Санкт-Петербургский политехнический университет имени Петра Великого

Физико-механический институт

Высшая школа прикладной математики и физики

Курсовая работа на тему "Исследование алгоритмической сложности и приближенных алгоритмов задачи оптимизации MAX Linear Equasions over F2" по дисциплине "Проектирование алгоритмов"

Выполнил:

Студент: Габдушев Рушан
Группа: 5040102/30201

Принял:

к. ф.-м. н.
Пастор Алексей Владимирович

2024 г.

Содержание

Список иллюстраций

Список алгоритмов

1 Введение

Задача **MAX Linear Equasions** является комбинаторной задачей, целью которой является одновременное выполнение наибольшего числа линейных ограничений. Такая задача часто возникает в таких прикладных областях, как, например, распознавание образов и искусственные нейронные сети. В данной работе рассматривается вариация данной задачи с коэффициентами и неизвестными из поля $\mathbb{F}_2$, то есть поля из двух элементов "0" и "1", на котором определены операции сложения и умножения следующим образом:

+	0	1
0	0	1
1	1	0

×	0	1
0	0	0
1	0	1

Рис. 1: Таблицы значений операций в поле $\mathbb{F}_2$

Таким образом, каждая неизвестная либо входит в уравнение с коэффициентом 1, либо не входит в него.

В различных источниках варьируются обозначения данной задачи. Встречаются обозначения:

- **MAX Linear Equasions Over** $\mathbb{F}_2$,
- **MAX Satisfying Linear Subsystem Over** $\mathbb{F}_2$,
- **MAX Feasible Linear Subsystem Over** $\mathbb{F}_2$,
- **E-Lin-2**.

Далее в работе будет использоваться обозначение **MAX Feasible Linear Subsystem Over** $\mathbb{F}_2$ или **MAX FLS**$_{\mathbb{F}_2}$.

2 Постановка задачи

Для доказательства некоторых фактов в дальнейшем потребуется несколько различных постановок данной задачи. В данной главе рассматриваются используемые постановки задачи, а также доказывается их эквивалентность.

2.1 Исходная постановка задачи

Дана система n линейных уравнений с m неизвестными с коэффициентами из поля $\mathbb{F}_2$. Требуется присвоить этим неизвестным значения из поля $\mathbb{F}_2$ так, чтобы они удовлетворяли как можно большему числу уравнений.

2.2 Постановка в форме задачи оптимизации

Дана система n линейных уравнений от m неизвестных: $A \in {\mathbb{F}_2}^{n \times m}, b \in {\mathbb{F}_2}^n$. Требуется найти $x \in {\mathbb{F}_2}^m$ такой, что в системе $Ax = b \bmod 2$ выполняется наибольшее число уравнений, т.е.

$$\begin{gathered} |S(x')| \leqslant |S(x)|, \forall x' \in {\mathbb{F}_2}^m, \\ \text{где } S(x) = \left\{\, s \in \{1..n\} \mid \sum_{j=1}^{m} a_{sj} x_j \equiv b_s \bmod 2 \,\right\} \end{gathered} \tag{1}$$

Обозначим (A, b) – множество уравнений в системе $Ax = b \bmod 2$.
$\delta(A, b)(x) \in \mathbb{Z}$ – число выполненных уравнений в системе (A, b), при значении неизвестных x. Тогда (1) можно переписать как

$$\delta(A, b)(x) = \max_{x' \in {\mathbb{F}_2}^m} (\delta(A, b)(x')). \tag{2}$$

2.3 Постановка в форме задачи распознавания

Дана система из n линейных уравнений от m неизвестных: $A \in {\mathbb{F}_2}^{n \times m}, b \in {\mathbb{F}_2}^n$ и число $B \in \{1..n\}$. Существует ли $x \in {\mathbb{F}_2}^m$ такой, что в системе $Ax = b \bmod 2$ выполняется не меньше B уравнений, т.е.

$$\begin{gathered} \exists x \in {\mathbb{F}_2}^m : |S(x)| \geqslant B, \\ \text{где } S(x) = \left\{\, s \in \{1..n\} \mid \sum_{j=1}^{m} a_{sj} x_j \equiv b_s \bmod 2 \,\right\} \end{gathered} \tag{3}$$

Или в наших обозначениях

$$\exists x \in {\mathbb{F}_2}^m : \delta(A, b)(x) \geqslant B. \tag{4}$$

2.4 Эквивалентность постановок

Докажем теперь, что постановки распознавания и оптимизации равнозначны.

