УДК 519.854.3

ЭКСПЕРИМЕНТАЛЬНЫЙ АНАЛИЗ МЕТОДА ОТСЕЧЕНИЯ ГОМОРИ ДЛЯ ЗАДАЧ ЦЕЛОЧИСЛЕННОГО ПРОГРАММИРОВАНИЯ

Никуленко М.А.

студент,

Российский университет транспорта (МИИТ),

Москва, Россия

Иванова А.П.

к.ф.-м.н., доцент

Российский университет транспорта (МИИТ),

Москва, Россия

Аннотация:

В статье представлено экспериментальное исследование метода отсечения Гомори для решения задач целочисленного линейного программирования. Разработан программный комплекс, включающий генератор тестовых задач с двумя переменными и тремя ограничениями, формирующими пятиугольную область допустимых решений, а также модуль реализации метода Гомори на основе симплекс-метода. Проведены вычислительные эксперименты, позволившие оценить среднее, минимальное и максимальное число отсечений, а также дисперсию этого показателя. Показано, что для подавляющего большинства задач требуется не более двух отсечений, однако при увеличении коэффициентов ограничений растёт разброс сложности задач и максимальное число итераций. Полученные результаты подтверждают высокую практическую эффективность метода Гомори для задач малой размерности и могут быть использованы при настройке гибридных алгоритмов дискретной оптимизации.

Ключевые слова: целочисленное линейное программирование, метод отсечения Гомори, симплекс-метод.

EXPERIMENTAL ANALYSIS OF THE GOMORRI CUT METHOD FOR INTEGER PROGRAMMING PROBLEMS

Nikulenko M.A.

student,

Russian University of Transport (MIIT),

Moscow, Russia

Ivanova A.P.

Ph.D., Associate Professor

MIREA – Russian Technological University,

Russian University of Transport (MIIT),

Moscow, Russia

Abstract: The article presents an experimental study of the Gomory cut-off method for solving integer linear programming problems. A software package has been developed, which includes a test problem generator with two variables and three constraints forming a pentagonal region of admissible solutions, as well as a module for implementing the Gomory method based on the simplex method. Computational experiments have been carried out, which allowed to estimate the average, minimum and maximum number of cuts, as well as the variance of this indicator. It is shown that for the vast majority of problems, no more than two cutoffs are required, but as the constraint coefficients increase, the range of problem complexity and the maximum number of iterations increase. The results obtained confirm the high practical efficiency of the Gomory method for low-dimensional problems and can be used in the configuration of hybrid discrete optimization algorithms.

Keywords: integer linear programming, Gomory's cut method, and the simplex method.

Введение

Задачи целочисленного линейного программирования (ЗЦЛП) занимают центральное место в современной прикладной математике, исследовании

операций и компьютерных науках. Необходимость учёта целочисленности переменных возникает при моделировании многих практических задач: от распределения ресурсов и планирования производства до логистики, размещения объектов и построения расписаний. В отличие от задач линейного программирования (ЗЛП), эффективно решаемых, например, симплекс-методом [1, 2], наличие требования целочисленности коренным образом меняет структуру задачи и приводит к резкому росту вычислительной сложности. Многие задачи целочисленного программирования являются NP-трудными, что делает разработку и анализ эффективных методов их решения одной из ключевых задач дискретной оптимизации [3].

Одним из первых методов решения задач целочисленного линейного программирования является метод отсечения Гомори. Предложенный Ральфом Гомори в конце 1950-х годов, этот метод основан на элегантной идее последовательного уточнения области допустимых решений [4, 5]. На каждой итерации к исходной системе линейных ограничений добавляется новое, специальным образом сконструированное неравенство – отсечение Гомори. Это отсечение отсекает текущее оптимальное, но нецелочисленное решение задачи, не исключая при этом ни одного допустимого целочисленного вектора. Итеративный процесс продолжается до тех пор, пока оптимальное решение не станет целочисленным.

