\mathbf{I}$. Алгоритм $\mathbf{A}$ решения массовой задачи $\mathbf{\Pi}$ (не обязательно распознавания свойств) называется *псевдополиномиальным*, если для некоторого полинома $p(\cdot,\cdot)$ выполнено $t_{\mathbf{A}}(e(\mathbf{I})) < p(|\mathbf{I}|,\mathrm{num}(\mathbf{I}))$ $\forall \mathbf{I} \in \mathbf{\Pi}$.

Примером псевдополиномиального алгоритма является алгоритм динамического программирования для решения **ЗР**. Для многих других задач (в частности, **КМ**) псевдополиномиальных алгоритмов не известно. Чтобы выделить класс таких задач, введем понятие *полиномиального сужения* массовой задачи $\mathbf{\Pi}$ как множества тех индивидуальных задач, числовые параметры которых не превосходят полинома от длины входа,

$$\mathbf{\Pi}_{p(\cdot)} \doteq \{\mathbf{I} \in \mathbf{\Pi} |\ \mathrm{num}(\mathbf{I}) < p(|\mathbf{I}|)\}.$$

Определение 9. Массовая задача $\mathbf{\Pi}$ распознавания свойств называется *сильно* **NP**-*полной*, если ее полиномиальное сужение **NP**-полно, т.е. $\exists p(\cdot)$ — полином: $\mathbf{\Pi}_{p(\cdot)} \in \mathbf{NPC}$.

Примерами сильно **NP**-полных задач являются **ВЫП** и **3-ВЫП** (как совпадающие со своими полиномиальными сужениями), **БЛН** (поскольку **ВЫП** была сведена к ее полиномиальному сужению, в котором модули правых частей не превышают n), **ЦЛН** (как обобщение **БЛН** в отличие от **ЗР**), а также **КМ** [1, с. 123-124].

Теорема 4. Если $\mathbf{P} \neq \mathbf{NP}$, то ни для какой сильно **NP**-полной задачи не существует псевдополиномиального алгоритма решения.

Доказательство проведем от противного. Пусть ДМТ $\mathbf{A}$ решает сильно **NP**-полную задачу $\mathbf{\Pi}$ и $\forall \mathbf{I} \in \mathbf{\Pi}$ $t_{\mathbf{A}}(e(\mathbf{I})) < p'(|\mathbf{I}|,\mathrm{num}(\mathbf{I}))$ для полинома $p'(\cdot,\cdot)$. Тогда $\forall \mathbf{I} \in \mathbf{\Pi}_{p(\cdot)}$ $t_{\mathbf{A}}(e(\mathbf{I})) < p'(|\mathbf{I}|, p(|\mathbf{I}|)) = p''(|\mathbf{I}|)$, т.е. $\mathbf{\Pi}_{p(\cdot)} \in \mathbf{P}$ — противоречие с $\mathbf{\Pi}_{p(\cdot)} \in \mathbf{NPC}$ или утверждением 6.

Сильно **NP**-полные задачи считаются наиболее трудными для счета среди всех задач класса **NP**. Далее мы покажем, что для подобных задач в оптимизационной постановке отсутствуют эффективные алгоритмы поиска даже приближенного решения. Рекомендуемым подходом к их решению является разбиение на подзадачи $\mathbf{\Pi}'$: $\mathbf{D}(\mathbf{\Pi}') \subseteq \mathbf{D}(\mathbf{\Pi})$, $\mathbf{Y}(\mathbf{\Pi}') = \mathbf{Y}(\mathbf{\Pi}) \cap \mathbf{D}(\mathbf{\Pi}')$, анализ сложности подзадач

и разработка схем перебора (см. в §§10,12) для $\mathbf{\Pi}' \in \mathbf{NPC}$. При этом для сильно **NP**-полных подзадач не удается использовать метод динамического программирования в качестве способа перебора (ибо, по-видимому, реализующие его алгоритмы псевдополиномиальны) и следует ориентироваться на схему метода ветвей и границ (§§10,11).

§4. Приближенное решение задач комбинаторной оптимизации

Определение задачи комбинаторной оптимизации и приближенного алгоритма ее решения. Утверждение о разнице между приближенным и точным оптимумом для задачи о рюкзаке. Определение ε-приближенного алгоритма и полностью полиномиальной приближенной схемы (ППТС). Связь между существованием ППТС и псевдополиномиальностью. Теорема об отсутствии ППТС для задач оптимизации, соответствующих сильно **NP***-полным задачам распознавания свойств. Пример задачи о коммивояжере.*

Важный класс массовых задач образуют *задачи дискретной (комбинаторной) оптимизации.* Для оптимизационной постановки задачи $\mathbf{\Pi}$ решением каждой индивидуальной задачи $\mathbf{I} \in \mathbf{\Pi}$ является произвольная реализация оптимума

$$Opt_{\mathbf{\Pi}}(\mathbf{I}) \doteq \max_{z \in S_{\mathbf{\Pi}}(\mathbf{I})} f_{\mathbf{\Pi}}(\mathbf{I}, z),$$

т.е. такая точка $z^*(\mathbf{I}) \in S_{\mathbf{\Pi}}(\mathbf{I})$, для которой $f_{\mathbf{\Pi}}(\mathbf{I}, z^*(\mathbf{I})) = Opt_{\mathbf{\Pi}}(\mathbf{I})$. Здесь $S_{\mathbf{\Pi}}(\mathbf{I})$ — область допустимых значений дискретной (целочисленной) переменной z, $f_{\mathbf{\Pi}}(\mathbf{I}, \cdot) : S_{\mathbf{\Pi}}(\mathbf{I}) \to \mathbf{Z}$ — целевая функция индивидуальной задачи $\mathbf{I}$ оптимизации, знак max в постановке задачи может быть заменен на min.

Будем обозначать σ_S и σ_f те компоненты входного слова $\sigma = e(\mathbf{I})$, определяющего параметры индивидуальной задачи $\mathbf{I} \in \mathbf{\Pi}$, которые задают допустимую область (ограничения задачи) и функцию цели соответственно. Например, для **ЗР** имеем $f_{\mathbf{ЗР}}(\sigma, z) = \langle c, z \rangle$, $S_{\mathbf{ЗР}}(\sigma) = \{z = (z_1, \ldots, z_n) | z_j \in \{0,1\}\ \forall j = \overline{1,n}$ и $\langle w, z \rangle \le K\}$, $\sigma_S = (n, w, K)$ и $\sigma_f = c$. Здесь и далее знак $\langle \cdot, \cdot \rangle$ обозначает скалярное произведение векторов.

Определение 10. Алгоритм $\mathbf{A}$ называется *приближенным алгоритмом решения массовой задачи* $\mathbf{\Pi}$ *оптимизации*, если $\forall \mathbf{I} \in \mathbf{\Pi}$ он находит некоторую точку из допустимой области $z_{\mathbf{A}}(\mathbf{I}) \in S_{\mathbf{\Pi}}(\mathbf{I})$, принимаемую за приближенное решение. Значение $f_{\mathbf{\Pi}}(\mathbf{I}, z_{\mathbf{A}}(\mathbf{I}))$ называется приближенным значением оптимума и обозначается $A(\mathbf{I})$.

Говорить об абсолютной погрешности приближенного алгоритма решения массовой задачи оптимизации (на классе всевозможных индивидуальных задач) не имеет большого смысла, как показывает

Утверждение 11. Если $\mathbf{P} \neq \mathbf{NP}$, то ни для какой константы $C > 0$ не существует полиномиального приближенного алгоритма $\mathbf{A}$ решения $\mathbf{ЗР}$ с оценкой $|Opt_{\mathbf{ЗР}}(\mathbf{I}) - A(\mathbf{I})| \leq C \quad \forall \mathbf{I} \in \mathbf{ЗР}$.

Доказательство проведем от противного. Пусть найдены такие C и $\mathbf{A}$. Построим алгоритм $\mathbf{A}'$ следующим образом: $\forall \mathbf{I} \in \mathbf{ЗР}$ домножим все коэффициенты c_j на $C+1$ — получим индивидуальную задачу $\mathbf{I}' \in \mathbf{ЗР}$, к которой применим алгоритм $\mathbf{A}$ и разделим полученный ответ на $C+1$, т.е. $A'(\mathbf{I}) = A(\mathbf{I}')/(C+1)$. Очевидно, $Opt_{\mathbf{ЗР}}(\mathbf{I}') = (C+1)Opt_{\mathbf{ЗР}}(\mathbf{I})$ и из полиномиальности алгоритма $\mathbf{A}$ вытекает полиномиальность $\mathbf{A}'$. При этом его точность равна $|Opt_{\mathbf{ЗР}}(\mathbf{I}) - A'(\mathbf{I})| = |Opt_{\mathbf{ЗР}}(\mathbf{I}') - A(\mathbf{I}')|/(C+1) \leq C/(C+1) < 1$, т.е. равна нулю (так как все значения целевой функции целые). Получили полиномиальный алгоритм точного решения $\mathbf{ЗР}$. Проверка $Opt_{\mathbf{ЗР}}(\mathbf{I}) \geq B$ полиномиальна, значит, построили и полиномиальный алгоритм решения $\mathbf{ЗР}$ в постановке распознавания свойств, что с учетом универсальности последней противоречит утверждению 6.

Определение 11. Приближенный алгоритм $\mathbf{A}$ решения массовой задачи $\mathbf{\Pi}$ оптимизации называется *ε-приближенным алгоритмом* решения $\mathbf{\Pi}$ для некоторого $\varepsilon > 0$, если

$$\forall \mathbf{I} \in \mathbf{\Pi} \qquad \frac{|Opt_{\mathbf{\Pi}}(\mathbf{I}) - A(\mathbf{I})|}{|Opt_{\mathbf{\Pi}}(\mathbf{I})|} < \varepsilon,$$

т.е. его относительная погрешность не превосходит ε.

Для ε-приближенных алгоритмов приведем следующий результат [2, с. 439], доказательство которого основано на методе динамического программирования и в данном курсе опускается.

Теорема 5. Пусть для задачи $\mathbf{\Pi}$ оптимизации
1) существует псевдополиномиальный алгоритм ее решения;
2) $\forall \mathbf{I} \in \mathbf{\Pi} \quad |Opt_{\mathbf{\Pi}}(\mathbf{I})| < p_1(|\mathbf{I}|, \mathrm{num}(\mathbf{I}))$ и $\mathrm{num}(\mathbf{I}) < p_2(|\mathbf{I}|, Opt_{\mathbf{\Pi}}(\mathbf{I}))$

для некоторых полиномов $p_1(\cdot,\cdot)$, $p_2(\cdot,\cdot)$;
3) $\forall\sigma = e(\mathbf{I})$, $\mathbf{I} \in \mathbf{\Pi}$: параметры σ_S, задающие ограничения, и σ_f, задающие целевую функцию, не пересекаются, и $\forall z \in S_{\mathbf{\Pi}}(\sigma)$ функция цели $f_{\mathbf{\Pi}}(\sigma, z)$ линейно зависит от параметров σ_f;

тогда $\exists p(\cdot,\cdot)$ — полином: $\forall\varepsilon > 0$ $\exists$ ε-приближенный алгоритм $\mathbf{A}_\varepsilon$ решения $\mathbf{\Pi}$ с временно́й сложностью $T_{\mathbf{A}_\varepsilon}(|\mathbf{I}|) < p(|\mathbf{I}|, 1/\varepsilon)$.

Теорема 5 справедлива, например, для **ЗР** (сравните результат с утверждением 11). Набор алгоритмов $\{\mathbf{A}_\varepsilon\}$, определенный в теореме 5, называется *полностью полиномиальной приближенной схемой* (ППТС) решения задачи $\mathbf{\Pi}$ оптимизации. Наличие ППТС — лучшее, чего можно ожидать при решении **NP**-трудных задач. К сожалению, в целом ряде случаев на это нельзя рассчитывать, так как имеется

Теорема 6. Если для $\mathbf{\Pi}$ оптимизации соответствующая ей $\mathbf{\Pi}$ распознавания свойств является сильно **NP**-полной и $\exists p'(\cdot)$ — полином: $|Opt_{\mathbf{\Pi}}(\mathbf{I})| < p'(\mathrm{num}(\mathbf{I}))$ $\forall\mathbf{I} \in \mathbf{\Pi}$, то при условии, что $\mathbf{P} \neq \mathbf{NP}$, для $\mathbf{\Pi}$ не существует ППТС.

Доказательство проведем от противного. Пусть ППТС существует. Построим алгоритм $\mathbf{A}'$ следующим образом. Для любой индивидуальной задачи $\mathbf{I}$ $\mathbf{A}'$ вызывает $\mathbf{A}_\varepsilon$ с $\varepsilon = 1/(p'(\mathrm{num}(\mathbf{I}))+1)$. Тогда по определению ε-приближенного алгоритма $\mathbf{A}_\varepsilon$ $|Opt_{\mathbf{\Pi}}(\mathbf{I}) - A'(\mathbf{I})| < |Opt_{\mathbf{\Pi}}(\mathbf{I})|/(p'(\mathrm{num}(\mathbf{I})) + 1) < p'(\mathrm{num}(\mathbf{I}))/(p'(\mathrm{num}(\mathbf{I})) + 1)$ по условию теоремы. Но в левой части полученного неравенства было целое число, которое оказывается равным нулю как неотрицательное, меньшее 1. Таким образом, алгоритм $\mathbf{A}'$ точен, причем $T_{\mathbf{A}'}(|\mathbf{I}|) = T_{\mathbf{A}_\varepsilon}(|\mathbf{I}|) < p(|\mathbf{I}|, p'(\mathrm{num}(\mathbf{I})) + 1)$ по определению ППТС. Следовательно, алгоритм $\mathbf{A}'$ псевдополиномиален, что противоречит теореме 4.

Утверждение 12. Если $\mathbf{P} \neq \mathbf{NP}$, то ни для какого $\varepsilon > 0$ не существует полиномиального ε-приближенного алгоритма решения оптимизационной постановки задачи коммивояжера.