Пусть $I = (A, b) \in D_{\text{MAX FLS}_{\mathbb{F}_2}}$ – индивидуальная задача оптимизации, (I, B) – индивидуальная задача распознавания, $B \in \mathbb{Z}$.

Утверждение 1. Предположим, что существует алгоритм $\mathcal{B}$, который решает MAX $\mathrm{FLS}_{\mathbb{F}_2}$ в форме задачи оптимизации за полиномиальное время. Тогда существует и алгоритм $\mathcal{A}$, решающий за полиномиальное время задачу в форме распознавания.

Доказательство. В результате его работы алгоритма $\mathcal{B}$ мы получим $B' = OPT(I)$ и x, при котором достигается оптимум. Тогда если $B \leqslant B'$ – ответ на задачу распознавания “yes”, иначе “no”. □

Утверждение 2. Предположим, что существует алгоритм $\mathcal{A}$, который решает MAX $\mathrm{FLS}_{\mathbb{F}_2}$ в форме задачи распознавания за полиномиальное время. Тогда существует и алгоритм $\mathcal{B}$, находящий за полиномиальное время максимальный размер выполнимой подсистемы, и алгоритм $\mathcal{C}$, находящий за полиномиальное время соответствующие значения неизвестных.

Доказательство. Предположим теперь, что для задачи распознавания существует алгоритм $\mathcal{A}$ со сложностью $O(p(|I|), p(x) \in \mathbb{Z}[x]$. Сконструируем на его основе полиномиальный алгоритм для задачи оптимизации. Первая часть задачи по нахождению $OPT(I)$ решается двоичным поиском по B (Алгоритм 1).

Алгоритм 1 Нахождение $OPT(I)$ через задачу распознавания

$B_{low} \leftarrow 0$
$B_{high} \leftarrow n + 1$
while $B_{high} - B_{low} > 1$ **do** // Найдём $OPT(I)$ двоичным поиском
    $B_{mid} \leftarrow \lfloor \frac{B_{low}+B_{high}}{2} \rfloor$
    if $\mathcal{A}((A, b), B_{mid}) = yes$ **then**
        $B_{low} \leftarrow B_{mid}$
    else
        $B_{high} \leftarrow B_{mid}$
    end if
end while
$B \leftarrow B_{low}$

Сложность алгоритма поиска оптимума равна $O(\log_2(n)p(|I|))$.

Поиск аргумента произведём поочерёдным исключением неизвестных: если при некотором фиксированном значении для очередной неизвестной в результирующей системе существует выполнимая подсистема размера $OPT(I)$, то выбираем это значение, иначе противоположное (Алгоритм 2).

Алгоритм 2 Нахождение x через задачу распознавания

$B = \mathcal{B}((A, b))$
for $i \in \{1, \ldots, m\}$ **do**
    $b^{(0)} \leftarrow b$
    $b^{(1)} \leftarrow b$
    for $j \in \{1, \ldots, n\}$ **do**
        if $a_{ji} = 1$ **then**
            $a_{ji} \leftarrow 0$ // Исключаем неизвестную x_i.
            // В случае $x_i = 0$ b не изменяется.
            $b_j^{(1)} \leftarrow b_j^{(1)} - 1 \bmod 2$ // Обновляем b для случая $x_i = 1$.
        end if
    end for
    if $\mathcal{A}((A, b^{(0)}), B) = yes$ **then**
        $b \leftarrow b^{(0)}$
        $x_i \leftarrow 0$
    else
        $b \leftarrow b^{(1)}$
        $x_i \leftarrow 1$
    end if
end for
return x

Сложность алгоритма поиска аргумента при известном $OPT(I)$ равна $O(mp(|I|))$. Сложность алгоритма поиска оптимума и аргумента равна $O((m + \log_2(n))p(|I|))$. □

3 NP полнота задачи

3.1 Критерий NP-полноты

Основным способом доказательства NP-полноты задач является использование критерия NP-полноты [1]. В нашей ситуации, для использования данного критерия требуется выполнение 2 этапов:

- Доказать, что MAX $\mathrm{FLS}_{\mathbb{F}_2}$ относится к классу NP;
- Свести известную NP-полную задачу к MAX $\mathrm{FLS}_{\mathbb{F}_2}$.

3.2 Доказательство NP-полноты

Теорема 3. MAX $\mathrm{FLS}_{\mathbb{F}_2} \in \mathrm{NPC}$.

Доказательство. $\mathrm{MAX\,FLS}_{\mathbb{F}_2} \in \mathrm{NP}$, так как в качестве подсказки достаточно указать значения неизвестных x.