Настоящая работа посвящена экспериментальному исследованию количества отсечений, генерируемых методом Гомори при решении случайным образом сгенерированных ЗЦЛП. В рамках исследования была разработана программная реализация, объединяющая симплекс-метод и алгоритм построения правильного отсечения Гомори, а также генератор тестовых задач. Основная цель работы – провести вычислительные эксперименты для оценки среднего числа отсечений и проанализировать зависимость этого показателя от ключевых характеристик решаемых задач. Результаты исследования позволяют сделать выводы о практической сходимости метода Гомори и могут быть полезны при настройке гибридных алгоритмов целочисленной оптимизации.

1. Постановка задачи

Рассмотрим ЗЦЛП в общем виде [3]:

$$f(x)=\sum_{j=1}^{n}c_j x_j \to \max, \quad (1)$$

$$\sum_{j=1}^{n}a_{ij}x_j \le b_j,\ i=1,2,...,m, \quad (2)$$

$$x_j \ge 0,\ j=1,2,...,n, \quad (3)$$

$$x_j \in \mathrm{Z},\ j=1,2,...,n. \quad (4)$$

Метод отсечения основан на следующем: сначала решается задача (1)-(3) без условия целочисленности (4). Если её решение является целым, то оптимальное решение задачи (1)-(4) получено, в противном случае к системе ограничений (2) добавляется новое неравенство, которое должно обладать тремя свойствами: должно быть линейным, должно отсекать найденный нецелочисленный план, не должно отсекать ни одного целочисленного плана. Дополнительное неравенство, обладающее перечисленными тремя свойствами, называется правильным отсечением Гомори. Далее переходим к решению новой задачи с добавленным ограничением. Процесс заканчивается, когда очередной план оказывается целочисленным.

Пусть на некотором шаге симплекс-метода получено оптимальное решение, в котором базисные переменные выражаются через свободные следующим образом:

$$\begin{cases} x_1=\beta_1-\alpha_{1,m+1}x_{m+1}-...-\alpha_{1,n}x_n, \\ ..., \\ x_m=\beta_m-\alpha_{m,m+1}x_{m+1}-...-\alpha_{m,n}x_n, \end{cases}$$

$$x=(\beta_1,\beta_2,...,\beta_m,0,...,0).$$

Предположим, что некоторое β_i не является целым, тогда можно доказать, что правильным отсечением является неравенство [6,7]:

$$\{\beta_i\}-\{\alpha_{i,m+1}\}x_{m+1}-...-\{\alpha_{i,n}\}x_n \le 0,$$

где $\{z\}$ – дробная часть числа z ($\{z\} = z - [z]$, где $[z]$ – целая часть числа z).

Далее неравенство (5) заменяется на уравнение путём введения дополнительной переменной $x_{n+1} \geq 0$:

$$\{\beta_i\} - \{\alpha_{i,m+1}\}x_{m+1} - \ldots - \{\alpha_{i,n}\}x_n + x_{n+1} = 0.$$

2. Вычислительный эксперимент

2.1. Модуль генерации задач

Для проведения вычислительного эксперимента необходимо обеспечить возможность генерации тестовых задач с контролируемыми параметрами. Поскольку исследование направлено на анализ количества отсечений, а не на проверку работоспособности метода в общем случае, было принято решение использовать задачи с двумя переменными. Это позволяет, во-первых, визуализировать область допустимых решений и проверять корректность работы алгоритма графически, а во-вторых, упрощает анализ геометрической интерпретации отсечений.

Сформулируем требования к генерируемым задачам. Генерируемая задача целочисленного линейного программирования должна удовлетворять следующим условиям:

– $n = 2$ – две переменные ($x_1 \geq 0$, $x_2 \geq 0$), что соответствует плоскости для графического анализа,

– $m = 3$ – три линейных ограничения вида (2), при этом прямые, соответствующие этим ограничениям должны пересекаться таким образом, чтобы область допустимых решений представляла собой пятиугольник. Такая форма выбрана намеренно: она достаточно сложна, чтобы процесс отсечений был содержательным.

Коэффициенты целевой функции $c_j > 0$ генерируются случайным образом из небольшого диапазона (чтобы избежать слишком больших чисел).

Опишем алгоритм генерации пятиугольника.