Доказательство см. в [2, с. 440–441].

Для частного случая **КМ**, в котором функция $d(\cdot,\cdot)$ расстояния между городами удовлетворяет неравенству треугольника, известен 0.5-приближенный полиномиальный алгоритм Кристофидеса [2, с. 429–432] (решения **КМ** оптимизации).

2. ОСНОВЫ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Литература:

3. Хачиян Л. Г. Сложность задач линейного программирования. М.: Знание, 1987, N 10.

§5. Понятие о сложности задачи линейного программирования (ЛП)

Определение основной задачи ЛП (озЛП). Принцип граничных решений и геометрическое описание симплекс-метода. Алгебраическая и битовая сложность методов ЛП. Результаты по сложности задач, близких к ЛП. Теорема о границах решений задач ЛП с целыми коэффициентами. Теорема о мере несовместности систем линейных неравенств с целыми коэффициентами.

1. Согласно [3] линейное программирование — это раздел прикладной математики, изучающий теорию, приложения и методы решения конечных систем линейных неравенств с конечным числом вещественных неизвестных $x_1, \ldots, x_n$:

$$\left.\begin{array}{rl} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \le b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \le b_m \end{array}\right\} \qquad (1)$$

или в сокращенной записи $Ax \le b$. Считаем, что матрица A не содержит нулевых строк a_i. *Основная задача ЛП (озЛП)* состоит в нахождении такого решения (1), которое максимизирует заданную линейную функцию $\langle c, x\rangle \doteq c_1x_1 + c_2x_2 + \ldots + c_nx_n$ вектора неизвестных x по всем вещественным x, удовлетворяющим системе (1):

$$\max_{x \in \mathbf{R}^n:\ Ax \le b} \langle c, x\rangle; \qquad (2)$$

озЛП (2) с n неизвестными и m ограничениями называется задачей размерности (n, m) и задается числовой таблицей

$$\left|\begin{array}{ccccc} a_{11} & a_{12} & \ldots & a_{1n} & b_1 \\ a_{21} & a_{22} & \ldots & a_{2n} & b_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{m1} & a_{m2} & \ldots & a_{mn} & b_m \\ c_1 & c_2 & \ldots & c_n & 0 \end{array}\right| \qquad (3)$$

своих коэффициентов. В частном случае $c = \bar{0}$ задача (2) эквивалентна (1), так что умение решать озЛП предполагает умение решать системы линейных неравенств (ЛН). В §7 будет показано обратное сведение. Вообще говоря, в форме (2) может быть представлена любая задача ЛП с ограничениями равенствами и неравенствами, в том числе *каноническая задача* ЛП

$$\max_{Ax=b,\ x\geq\bar{0}} \langle c, x\rangle.$$

(Здесь и далее черта сверху будет использоваться для выделения вектора в отличие от похожего числа.)

УПРАЖНЕНИЕ 5. Представить каноническую задачу ЛП в форме озЛП.

Несмотря на то, что формально задачи ЛП не являются дискретными ($x \in \mathbf{R^n}$), их решение нетрудно свести к перебору конечного числа угловых точек (вершин полиэдра (1), задающего ограничения) на основании *принципа граничных решений*:

если задача (2) имеет решение, то найдется такая подматрица A_I матрицы A, что любое решение системы уравнений $A_I x = b_I$, т.е.

$$\{a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n = b_i |\ i \in I\},$$

реализует максимум в (2).

Отметим, что для невырожденных A_I решение соответствующей системы уравнений $A_I x = b_I$, удовлетворяющее ограничениям (1), является угловой точкой (1). Из принципа граничных решений следует, что если угловая точка (1) существует, то разрешимая задача (2) имеет решение и в угловой точке (1), т.е. она эквивалентна максимизации $\langle c, x\rangle$ на конечном множестве вершин полиэдра (1). Процедура решения системы линейных уравнений методом Гаусса требует не более полинома 3-й степени от m, n (точнее, $\max(m,n)[\min(m,n)]^2$) арифметических операций с элементами A и b. Однако число возможных подматриц матрицы A экспоненциально, и метод полного их перебора не эффективен.

В 1820-х гг. Ж. Фурье и затем в 1947 г. Дж. Данциг предложили метод направленного перебора смежных вершин (1) — в направлении возрастания целевой функции (2) — *симплекс-метод*. Хотя каждый шаг симплекс-метода (представляющий собой определенную

процедуру пересчета элементов симплекс-таблицы (3)) ограничен по порядку числом mn арифметических операций, в настоящее время для всех известных вариантов симплекс-метода приведены примеры, экспоненциальные по числу итераций, когда перебирается более $2^{\min(n,m/2)}$ вершин, но доказательство невозможности построить полиномиальный симплекс-метод также отсутствует. Подчеркнем, что на практике симплекс-метод не показывает данной оценки ("плохие" примеры довольно редки). Можно построить алгоритм решения задачи ЛП с оценкой $f(n)m$ арифметических операций (над числами, записанными в (3)), где $f(\cdot)$ растет быстрее экспоненты. Алгоритм с полиномиальной оценкой одновременно по n и m не известен и вряд ли будет построен.

Теперь заметим, что функция, оценивающая число арифметических операций в зависимости от n и m, не учитывает длину кода элементов (3), а только их количество и поэтому не является временно́й сложностью алгоритма. Указанная функция носит название *алгебраической сложности* в отличие от *битовой сложности* — функции, оценивающей число арифметических операций с битами (или с конечными порциями — по размеру машинного регистра) цифровой записи параметров индивидуальной задачи ЛП в зависимости от длины входного слова, т.е. от n, m и длин l кодов чисел в симплекс-таблице. Очевидно, битовая сложность алгоритма соответствует его временно́й сложности (см. §1). Входные коэффициенты задачи ЛП обычно рациональны, поэтому далее условимся считать их целыми, тогда l — длина записи максимального коэффициента в (3) — конечна. Набор (n, m, l) называется битовой размерностью задачи ЛП. Вопрос о существовании алгоритма ЛП с полиномиальной битовой сложностью был решен Л. Г. Хачияном в 1978 г., и тем самым была доказана полиномиальность задач ЛП. Основные моменты этого доказательства излагаются в следующем пункте и §6. Здесь же укажем на отличие классов сложности задачи ЛП и других линейных задач.

Метод Гаусса решения системы линейных алгебраических уравнений имеет полиномиальную алгебраическую сложность, т.е. является *сильнополиномиальным*. Для ЛП вопрос о существовании сильнополиномиального алгоритма открыт. Кроме того, задача решения

системы линейных уравнений принадлежит классу **NC**, а аналогичный результат для ЛП означал бы равенство **NC=P**, ожидать которое нет оснований.

Из полиномиальности ЛП вытекает полиномиальность задачи **ЛН** (существует ли решение системы ЛН): **ЛН** $\in$ **P**. Аналогичные задачи с дополнительным ограничением целочисленности или булевости решения **NP**-полны: **ЦЛН**, **БЛН** $\in$ **NPC** (см. §2), т.е. полиномиальные алгоритмы для них вряд ли будут построены.

Существует неполиномиальное обобщение ЛП — задача проверки истинности высказываний вида

$$\mathbf{Q_1}x_1 \ldots \mathbf{Q_n}x_n\ \mathbf{F}(\langle a_1, x\rangle \le b_1, \ldots, \langle a_m, x\rangle \le b_m),$$

где $\mathbf{Q_i} \in \{\forall, \exists\}$, а $\mathbf{F}(\cdot, \ldots, \cdot)$ — предложение, составленное из линейных неравенств с помощью связок $\&, \vee, \neg$ (и, или, отрицание). Доказано, что любой алгоритм, решающий эту массовую задачу, имеет не менее чем экспоненциальную сложность. Тот же результат будет и при замене равенствами всех неравенств в постановке задачи.

2. Рассмотрим некоторые свойства задач ЛП с целыми коэффициентами. Для любой целочисленной матрицы D введем параметр

$$\Delta(D) \doteq \max_{\{D' \,-\, \text{квадратная подматрица } D\}} |\det D'|.$$

Будем обозначать через $[A|b]$ матрицу, составленную из A и вектора-столбца $b \in \mathbf{Z^m}$, дописанного справа. Здесь и далее $\mathbf{Z^m}$ — m-мерное пространство целочисленных векторов, $\mathbf{Z^m_+}$ — его неотрицательный ортант.

Теорема 1 *(о границах решений).* Если озЛП (2) размерности (n, m) с целыми коэффициентами разрешима, то у нее существует рациональное решение x^* в шаре $\|x\| \le n^{1/2}\Delta([A|b])$ и значением озЛП (2) $d^* \doteq \langle c, x^*\rangle$ является рациональное число t/s со знаменателем, ограниченным величиной $\Delta(A)$.

Доказательство. На основании принципа граничных решений $\exists A_I \subseteq A$: по правилу Крамера $|x^*_j| = |\det A^j_I / \det A_I| \le \Delta([A|b])$, ибо $|\det A_I| \ge 1$, а определитель матрицы A^j_I, полученной из A_I заменой

j-го столбца на $\pm b_I$, не превышает по модулю $\Delta([A|b])$. Отсюда для евклидовой нормы x^* получаем требуемую оценку. С учетом целочисленности вектора c знаменатель d^* может быть выбран равным знаменателю x_j^* $\forall j$, и 2-е утверждение теоремы следует из определения $\Delta(A) \geq |\det A_I|$.

Определение 1. Точка x^ε называется ε-приближенным решением системы линейных неравенств (1), если

$\langle a_i, x^\varepsilon \rangle \leq b_i + \varepsilon \quad \forall i = \overline{1,m}$, где a_i — i-я строка матрицы A,

или в матричной записи, обозначая $\mathbf{e}$ — вектор-столбец из единиц,

$$Ax^\varepsilon \leq b + \varepsilon \mathbf{e}. \qquad (1_\varepsilon)$$

Теорема 2 *(о мере несовместности)*. Если система ЛН (1) имеет ε_1-приближенное решение для $\varepsilon_1 \doteq 1/[(n+2)\Delta(A)]$, то эта система разрешима, т.е. имеет точное решение x^0.

Доказательство. Обозначим через ε^* минимальное ε, при котором система (1_ε) имеет решение (по условию $\varepsilon^* \leq \varepsilon_1$):

$$\varepsilon^* \doteq \min_{(x,\varepsilon):\ Ax \leq b + \varepsilon \mathbf{e}} \varepsilon.$$

Допустим, что утверждение теоремы не верно, тогда $\varepsilon^* > 0$. Задача определения ε^* является (с учетом равенства $\min(\cdot) = -\max(-\cdot)$) озЛП с целевым вектором $c = (0, \ldots, 0, -1)$, $n+1$ переменными (x, ε) и ограничениями $Ax - \varepsilon \mathbf{e} \leq b$. Следовательно, по теореме 1 ε^* может быть представлена в виде дроби со знаменателем, не превышающим $\Delta([A| - \mathbf{e}]) \leq (n+1)\Delta(A)$, т.е. $\varepsilon^* \geq 1/[(n+1)\Delta(A)] > \varepsilon_1$ — пришли к противоречию с определением ε^*.

Аналогичное утверждение справедливо и для озЛП.

Определение 2. Точка x_ε^* называется ε-приближенным решением озЛП (2), если она является ε-приближенным решением системы (1) и реализует максимум в (2) с ε-точностью:

$$\langle a_i, x_\varepsilon^* \rangle \leq b_i + \varepsilon \quad \forall i = \overline{1,m} \text{ и } \langle c, x_\varepsilon^* \rangle \geq d^* - \varepsilon.$$

Теорема 2 * *(о мере несовместности)*. Если озЛП (2) имеет ε_2-приближенное решение для $\varepsilon_2 \doteq 1/(2n^2\Delta^3(A))$, то эта задача имеет точное решение x^*.

Доказательство см. в [3, с. 21].

§6. Метод эллипсоидов

Полиномиальный алгоритм округления ε_1-приближенного решения системы линейных неравенств. Метод эллипсоидов ε_2-приближенного решения озЛП. Оценка сложности метода эллипсоидов. Полиномиальность ЛП.

1. Имея ε-приближенное решение (1) с $\varepsilon \le \varepsilon_1$, можно (на основании теоремы 2, §5) быть уверенным в существовании точного решения системы линейных неравенств. Оказыватся, процедура получения x^0 из x^{ε_1} является полиномиальной. Соответствующий *алгоритм округления* ε_1-приближенного решения системы (1) до точного был указан Л. Г. Хачияном и состоит в следующем.

Присвоим $x^1 := x^{\varepsilon_1}$ и подставим x^1 в (1). Разобьем множество $M \doteq \{1, \dots, m\}$ индексов неравенств в системе на два подмножества

$$M(x^1) \doteq \{i : \ |\langle a_i, x^1\rangle - b_i| \le \varepsilon_1\},$$
$$M \setminus M(x^1) \doteq \{i : \ \langle a_i, x^1\rangle - b_i \le -\varepsilon_1\}.$$

Найдем решение x'^1 системы равенств $A_{M(x^1)}x = b_{M(x^1)}$ (существует по теореме 2). Пусть x'^1 не является точным решением (1), т.е. в x'^1 не выполнилось i-е неравенство для какого-либо $i \notin M(x^1)$. Тогда введем множество индексов невыполненных неравенств $M^+ \doteq \{i|\ \langle a_i, x'^1\rangle > b_i\} \subseteq M \setminus M(x^1)$ и рассмотрим на отрезке $[x^1, x'^1]$ ближайшую к x'^1 точку, в которой еще выполнены все неравенства для $i \in M^+$ (в x^1 они выполнены с ε_1-запасом). А именно определим

$$\tau \doteq \min_{i\in M^+} \frac{b_i - \langle a_i, x^1\rangle}{\langle a_i, x'^1\rangle - \langle a_i, x^1\rangle}, \quad i_1 \doteq \arg\min_{i\in M^+} \frac{b_i - \langle a_i, x^1\rangle}{\langle a_i, x'^1\rangle - \langle a_i, x^1\rangle}$$

и присвоим $x^2 := (1-\tau)x^1 + \tau x'^1$. Имеем $M(x^2) \supseteq M(x^1) \cup \{i_1\}$, ибо неравенства с индексами из $M(x^1)$ ε_1-приближенно выполнялись как равенства на всем отрезке $[x^1, x'^1]$, а неравенство с индексом $i_1 \in M^+$, не выполненное в точке x'^1, выполняется в x^2 как равенство по построению. Таким образом, $M(x^2) \supset M(x^1)$, но $|M(x)| \le m$, поэтому, повторяя указанную процедуру с заменой x^1 на x^2 и т.д., придем не более чем через $\max(n, m)$ шагов к тому, что решение x' соответствующей системы равенств окажется x^0 — решением (1).