Докажем $\mathrm{HM} \propto \mathrm{MAX\,FLS}_{\mathbb{F}_2}$.

Рассмотрим граф $G = (V, E)$ из задачи HM. На его основе построим систему уравнений A следующим образом:

Для каждой вершины $u_i \in V$ добавим по одной неизвестной x_i и уравнение:

$$(A_i^V, b_i^V) = \{x_i \equiv 1 \bmod 2\} \tag{5}$$

Для каждого ребра $e_k = \{u_i, u_j\} \in E$ добавим две неизвестные p_k, q_k и блок из пяти уравнений (A_k^E, b_k^E)

$$(A_k^E, b_k^E) = \left\{ \begin{array}{rcl} p_k & \equiv & 1 \bmod 2, \\ q_k & \equiv & 1 \bmod 2, \\ p_k + q_k & \equiv & 1 \bmod 2, \\ x_i + p_k & \equiv & 1 \bmod 2, \\ x_j + q_k & \equiv & 1 \bmod 2 \end{array} \right\} \tag{6}$$

Отметим, что при фиксированных значениях для неизвестных x_i, x_j и свободно выбираемых значениях для p_k, q_k можно получить следующее число выполненных уравнений:

$$\begin{array}{rrcl} x_i = x_j = 0: & \delta(A_k^E, b_k^E) & \in & \{0, 3, 4\} \\ x_i \neq x_j: & \delta(A_k^E, b_k^E) & \in & \{1, 2, 3, 4\} \\ x_i = x_j = 1: & \delta(A_k^E, b_k^E) & \in & \{2, 3\} \end{array} \tag{7}$$

Итоговое множество уравнений системы:

$$(A, b) = \bigcup_{i=1}^{|V|} (A_i^V, b_i^V) \cup \bigcup_{k=1}^{|E|} (A_k^E, b_k^E) \tag{8}$$

Докажем сведение:

1. Докажем, что если в графе $G = (V, E)$ существует независимое множество $I \subseteq V$, то существует такой x, что $\delta(A, b)(x) \geqslant |I| + 4|E|$. Составим x следующим способом:

для каждой вершины $u_i \in V$

$$x_i = \begin{cases} 1, & u_i \in I \\ 0, & u_i \notin I \end{cases} \tag{9}$$

для каждого ребра $e_k = \{u_i, u_j\} \in E$

$$p_k, q_k = \begin{cases} 1, 1, & u_i, u_j \notin I \\ 0, 1, & u_i \notin I, u_j \in I \\ 1, 0, & u_i \in I, u_j \notin I \end{cases} \tag{10}$$

Видим, что в любом из рассмотренных случаев $\delta(A_k^E, b_k^E)(x) = 4$. Отметим, что случай $u_i, u_j \in I$ невозможен, так как I – независимое множество.

Получаем

$$\delta(A, b)(x) = \sum_{i=1}^{|V|} \delta(A_i^V, b_i^V)(x) + \sum_{k=1}^{|E|} \delta(A_k^E, b_k^E)(x) = |I| + 4|E|. \qquad (11)$$

2. Покажем теперь, что если существует такой x, что

$$\delta(A, b)(x) \geqslant B + 4|E|, \qquad (12)$$

то в графе $G = (V, E)$ существует независимое множество $I \subseteq V$ и $|I| \geqslant B$.

Обозначим $I' \subseteq V$ – множество вершин графа, таких что $\forall u_i \in I' : x_i = 1$. Обозначим $M(I') \subseteq E$ – множество рёбер графа между вершинами I', то есть $M(I') = \{e_k \in E \mid e_k = \{u_i, u_j\}; u_i, u_j \in I'\}$. Тогда из (7)

$$\delta(A_k^E, b_k^E) \leqslant \begin{cases} 4, & e_k \in E \setminus M(I') \\ 3, & e_k \in M(I') \end{cases} \qquad (13)$$

Отсюда

$$\delta(A, b)(x) \leqslant |I'| + 4|E \setminus M(I')| + 3|M(I')| = |I'| + 4|E| - |M(I')| \qquad (14)$$

Тогда из (12) и (14):

$$B + 4|E| \leqslant |I'| + 4|E| - |M(I')| \qquad (15)$$

$$|I'| - |M(I')| \geqslant B \qquad (16)$$

Тогда из I' можно исключить до $|M(I')|$ вершин и оно всё ещё будет требуемого размера. Исключение из I' любой вершины, инцидентной ребру из $M(I')$ уменьшит число таких рёбер как минимум на одно. Значит можно исключить не более $|M(I')|$ вершин и получить независимое множество размером не меньше B.