Прямая генерация случайных ограничений, образующих выпуклый пятиугольник в первой четверти плоскости $x_1 0 x_2$, является нетривиальной

задачей. Простой перебор случайных прямых часто приводит к пустой или неограниченной области допустимых решений.

Для гарантированного получения области требуемой формы был разработан алгоритм, основанный на вершинном задании пятиугольника. Идея заключается в том, чтобы вначале сгенерировать координаты пяти вершин, а затем по ним построить уравнения прямых, содержащих стороны пятиугольника. Заметим, что координаты вершин генерируются как псевдослучайные целые равномерно распределённые числа в заданных диапазонах.

Обозначим вершины пятиугольника в порядке обхода против часовой стрелки как *A, B, C, D, E*. С учётом требований неотрицательности переменных и расположения в первой четверти координатной плоскости, фиксируются следующие особенности:

Вершина *A* всегда находится в начале координат: $A(0,0)$.

Вершина *B* лежит на положительной полуоси x_1: $B(x_B,0)$, где $x_B > 0$.

Вершина *E* лежит на положительной полуоси x_2: $E(0,y_E)$, где $y_E > 0$.

Вершины *C* и *D* являются внутренними вершинами пятиугольника и имеют положительные координаты: $C(x_C,y_C)$ и $D(x_D,y_D)$, где $x_C, y_C, x_D, y_D > 0$.

Для того чтобы гарантированно получить пятиугольник, необходимо правильно определить взаимное расположение вершин. Ключевую роль играет отрезок *CD* (будущая сторона пятиугольника) и положение вершин *B* и *E* относительно него.

Шаг 1. Генерация вершин *C* и *D*.

Чтобы пятиугольник имел вытянутую форму, вершины *C* и *D* генерируются в противоположных углах области. В данной работе используются следующие диапазоны (выбор обусловлен стремлением получить пятиугольник, пригодный для визуализации):

Вершина *C*: $x_C \in [14,20]$, $y_C \in [1,7]$.

Вершина *D*: $x_D \in [1,7]$, $y_D \in [14,20]$.

Шаг 2. Построение прямой *CD* и определение положения вершин *B* и *E*.

Соединив точки *C* и *D*, получим прямую, которая будет являться одним из ограничений. Уравнение этой прямой имеет вид

$$y = k_{CD}x + b_{CD}, \quad \text{где} \quad k_{CD} = \frac{y_D - y_C}{x_D - x_C}, \quad b_{CD} = y_C - k_{CD}x_C.$$

Шаг 3. Определение вспомогательной точки.

Для определения координат вершины *B* сначала генерируется вспомогательная точка (x_V,y_V), которая будет лежать выше прямой *CD* и между проекциями точек *D* и *C* на ось абсцисс. Координаты вспомогательной точки генерируются в интервалах $[x_D + 1, x_C - 1]$ и $[k_{CD}x_B + b_{CD} + 1, y_D - 1]$ соответственно, что гарантирует расположение точки выше отрезка *CD*.

Шаг 4. Генерация вершины B.

Вершина *B* лежит на оси x_1. Строим прямую через вспомогательную точку (шаг 3) и через точку *C*. Находим пересечение этой прямой с осью x_1. Это и будет координата точки *B*, а получившаяся прямая – новым ограничением.

Шаг 5. Генерация вершины *E*.

Вершина *E* лежит на оси x_2. Строим прямую через вспомогательную точку (шаг 3) и через точку *D*. Находим пересечение этой прямой с осью x_2. Это и будет координата точки *E*, а получившаяся прямая – новым ограничением.

После того как координаты всех пяти вершин определены, три искомых ограничения строятся как прямые, проходящие через следующие пары вершин:

– ограничение 1: сторона *BC*,

– ограничение 2: сторона *CD*,

– ограничение 3: сторона *DE*.

Для каждой прямой вычисляются коэффициенты *a*, *b* и *c* в уравнении $ax_1 + bx_2 = c$ по формуле прямой, проходящей через две точки.

2.2. Модуль реализации метода Гомори

Вторым ключевым компонентом программного комплекса является модуль, реализующий непосредственно метод отсечения Гомори. Он построен на основе класса gomori, который использует разработанный ранее класс

simplex_method для решения вспомогательных задач линейного программирования (код программы, разработанной авторами, выложен на https://github.com/MatsueSS/Method-Gomori).