С учетом полиномиальности задачи решения систем уравнений предложенный алгоритм округления полиномиален.

Аналогичный алгоритм имеется и для округления ε_2-приближенного решения озЛП $x^*_{\varepsilon_2}$ до точного x^* (см. [3, с. 21]). Поэтому для построения полиномиального алгоритма решения озЛП осталось указать полиномиальный алгоритм поиска ε_2-приближенного решения озЛП в шаре $\|x\| \le n^{1/2}\Delta$ или удостоверения, что такого решения нет (по теоремам $1{,}2^*$ из §5). Требуемый алгоритм, основанный на *методе эллипсоидов*, который предложили в 1976–77 гг. Д. Б. Юдин и А. С. Немировский и (независимо) Н. З. Шор, приводится в следующих пунктах.

Здесь и далее $\Delta \doteq \Delta(D)$, где матрица D задается таблицей (3).

2. Пусть E — произвольный эллипсоид в $\mathbf{R^n}$ с центром ξ и ненулевого объема $\mathrm{vol}E$. Рассмотрим $(n-1)$-мерную плоскость, заданную вектором g нормали и проходящую через центр ξ эллипсоида E. Обозначим через $E^-(g)$ один из двух полуэллипсоидов, на которые разбивает E данная плоскость, $E^-(g) = E \cap \{x|\ \langle g, x-\xi\rangle \le 0\}$.

Утверждение 1. Полуэллипсоид $E^-(g)$ эллипсоида E можно целиком заключить в новый эллипсоид E', имеющий объем, строго меньший E,

$$\frac{\mathrm{vol}E'}{\mathrm{vol}E} < e^{-1/(2n+2)}, \qquad (*)$$

и E' можно вычислить по $E^-(g)$ за $O(n^2)$ арифметических операций.

Доказательство. Пусть E — единичный шар с центром в точке $\bar{0}$: $E = \{x \in \mathbf{R^n} :\ \|x\| \le 1\}$, а $E^-(g) = E \cap \{x_n \ge 0\}$. Поместим центр E' в точку $\xi' = (0, \ldots, 0, \frac{1}{n+1})$, тогда

$$E' = \{x|\ (x_1^2 + \ldots + x_{n-1}^2)/\beta^2 + (x_n - \frac{1}{n+1})^2/\alpha^2 \le 1\},$$

где $\alpha \doteq 1 - 1/(n+1) < e^{-1/(n+1)}$, $\beta^2 \doteq 1 + 1/(n^2-1) < e^{1/(n^2-1)}$. Отношение объемов равно произведению полуосей $\alpha\beta^{n-1} < e^{-1/(2n+2)}$, отсюда получаем $(*)$, ибо любой эллипсоид можно превратить в шар афинным преобразованием координат, сохраняющим объем. Действительно, будем представлять произвольный эллипсоид E с помощью его центра ξ и матрицы Q $(n \times n)$, задающей указанное преобразование: $E = \{x|\ x = \xi + Qy,\ \|y\| \le 1\}$. Обозначим $\eta \doteq Q^T g/\|Q^T g\|$, где верхний индекс ^{T} — знак транспонирования. Тогда ξ' и Q', представляющие эллипсоид E' минимального объема, описанный вокруг

полуэллипсоида $E^-(g)$, пересчитываются по формулам

$$\xi' = \xi - \frac{1}{n+1} Q\eta, \quad Q' = \frac{n}{\sqrt{(n^2-1)}} \{Q + (\sqrt{\frac{n-1}{n+1}} - 1) Q\eta\eta^T\}$$

за $O(n^2)$ арифметических операций.

3. *Метод эллипсоидов получения ε-приближенного решения озЛП.* Положим $\varepsilon := \varepsilon_2 \doteq 1/(2n^2\Delta^3)$. Введем множество ε-приближенных решений озЛП в шаре радиуса $R \doteq n^{1/2}\Delta$ с центром в $\bar{0}$: $X^*_\varepsilon \doteq \{x |\ \langle a_i, x\rangle \le b_i + \varepsilon \ \ \forall i = \overline{1,m}, \ \ \langle c, x\rangle \ge d^* - \varepsilon, \ \ \|x\| \le R\}$. Выберем указанный выше шар в качестве начальной итерации для эллипсоида $E \supset X^*_\varepsilon$. Рассмотрим произвольную итерацию.

Проверяем, является ли центр ξ эллипсоида E ε-приближенным решением. Если да, то алгоритм заканчивает свою работу, в противном случае строим эллипсоид E' для очередной итерации как минимальный по объему эллипсоид, содержащий полуэллипсоид $E^-(g)$ (см. п.2), где вектор g определяется следующим образом. Так как $\xi \notin X^*_\varepsilon$, то либо

1^0) $\exists i: \ \ \langle a_i, \xi\rangle > b_i + \varepsilon$, и тогда $g := a_i$, либо

2^0) $\langle c, \xi\rangle < d^* - \varepsilon$ и $g := -c$.

Убедимся, что при этом $X^*_\varepsilon \subset E'$. Действительно, для варианта 1^0 $\forall x \in X^*_\varepsilon \ \ \langle a_i, x\rangle \le b_i + \varepsilon < \langle a_i, \xi\rangle$, т.е. $X^*_\varepsilon \subset E \cap \{x |\ \langle a_i, x - \xi\rangle \le 0\} = E^-(a_i) \subset E'$; и аналогично получим для варианта 2^0

$$X^*_\varepsilon \subset E \cap \{x |\ \langle c, x - \xi\rangle \ge 0\} = E^-(-c) \subset E'.$$

Теперь с $E := E'$ возвращаемся к началу итерации (на новый шаг).

Оценим число итераций метода эллипсоидов. Покажем, что X^*_ε содержит шар радиуса $r/2$, где $r \doteq \varepsilon/(hn^{1/2}) < R, \ \ h \ge |a_{ij}|, |c_j|$ (*h высота задачи*). Пусть x^* — точное решение в X^*_ε. Из $\|x^* - x\| \le r$ следует $|\langle a_i, x\rangle - \langle a_i, x^*\rangle| \le \|a_i\| \ \|x^* - x\| \le hn^{1/2}r = \varepsilon \ \ \forall i \in M$ и $|\langle c, x\rangle - \langle c, x^*\rangle| \le \|c\| \ \|x^* - x\| \le hn^{1/2}r$, т.е. указанный выбор r гарантирует, что все такие x будут ε-приближенными решениями. Поскольку $\|x^*\| \le R$, то множество тех из рассматриваемых x, для которых $\|x\| \le R$ (т.е. пересечение шаров радиуса r и R, включающее центр первого), содержит шар радиуса $r/2$. Этот шар и принадлежит X^*_ε. Таким образом, объем X^*_ε больше объема n-мерного

шара радиуса $r/2$. Значит, объем эллипсоида, построенного последним, например E^k для k-й итерации, не должен оказаться меньше объема этого шара. Отсюда и из утверждения 1 получаем для k соотношение

$$\left(\frac{r}{2R}\right)^n \le \frac{\mathrm{vol}X^*_\varepsilon}{\mathrm{vol}E^1} \le \frac{\mathrm{vol}E^k}{\mathrm{vol}E^1} < e^{-k/(2n+2)},$$

из которого k (по определению r, R, ε, h и Δ) не превосходит

$$2n^2\ln(Rnh/\varepsilon) < 2n^2\ln(2n^{3.5}\Delta^5) < 10n^2\ln(n\Delta).$$

УПРАЖНЕНИЕ 6. Оценить по порядку битовую длину L входа озЛП: доказать, что $L > O(\ln(n\Delta))$.

Следовательно, число итераций метода эллипсоидов $k < O(n^2)L$, и с учетом $O(n^2+nm)$ арифметических операций для каждой итерации получим оценку $O(n^3(n+m)L)$ для числа арифметических операций, достаточного методу эллипсоидов при поиске ε_2-приближенного решения озЛП. Алгоритм округления ε_2-приближенного решения до точного этой оценки не портит (см.[3, с. 21]). Можно также показать, что при реализации метода эллипсоидов и алгоритма округления все арифметические операции достаточно проводить с числами двоичной длины, ограниченной $O(L)$. При этом ошибки, возникающие за счет конечности числа разрядов (ошибки округлений), поглощаются путем некоторого дополнительного увеличения ("раздутия") описанного эллипсоида E' на каждой итерации [3, с. 24], что не влияет на порядок оценки для общего числа итераций. В результате временна́я сложность такой процедуры решения озЛП оказывается полиномом от длины входа и справедлива

Теорема 3. Задача ЛП с целыми коэффициентами разрешима за полиномиальное от длины входа время.

Следствием данной теоремы является

Утверждение 2. **ЛН $\in$ P**.

Подчеркнем, что несмотря на доказанную полиномиальность, метод эллипсоидов не может конкурировать с симплекс-методом при практическом решении задач ЛП (реально он применяется в выпуклом квадратичном программировании), поскольку полученная оценка числа его итераций достигается на любых индивидуальных

задачах, даже если в качестве начального приближения взять решение. Тогда как симплекс-метод для "хороших" (невырожденных) задач дает оценку $O(n^3)$, на порядок меньшую, чем метод эллипсоидов, и за одну итерацию может подтвердить, что начальное приближение является решением. Тем не менее сам факт полиномиальности ЛП инициировал поиск новых методов ЛП, что привело к созданию целого класса эффективных методов математического программирования — *методы внутренней точки* — и позволило построить конкурентоспособные полиномиальные алгоритмы ЛП. Идея их построения будет изложена в следующем параграфе, где также приводятся необходимые сведения по теории ЛП, начиная с ЛН.

§7. Теория двойственности ЛП. Идея метода Кармаркара

Следствия систем ЛН. Афинная лемма Фáркаша /без доказательства/. Лемма Фаркаша о неразрешимости. Теорема двойственности ЛП. Сведение озЛП к однородной системе уравнений с ограничением положительности. Идея метода Кармаркара и его отличие от симплекс-метода.

1. Система ЛН (1) называется *разрешимой*, если $\exists x$: $Ax \le b$, и *неразрешимой* — в противном случае. ОзЛП (2) разрешима, когда разрешима система (1) и максимум в (2) достигается.

Определение 3. Линейное неравенство

$$\langle c, x\rangle \le d \tag{4}$$

является *следствием* разрешимой системы линейных неравенств (1), если для любого x, удовлетворяющего (1), выполнено (4).

Способ получения неравенств-следствий довольно прост: выберем произвольные $\lambda_i \ge 0 \ \ \forall i \in M$, домножим на λ_i каждое i-е неравенство системы (1) и сложим; получим для вектора

$$c = \sum_{i\in M} \lambda_i a_i \quad \text{и любого числа} \quad d \ge \sum_{i\in M} \lambda_i b_i,$$

что (4) будет следствием (1). Оказывается, других следствий у ЛН не бывает. А именно справедлива

Лемма *Фа́ркаша (афинная).* Линейное неравенство (4) является следствием разрешимой в вещественных переменных системы ЛН (1) тогда и **только тогда**, когда существует вектор $\lambda \in \mathbf{R^m}$:

$$c = \sum_{i \in M} \lambda_i a_i, \quad d \geq \sum_{i \in M} \lambda_i b_i, \quad \lambda_i \geq 0 \ \ \forall i \in M. \tag{5}$$

(Схему доказательства см. в [3, с. 18].)

Для неразрешимой системы ЛН (1) можно формально считать следствием (1) заведомо неверное неравенство $\langle \bar{0}, x \rangle \leq -1$ и далее пользоваться афинной леммой Фаркаша, как показывает

Лемма Фаркаша *о неразрешимости.* Система ЛН (1) неразрешима тогда и только тогда, когда разрешима система

$$\sum_{i \in M} \lambda_i a_i = \bar{0}, \quad \sum_{i \in M} \lambda_i b_i \leq -1, \quad \lambda_i \geq 0 \ \ \forall i \in M. \tag{6}$$

Доказательство. Пусть (1) неразрешима, тогда из разрешимости системы $\langle a_i, x \rangle + x_{n+1} \leq b_i \ \ \forall i \in M$ должно следовать, что $x_{n+1} \leq -\varepsilon < 0$, т.е. следствием этой системы является неравенство $\langle (0, \ldots, 0, 1/\varepsilon), (x, x_{n+1}) \rangle \leq -1$ и из афинной леммы Фаркаша получаем (6) (а также в дополнение $\sum \lambda_i = 1/\varepsilon$). Если же (6) разрешима, то указанное выше неравенство $\langle \bar{0}, x \rangle \leq -1$ оказывается следствием (1) и должно выполняться для всех x, удовлетворяющих (1), значит, таких не существует.

Теперь мы можем доказать основной теоретический результат ЛП — теорему двойственности, на которой базируются как методы решения задач ЛП, так и способы анализа решения, и которая фактически дает необходимые и достаточные условия оптимальности в ЛП. Наличие двойственности, обусловив хорошую характеризацию задачи **ЛН**, предопределило полиномиальность ЛП.

