Алгоритм Гавиша-Пиркуля

Вступление

Данная статья описывает алгоритмы, разработанные Гавишем и Пиркулем, для решения задачи о ранце со многими ограничениями.

1. Релаксации многомерной задачи о ранце

В данной главе приведены различные релаксации задачи, включающие Лагранжеву, Суррогатную и Композитную релаксации, а так же приводятся соотношения между этими релаксациями.

Многомерная задача о ранце формулируется так:

$$Z_{ip} = \max\{cx|Ax \leq b, x \in \{0,1\}\} \tag{1}$$

где A это матрица веса предметов для каждого ограничения размера $m \times n$, b это вектор ограничений и c это вектор стоимости предметов.

Возможны различные релаксации данной задачи. Одна из самых распространненых релаксаций это ослабление ограничения на целостность x . Обозначим ее Z_{lp} :

$$Z_{lp} = \max\{cx|Ax \leq b, x \in [0 \dots 1]\} \tag{2}$$

Лагранжева релаксация получается выбором одного из ограничений (допустим k) как активного и добавления произведения оставшихся ограничений на вектор $\lambda, \lambda > 0$ к целевой функции:

$$L(\lambda) = \max\{cx + \lambda(\overline{b_k} - B_k x)|a_k x \leq b_k, x \in \{0,1\}\} \tag{3}$$

где B_k это матрица размера $(m-1) \times n$, полученная удалением строки k из матрицы A, а $\overline{b_k}$ это вектор, полученный удалением k-го элемента из вектора ограничений b. Полученные границы при решении Лагранжевой релаксации зависят от вектора λ , и ключевой задачей является нахождения такого вектора λ^*, что:

$$L(\lambda^*) = \min_{\lambda}\{L(\lambda)\} \tag{4}$$

Альтернативный подход это суррогатная релаксация:

$$S(\mu) = \max\{cx|\mu(Ax - b) \leq 0, x \in \{0,1\}\} \tag{5}$$

Как и в случае с Лагранжевой релаксацией, ключевой задачей является нахождение вектора μ^*, такого, что:

$$S(\mu^*) = \min_{\mu}\{S(\mu)\} \tag{6}$$

Композитная релаксация это комбинация двух описанных релаксаций:

$$C(\mu, \lambda) = \max\{cx - \lambda(Ax - b) | \mu(Ax - b) \leq 0, x \in \{0, 1\}\} \tag{7}$$

Опять же, задача состоит в нахождении таких векторов $\tilde{\mu}, \tilde{\lambda}$, что:

$$C(\tilde{\mu}, \tilde{\lambda}) = \min_{\mu,\lambda}\{C(\mu, \lambda)\} \tag{8}$$

Отношения между этими релаксациями хорошо известны и приведодятся в следующей лемме:

Лемма 1. *Следующие неравенства справедливы для приведенных релаксаций:*

$$Z_{lp} \geq L(\lambda^*) \geq S(\mu^*) \geq C(\tilde{\mu}, \tilde{\lambda}) \geq Z_{ip} \tag{9}$$

2. Нахождение множителей

Важнейшей проблемой эффективного использования Лагранжевой, Суррогатной и Композитной релаксацийявляется нахождение хороших векторов множителей. Для заданнного набора множителей каждая релаксация сводится к решению классической (одномерной) задачи о ранце, которая можетбыть решена с использованием любого существующего эффективного алгоритма.

Методы субградиентной оптимизации и различные методы корректировки множителей были признаны эффективными эвристиками для получения хороших множителей. Опыт вычисления с использованием этих методов показывает, что процедуры субградиентной оптимизации, как правило, занимают больше времени, но относительно стабильны и в среднем дают лучшие оценки. Это итерационный метод, который генерирует последовательность векторов по следующему правилу:

$$\lambda^{k+1} = \lambda^k + t_k(AX^k - b) \tag{10}$$

где X^k это оптимальное решение для $L(\lambda^k)$ и t_k это положительный скалярный размер шага. Оценка Лагранжевой релаксации, представленные в этой статье вычисляются с использованием метода субградиентной оптимизации со следующим шагом:

$$t_k = \delta_k(L(\lambda^k) - Z_f)/\|Ax^k - b\|^2 \tag{11}$$

где Z_f это значение допустимого решения, а δ_k это скаляр, удовлетворяющий отношению $0 \leq \delta_k \leq 2$. Последовательность δ_k задается инициализацией стартового значение $\delta_0 = 2$ и уменьшением значение δ_k вдвое каждый раз, когда за последние 15 итераций граница не улучшается. В качестве активного ограничения выбирается ограничение, которое дает наименьшую границу при решении одномерной задачи, а остальные ограничения игнорируются.