2.2.1. Архитектура и взаимодействие с симплекс-методом

Класс gomori принимает на вход исходные данные задачи (матрицу ограничений, вектор правых частей, целевую функцию и знаки ограничений) и параметр оптимизации (минимизация/максимизация). Конструктор класса инициализирует внутренний экземпляр класса simplex_method и сразу вызывает симплекс-метод для получения оптимального решения непрерывной задачи. Такой подход позволяет отделить логику симплекс-метода от логики построения отсечений, что существенно упрощает отладку и модификацию программы.

2.2.2. Проблема машинной точности и работа с погрешностями

При реализации численных методов на ЭВМ неизбежно возникает проблема представления вещественных чисел. Значения, которые математически являются целыми (например, 4), в памяти компьютера могут представляться как 3.9999999999 или 4.0000000001. Это создает серьёзные трудности для метода Гомори, поскольку алгоритм должен определять, является ли решение целочисленным, и выделять дробные части для построения отсечений. Было применено использование эпсилон-значений 10^{-10} и реализован метод выделения дробной части.

2.2.3. Построение правильного отсечения

Ключевым элементом метода является построение отсечения Гомори по строке симплекс-таблицы с максимальной дробной частью свободного члена. Этот процесс реализован в функции add_new_restrict и выполняется в несколько этапов: выбор ведущей строки, формирование коэффициентов отсечения, добавление новой переменной.

2.2.4. Восстановление допустимости

Происходит выбор ведущей строки – выбирается строка с отрицательной правой частью. Выбор ведущего столбца – среди остальных коэффициентов

выбранной строки ищется столбец, минимизирующий отношение f_i / a_{ij}. После этого выполняются симплекс-преобразования.

2.2.5. Основной цикл метода Гомори

Основной алгоритм реализован в функции method_gomori и представляет собой итерационный процесс. На каждой итерации проверяется наличие дробных первоначальных переменных, добавляется новое отсечение и восстанавливается допустимость, выполняется оптимизация симплекс-методом для получения оптимального решения, увеличивается счётчик итераций.

Процесс завершается, когда все первоначальные переменные становятся целочисленными (с учётом погрешности).

3. Результаты вычислительного эксперимента

После реализации программного комплекса, включающего генератор задач и решатель на основе метода Гомори, необходимо провести серию вычислительных экспериментов. Основная цель данных экспериментов – оценить среднее количество отсечений, требуемое для получения оптимального целочисленного решения, и выявить зависимость этого показателя от различных характеристик решаемых задач.

С помощью модуля PentagonGenerator создаётся новая задача целочисленного программирования с двумя переменными и тремя дополнительными ограничениями, образующими область допустимых решений в виде пятиугольника. Задача решается симплекс-методом без учёта целочисленности. Если полученное решение оказывается целочисленным, задача не учитывается при подсчёте количества отсечений. Если решение нецелочисленное, запускается итерационный процесс добавления отсечений. На каждом шаге фиксируется факт добавления нового ограничения. После получения целочисленного решения записывается количество итераций (отсечений), потребовавшееся для данной задачи. Шаги повторяются 100 000 раз. Подсчитывается среднее количество отсечений, минимальное и максимальное число отсечений в выборке, дисперсия для оценки разброса значений.

В первой серии экспериментов было сгенерировано 100000 задач так, чтобы они были целыми, с параметрами, описанными в разделе 2, результаты представлены в таблице 1.

Таблица 1. – результаты эксперимента, количество учитываемых в подсчётах задач равно 100000. Авторская разработка

Номер эксперимента	Количество задач	Среднее	Дисперсия	Количество отсечений	
				Минимум	Максимум
1	2 404 153	1.61144	0.492481	1	3
2	653 741	1.76022	2.75053	1	7
3	694 982	1.28048	0.201811	1	2
4	1 746 975	1.27091	0.240158	1	3
5	1 297814	1.2535	0.189238	1	2