Определение 4. *Двойственной* к задаче ЛП на максимум с ограничениями неравенствами в форме озЛП (2) называется следующая задача ЛП на минимум с ограничениями в канонической форме:

$$\min \{ \sum_{i \in M} \lambda_i b_i \mid \sum_{i \in M} \lambda_i a_i = c, \ \lambda_i \geq 0 \ \forall i \in M \}, \quad \text{или} \quad \text{в краткой записи}$$

$$\min_{\lambda A = c,\ \lambda \geq \bar{0}} \langle \lambda, b \rangle. \tag{7}$$

Для того, чтобы построить двойственную к произвольной задаче ЛП, надо представить ее в форме озЛП, применить формулу (7), а затем вернуться к обозначениям исходной задачи.

УПРАЖНЕНИЕ 7. Показать, что двойственная задача к двойственной задаче ЛП совпадает с прямой задачей ЛП: представить (7) в форме озЛП (аналогично упражнению 5), выписать двойственную к полученной задаче и свести ее к (2).

Теорема 4 (*двойственности ЛП*). Задача ЛП разрешима тогда и только тогда, когда разрешима двойственная к ней. В случае разрешимости оптимальные значения целевых функций в обеих задачах совпадают, т.е. $d^* = d^{**}$, где d^* — значение (2), d^{**} — значение (7).

Доказательство проведем для случая озЛП, поскольку любая задача ЛП адекватно представляется в такой форме.

Пусть задача (2) разрешима, тогда (4) является следствием (1) $\forall d \geq d^*$ и не является $\forall d < d^*$, что по афинной лемме Фаркаша эквивалентно разрешимости (5) при $d \geq d^*$ и неразрешимости (5) при $d < d^*$, т.е. $d^* = \min\{d|\ (5)\}$, а это и есть значение (7).

И наоборот, из разрешимости (7) следует неразрешимость (6), ибо в противном случае min в (7) обращался бы в $-\infty$ (так как прибавление решения (6) к решению (7) дает допустимую точку и уменьшает значение целевой функции (7)). Отсюда получаем разрешимость (1) по лемме Фаркаша о неразрешимости. Кроме того, разрешимость (7) означает разрешимость (5) для любого $d \geq d^{**}$, так что (4) оказывается следствием (1) для $d \geq d^{**}$, и поэтому d^{**} ограничивает сверху значение (2), т.е. максимум в (2) достигается. Таким образом получили разрешимость задачи (2) и можем вернуться к началу доказательства для установления равенства $d^* = d^{**}$.

Из теоремы 4 непосредственно получаем

Утверждение 3. Задача ЛП оптимизации эквивалентна решению системы линейных неравенств.

Действительно, озЛП (2) эквивалентна задаче ЛП (7) и обе они эквивалентны системе ЛН относительно неизвестных (x, λ):

$$Ax \leq b,\ \langle c, x \rangle = \langle b, \lambda \rangle,\ \lambda A = c,\ \lambda \geq \bar{0}. \tag{8}$$

Утверждение 4. Задача ЛП оптимизации эквивалентна решению системы линейных уравнений в неотрицательных переменных.

Доказательство. От системы ЛН (8) переходим к ограничениям в канонической форме аналогично упражнениям 5,7.

Утверждение 5. Задача ЛП эквивалентна поиску неотрицательного ненулевого решения однородной системы линейных уравнений.

Доказательство. На основании утверждения 4 озЛП сводится к некоторой системе ЛН (с целыми коэффициентами) относительно вектора вещественных неизвестных y:

$$\hat{P}y = \hat{q}, \quad y \geq \bar{0}, \tag{9}$$

пусть $\hat{P}$ — матрица $(K \times (N-1))$. Введем параметр $\hat{R}$, мажорирующий координаты какого-то решения (9) (по теореме о границах решений), если система (9) разрешима. Добавим к (9) неравенство

$$\langle y, \mathbf{e} \rangle = y_1 + \ldots + y_{N-1} \leq N\hat{R},$$

которое превратим в равенство с помощью дополнительной переменной y_N: $\langle \hat{y}, \mathbf{e} \rangle = y_1 + \ldots + y_{N-1} + y_N = N\hat{R}$, а (9) перепишется как $[\hat{P}|\bar{0}]\hat{y} = \hat{q}, \quad \hat{y} \geq \bar{0}$. Теперь сделаем замену переменных $x := \hat{y}/\hat{R}$ и обозначим $P \doteq N\hat{R}[\hat{P}|\bar{0}] - [\hat{q}|\hat{q}|\ldots|\hat{q}]$. Придем к однородной системе $Px = \bar{0}$ с дополнительными ограничениями $x = (x_1, \ldots, x_N) \geq \bar{0}$, $\langle x, \mathbf{e} \rangle = N$, что соответствует системе $Px = \bar{0}, \quad x \geq \bar{0}, \quad \langle x, \mathbf{e} \rangle > 0$ с решениями-лучами $tx^0 \quad \forall t > 0$, любое из которых пересчитывается в решение исходной системы.

2. *Метод Кармаркара* (N. Karmarkar, 1984 г.). Воспользуемся утверждением 5 и обозначениями, введенными при его доказательстве. Пусть $p(x) \doteq (\langle p_1, x \rangle)^2 + \ldots + (\langle p_K, x \rangle)^2$, где p_i — строки P. Тогда $p(x) = 0$ эквивалентно $Px = \bar{0}$. Введем *функцию Кармаркара*

$$k(x) \doteq \frac{[p(x)]^{N/2}}{x_1 x_2 \cdot \ldots \cdot x_N}.$$

Применяя теорему 2 и алгоритм округления к задаче решения (9), можно показать, что для точного ее решения достаточно найти такой $\hat{x} > \bar{0}$, для которого $k(\hat{x}) \leq 1/[3(\Delta(\hat{P}))^N]$ [3, с. 25–26].

Полиномиальный алгоритм поиска нужного приближенного $\hat{x}$ приводится в [3, с. 26–28], и мы не будем его описывать. Отметим

только, что аналогичный алгоритм может быть построен на основании применения метода Ньютона (см. в разд.3) к задаче минимизации функции Кармаркара или ей подобных. В результате получаем целый класс полиномиальных алгоритмов ЛП, которые на практике оказываются сравнимыми с симплекс-методом, не имея теоретических недостатков последнего. Предложенные алгоритмы строятся на принципиально новой идее: не дискретной, а непрерывной трактовки задачи ЛП, когда вместо перебора конечного числа угловых точек осуществляют поиск решения в исходном пространстве вещественных переменных, и траектории алгоритмов не проходят через угловые точки. Напомним, что метод эллипсоидов также не ориентируется на угловые точки многогранника ограничений. Характерно, что именно такой уход от дискретного программирования позволил построить полиномиальные алгоритмы ЛП. Поэтому далее будет дан некоторый обзор основных подходов к решению непрерывных задач оптимизации.

Замечание. Если бы речь шла о непосредственном поиске точного решения задачи ЛП указанными методами, то нельзя было бы гарантировать конечношаговость (не то, что полиномиальность) соответствующих алгоритмов. Для их применения существенной является возможность остановки в приближенном решении благодаря наличию полиномиального алгоритма округления. Но поскольку для его работы требуется начальное приближение из определенной окрестности решения, зависящей от длины l или высоты h, или длины входа L конкретной задачи ЛП, то и число итераций алгоритмов, базирующихся на рассматриваемом принципе, зависит от числа цифр в записи элементов матрицы ограничений. Так что не удается использовать данную идею для построения сильнополиномиальных алгоритмов ЛП, кроме как в частных случаях ограниченности элементов матрицы (например, в задачах на графах и сетях, где $a_{ij} = 0, \pm 1$).

3. ЭЛЕМЕНТЫ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ (МП)

Литература:

4. Карманов В. Г. Математическое программирование. М.: Наука, 1986.

5. Сухарев А. Г., Тимохов А. В., Федоров В. В. Курс методов оптимизации. М.: Наука, 1985.

6. Мину М. Математическое программирование. М.: Наука, 1990.

§8. Обзор идей МП

Классификация задач МП. Преимущества выпуклого случая. Понятие о градиентных и Ньютоновских методах минимизации. Условная оптимизация, способы освобождения от ограничений (методы барьеров и штрафов).

1. Задача ЛП, как и задача минимизации функции Кармаркара, является частным случаем задачи МР:

$$\min_{x \in X} f(x). \tag{1}$$

Здесь требуется найти $\arg\min\limits_{x \in X} f(x) \in \mathrm{Arg}\min\limits_{x \in X} f(x)$, т.е.

$$x^* \in X^* \doteq \{x^* \in X |\ f(x^*) \le f(x) \ \ \forall x \in X\}, \ \text{ и } f^* = f(x^*). \tag{2}$$

Любой такой x^* называется *решением* (1); f^* — *значение* (1), или *оптимальное значение* целевой функции f в задаче (1), X — *множество ограничений* или *допустимое множество*.

В зависимости от природы множества X задачи оптимизации классифицируются как: дискретные (комбинаторные) — X конечно или счетно, целочисленные — $x_j \in \mathbf{Z}$, булевы — $x_j \in \mathbf{B}$, вещественные (непрерывные) — $X \subseteq \mathbf{R^n}$, бесконечномерные или в функциональном пространстве, например, когда X — подмножество гильбертова пространства $\mathbf{L_2}$, и т.п. В данном разделе будем по преимуществу рассматривать задачи с вещественными переменными, которые собственно и называются (традиционно) *задачами математического программирования (ЗМП)*. Если $X \subset \mathbf{R^n}$, то говорим о задаче *условной* оптимизации (при условии $x \in X$), иначе ($X = \mathbf{R^n}$) получаем задачу *безусловной* оптимизации.

Для ЗМП минимум в (1) достигается в условиях теоремы Вейерштрасса (f непрерывна, X компактно или для некоторого $\hat{x} \in X$ ограничено *множество Лебега* функции f — $\{x \in X | f(x) \leq f(\hat{x})\}$).

Кроме разделения на условные и безусловные, ЗМП классифицируются по свойствам целевой функции и множества ограничений соответственно на задачи ЛП, выпуклого программирования, гладкие или негладкие и др. Для каждого из классов ЗМП разрабатываются свои численные методы их решения. С точки зрения численных методов существенно также разделение на *локальную* и *глобальную* оптимизацию. В определении (2) речь идет о глобальном минимуме, который, однако, найти не просто, и поэтому задачу стараются свести к дискретной оптимизации на множестве локальных минимумов.

Определение 1. Точка $x^0 \in X$ называется точкой *локального* минимума в ЗМП (1), если $\exists \varepsilon > 0 : \quad f(x^0) \leq f(x) \ \ \forall x \in X \cap \mathbf{O}_\varepsilon(x^0)$. Здесь и далее $\mathbf{O}_\varepsilon(x)$ обозначает ε-окрестность точки x.

Для поиска локального минимума применяются специальные методы, которые при определенных предположениях оказываются эффективными. Тогда как общая задача глобальной оптимизации является **NP**-трудной. Действительно к ней сводится **NP**-полная.

Утверждение 1. **ЦЛН**$\propto$ЗМП.

Доказательство. Поскольку задача ЛН является частным случаем задачи ЛП, то для сведения **ЦЛН** к ЗМП достаточно представить условие целочисленности переменных в виде ограничений (неравенств) на вещественные переменные, что нетрудно сделать, например, так: $\{x_j \in \mathbf{Z}\}$ эквивалентно $\{x_j \in \mathbf{R} | \sin^2(\pi x_j) \leq 0\}$.

Поэтому методы глобальной оптимизации будут рассмотрены в разд.4, а в данном параграфе остановимся на поиске локального экстремума. Отметим, что для ряда экстремальных постановок задач физики точки локального экстремума имеют самостоятельное значение. Кроме того, существует целый класс ЗМП, для которого локальный экстремум совпадает с глобальным минимумом, это — задачи выпуклого программирования.

Определение 2. Функция f называется *выпуклой на* X, если ее надграфик $\mathrm{epigraf}_X f \doteq \{(x, y) | \ y \geq f(x), \ x \in X\}$ — выпуклое множество. Функция, выпуклая на всей области определения, называется выпуклой. Множество называется выпуклым, если вместе с любыми

двумя своими точками оно содержит отрезок, их соединяющий.

Утверждение 2. Любая точка локального минимума выпуклой функции является точкой ее глобального минимума.

Доказательство. Пусть $f(x^0) > f(x^*)$. Тогда $f(x^0) > f(x)$ для всех точек x полуинтервала $(x^0, x^*]$ (по определению 2), а значит, и в некоторой окрестности x^0 — противоречие с определением 1.

Для решения задач выпуклого программирования применим метод эллипсоидов, причем в гладком случае отсечение полуэллипсоида проводится на основе градиента невыполненного ограничения в полной аналогии с алгоритмом из §6. Поэтому задача поиска ε-приближенного решения задачи выпуклого программирования оказывается полиномиально разрешимой. Для *острых* задач выпуклого программирования — когда функция цели убывает в окрестности минимума не медленнее некоторой линейной функции — можно получить и точное решение.

2. Общими методами локальной оптимизации (для произвольного, не обязательно выпуклого, случая) являются *методы локального спуска.*

Определение 3. Вектор $h \in \mathbf{R^n}$ называется *направлением убывания* функции f в точке x, если $f(x+\alpha h) < f(x)$ для всех достаточно малых $\alpha > 0$.

Утверждение 3. Пусть f дифференцируема в точке x. Тогда, если $\langle \mathrm{grad} f(x), h\rangle < 0$, то h — направление убывания функции f в точке x, а если h — направление убывания функции f в точке x, то $\langle \mathrm{grad} f(x), h\rangle \le 0$.

Доказательство. Из условия дифференцируемости f имеем для достаточно малых $\alpha > 0$: $f(x+\alpha h) - f(x) = \langle \mathrm{grad} f(x), \alpha h\rangle + o(\alpha) = \alpha\{\langle \mathrm{grad} f(x), h\rangle + o(\alpha)/\alpha\}$. Очевидно, последняя добавка не изменит знака выражения в фигурных скобках, если скалярное произведение строго отрицательно или строго положительно. Отсюда автоматически вытекает требуемое утверждение.