2.1. Процедура нахождения суррогатных множителей

Гавиш и Пиркуль предложили новую процедуру поиска множителей в суррогатной релаксации. Приводится алгоритм для нахождения множителей для двумерной задачи, а затем формулируется его обобщение для многомерного случая.

Алгоритм для двумерного случая основан на следующей суррогатной задаче:

$$S(\psi, \mu) = \max\{cx | (\psi, \mu) \cdot (Ax - b) \leq 0, x \in \{0, 1\}\} \tag{12}$$

где ψ и μ это скаляры.

Предполагается, что ограничения линейно независимы, первое ограничение нарушается решение $S(0, 1)$, а второе ограничение нарушается решением $S(1, 0)$ (В противном случае случае оптимальное решение проблемы найдено). Следующие леммы были доказаны в предыдущей работе Гавиша и Пиркуля, посвященной многомерной задаче о ранце:

Лемма 2. *Пусть x^* оптимальное решение $S(1, \mu)$. Тогда x^* может нарушать не более одного ограничения в двумерной задаче*

Лемма 3. *Пусть μ' наименьшее значение μ, такое, что решение $S(1, \mu)$ удовлетворяет второму ограничению. Тогда для любых μ_1, μ_2, таких, что $0 \leq \mu_1 \leq \mu_2 \leq \mu', S(1, \mu_1) \geq S(1, \mu_2)$*

Лемма 4. *Пусть μ'' наименьшее значение μ, такое, что решение $S(1, \mu)$ не удовлетворяет второму ограничению. Тогда для любых μ_1, μ_2, таких, что $\mu'' \leq \mu_1 \leq \mu_2 \leq \infty, S(1, \mu_1) \leq S(1, \mu_2)$*

Из этих лемм следует, что когда $\mu' \neq \mu''$ любое значение $\mu \in (\mu'; \mu'')$ является оптимальный множителем. Если $\mu' = \mu''$ оптимальное решение равняется $\mu' = \mu''$. Эти наблюдения используются в следующем алгоритме для нахождение ϵ-окрестности оптимального множителя.

Алгоритм 1. (Вычисление ϵ-окрестности оптимальных множителей)

Шаг 1. Пусть μ_H и μ_L это два множителя, такие, что решение $S(1, \mu)$ удовлетворяет первому ограничению при $\mu = \mu_L$ и не удовлетворяет первое ограничение при $\mu = \mu_H$.

Шаг 2. Если $\mu_H - \mu_L \leq \epsilon$ СТОП. (ϵ-диапазон, который включает оптимальное решение был определен)

Шаг 3. Пусть $\mu = \mu_L + (\mu_H - \mu_L)/2$. Если в решении $S(1, \mu)$:

(i) Оба ограничения удовлетворяются СТОП. (Оптимальное решение было получено)

(ii) Только первое ограничение удовлетворяется то $\mu_L = \mu$, перейти в Шаг 2.

(iii) Первое ограничение не удовлетворяется то $\mu_H = \mu$, перейти в Шаг 2.

Алгоритм начинается с начального набора множителей и использует деление пополам чтобы сократить расстояние между множителями.

Алгоритм 1 может быть включен в эвристику для вычисления множителей у задач, имеющих более двух ограничений. Этот метод определяет направление спуска. Поиск в этом направлении выполняется с помощью Алгоритма 1, в то время как остальные множители остаются постоянными. Полученный набор множителей используется для определения нового направления и поиск повторяется в этом направлении. Следующая лемма определяет возможное направление поиска:

Лемма 5. *Пусть x' 'это решение $S(\mu)$ и x'' это решение $S(\mu + \epsilon e_j)$, где $\epsilon > 0$ и e_j это единичный вектор. Тогда, если $a_j x' > b_j$ и $a_j x'' > b_j$ увеличение j-го элемента μ может улучшить значение целевой функции*

Доказательсво. Определим задачу с двумя ограничениями, где первое ограничение это $(\mu A)x \leq \mu b$ и второе ограничение это $a_j x \leq \mu b_j$. Доказательство следует из Леммы 3.

Из Леммы 5 следует, что ограничения, которые нарушает текущее решение указывает на потенциальное направление поиска. Среди неудовлетворяющих ограничений выбирается ограничение с наибольшим нарушением.

Алгоритм 2. (Вычисление суррогатных множителей для многомерной задачи)
Шаг 1. Определим ограничение, которое дает наименьшее значение целевой функции, когда остальные ограничения игнорируются. Назовем это ограничение как ограничение 1.
Шаг 2. Определим ограничение, которое нарушается больше всех при решении одномерной задачи о ранце с ограничением 1. Назовем это ограничение ограничением 2.
Шаг 3. Применим Алгоритм 1 к двумерное задаче с ограничениями 1 и 2. Если целевая функция не уменьшится в Алгоритме 1 или, если число итераций достигнет верхнего предела СТОП.
Шаг 4. Пусть суррогатное ограничение, определенное в Шаге 3 будет ограничением 1. Перейдем в Шаг 2.