3. Из пунктов **1** и **2** выходит, что в задаче НМ для $G = (V, E)$ существует независимое множество размера не меньше B тогда и только тогда, когда в задаче MAX $\mathrm{FLS}_{\mathbb{F}_2}$ для системы (A, b) существует $x \in {\mathbb{F}_2}^{|V|+2|E|} : \delta(A, b)(x) \geqslant B + 4|E|$.

4. MAX $\mathrm{FLS}_{\mathbb{F}_2} \in$ NPC

- Задача НМ – NP-полная [3];
- MAX $\mathrm{FLS}_{\mathbb{F}_2} \in$ NPC;
- НМ $\propto$ MAX $\mathrm{FLS}_{\mathbb{F}_2}$;
- По лемме о критерии NP-полноты [1] задача MAX $\mathrm{FLS}_{\mathbb{F}_2}$ является NP-полной.

□

4 Анализ некоторых частных случаев

4.1 Ограничение числа неизвестных в одном уравнении

Рассмотрим случай с ограничением на максимальное число неизвестных в одном уравнении. То есть:

$$\sum_{i=1}^{m} a_{ji} \leqslant K \ \forall j \in \{1, \dots, n\}, K \in \mathbb{Z} \tag{17}$$

В случае $K = 1$ система содержит только уравнения вида $x_i \equiv 0 \bmod 2$ и $x_i \equiv 1 \bmod 2$ и может быть разбита на m независимых тривиальных подсистем $J_i = \{\, j \in \{1, \dots, n\} \mid a_{ji} = 1 \,\}$ где $i \in \{1, \dots, m\}$. Поэтому существует тривиальный полиномиальный алгоритм (Алгоритм 3).

Алгоритм 3 Полиномиальный алгоритм для случая уравнений от одной неизвестной

for $i \in \{1, \dots, m\}$ **do**
  $J_i^{(0)} = \{\, j \in \{1, \dots, n\} \mid a_{ji} = 1, b_j = 0 \,\}$
  $J_i^{(1)} = \{\, j \in \{1, \dots, n\} \mid a_{ji} = 1, b_j = 1 \,\}$
  // Присваиваем значение, которое удовлетворит наибольшее число уравнений J_i.
  if $|J_i^{(0)}| \geqslant |J_i^{(1)}|$ **then**
    $x_i \leftarrow 0$
  else
    $x_i \leftarrow 1$
  end if
  for $j \in J_i = \{\, j \in \{1, \dots, n\} \mid a_{ji} = 1 \,\}$ **do**
    $a_{ji} \leftarrow 0$
    $b_j \leftarrow b_j - x_i \bmod 2$
  end for
end for
$B \leftarrow n - \sum_{i=1}^{n} b_i$
return B, x

Считая, что подсчёт $|J_i^{(0)}|$ и $|J_i^{(1)}|$ выполняется за $O(n)$, алгоритм имеет временную сложность $O(mn)$.

Теорема 4. MAX $\text{FLS}_{\mathbb{F}_2}$ остаётся NPC при ограничении на максимальное число неизвестных в уравненнии $K \geqslant 2$.

Доказательство. Заметим, что при доказательстве NP-полноты общей задачи не были использованы уравнения, зависящие более чем от 2 неизвестных. □

4.2 Ограничение числа вхождения неизвестных в систему

Рассмотрим случай с ограничением на максимальное общее число вхождений каждой неизвестной. То есть:

$$\sum_{j=1}^{n} a_{ji} \leqslant K \ \forall i \in \{1, \ldots, m\}, K \in \mathbb{Z} \tag{18}$$

Снова обращаясь к сведению HM $\propto$ MAX FLS$_{\mathbb{F}_2}$, заметим, что для задачи HM на графе G полученная при сведении система подходит под рассматриваемое условие для $K = \Delta(G) + 1$. Задача HM $\in$ NPC при $\Delta(G) \geqslant 3$ [?], а значит задача остаётся NPC при $K > 3$.

Покажем, что задача остаётся NPC при $K = 3$. Для этого рассмотрим задачу MAX CUT.

Дан неориентированный граф $G = (V, E)$. Необходимо найти такое подмножество вершин $S \subset V$, которое образует наибольший разрез. То есть максимизирующий $|E'(S)|$, где

$$E'(S) = \{\, e = \{u, v\} \in E \mid u \in S, v \in V \setminus S \,\}. \tag{19}$$

Теорема 5. MAX CUT $\in$ NPC и остаётся NPC при ограничении $\Delta(G) = K$, если $K \geqslant 3$.