Таким образом, направление локального убывания дифференцируемой функции должно составлять острый угол с ее антиградиентом, который является в смысле линейного приближения наилучшим направлением убывания. Для мнемоники приведем эпиграф к главе, посвященной градиентным методам минимизации, из 1-го издания

книги Ф. П. Васильева *Численные методы решения экстремальных задач*: "Вот кто-то с горочки спустился — антиградиент!"

Если $\mathrm{grad} f(x) = 0$, то x будет *стационарной точкой*. Отметим, что в условной оптимизации равенство нулю градиента уже не является необходимым условием минимума (соответствующие условия будут рассмотрены в §9). Но в более простом случае $X = \mathbf{R^n}$ можно, двигаясь небольшими шагами в направлении антиградиента функции f в текущей точке, прийти в стационарную точку, как правило, локального минимума. Так мы получаем идею *градиентного метода безусловной минимизации*, задаваемого итеративной процедурой

$$x^{t+1} = x^t - \alpha_t \mathrm{grad} f(x^t), \quad t = 1, 2, \ldots, \quad \forall x^1 \in \mathbf{R^n}.$$

Параметр α_t называется шаговым множителем и может выбираться, исходя из различных соображений, разными способами.

1) *Пассивные способы* — $\{\alpha_t\}$ выбирается заранее.
Постоянный шаг — $\alpha_t = \alpha_0$ для достаточно малых α_0.
Убывающий шаг (если α_0 не известно или при наличии помех) — $\alpha_t \downarrow 0, \ \sum \alpha_t = \infty, \ \sum \alpha_t^2 < \infty,$ например $\alpha_t = 1/t$.

2) *Адаптивные способы* — $\{\alpha_t\}$ зависит от реализующейся $\{x^t\}$.
Метод скорейшего спуска — $\alpha_t \in \mathrm{Arg\,min}_{\alpha > 0} f(x^t - \alpha\, \mathrm{grad} f(x^t))$.
Метод дробления шага (деления пополам) — если $f(x^{t+1}) > f(x^t)$, то возврат к t-й итерации с $\alpha_t := \alpha_t/2$. (Возможно и увеличение шага при стабильном убывании f, т.е. приближенный скорейший спуск.)
Правило Армихо — путем дробления шага добиваемся для α_t выполнения условия $f(x^t - \alpha_t \mathrm{grad} f(x^t)) - f(x^t) \le -\varepsilon \alpha_t \|\mathrm{grad} f(x^t)\|^2$.

В общем случае дифференцируемой, ограниченной снизу f можно получить сходимость градиентного метода к множеству стационарных точек, а при дополнительных предположениях доказывается (за исключением варианта с убывающим шагом) *линейная скорость сходимости*, которая в выпуклых задачах означает $\|x^{t+1} - x^*\| \le q\|x^t - x^*\|$ для некоторого $0 < q < 1$. Указанная линейная оценка объясняется тем, что в процессе минимизации градиентным методом используется линейная аппроксимация целевой функции на каждом шаге. Более высокую скорость сходимости получают для методов, основанных на квадратичной аппроксимации, в предположении дважды дифференцируемости f. Типичным примером здесь является *метод Ньютона*.

Пусть $f \in \mathbf{C^2(R^n)}$, разложим функцию f в ряд Тейлора в окрестности текущей точки x^t:

$$f(x)-f(x^t) = \langle \mathrm{grad} f(x^t), x-x^t\rangle + \frac{1}{2}\langle f''(x^t)(x-x^t), x-x^t\rangle + o(\|x-x^t\|^2).$$

Выберем x^{t+1} из условия минимизации квадратичной аппроксимации $f(x)$ в точке x^t, т.е. квадратичной части приращения $f(x)-f(x^t)$, получим метод Ньютона:

$$x^{t+1} = x^t - (f''(x^t))^{-1}\mathrm{grad} f(x^t), \quad t = 1, 2, \ldots,$$

где начальное приближение x^1 должно находиться достаточно близко к точке оптимума x^*. В таком случае (и при дополнительных предположениях, более сильных, чем для приведенной ранее оценки скорости сходимости градиентного метода) для метода Ньютона будет справедлива *квадратичная скорость сходимости*

$$\|x^{t+1} - x^*\| \le Q\|x^t - x^*\|^2, \quad \text{т.е.} \quad \|x^{t+1} - x^*\| \le \frac{1}{Q}(Q\|x^1 - x^*\|)^{2^t},$$

что предполагает $\|x^1 - x^*\| < 1/Q$ (оценку для Q см., например, в [5, с. 192]). Еще раз подчеркнем, что градиентный метод в отличие от ньютоновского сходится при любом начальном приближении. Из определения метода Ньютона также следует требование невырожденности матрицы вторых производных (гессиана) функции f.

Нетрудно видеть, что полученная формула метода Ньютона решения задач безусловной минимизации совпадает с формулой метода Ньютона решения системы уравнений $\mathrm{grad} f(x) = 0$, соответствующей необходимым условиям экстремума.

3. Для задач условной минимизации, например $\min\limits_{x \in [1,2]} x^2$,

предложенные методы нуждаются в модификации. В частности, для приведенного примера, когда множество X имеет достаточно простую структуру, указанные выше формулы совмещаются с процедурой проектирования на X на каждом шаге метода. Так приходим к методу *проекции градиента*

$$x^{t+1} = \mathrm{Pr}_X\{x^t - \alpha_t \mathrm{grad} f(x^t)\}, \quad t = 1, 2, \ldots, \quad \forall x^1 \in \mathbf{R^n}.$$

Для более сложных множеств X, допустим, задаваемых ограничениями неравенствами

$$X \doteq \{x \in \mathbf{R^n} \mid g_i(x) \le 0 \quad \forall i \in M\}, \tag{3}$$

универсальным способом освобождения от ограничений является их штрафование. А именно для достаточно большой константы $C > 0$ вместо задачи условной минимизации (1),(3) рассматривают задачу безусловной минимизации оштрафованной целевой функции

$$\min_{x \in \mathbf{R^n}} \{f(x) + C \sum_{i \in M} [g_i^+(x)]^p\}, \quad \text{где} \quad \sum_{i \in M} [g_i^+(x)]^p \; -$$

это *функция штрафа (штрафная функция)* для ограничений неравенств, $g^+(\cdot) \doteq \max[0, g(\cdot)]$ — *срезка* g, параметр штрафа $p \ge 1$. (Другие виды функций штрафа см. в [4,5].) В условиях непрерывности функций f, g_i, непустоты X и ограниченности множества Лебега функции f можно доказать, что с ростом константы штрафа

$$\lim_{C \uparrow \infty} \min_{x \in \mathbf{R^n}} \{f(x) + C \sum_{i \in M} [g_i^+(x)]^p\} = f^*. \tag{4}$$

Если $p = 1$ (функция-срезка и, следовательно, штрафная функция является острой), то $\exists C^* : \quad \min\{f(x) + C^* \sum_{i \in M} g_i^+(x)\} = f^*$ (существует *точный штраф*). Однако при $p > 1$ — *гладкий штраф* подобное равенство означало бы несущественность ограничений $x \in X$ (точка безусловного минимума и так находится в X).

Утверждение 4. Пусть $f, g_i \in \mathbf{C^1(R^n)}$, выпуклы, $p > 1$ и $\exists C^*$: $x^C \doteq \arg\min\{f(x) + C^* \sum [g_i^+(x)]^p\} \in X$, тогда

$$x^C \in \operatorname{Arg} \min_{x \in \mathbf{R^n}} f(x), \text{ т.е. } \min_{x \in \mathbf{R^n}} f(x) = \min_{x \in X} f(x).$$

Доказательство. Так как x^C — точка безусловного экстремума дифференцируемой функции, то градиент оштрафованной функции цели в ней равен нулю: $\operatorname{grad} f(x^C) + C^* p \sum [g_i^+(x^C)]^{p-1} \operatorname{grad} g_i(x^C) = 0$. Но из условия $x^C \in X$ все выражения в квадратных скобках, а значит, и второе слагаемое равны нулю. Отсюда следует $\operatorname{grad} f(x^C) = 0$, т.е. необходимое условие экстремальности точки x^C для задачи безусловной оптимизации, которое в выпуклом случае оказывается и

достаточным (см. утверждение 2). Поэтому x^C — точка безусловного минимума f. Но $x^C \in X$, так что x^C — и точка условного минимума f на X, ибо безусловный минимум не превышает условного. Утверждение доказано.

Таким образом, для гладкого штрафа не удается свести задачу условной минимизации к безусловной, тем не менее формула (4) позволяет итеративно комбинировать метод штрафов и градиентный метод в следующей процедуре: $\forall x^1 \in \mathbf{R^n}$

$$x^{t+1} = x^t - \alpha_t \{\mathrm{grad} f(x^t) + C_t p \sum_{i \in M} [g_i^+(x^t)]^{p-1} \mathrm{grad} g_i(x^t)\}, \quad t = 1, 2, \ldots,$$

которая сходится при определенных соотношениях между $\{\alpha_t\}$ и $\{C_t\}$, в частности для убывающего шага при $\sum \alpha_t^2 C_t^2 < \infty$ (например, $\alpha_t = 1/t, \quad C_t < \sqrt{t}$).

Утверждение 4 показывает, что траектории метода штрафа проходят, вообще говоря, вне множества ограничений X, хотя и сходятся к данному множеству. Из-за этого рассмотренный метод иногда также называют методом внешних штрафов в отличие от методов *внутренней точки*, или *барьеров*. Типичным примером применения метода барьеров является описанный в §7 метод Кармаркара, когда задача (9), эквивалентная задаче условной минимизации

$$\min_{x \geq \bar{0}, \ \sum x_j = N} p(x),$$

сводится к безусловной минимизации специальной барьерной функции $k(x)$, не позволяющей методу Ньютона выйти за ограничения $x > 0$, если в этих ограничениях выбрано начальное приближение. Различные виды барьерных функций см. в [4,5] — для них характерно быстрое возрастание при приближении изнутри к границе множества ограничений (тогда как штрафная функция стремится к нулю при приближении к множеству ограничений — извне). Для решения общей задачи МП (1),(3) с ограничениями неравенствами методу Кармаркара соответствует использование вместо рассмотренной выше штрафной функции, основанной на срезке, *логарифмической* барьерной функции, равной

$$-\frac{1}{C} \sum_{i \in M} \ln[-g_i(x)]$$

при $g_i(x) < 0\ \forall i \in M$ и $+\infty$ в противном случае. Эта функция также прибавляется к целевой, и справедливо соотношение, аналогичное (4).

Другие способы сведения задач условной оптимизации к безусловной, основанные на методе *множителей Лагранжа*, будут вытекать из результатов следующего параграфа.

§9. Двойственность в МП

Необходимые условия локального минимума обобщенно дифференцируемых функций при ограничениях неравенствах. Теорема Куна-Таккера. Понятие о регулярности ограничений неравенств в задаче МП. Метод множителей Лагранжа.

1. В этом параграфе будем рассматривать задачу условной оптимизации (1) с $X \subset \mathbf{R^n}$, $X \neq \emptyset$, по преимуществу, с ограничениями неравенствами (3). Как уже отмечалось, условие равенства нулю градиента для таких задач может не иметь никакого отношения к точкам условного экстремума. Поэтому выведем соответствующие необходимые условия для рассматриваемого случая. Вначале они будут даны в достаточно общей форме, допускающей применение для широкого класса задач МП (кусочно-гладких и при произвольным образом заданных ограничениях, а также не обязательно конечномерных). Затем проведем конкретизацию для ограничений (3). Для обычных задач МП (конечномерных, с непрерывно дифференцируемыми функциями) справедливы все дальнейшие построения и выводы при замене знака ∇ обычным градиентом. Таким образом, основой обобщения является следующее

Определение 4. Функция f называется *дифференцируемой по Адамару* в точке $x \in \mathbf{R^n}$, если существует вектор $\nabla f(x) \in \mathbf{R^n}$, такой что $\forall y \in \mathbf{R^n}$ выполнено:

$$\lim_{(\tau, y') \to (+0, y)} \frac{f(x + \tau y') - f(x)}{\tau} = \langle \nabla f(x), y \rangle.$$

Для бесконечномерных задач, когда f — функционал: $E \to \mathbf{R^1}$, где E некоторое функциональное пространство, требуется: $\nabla f(x) \in E'$ для пространства E', сопряженного к E, и $x, y \in E$. В гладком случае $\nabla f(x) = \mathrm{grad} f(x)$ и можно положить y' тождественно равным y.

В безусловной оптимизации существенную роль играли направления спуска (убывания целевой функции). В условной оптимизации, кроме убывания целевой функции, требуется отслеживать еще и невыход за ограничения. Поэтому вводится понятие *возможного* или *допустимого* направления в точке $x \in X$ для множества ограничений X как такого вектора y, для которого $\exists \tau^0 > 0 : \quad x + \tau y \in X \quad \forall \tau \in [0, \tau^0]$. Замыкание множества всех допустимых направлений в точке x для X дает следующее

Определение 5. *Контингентным конусом* к множеству X в точке x называется множество векторов

$$K(X, x) \doteq \{y | \ \exists \ \{(\tau_t, y^t)\}_{t=1}^{\infty} : \ (\tau_t, y^t) \to (+0, y), \quad x + \tau_t y^t \in X \quad \forall t\}.$$

Очевидно, для $\hat{x} \notin X \quad K(X, \hat{x}) = \emptyset$, а для $x' \in \mathrm{int} X \quad K(X, x') = \mathbf{R^n}$. Для $x \in \partial X$ в случае гладкой границы конус $K(X, x)$ называется также *конусом касательных* и соответствует касательным направлениям для ограничений-равенств.

Теорема 1 (*общий вид необходимых условий локального минимума в задаче (1)*). Пусть функция f дифференцируема по Адамару, $X \subset \mathbf{R^n}, \quad X \neq \emptyset, \quad x^0$ — точка локального минимума f в задаче (1), тогда $\forall y \in K(X, x^0) \quad \langle \nabla f(x^0), y \rangle \geq 0$.