В каждой итерации Алгоритма-1 решается одномерная задача о ранце. Это самый затратный шаг в алгоритме. Возможно ускорить этот шаг, заменив 0-1 ранец непрерывным ранцем. Эта модифицированная версия Алгоритма-1 будет обозначаться Алгоритм-1А и больше не гарантирует нахождение эпсилон-окрестности оптимальных суррогатных множителей. Фактически, непрерывный ранец, сформированный с использованием нового множителя, решает задачу LP релаксации. Модифицированная версия Алгоритма-2, которая использует Алгоиртм-1А в 3-ем шаге будет обозначаться Алгоритм-2А и может быть рассмотрена как эвристика, которая находит суррогатные множители для целочисленной задачи. Но также, он может рассматриваться как эвристика, которая пытается решить задачу релаксации линейного программирования.

Лемма 6. *Решение непрерывной релаксации суррогатной задачи, которая формируется с использованием оптимальных двойственных переменных линейного программирования в качестве множителей, также является оптимальным решением для задачи линейного программирования (2).*

Лемма 6 подразумевает, что Алгоритм-2А является эвристикой, которая ищет оптимальные двойственные переменные для задачи линейного программирования. Суррогатные границы, полученные при использовании двойственных переменных задачи линейного программирования, всегды равны или более жестки чем граница линейного программирования. Поэтому ожидается, что во многих случаях множители, полученные при использовании Алгоритма-2А, приведут к более жестким границам по сравнению с границами, полученными линейным программированием. Можно показать, что для заданного значения ϵ Алгоритм-1А имеет временную сложность $O(n \log(\epsilon))$ и Алгоритм-2А $O(n \cdot \max\{\log(\epsilon), m\})$

2.2. Нахождение композитных множителей

Методы направленного поиска, разработанные для нахождения лагранжевых и суррогатных множителей, были основаны на том факте, что $L(\lambda)$ и $S(\mu)$ являются выпуклыми и квазивыпуклыми. Было показано, что композитная релаксация не обладает ни одной из этих

двух характеристик, и аналогичные методы поиска неизбежно приведут к неудаче в поиске оптимальных или почти оптимальных множителей.

Была разработана эвристика для определения множителей для композитной релаксации. Она начинается с инициализации Лагранжевых множителей, которые равны нулю, и ищет наилучшее множество суррогатных множителей. Затем суррогатные множители используются для формирования суррогатного ограничения и определяется множесвто Лагранжевых множителей. Алгоритм-2 или Алгоиртм-2А могут быть использованы для вычисления суррогатных множителей. Для определения множителей Лагранжа может быть применена процедура субградиентной оптимизации или процедура двойного спуска. Двойной спуск состоит из фиксации всех элементов вектора множителей Лагранжа, кроме одного, к их последнему значению и поиску наилучшего значения оставшегося. Поскольку это одномерный поиск над выпуклой функцией, прямой поиск позволит найти нужный множитель.

3. Анализ чувствительности и уменьшение числа переменных

Многие эффективные алгоритмы для решения одномерного ранца обязаны своим успехом тому факту, что большинство переменных может быть зафиксированы с помощью процедуры сокращения, оставляя небольшую проблему для метода ветвей и границ. Эффективность сокращения после анализа чувствительности напрямую связана с плотностью нижних границ, используемых в анализе. Поскольку границы для многомерного ранца не такие плотные, как границы для случая с одним ограничением, ожидается, что сокращение не будет столь эффективным.

Пусть x^* это решение $S(\mu)$. Можем определить $S(\mu|x_i = 1 - x_i{}^*)$ как задачу, полученную фиксасцией переменной x_i, где $x_i{}^*$ это i-ый элемент x^*. Пусть $\overline{S}(\mu)$ и $\overline{S}(\mu|x_i = 1 - x_i{}^*)$ это задачи, полученные отбрасыванием необходимости целочисленности x. Тогда следующие штрафы могут быть определены:

$$\delta_{i_1} = \overline{S}(\mu|x_i = 1 - x_i{}^*) + x_i^* c_i - Z_F, \tag{13}$$

$$\delta_{i_2} = S(\mu|x_i = 1 - x_i{}^*) + x_i^* c_i - Z_F \tag{14}$$

где Z_F это лучшее известное допустимое решение.

Вначале вычисляется δ_{i_1}. Если δ_{i_1} меньше или равно нулю, тогда переменная x_i фиксируется со значением x_i^*. Только после этого вычисляется значение δ_{i_2} и продолжается сокращение задачи.