Доказательство. [4]. □

Теорема 6. MAX CUT $\propto$ MAX FLS$_{\mathbb{F}_2}$

Доказательство. Рассмотрим граф $G = (V, E)$ из задачи MAX CUT. На его основе построим систему уравнений A следующим образом:

Для каждой вершины $u_i \in V$ добавим по одной неизвестной x_i.

Для каждого ребра $e_k = \{u_i, u_j\} \in E$ добавим уравнение

$$x_i + x_j \equiv 1 \bmod 2. \tag{20}$$

Заметим, что если

$$x_i = \begin{cases} 1, & u_i \in S \\ 0, & u_i \notin S \end{cases}, \tag{21}$$

то

$$e_k = \{u_i, u_j\} \in E'(S) \Leftrightarrow \{\, x_i + x_j \equiv 1 \bmod 2 \,\} \text{ выполнено.} \tag{22}$$

Тогда

$$|E'(S)| = B \Leftrightarrow \delta(A, b)(x) = B. \tag{23}$$

□

Заметим, что при $\Delta(G) = K$ система, полученная после сведения задачи MAX CUT к MAX $\text{FLS}_{\mathbb{F}_2}$ включает каждую неизвестную не более K раз.

Теорема 7. MAX $\text{FLS}_{\mathbb{F}_2}$ остаётся NPC при ограничении на максимальное общее число вхождений каждой неизвестной $K = 3$.

Доказательство. Рассмотрим задачу MAX CUT для $\Delta(G) = 3$. По теореме 5 задача является NPC, а по теореме 6 она может быть сведена к задаче MAX $\text{FLS}_{\mathbb{F}_2}$, удовлетворяющей ограничениям. □

5 Точный экспоненциальный алгоритм

Данный алгоритм является простым переборным алгоритмом, который производит перебор всех 2^m возможных комбинаций значений неизвестных, закодированных бинарным представлением числа h. Для каждой комбинации за линейное от длины входа время $O(nm)$ вычисляется $b - Ax' \bmod 2$, и подсчитывается число выполненных уравнений.

Таким образом, итоговая временная сложность алгоритма $O(nm2^m)$.

Алгоритм 4 Переборный алгоритм

$B \leftarrow 0$
for $h \in \{0, \ldots, 2^m - 1\}$ **do** // Перебор всех комбинаций значений x через двоичное представление h
    for $i \in \{1, \ldots, m\}$ **do**
        $x'_i \leftarrow ((h - 1)$ **div** $2^{i-1})$ **mod** 2 // Получение бита i числа h
    end for
    $c \leftarrow b - Ax' \bmod 2$
    $B' \leftarrow n - \sum_{i=1}^{n} c_i$
    if $B' \geqslant B$ **then**
        $x \leftarrow x'$
        $B \leftarrow B'$
    end if
end for
return B, x

6 Полиномиальный приближенный алгоритм

6.1 2-приближенный алгоритм

Рассмотрим алгоритм $\mathcal{A}$ (Алгоритм 5).

Данный алгоритм в каждой итерации цикла удаляет из I один элемент, поэтому заканчивает свою работу после m итераций. Итерация состоит из

Алгоритм 5 Полиномиальный приближенный алгоритм

$I \leftarrow \{1, \dots, m\}$
while $I \neq \varnothing$ **do**
  if $\exists i' \in I : J_{i'} = \{\, j \in \{1, \dots, n\} \mid (a_{jk} = 1) \Leftrightarrow (k = i') \,\} \neq \varnothing$ **then**
    // Если есть уравнения, содержащие ровно одну неизвестную, выберем любую из таких неизвестных (или с наименьшим индексом для детерминированности).
    $i \leftarrow i'$
    $J_i^{(0)} = \{\, j \in J_i \mid b_j = 0 \,\}$
    $J_i^{(1)} = \{\, j \in J_i \mid b_j = 1 \,\}$
    // Присваиваем значение, которое удовлетворит не меньше половины уравнений J_i.
    if $|J_i^{(0)}| \geqslant |J_i^{(1)}|$ **then**
      $x_i \leftarrow 0$
    else
      $x_i \leftarrow 1$
    end if
  else
    $i \in I$ // Выбираем любую оставшуюся неизвестную (или с наименьшим индексом для детерминированности).
    $x_i \leftarrow \mathrm{rand}(\{0, 1\})$ // Присваиваем случайное значение (или всегда 0 для детерминированности).
  end if
  // Пересчитываем систему с учётом значения x_i.
  for $j \in \{1, \dots, n\}$ **do**
    if $a_{ji} = 1$ **then**
      $a_{ji} \leftarrow 0$
      $b_j \leftarrow b_j - x_i \bmod 2$
    end if
  end for
  $I \leftarrow I \setminus \{i\}$
end while
$B \leftarrow n - \sum_{i=1}^{n} b_i$
return B, x