Доказательство. Выберем $y \in K(X, x^0)$. Для соответствующих ему по определению 5 $\{\tau_t, y^t\}$ выполнено $x^0 + \tau_t y^t \in X$, и, начиная с достаточно большого $t, \quad x^0 + \tau_t y^t \in X \cap \mathbf{O}_\varepsilon(x^0)$ (ибо $\tau_t \to 0$), следовательно, по определению 1 $f(x^0 + \tau_t y^t) \geq f(x^0)$. В пределе получим

$$\lim_{(\tau, y') \to (+0, y)} \frac{f(x^0 + \tau y') - f(x^0)}{\tau} = \lim_{(\tau_t, y^t) \to (+0, y)} \frac{f(x^0 + \tau_t y^t) - f(x^0)}{\tau_t} \geq 0,$$

и требуемое соотношение вытекает из определения 4.

Содержательно данные условия означают, что среди допустимых направлений в точке локального минимума не должно быть направлений убывания целевой функции (см. утверждение 3 §8). Однако в таком общем виде этими условиями неудобно пользоваться.

Конкретизируем полученные условия для ограничений неравенств, когда X задается формулой (3). Введем $\forall x \in X$ множество

индексов $J(x) = \{i \in M |\ g_i(x) = 0\}$ — *активных ограничений* в точке x, т.е. таких неравенств из (3), которые в этой точке выполнены как равенства. И определим множество (конус)

$$G(x) \doteq \{y \in \mathbf{R^n} |\ \langle \nabla g_j(x), y\rangle \le 0 \quad \forall j \in J(x)\}.$$

Определение 6. Множество X для ограничений неравенств (3) называется *регулярным в точке* $x \in X$, если $G(x) \subseteq K(X, x)$.

Теорема 2 *(необходимые условия локального минимума с ограничениями неравенствами).* Пусть функции f, g_i $\forall i \in M$ дифференцируемы по Адамару, $X \neq \emptyset$, x^0 — точка локального минимума f в задаче (1),(3) и множество X регулярно в точке x^0. Тогда

$$\exists \lambda_j \ge 0: \quad \nabla\{f(x^0) + \sum_{j \in J(x^0)} \lambda_j g_j(x^0)\} = 0. \tag{5}$$

Доказательство. По теореме 1 и из определения регулярности X в x^0 следует, что $\langle \nabla f(x^0), y\rangle \ge 0$ для всех y, удовлетворяющих условию $\langle \nabla g_j(x^0), y\rangle \le 0 \quad \forall j \in J(x^0)$. Значит, по определению 3 §7, линейное неравенство $\langle \nabla f(x^0), y\rangle \ge 0$ является следствием системы линейных неравенств $\{\langle \nabla g_j(x^0), y\rangle \le 0 \quad \forall j \in J(x^0)\}$. Приведя это неравенство к стандартному виду $\langle -\nabla f(x^0), y\rangle \le 0$ и применив афинную лемму Фаркаша (§7), получим, что

$$\exists \lambda_j \ge 0: \quad -\nabla f(x^0) = \sum_{j \in J(x^0)} \lambda_j \nabla g_j(x^0).$$

Таким образом, для регулярных ограничений необходимым условием локального минимума в гладкой задаче (1),(3) является равенство нулю дифференциала функции в фигурных скобках в (5) для хоть каких-нибудь $\lambda_j \ge 0$. Чтобы не записывать в явном виде множество активных ограничений, вводят *функцию Лагранжа*

$$L(\lambda, x) \doteq f(x) + \sum_{j \in M} \lambda_j g_j(x) \doteq f(x) + \langle \lambda, \bar{g}(x^0)\rangle$$

(регулярной) задачи (1),(3), где вектор-функция $\bar{g}(\cdot) \doteq (g_j(\cdot) |\ j \in M)$. Из теоремы 2 следует, что равенство нулю дифференциала функции Лагранжа для $\lambda_j \ge 0$ также является необходимым условием

локального минимума в регулярной задаче (1),(3), ибо *множители Лагранжа* λ_j, соответствующие неактивным ограничениям, можно взять равными нулю. Последнее условие записывается как

$$\langle \lambda, \bar{g}(x^0) \rangle = 0 \tag{6}$$

и называется *условием дополняющей нежесткости*. Итак, доказана

Теорема 3 *(принцип оптимальности Лагранжа).* В предположениях теоремы 2 для задачи (1),(3) существует неотрицательный вектор множителей Лагранжа $\lambda \geq \bar{0}$, такой, что для x^0 выполнены *условия оптимальности*: $\nabla_x L(x^0, \lambda) = \bar{0}$ и (6).

Для выпуклых задач (1),(3) данные необходимые условия являются в регулярном случае и достаточными, и может быть доказана

Теорема 4 (*Куна, Таккера*). Если в задаче (1),(3) функции $f, g_j \in \mathbf{C^1(R^n)}$ выпуклы и множество X регулярно (в любой точке), то x^* — точка оптимума в этой задаче тогда и только тогда, когда в ней выполнены условия оптимальности для $\lambda \geq \bar{0}$.

Доказательство. Необходимость следует из предыдущих теорем, покажем достаточность. Для данного λ в точке x^* выполнено условие экстремальности x^* для функции $L(\cdot, \lambda)$. С учетом неотрицательности λ эта функция выпукла по x, значит, x^* является точкой ее минимума (см. утверждение 2 §8). Отсюда и из условия дополняющей нежесткости получим, что $f(x^*) = f(x^*) + \langle \lambda, \bar{g}(x^*) \rangle = L(x^*, \lambda) \leq L(x, \lambda) \doteq f(x) + \langle \lambda, \bar{g}(x) \rangle \leq f(x)\ \forall x \in X$ (ибо $g_j(x) \leq 0$ для x, удовлетворяющих ограничениям), что и требуется в определении (2).

Аналогичные теоремам 2–4 утверждения справедливы и для случая, когда X задается ограничениями-равенствами, и для смешанных систем ограничений равенств и неравенств: $g_j(x) \leq 0$, $g_i(x) = 0$. Только на соответствующие ограничениям-равенствам множители Лагранжа λ_i не надо накладывать условия неотрицательности, а на условие дополняющей нежесткости эти ограничения не влияют (в случае ограничений-равенств вообще опускаем (6) и приходим к классическому *правилу множителей Лагранжа*).

2. Теперь вспомним, что полученные условия являются значимыми лишь в предположении регулярности ограничений, для которого определение 6 не дает конструктивного способа проверки. В данном

пункте будут рассмотрены некоторые достаточные условия регулярности ограничений неравенств (3) для гладких задач.

Кроме $G(x)$, определенного в п.**1**, введем также множество

$$G^0(x) \doteq \{y \in \mathbf{R^n} \mid \langle \nabla g_j(x), y\rangle < 0 \quad \forall j \in J(x)\},$$

отличающееся заменой нестрогого неравенства строгим. Но это множество уже включается в контингентный конус.

Утверждение 5. В предположении дифференцируемости по Адамару (или непрерывной дифференцируемости) функций g_j, задающих ограничения (3), $G^0(x) \subset K(X,x) \quad \forall x \in X$.

Доказательство (от противного). Пусть существует направление $y \in G^0(x)$, не входящее в $K(X,x)$, т.е. для любой последовательности, фигурирующей в определении 5, найдется подпоследовательность $(\tau_t, y^t) \to (+0, y)$: $x + \tau_t y^t \notin X$, следовательно, $\forall t \ \exists$ индекс j, такой что $g_j(x + \tau_t y^t) > 0$. Возможных индексов — конечное число, а различных t бесконечно много, значит, найдется ограничение, пусть i-е, которое нарушается бесконечное число раз. Рассмотрим соответствующую подпоследовательность $\{t_k\}$: $g_i(x+\tau_{t_k} y_{t_k}) > 0$ и, устремляя $k \to \infty$, получим, что $g_i(x) \geq 0$. Но из условия $x \in X$ справедливо обратное неравенство, откуда следует равенство, т.е. $i \in J(x)$. Однако для этого i по определению 4 будем иметь $\langle \nabla g_i(x), y\rangle \doteq$

$$\doteq \lim_{(\tau, y') \to (+0, y)} \frac{g_i(x + \tau y') - g_i(x)}{\tau} = \lim_{k \to \infty} \frac{g_i(x + \tau_{t_k} y_{t_k}) - g_i(x)}{\tau_{t_k}} \geq 0.$$

Пришли к противоречию с $y \in G^0(x)$.

Отсюда получаем следующее *условие регулярности*:

$$G(x) = \overline{G^0(x)}. \tag{7}$$

Здесь и далее черта над множеством обозначает его замыкание.

Утверждение 6. В сделанных предположениях условие (7) обеспечивает регулярность X в точке x.

Для доказательства достаточно заметить, что множество $K(X,x)$ является замкнутым, а включение $G^0(x) \subset K(X,x)$ приводит к $\overline{G^0(x)} \subseteq K(X,x)$ после взятия операции замыкания.

Утверждение 7. Достаточным для (7) является

$$G^0(x) \neq \emptyset. \tag{8}$$

Доказательство. Из (8) для алгебраической суммы G и G^0 следует: $G + G^0 \subseteq G^0$, т.е. $\overline{G + G^0} \subseteq \overline{G^0}$, а $\overline{G^0} \supseteq \bar{0}$ дает $G + \overline{G^0} \supseteq G$. И из линейности оператора замыкания и замкнутости G получаем (7).

Для выпуклых X выполнение (8) и, следовательно, регулярность (в любой точке) ограничений (3) гарантируется *условием Слэйтера* ($\exists x' \in X : \quad g_i(x') < 0 \quad \forall i \in M$). Линейные ограничения всегда регулярны (множество G совпадает с контингентным конусом), хотя условие Слэйтера или (8) для них может не выполняться.

Другие типы условий регулярности, а также условия регулярности для смешанных систем ограничений равенств и неравенств см. в [4–6]. В частности, классическим условием регулярности для ограничений-равенств является линейная независимость градиентов ограничений в экстремальной точке.

УПРАЖНЕНИЕ 8. Получить теорему двойственности ЛП как следствие теоремы Куна-Таккера (для случая озЛП).

Условия оптимальности служат основным инструментом теоретического исследования задач условной оптимизации. Чтобы численно (приближенно) найти условный экстремум с их помощью, применяют методы безусловной оптимизации для поиска седловой точки функции Лагранжа или комбинируют штрафную функцию с функцией Лагранжа для получения точного гладкого штрафа. К сожалению, все эти методы останавливаются в первом попавшемся локальном экстремуме. Глобальный оптимум можно искать, перебирая локальные оптимумы, но для задач неодномерной минимизации не понятно, как находить все локальные оптимумы. Некоторые из существующих подходов к решению задач глобальной оптимизации приводятся в следующем параграфе.

4. СПОСОБЫ РЕШЕНИЯ ПЕРЕБОРНЫХ ЗАДАЧ

Литература:

2. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. М.: Мир, 1985.

6. Мину М. Математическое программирование. М.: Наука, 1990.

§10. Глобальная оптимизация. Метод ветвей и границ

Случайный и последовательный перебор. Метод ветвей и границ в глобальной оптимизации. Описание и стратегии метода.

1. Как уже отмечалось ранее, задачи глобальной оптимизации (т.е. в невыпуклом случае задачи оптимизации вообще) являются переборными. Переборные алгоритмы не эффективны (в расчете на худшую задачу), поэтому успех в решении каждой конкретной задачи существенным образом зависит от способа организации перебора. Если мы готовы оставить возможность или невозможность решения нашей задачи на волю случая, то естественно использовать случайный перебор. Этот способ перебора обычно является самым простым и, как правило, экономит память. Для задачи поиска глобального минимума ему соответствует следующий *метод Монте-Карло*.

Пусть решается задача (1) из §8, где (для упрощения изложения) множество ограничений X — единичный n-мерный куб. Выбираем в соответствии с равномерным распределением на X случайные точки x^t, в которых вычисляем значение целевой функции, запоминаем текущее наименьшее значение — *рекорд* — и реализующую его точку. Тогда $\forall \varepsilon > 0$ вероятность

$$\mathbf{P}(|\min_t f(x^t) - f^*| > \varepsilon) \to 0 \text{ при } t \to \infty.$$

Сходимость такого метода будет довольно медленной. При этом не известно, на каком расстоянии от точки минимума находится полученная реализация.

Сузим класс рассматриваемых задач (1), предположив вдобавок к предыдущему, что функция цели липшицева на X с константой L: $f \in \mathrm{Lip}(X, L)$, т.е. $|f(x) - f(x')| \le L\|x - x'\|$ $\forall x, x' \in X$. И не рассчитывая найти точное решение, зададимся подходящим $\varepsilon > 0$ с

целью поиска ε-приближенного решения x^ε : $f(x^\varepsilon) \le f(x^*) + \varepsilon$. (На близость x^ε и x^* никаких условий не накладывается.)

Теперь мы можем применять методы детерминированного перебора. Пассивный (не использующий при выборе очередной точки информацию, полученную для предыдущих) способ поиска приводит к полному перебору: разобьем X на подкубы X^j так, чтобы $\forall x, x' \in X^j : \|x - x'\| \le \delta \doteq \varepsilon/L$, в каждом X^j берем произвольную точку x^j и полагаем

$$f(x^\varepsilon) \doteq \min_j f(x^j).$$

Очевидно, x^ε и есть искомое ε-приближенное решение. (Действительно, $\forall j,\ \forall x \in X^j\ \ f(x^\varepsilon) \le f(x^j) \le f(x) + \varepsilon$ по условию Липшица, и, в частности, для $x = x^*$ имеем $f(x^\varepsilon) \le f(x^*) + \varepsilon$ — соответствие с определением.) Однако сторона каждого j-го подкуба равна $\varepsilon/(L\sqrt{n})$, а всего подкубов и, следовательно, вычислений значений целевой функции будет $(L\sqrt{n}/\varepsilon)^n$ в любом случае, что́ не мыслимо даже для десятка переменных. Поэтому разрабатываются методы последовательного перебора, позволяющие учитывать уже вычисленные значения и адаптироваться к нехудшему случаю.