4. Метод ветвей и границ

Несмотря на эффективность анализа чувствительности в уменьшении масштаба рассматриваемой проблемы, уменьшенные проблемы по-прежнему имеют значительный размер. Следовательно, для решения этих уменьшенных проблем требуется эффективный алгоритм ветвей и границ. Был разработан общий алгоритм ветвей и границ для экспериментов с различными схемами ветвления, выбора переменной для ветвления и вычисления границы в узле. Сначала представлен общий алгоритм, а затем подробно объяснен каждый шаг. Предполагается, что перед входом в стадию метода ветвей и границ проводится анализ чувствительности и новая, уменьшенная проблема сформирована. Также предполагается, что

штрафы, вычисленные на этапе анализа чувствительности, доступны для использования в алгоритме.

Алгоритм МВГ.

I. Инициализация. Пусть x^* это лучшее известное допустимое решение и Z^* - соответствующее значение целевой функции. Определим S как множество всех свободных переменных, S_F как множество всех зафиксированных переменных, а P как множество всех нерассмотренных узлов. Положим $S = \{\text{Все переменные}\}, S_F = \emptyset, P = \{\text{Узел 0}\}$

II. Разделение. Если $P = \emptyset$ перейти в VI. Если $S = \emptyset$ перейти в III. Иначе, выберите переменную x_j в соответствии с правилом разделения. Сгенерируйе два новых узла с $x_j = 0$ и $x_j = 1$ соответственно. Обновите $S, S_F P$. Если фиксация в единице $x_j = 1$ ведет к нарушение какого либо ограничения удалите соответствующий узел из P

III. Ветвление. Выберите узел из P в соответствии с правилом ветвления.

IV. Вычисление границы. Решите, необходимо ли расчитывать новую границу, и, если необходимо, определите тип расчитываемой границе в зависимости от используемой схемы. Если граница не расчитана перейти в (II). Если граница найдена и соответствубщее решение допустимо установите узел и обновите P, Если граница меньше Z^*, обновите Z^* и x^* соответственно. Перейдите в (III). Если решение, связанное с границей недопустимо, и граница больше Z^*, установите узел и перейдите в (III). Если граница меньше чем Z^* продолжите.

V. Чувствительный анализ. Решите, необходим ли чувствительный анализ. Если да, выполните анализ и обновите $S, S_F P$ соответственно. Обновите порядок разделения, перейдите в (II) *VI. Завершение.* x^* это оптимальное решение и Z^* - соответствующее значение функции данного решения.

Правило определения границы - в каждом узле МВГ доступно 4 опции для определения границы:

(1) - Не вычислять новую границу

(2) - Решить $\overline{S}(\mu)$ с текущим вектором множителей.

(3) - Решить $S(\mu)$ с текущим вектором множителей.

(4) - Вычислить новый набор суррогатный множителей, используя Алгоритм-2А и решить $S(\mu^*)$

Комбинация этих опций используется как основа для описанного алгоритма. Из того, что $\min_\mu S(\mu) \leq S(\mu') \leq \overline{S}(\mu')$ следует, что при увеличении частоты обновление вектором множителей, число рассмотренных узлов в алгоритме будет уменьшаться. Таким образом, необходим компромисс между временем, затраченным на вычисление новых множителей и времени, затраченным на построение большего дерева ветвления.

Нет необходимости вычислять границу в каждом узле дерева, так как узлы, которые не приведут к улучшению границы могут быть обнаружены априори. Если x^* это решение $S(\mu^*)$, когда новый узел создается фиксацией x_i к x_i^k, то граница вычисленная в новом узле будет такая же, как у узла-родителя.

Правило разделения - Разделение осуществляется выбором свободной переменной и фиксацией этой переменной значениями 0 и 1. Существуют два правила для выбора переменной ветвления:

(1) - Правило отношений. Расположите переменные в порядке уменьшения удельной стоимости $c_i/(\mu A)$ и используйте это расположение как порядок разделения. Заметьте, что порядок нужно обновлять каждый раз, когда вычисляется новый вектор μ

(2) - Правило штрафов. Используйте штраф δ_{i_2} из чувствительного анализа и расположите переменны в порядке уменьшения штрафа. Это расположение используется как порядок

разделения.

Правило ветвления - ...

Анализ чувствительности - Анализ чувствительности и тесты на сокращение могут быть выполнены на различных уровнях дерева, чтобы исправить некоторые свободные переменные и определить новые штрафы, которые затем могут быть использованы для обновления порядка разделения во втором правиле. Анализ чувствительности потенциально наиболее эффективен после повторной оптимизации вектора множителей, хотя этот анализ можно выполнить и в других узлах дерева.