трех частей – поиск уравнения от одной неизвестной ($O(nm)$), подсчёт всех уравнений только от этой неизвестной ($O(nm)$), пересчёт системы с подменой неизвестной на её значение ($O(nm)$). Таким образом, алгоритм имеет временную сложность $O(nm^2)$ и, очевидно, $O(nm)$ сложность по памяти (матрица A, вектора x, b, множество индексов I).

Теорема 8. Алгоритм $\mathcal{A}$ – 2-приближенный.

Доказательство. Оценим точность приближения решения. Обозначим T_i – число образующихся после итерации i новых уравнений вида $0 \equiv 0 \bmod 2$, а F_I – вида $0 \equiv 1 \bmod 2$. Если на итерации алгоритма есть уравнения, содержащие ровно одну неизвестную, то одна из таких неизвестных исключается, таким образом, что образуются тривиальные уравнения. При этом $T_i \geqslant F_i$. Если же уравнений с одной неизвестной нет, то исключение любой из неизвестных не создаст тривиальных уравнений и $T_i = F_i = 0$.

$$\begin{gathered} B = \sum_{i=1}^{m} T_i \geqslant \sum_{i=1}^{m} F_i = n - \sum_{i=1}^{m} T_i = n - B \\ 2B \geqslant n \\ OPT(I) \leqslant n \leqslant 2B \\ \frac{OPT(I)}{B} \leqslant 2 \end{gathered} \tag{24}$$

□

Теорема 9. Погрешность алгоритма $\mathcal{A}$ равна 2.

Доказательство. Покажем теперь, что

$$2 = \inf\{\, \alpha \geqslant 1 \mid \forall I \in D_{\text{MAX FLS}_{\mathbb{F}_2}} (R_{\mathcal{A}}(I) \leqslant \alpha) \,\}. \tag{25}$$

Рассмотрим систему

$$\begin{aligned} &\left.\begin{aligned} x_1 + x_2 &\equiv 0 \bmod 2 \\ x_1 + x_2 + x_3 &\equiv 0 \bmod 2 \end{aligned}\right\} \times 1 \\ &\left.\begin{aligned} x_1 + x_3 + x_4 &\equiv 0 \bmod 2 \\ x_2 + x_3 + x_4 &\equiv 1 \bmod 2 \end{aligned}\right\} \times N \end{aligned} \tag{26}$$

Очевидно, что для данной системы $OPT(I) = 2N + 1$ и достигается, например, при $x = (0, 1, 1, 1)$. Однако, алгоритм $\mathcal{A}$ выполнится следующим образом:

Итерация 1: $i = 1, x_i = 0$

$$\begin{aligned} &\left.\begin{aligned} x_2 &\equiv 0 \bmod 2 \\ x_2 + x_3 &\equiv 0 \bmod 2 \end{aligned}\right\} \times 1 \\ &\left.\begin{aligned} x_3 + x_4 &\equiv 0 \bmod 2 \\ x_2 + x_3 + x_4 &\equiv 1 \bmod 2 \end{aligned}\right\} \times N \end{aligned} \tag{27}$$

Итерация 2: $i = 2, x_i = 0$

$$\left.\begin{aligned} 0 &\equiv 0 \bmod 2 \\ x_3 &\equiv 0 \bmod 2 \end{aligned}\right\} \times 1$$
$$\left.\begin{aligned} x_3 + x_4 &\equiv 0 \bmod 2 \\ x_3 + x_4 &\equiv 1 \bmod 2 \end{aligned}\right\} \times N \tag{28}$$

Итерация 3: $i = 3, x_i = 0$

$$\left.\begin{aligned} 0 &\equiv 0 \bmod 2 \\ 0 &\equiv 0 \bmod 2 \end{aligned}\right\} \times 1$$
$$\left.\begin{aligned} x_4 &\equiv 0 \bmod 2 \\ x_4 &\equiv 1 \bmod 2 \end{aligned}\right\} \times N \tag{29}$$