Предположим, что уже вычислены значения функции в точках $x^1, \ldots, x^{j-1}$ и рекордным оказалось значение $f(x^r) = R$. Тогда, если $f(x^j) < f(x^r)$, то обновляем рекорд $r := j,\ R := f(x^j)$, а если $f(x^j) > f(x^r)$, то можно не вычислять значений функции на множестве $T_j(R) \doteq \{x \in X\colon \|x - x^j\| \le (f(x^j) - R)/L\}$, так как это не даст нового рекорда (ибо $\forall x \in T_r\ \ f(x^j) - f(x) \le L\|x - x^j\| \le f(x^j) - f(x^r)$, т.е. $f(x) \ge f(x^r) = R$, и значит, среди них нет глобально-оптимального решения). Обновление рекорда в принципе позволяет "отбросить" аналогичные множества $T_i(R)$ для $i = 1, \ldots, j-1$.

Естественно, в $T_i,\ T_j$ могут попасть и точки x^k с уже вычисленным значением $f(x^k)$ (которые таким образом вычислялись зря). Поэтому хотелось бы так организовать перебор, чтобы по возможности уменьшить число подобных "зряшных" вычислений. К сожалению, оптимальной стратегии организации перебора для многомерных задач нет. Использование случайных точек x^i приводит к проблеме хранения и обновления сложного множества $\cup T_i(R)$ заведомо не оптимальных точек. Метод послойного перебора дает возможность сокращения лишь по одной переменной. Для задач большой размерности

предлагается (различными авторами) следующий метод перебора по схеме *ветвей и границ*.

2. *Метод ветвей и границ (МВГ) для глобальной минимизации.* Пусть x^1 — центр куба X. Вычисляем $f(x^1)$ и присваиваем это значение рекорду $R := f(x^1)$. Разбиваем куб на 2^n одинаковых подкубов X^{1i} со стороной $1/2$ и вычисляем значения целевой функции в их центрах: $f(x^{1i})$, $i = 1, \ldots, 2^n$, обновляя по ходу вычислений значение рекорда $R := \min_i f(x^{1i})$. Проверяем выполнение условия $X^{1i} \subseteq T_{1i}(R)$ для $i = 1, \ldots, 2^n$ и отбрасываем соответствующие подкубы. Каждый из оставшихся разбиваем на 2^n одинаковых подкубов X^{2ij} со стороной $1/4$ и поступаем, как прежде. На любом шаге у нас формируется множество $\mathbf{K}$ "кубиков" со сторонами 2^{-l}, $l \geq 2$, целое. Правило выбора очередного кубика для разбиения называется *правилом ветвления* — возможные варианты приводятся ниже. Кубики со стороной не больше $\varepsilon/(L\sqrt{n})$ исключаются из множества $\mathbf{K}$ — дробление кубика заканчивается. Также исключаются кубики, попавшие в множество $T_k(R)$ (с индексом k — номером кубика) для текущего значения рекорда, — *правило отсечения ветвей*. Рекорд обновляется при получении меньшего значения целевой функции (*правило получения границ, т.е. оценок*). Значения целевой функции вычисляются в центре каждого нового подкубика, включаемого в $\mathbf{K}$ после разбиения выбранного для этого кубика. Алгоритм останавливается, когда $\mathbf{K}$ пусто.

Указанная терминология и название метода определяются тем, что визуально данная схема перебора представляется в виде графа-дерева, корневая вершина которого соответствует кубу X, вершины первого яруса — подкубам X^{1i}, вершины второго яруса — кубикам X^{2ij}, подсоединенным к своим *порождающим* вершинам X^{1i} 1-го яруса, и т.д. Если кубик исключается из $\mathbf{K}$, его вершина *закрывается* — из нее не будут идти ветви на следующий ярус. Если кубик еще не включен в $\mathbf{K}$, его вершина еще *не раскрыта*. Порядок закрытия вершины определяется правилом отсечения (своим для каждой массовой задачи — см. также в §11), порядок раскрытия — правилом ветвления (своим для каждой индивидуальной задачи). Различают два вида правил ветвления по типу построения дерева решений (выбора вершин для раскрытия): "в ширину", когда сначала раскрываются

все вершины одного яруса до перехода к следующему, и "в глубину" — всякий раз раскрывается лишь одна (обычно с лучшим значением рекорда) вершина на ярусе до конца ветви. На практике реализуют некоторую смесь, например, первое правило, пока хватает машинной памяти (в **K** не слишком много элементов), затем переключаемся на второе. Предпочтительность той или иной стратегии ветвления оценивается каждым вычислителем по-своему, исходя из главной задачи метода ветвей и границ — быстрее получить лучший рекорд, чтобы отсечь больше ветвей.

В рассматриваемой задаче есть хороший способ улучшения рекорда — локальная оптимизация (см. в §8). Ее имеет смысл проводить из текущей точки, в которой произошло обновление рекорда, например, делая несколько шагов градиентного метода. При этом расположение кубиков менять не надо, просто увеличивается шанс сокращения перебора (отбрасывания бóльших кубиков).

Отметим, что в худшем случае $f = const$ ($\cup T_i = \emptyset$) — не удается отбросить ни одной точки x — и приходим к полному перебору; т.е. указанная в п.1 экспоненциальная оценка точна на классе всех липшицевых функций.

§11. Целочисленное линейное программирование (ЦЛП)

Отличие задач ЦЛП и ЛП: существенная нелинейность ограничений типа целочисленности. Неэффективность округления решения ЛП до ближайшего целого. Случай вполне унимодулярной матрицы ограничений. МВГ в ЦЛП. МВГ для булева линейного программирования (БЛП).

1. По-видимому, наиболее важным классом задач глобальной оптимизации являются задачи ЦЛП. Эти задачи формулируются как задачи ЛП с дополнительным ограничением целочисленности переменных. Последнее ограничение, какими бы способами от него ни избавляться, "портит" свойство выпуклости (и полиномиальности) задачи ЛП. Например, выразив условие целочисленности в форме ограничений неравенств, рассмотренной в доказательстве утверждения 1 §8, и сняв их методом штрафов, придем к задаче глобальной оптимизации, имеющей не меньше локальных экстремумов, чем вариантов для целочисленных переменных в исходной ЦЛП. Поэтому

на практике удается решать задачи ЦЛП только небольшой размерности или с ограничениями целочисленности не на все, а лишь на несколько переменных.

Существует частный класс задач ЦЛП, в которых ограничение целочисленности оказывается несущественным.

Определение 1. Матрица называется *вполне унимодулярной*, если определитель любой ее невырожденной квадратной подматрицы равен по модулю 1.

Утверждение 1. Если матрица ограничений разрешимой задачи ЛП с целыми коэффициентами вполне унимодулярна, то у нее существует целочисленное решение.

Доказательство очевидно из принципа граничных решений (§5) и правила Крамера (см. доказательство теоремы 1 §5).

Утверждение 2. Матрица A вполне унимодулярна тогда и только тогда, когда для любого целочисленного вектора b все вершины многогранника $Ax \le b,\ x \ge \bar{0}$ являются целочисленными.

Доказательство в одну сторону аналогично предыдущему, в другую сторону см. ссылку в [2, с. 333].

Таким образом, вполне унимодулярными матрицами ограничений в принципе ограничивается класс задач ЦЛП, эквивалентных ЛП и, следовательно, допускающих эффективное решение. Отметим, что указанный класс, хотя и чрезвычайно узок с формальной точки зрения (элементами матрицы A могут быть только 0, 1 и -1, причем по большей части 0), соответствует достаточно широкому классу практических задач оптимизации на графах и сетях (одно- и двухпродуктовые сети, двудольные графы и т.п.).

Приведем без доказательства еще одно полезное утверждение, позволяющее в некоторых случаях получать приближенное решение ЦЛП путем решения ЛП.

Утверждение 3. Если все элементы симплекс-таблицы a_{ij}, b_i, c_j натуральные числа, то для любого решения x^0 задачи ЛП

$$\max_{Ax \le b,\ x \ge \bar{0}} \langle c, x \rangle$$

вектор $\lfloor x^0 \rfloor$, составленный из компонент $\lfloor x_j^0 \rfloor$, будет допустимым в данной задаче. При этом для решения x^* соответствующей задачи

ЦЛП

$$\max_{Ax \le b,\ x \in \mathbf{Z}^n_+} \langle c, x \rangle$$

очевидна оценка $|\langle c, \lfloor x^0 \rfloor \rangle - \langle c, x^* \rangle| \le \langle c, \bar{1} \rangle$.

Условие положительности исходных данных выполняется для некоторых экономических задач. Такой же результат можно получить для ряда многопродуктовых потоковых задач на сетях и других линейных задач максимизации с положительным c, в которых допустимое множество вместе с любой точкой x содержит и все x' с компонентами $x'_j \in [0, x_j]$. Однако поиск x^* по $\lfloor x^0 \rfloor$ может потребовать перебора 2^n вариантов округления компонент x^0.

К сожалению, в общем случае и перебора всех возможных вариантов округления компонент решения непрерывной задачи ЛП оказывается недостаточно для получения решения ЦЛП (например, при $n = 2$, если для положительного c рассмотреть систему ограничений $-9x_1 + 10x_2 \le 0,\ -8x_1 + 10x_2 \le -1$). Таким образом, поиск решения ЦЛП может потребовать очень большого перебора целочисленных точек, и возникает та же, что и в §10, задача организации перебора с целью попытаться его сократить в случае не самой плохой задачи. Одним из достаточно употребительных методов перебора здесь является метод ветвей и границ, который для ЦЛП будет рассмотрен в п.2. Другие методы см. в [2,6].

2. *Метод ветвей и границ для ЦЛП.* Рассматривается задача

$$\max_{z \in \mathbf{Z}^n:\ Az \le b} \langle c, z \rangle, \tag{1}$$

решением которой является целочисленный вектор z^*.

В корневой вершине метода вместо задачи (1) решается озЛП

$$\max_{x \in \mathbf{R}^n:\ Ax \le b} \langle c, x \rangle, \tag{2}$$

решением которой является вектор x^0. Если x^0 оказался целочисленным, то $z^* := x^0$ — решение задачи (1) закончено. Иначе $\exists x^0_j \notin \mathbf{Z}$ и осуществляем ветвление по j-й компоненте следующим образом.

Из вершины выходят две ветви, и на новом ярусе к ограничениям озЛП, решаемой в порождающей вершине, добавляется ограничение

$x_j \le \lfloor x_j^0 \rfloor$ для 1-й ветви или $x_j \ge \lceil x_j^0 \rceil$ для 2-й ветви. Значение максимума в исходной задаче ЦЛП (1), очевидно, равно максимальному из значений подзадач ЦЛП на каждой ветви. Но, как и ранее, вместо подзадачи ЦЛП рассматривается подзадача без ограничения целочисленности. Такая озЛП и решается в очередной порожденной вершине в случае ее раскрытия, обозначим решение через x^k.

Если x^k — целочисленное, то вершина закрывается, а значение $\langle c, x^k \rangle$ функции цели сравнивается с рекордом для его обновления или, по первому разу, присваивается рекорду, и точка x^k — допустимая точка в задаче (1) — запоминается. После получения рекорда может быть закрыта любая раскрытая вершина, для которой оптимальное значение целевой функции окажется меньше рекорда. Действительно, поскольку максимум по большему множеству не меньше максимума по меньшему, то значение озЛП дает оценку сверху (*границу*) значения соответствующей целочисленной подзадачи, и когда верхняя оценка не превышает рекорда, бессмысленно пытаться увеличить рекорд на данной ветви.

Другим случаем закрытия вершины (отсечения ветви) является неразрешимость поставленной озЛП и, следовательно, той же подзадачи ЦЛП.

Если x^k — нецелочисленное, то $\exists x_i^k \notin \mathbf{Z}$, и осуществляем ветвление по i-й компоненте описанным выше способом. Процедура заканчивается после закрытия всех вершин, тогда значение (1) равно текущему рекорду, либо рекорд остался неопределенным и задача (1) не имеет решения.

Выбор стратегии ветвления в ЦЛП играет не меньшую роль, чем в глобальной оптимизации. Отсутствие рекорда приводит к лишнему перебору, но процедура ветвления “в глубину” может вместо рекорда дать несовместную систему ограничений. Кроме того, для нескольких нецелых компонент x^k не понятно, по какой из них лучше осуществлять ветвление: по новой, которая не рассматривалась на предыдущих ярусах, или сначала перебрать все допустимые целые значения одной из компонент (по аналогии с БЛП — см. ниже). Последняя стратегия имеет смысл при наличии двусторонних ограничений на переменные.

3. *Метод ветвей и границ для БЛП.* Частным случаем задачи

(1) ЦЛП является задача БЛП

$$\max_{z \in \mathbf{B^n}:\ Az \le b} \langle c, z \rangle, \tag{3}$$

решение которой — вектор z^0 из булева куба.

Из результатов §2 (утверждения 8) вытекает **NP**-трудность БЛП и, следовательно, правомерность использования переборных схем для решения (3). В §12 будет показана схема динамического программирования для БЛП с неотрицательными коэффициентами, а для произвольных задач (3) применима схема предыдущего пункта, которая несколько упрощается за счет дополнительного ограничения $0 \le z_i \le 1$, превращающего ЦЛП в БЛП. А именно, после замены $\mathbf{Z^n}$ на $\mathbf{B^n}$, при ветвлении в новые подзадачи добавляется вместо ограничений неравенств условие равенства 0 (для одной ветви) или 1 (для другой) той переменной, по которой осуществляется ветвление. Таким образом указанная переменная становится булевой во всех нижних ярусах, т.е. по ней не придется вновь проводить ветвление, а значит, на n-м ярусе решение (3) будет закончено. Число раскрываемых вершин (или решений подзадач ЛП) при этом не превысит 2^{n+1}, что, конечно, тоже немало, но заметно меньше, чем для ЦЛП (сравнимо со случаем, предусмотренным утверждением 3).