Далее увеличим все значения параметров, участвующих в генерации точек, в 10 раз и сгенерируем 1 000 000 задач: для того, чтобы набрать требуемое количество ЗЛП с нецелочисленным оптимальным решением потребовалось сгенерировать 10386746 задач. Минимальное количество отсечений равно 1, максимальное – 20. Среднее количество выполненных отсечений 1.64 при дисперсии 3.36. Можно заметить, что среднее количество выполненных отсечений Гомори выросло: увеличение параметров задачи в 10 раз привело к росту среднего числа отсечений с 1.35 до 1.64 – около 21%. Дисперсия увеличилась почти в 7 раз – с 0.5 до 3.36. Это означает, что разброс сложности задач резко вырос. Также видно, что появились сложные задачи, т.к. максимальное количество отсечений увеличилось с 7 до 20.

Заключение. В настоящей работе было проведено экспериментальное исследование количества отсечений, генерируемых методом Гомори при решении задач целочисленного линейного программирования. В ходе выполнения работы были решены следующие задачи: реализован программный комплекс, включающий генератор задач с двумя переменными и тремя

ограничениями, образующими пятиугольную область допустимых решений (на основе вершинного задания координат); решатель, реализующий симплекс-метод и алгоритм построения правильных отсечений Гомори.

Проведена серия вычислительных экспериментов для оценки среднего числа отсечений и анализа зависимости этого показателя от различных факторов.

Метод Гомори показал высокую эффективность для задач с двумя переменными и тремя ограничениями. В подавляющем большинстве случаев (более 95%) требуется не более 2 отсечений, что подтверждает его практическую применимость.

Увеличение коэффициентов ограничений в 10 раз приводит к умеренному росту среднего числа отсечений (с 1.35 до 1.64), но резко увеличивает дисперсию и максимальное наблюдаемое значение (до 20). Это означает, что при работе с задачами, содержащими большие числа, следует учитывать возможность появления «трудных» экземпляров. Редкие «трудные» задачи существуют, но их доля крайне мала. Даже в масштабированном варианте задачи с числом отсечений более 10 составляют менее 1% выборки, а с числом отсечений более 5 – менее 5%. Это подтверждает известный из литературы факт: метод Гомори в худшем случае может требовать много итераций, однако на практике такие ситуации встречаются исключительно редко.

Библиографический список:

1. Кузнецов Ю.Н., Кузубов В.И., Волощенко А.Б. Математическое программирование. – М.: Высшая школа, 1980. – 300 с.

2. Сигал И.Х., Иванова А.П. Методы оптимизации. Начальный курс. Часть 2. Симплекс-метод и смежные вопросы, элементы теории двойственности, многокритериальная оптимизация. Курс лекций по дисциплине «Методы оптимизации» (учеб. пособ). – М.: МИИТ, 2006. – 104 с.

3. Сигал, И. Х. Введение в прикладное дискретное программирование: модели и вычислительные алгоритмы: учебное пособие / И. Х. Сигал, А. П. Иванова. – 2-е изд., испр. и доп. – М.: ФИЗМАТЛИТ, 2007. – 304 с.

4. Ralph E. Gomory. Outline of an algorithm for integer solutions to linear programs. Bulletin of the American Mathematical Society, vol. 64, no. 5, 1958, pp. 275-278.
5. R. E. Gomory. An algorithm for the mixed integer problem. Tech. Report RM-2597, RAND Corporation, 1960.
6. Исследование операций в экономике: Учеб. пособие для вузов / Н.Ш. Кремер, Б.А. Путко, И.М. Тришин, М.Н. Фридман; Под ред. проф. Н.Ш. Кремера. – М: ЮНИТИ, 2002. – 407 с.
7. Корбут А.А., Финкельштейн Ю.Ю. Дискретное программирование. – М.: Наука, 1969. – 368 с.
8. Кормен, Томас Х., Лейзерсон, Чарльз И., Ривест, Рональд Л., Штайн, Клиффорд. Алгоритмы: построение и анализ, 2-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2022. – 1296 с.
9. Бьерн Страуструп. Язык программирования C++. Специальное издание. Пер. с англ. – М.: Издательство Бином, 2011. – 1136 с.
10. Шилдт, Герберт. Полный справочник по C++, 4-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2006. – 800 с.