Итерация 4: $i = 4, x_i = 0$

$$\left.\begin{aligned} 0 &\equiv 0 \bmod 2 \\ 0 &\equiv 0 \bmod 2 \end{aligned}\right\} \times 1$$
$$\left.\begin{aligned} 0 &\equiv 0 \bmod 2 \\ 0 &\equiv 1 \bmod 2 \end{aligned}\right\} \times N \tag{30}$$

Таким образом

$$\frac{OPT(I)}{\mathcal{A}(I)} = \frac{2N+1}{N+2} \xrightarrow[N \to +\infty]{} 2 \tag{31}$$

$\square$

6.2 Нижняя оценка погрешности для приближенных алгоритмов

Для задачи MAX $\mathrm{FLS}_{\mathbb{F}_2}$ было показано [5], что при условии $P \neq NP$ не существует полиномиального приближенного алгоритма с точностью $2 - \epsilon$ для $\forall \epsilon > 0$.

7 Вариации задачи

В этой главе мы рассмотрим несколько вариаций постановки задачи, не являющиеся её частными случаями.

7.1 Взвешенные уравнения

Рассмотрим модификацию задачи: **MAX $\mathbf{WFLS}_{\mathbb{F}_2}$ (Weighted Feasible Linear Subsystem Over $\mathbb{F}_2$)**:

Дана система n линейных уравнений от m неизвестных: $A \in \mathbb{F}_2{}^{n \times m}, b \in \mathbb{F}_2{}^n$, а также вектор целочисленных весов $w \in \mathbb{Z}^n$. Требуется найти $x \in \mathbb{F}_2{}^m$ такой, что:

$$\begin{gathered}\sum_{s \in S(x')} w_s \leqslant \sum_{s \in S(x)} w_s, \forall x' \in \mathbb{F}_2{}^m, \\ \text{где } S(x) = \{\, s \in \{1..n\} \mid \sum_{j=1}^{m} a_{sj} x_j \equiv b_s \bmod 2 \,\}\end{gathered} \tag{32}$$

Обозначим $\delta(A, b, w)(x) \in \mathbb{Z}$ – сумма весов выполненных уравнений в системе (A, b), при значении неизвестных x. Тогда (32) можно переписать как

$$\delta(A, b, w)(x) = \max_{x' \in \mathbb{F}_2{}^m} (\delta(A, b, w)(x')). \tag{33}$$

В форме задачи распознавания необходимо доказать для $B \in \mathbb{Z}$, что

$$\exists x \in \mathbb{F}_2{}^m : \delta(A, b, w)(x) \geqslant B. \tag{34}$$

Теорема 10. Задачи MAX WFLS$_{\mathbb{F}_2}$ и MAX FLS$_{\mathbb{F}_2}$ полиномиально эквивалентны.

Доказательство. Пусть $I = ((A, b, w), B)$ – индивидуальная задача MAX WFLS$_{\mathbb{F}_2}$ в форме задачи распознавания. Составим новую систему (A', b'), таким образом, что уравнение под номером i из системы (A, b) входит в него ровно w_i раз.

Тогда для $\forall x \in \mathbb{F}_2{}^m$ верно

$$\delta(A, b, w)(x) = \sum_{s \in S(x)} w_s = \delta(A', b')(x), \tag{35}$$

где $S(x) = \{\, s \in \{1..n\} \mid \sum_{j=1}^{m} a_{sj} x_j \equiv b_s \bmod 2 \,\}$. Из равенства этих функций следует равенство величин решений, а значит

$$\delta(A, b, w)(x) \geqslant B \Leftrightarrow \delta(A', b')(x) \geqslant B. \tag{36}$$

□

Следствие 10.1. Если существует ϵ-приближенный алгоритм, решающий задачу MAX FLS$_{\mathbb{F}_2}$, то существует и ϵ-приближенный алгоритм, решающий задачу MAX WFLS$_{\mathbb{F}_2}$.

Доказательство. Пусть $\mathcal{A}$ – ϵ-приближенный алгоритм, решающий задачу MAX FLS$_{\mathbb{F}_2}$, $I = (A, b, w)$ – индивидуальная задача MAX WFLS$_{\mathbb{F}_2}$.

Пусть $\mathcal{B}$ – алгоритм решающий задачу MAX WFLS$_{\mathbb{F}_2}$, который составляет новую систему (A', b'), тем же образом, что и в доказательстве теоремы 10, и запускает для неё алгоритм $\mathcal{A}$.