§12. Метод динамического программирования (ДП)

Теоретические основы ДП. Общая схема метода. Метод ДП для БЛП с неотрицательными коэффициентами. Связь с МВГ.

1. Еще одной традиционно используемой схемой перебора является *метод динамического программирования (ДП)*. Один пример алгоритма ДП приводился в §4, где этод метод позволил построить псевдополиномиальный алгоритм решения задачи о рюкзаке. Вообще говоря, подобные алгоритмы и надеются получить путем применения схемы ДП. Однако ДП можно использовать не для произвольных оптимизационных задач. Класс подходящих задач опишем далее.

Определение 2. Функция f называется *разделяемой* на f_1 и f_2, если она представима в виде

$$f(x, y) = f_1(x, f_2(y)). \tag{4}$$

Определение 3. Функция f называется *разложимой* на f_1 и f_2, если она разделяема на f_1, f_2 и функция f_1 монотонно не убывает по последнему аргументу.

Теорема 1 *(оптимальности для разложимых функций).*

$$\min_{(x,y)} f(x,y) = \min_x f_1(x, \min_y f_2(y)),$$

и точно так же для max.

Доказательство проведем для случая минимума.
(Равенство будет вытекать из пары противоположных неравенств.)

По определению минимума $\min\limits_{x,y} f_1(x, f_2(y)) \le f_1(x^0, f_2(y^0)) \quad \forall x^0, y^0$

и, следовательно, для $y^0 := \arg\min\limits_y f_2(y)$, $x^0 := \arg\min\limits_x f_1(x, f_2(y^0))$,

что доказывает неравенство "$\le$". Аналогично, в силу неубывания f_1 по последнему аргументу, $f_1(x', \min_y f_2(y)) \le f_1(x', f_2(y')) \quad \forall x', y'$.

Положим $y' := \arg\min\limits_y f_1(x', f_2(y))$, $x' := \arg\min\limits_x \{\min\limits_y f_1(x, f_2(y))\}$.

Поскольку повторный min равен двойному, в правой части получили $\min_{x,y} f(x,y)$, чем доказали и неравенство "$\ge$".

Для задачи условной оптимизации теорема оптимальности для разложимых функций переписывается следующим образом:

$$\min_{(x,y)\in\Omega} f(x,y) = \min_{x:\ Y(x)\neq\emptyset} f_1(x, \min_{y\in Y(x)} f_2(y)), \tag{5}$$

где $Y(x) = \{y|\ (x,y) \in \Omega\}$. Указанная теорема используется для понижения размерности оптимизационных задач и в методе ДП.

Для начала рассмотрим задачу оптимизации, записанную в виде

$$f^* = \min_{g(x,y)\in\mathcal{E}_T\subset\mathbf{R^m}} f(x,y), \quad x \in X \subseteq \mathbf{R^n},\ y \in Y(x) \subseteq \mathbf{R^k}. \tag{6}$$

Здесь $\mathcal{E}_T$ называется множеством *терминальных состояний* системы по ассоциации с динамическими системами управления, для оптимизации которых было изобретено ДП. Например, для $g = (g_1, \ldots, g_m)$,

если ограничения задачи заданы в форме $g_i(x,y) \le 0 \quad \forall i = \overline{1,m}$, то $\mathcal{E}_{\mathcal{T}} = \mathbf{R}^{\mathbf{m}}_{-}$. Пусть f разложима (4), а g разделяема:

$$g(x,y) = h_2(y, h_1(x)), \quad h_1\colon \mathbf{R^n} \to \mathbf{R^m}, \quad h_2\colon \mathbf{R^{k+m}} \to \mathbf{R^m},$$

тогда введем $\forall x, y \quad E = h_1(x), \quad E' = h_2(y,E)$ и вычисляем g как $g(x,y) = E'$. Функции $h_1,\ h_2$ называются *функциями перехода*, векторы $E,\ E'$ — *состояниями системы*. Множество всех возможных состояний системы обозначается $\mathcal{E}$ и формально задается так:

1) $\mathcal{E} \supset h_1(X) \doteq \{h_1(x) |\ x \in X\}$,

2) $\forall E \in \supset h_1(X) \quad \{h_2(y,E) | y \in Y(X)\} \subset \mathcal{E}$

(множество в качестве аргумента означает объединение по всем аргументам из этого множества).

Рассмотрим для (6) семейство задач поиска

$$F_2(E) = \min_{y:\ h_2(y,E) \in \mathcal{E}_{\mathcal{T}}} f_2(y),$$

которые нужно решать $\forall E \in \mathcal{E}$. По теореме оптимальности

$$f^* = \min_{x \in X} f_1(x, F_2(h_1(x))).$$

В результате задача (6) свелась к последовательности $|\mathcal{E}| + 1$ оптимизационных задач меньшей размерности.

В методе ДП данная процедура применяется рекурсивно к задаче

$$F^* = \min_{g(x_1,\ldots,x_n) \in \mathcal{E}_{\mathcal{T}} \subset \mathbf{R^m}} f(x_1, \ldots, x_n) \tag{7}$$

для сведения к семейству одномерных задач следующим образом.

Пусть f последовательно разложима, т.е.

$$\begin{aligned} f(x_1,\ldots,x_n) &= f_1(x_1, \hat{f}_2(x_2,\ldots,x_n)), \\ \hat{f}_2(x_2,\ldots,x_n) &= f_2(x_2, \hat{f}_3(x_3,\ldots,x_n)), \\ \ldots\ldots\ldots\ldots\ldots\ldots &\quad \ldots\ldots\ldots\ldots\ldots\ldots \\ \hat{f}_{n-1}(x_{n-1}, x_n) &= f_{n-1}(x_{n-1}, f_n(x_n)), \end{aligned}$$

и все f_i монотонно не убывают по 2-му аргументу. Пусть g последовательно разделяема, т.е. $\exists \mathcal{E} \subset \mathbf{R^m}$, $\exists$ функции перехода $h_1, h_2, \ldots, h_n$:

$\forall x \in X = \otimes X_i \;\; g(x) = h_n(x_n, E_{n-1})$, $E_{n-1} = h_{n-1}(x_{n-1}, E_{n-2}), \ldots,$ $E_2 = h_2(x_2, E_1)$, $E_1 = h_1(x_1)$ и $E_i \in \mathcal{E} \;\; \forall i = \overline{1, n-1}$.

Обозначим $\forall i = \overline{2, n-1}, \quad \forall E \in \mathcal{E}$ через $\hat{h}_i(x_i, x_{i+1}, \ldots, x_n, E)$ функцию, определяемую рекуррентно равенствами: $E'_i = h_i(x_i, E)$, $E'_{i+1} = h_{i+1}(x_{i+1}, E'_i), \ldots, E'_n = h_n(x_n, E'_{n-1}) \doteq \hat{h}_i(x_i, x_{i+1}, \ldots, x_n, E)$. Заметим, что в случае $E = E_{i-1}$: $\hat{h}_i(x_i, x_{i+1}, \ldots, x_n, E_{i-1}) = g(x)$ и $E'_j = E_j \;\; \forall j \geq i$. В сделанных обозначениях справедливо *возвратное соотношение* для ограничений

$$\hat{h}_i(x_i, x_{i+1}, \ldots, x_n, E) = \hat{h}_{i+1}(x_{i+1}, \ldots, x_n, h_i(x_i, E)). \tag{8}$$

Тогда по определению

$$F^* = \min_{x \in X:\; \hat{h}_2(x_2, \ldots, x_n, h_1(x_1)) \in \mathcal{E}_\mathcal{T}} f_1(x_1, \hat{f}_2(x_2, \ldots, x_n)),$$

и по теореме оптимальности

$$F^* = \min_{x_1 \in X_1} f_1(x_1, F_2(h_1(x_1))), \tag{9}$$

где $\forall E_1 \in \mathcal{E} \;\; F_2(E_1) \doteq \min_{(x_2, \ldots, x_n):\; \hat{h}_2(x_2, \ldots, x_n, h_1(x_1)) \in \mathcal{E}_\mathcal{T}} \hat{f}_2(x_2, \ldots, x_n) =$

$$\text{(из (8))} \quad = \min_{(x_2, \ldots, x_n):\; \hat{h}_3(x_3, \ldots, x_n, h_2(x_2, E_1)) \in \mathcal{E}_\mathcal{T}} f_2(x_2, \hat{f}_3(x_3, \ldots, x_n)) =$$

$$= \min_{x_2 \in X_2} f_2(x_2, F_3(h_2(x_2, E_1)))$$

(последнее равенство следует из (5) с $x = x_2, \;\; y = (x_3, \ldots, x_n)$),
и т.д., полагая минимум по пустому множеству равным $+\infty$, имеем

$$F_i(E) \doteq \min_{\hat{h}_i(x_i, x_{i+1}, \ldots, x_n, E) \in \mathcal{E}_\mathcal{T}} \hat{f}_i(x_i, x_{i+1}, \ldots, x_n) \;\; \forall i, E \tag{10}$$

— семейство задач, в которое "погрузили" (7),

$$F_i(E_{i-1}) = \min_{x_i \in X_i} f_i(x_i, F_{i+1}(h_i(x_i, E_{i-1}))) \;\; \forall E_{i-1} \in \mathcal{E} \tag{11}$$

— *возвратное (функциональное) уравнение ДП* $\forall i = \overline{2, n-1}$,

$$F_n(E) = \min_{x_n \in X_n:\; h_n(x_n, E) \in \mathcal{E}_\mathcal{T}} f_n(x_n). \tag{12}$$

Алгоритм ДП:

$\forall E \in \mathcal{E}$ вычисляем $F_n(E)$ из (12),

последовательно для $i = n-1, \ldots, 2$ определяем $F_i(E)$ из (11),(10),

затем F^* из (9).

Число шагов алгоритма (решений задач одномерной минимизации) будет порядка $n|\mathcal{E}|$. Таким образом метод ДП имеет смысл применять для задач с не очень большим числом состояний ($|\mathcal{E}|$ мало́).

2. Примерами разложимых функций могут служить min, max, сумма, произведение (с неотрицательными коэффициентами) и т.п. Исходно метод ДП использовался для оптимизации динамических систем, что нашло отражение в применяемой терминологии. Так, $\mathcal{E}$ соответствует физическому пространству состояний (возможных координат траектории движения), x_i — управлению в момент времени t_i, воздействие управления на траекторию определяется функцией перехода в следующее состояние, на конечное состояние наложены ограничения принадлежности к $\mathcal{E}_{\mathcal{T}}$, начальное состояние фиксировано; $f_i(x_i, E)$ — стоимость управления системой, находящейся в состоянии E, f — стоимость всей траектории $E_1, \ldots, E_{n-1}$.

Соотношение (11) означает минимизацию стоимости "хвоста" траектории в каждый момент времени, что согласуется с принципом оптимальности, сформулированным Р. Бэллманом: оптимальная политика управления такова, что для любого начального состояния и любых решений (по выбору управления), принятых на начальных шагах, оставшиеся решения образуют оптимальную политику, начинающуюся с состояния, возникшего в результате этих решений. (Отметим, что в случае строгой монотонности f таким образом можно получить любое решение, в случае нестрогой монотонности — хотя бы одно).

Проиллюстрируем применение метода ДП на примере решения задач БЛП с неотрицательными коэффициентами (элементами симплекс-таблицы). Итак, вернемся к задаче (3)

$$F^* = \max_{z \in \mathbf{B^n}:\ Az \le b} \langle c, z \rangle$$

в предположении $a_{ij},\ b_i,\ c_j \in \mathbf{Z}_+$. Обозначим через $\bar{a}_j$ j-й столбец

матрицы A. Рассмотрим семейство задач поиска

$$F_k(E) \doteq \max_{z:\ z_j \in \{0,1\}\ \forall j=k,\ldots,n} \sum_{j=k}^{n} c_j z_j$$

$$\sum_{j=k}^{n} \bar{a}_j z_j \le b - E,$$

где $E \in \mathcal{E} \doteq \{E \in \mathbf{Z}_+^{\mathbf{m}} \mid E_i \le b_i\ \forall i = \overline{1,m}\}$, $k = \overline{1,n}$.

Очевидно, $F^* = F_1(0)$. Возвратное уравнение в данном случае:

$$F_k(E) = \max\{F_{k+1}(E),\ c_k + F_{k+1}(E + \bar{a}_k)\},$$

$$F_n(E) = \begin{cases} c_n, & E \le b - \bar{a}_n, \\ 0 & \text{иначе.} \end{cases}$$

Находим $\forall E \in \mathcal{E}$ $F_n(E)$ и соответствующие $x_n(E)$, затем для $k = n-1, \ldots, 2$ определяем $F_k(E)$ и реализующие их $x_k(E)$ из возвратного уравнения, вычисляем $F_1(0)$, $x_1(0)$ и далее $x_2(E^1), \ldots, x_n(E^{n-1})$ в зависимости от того, какие состояния $E^1, \ldots, E^{n-1}$ были в конечном счете использованы при вычислении $F_1(0)$, если посмотреть по всем шагам алгоритма.

Число шагов предложенного алгоритма равно n и на n-м шаге рассматривается $\min\{(b_1+1)\cdot\ldots\cdot(b_m+1),\ 2^{n-1}\}$ состояний, на $(n-1)$-м — минимум из левой части (равной $|\mathcal{E}|$) и 2^{n-2} и т.п. Так что при больших b метод ДП решает примерно столько же задач, сколько МВГ в худшем случае, однако решаемые задачи здесь проще (проверка ограничений вместо ЛП). Подчеркнем, что процедура ДП не дает способов сокращения перебора, тогда как удачный выбор стратегии ветвления в МВГ (например, на основе имеющейся у вычислителя дополнительной информации или эвристических соображений) позволяет (хотя и не гарантированно) решать задачи большей размерности. Отметим также отсутствие ограничения неотрицательности коэффициентов для работы МВГ. В принципе, возможно комбинирование обеих схем (см. [6]).

Содержание