Тогда из (35) следует

$$OPT(A, b, w) = OPT(A', b') \tag{37}$$

$$\frac{OPT(A, b, w)}{\mathcal{B}(A, b, w)} = \frac{OPT(A', b')}{\mathcal{A}(A', b')} \leqslant \epsilon. \tag{38}$$

□

Следствие 10.2. При условии $P \neq NP$, для задачи MAX WFLS$_{\mathbb{F}_2}$ не существует полиномиального приближенного алгоритма с точностью $2 - \epsilon$ для $\forall \epsilon > 0$.

7.2 Обязательные ограничения

Рассмотрим модификацию задачи: **MAX FLSME$_{\mathbb{F}_2}$ (Feasible Linear Subsystem With Mandatory Equasions Over $\mathbb{F}_2$)**:

Даны две системы из n и k линейных уравнений от m неизвестных: $A \in {\mathbb{F}_2}^{n\times m}, C \in {\mathbb{F}_2}^{k\times m}, b \in {\mathbb{F}_2}^{n}, d \in {\mathbb{F}_2}^{k}$. Требуется найти $x \in X = \{\, x \in {\mathbb{F}_2}^{m} \mid Cx \equiv d \bmod 2 \,\}$ такой, что в системе $Ax = b \bmod 2$ выполняется наибольшее число уравнений, т.е.

$$\begin{gathered} |S(x')| \leqslant |S(x)|, \forall x' \in X, \\ \text{где } S(x) = \{\, s \in \{1..n\} \mid \sum_{j=1}^{m} a_{sj} x_j \equiv b_s \bmod 2 \,\} \end{gathered} \tag{39}$$

Или в наших обозначениях

$$\begin{gathered} \delta(A, b)(x) = \max_{x' \in X} (\delta(A, b)(x')), \\ X = \{\, x \in {\mathbb{F}_2}^{m} \mid Cx \equiv d \bmod 2 \,\}. \end{gathered} \tag{40}$$

Отметим, что система $Cx \equiv d \bmod 2$ должна быть выполнима.

Теорема 11. Задачи MAX FLSME$_{\mathbb{F}_2}$ и MAX WFLS$_{\mathbb{F}_2}$ полиномиально эквивалентны.

Доказательство. Пусть $I = ((A, b, C, d), B)$ – индивидуальная задача MAX FLSME$_{\mathbb{F}_2}$ в форме задачи распознавания. Составим новую систему (A', b'), включающую все уравнения (A, b) и (C, d). Составим вектор весов $w \in \mathbb{Z}^{n+k}$ таким образом, что для уравнений из системы (A, b) вес равен 1, а для уравнений из (C, d) вес равен $n + 1$.

Таким образом, если в (A', b', w) не выполнено хотя бы одно обязательное уравнение, то

$$\delta(A', b', w)(x) < k(n+1). \tag{41}$$

Пусть $\exists x \in X = \{\, x \in \mathbb{F}_2{}^m \mid Cx \equiv d \bmod 2 \,\}$, такой что

$$\delta(A, b)(x) \geqslant B. \tag{42}$$

Тогда

$$\delta(A', b', w)(x) \geqslant k(n+1) + B. \tag{43}$$

Обратно, пусть $\exists x \in \mathbb{F}_2{}^m$, такой что

$$\delta(A', b', w)(x) \geqslant k(n+1) + B. \tag{44}$$

Тогда из (41) следует, что $Cx \equiv d \bmod 2$, а значит

$$\delta(A, b)(x) = \delta(A', b', w)(x) - k(n+1) \geqslant B. \tag{45}$$

□

Список литературы

[1] А. В. Пастор. *Теория алгоритмов Лекция 2. Класс NP и NP-полнота.* https://logic.pdmi.ras.ru/~pastor/fmf/algorithm2/lect2.pdf. [Online; accessed 15-December-2024]. 2024.

[2] Amaldi, Edoardo, and Viggo Kann. "The complexity and approximability of finding maximum feasible subsystems of linear relations." *Theoretical computer science* 147.1-2 (1995): 181-210.

[3] Juris Hartmanis. *Computers and intractability: a guide to the theory of npcompleteness (michael r. garey and david s. johnson).* В: Siam Review 24.1 (1982), с. 90.

[4] Papadimitriou, Christos, and Mihalis Yannakakis. "Optimization, approximation, and complexity classes."*Proceedings of the twentieth annual ACM symposium on Theory of computing.* 1988.

[5] Håstad, Johan. "Some optimal inapproximability results."*Journal of the ACM (JACM)* 48.4 (2001): 798-859.

[6] Ausiello, Giorgio, et al. Complexity and approximation: Combinatorial optimization problems and their approximability properties. https://www.csc.kth.se/~viggo/approxbook/. [Online; accessed 15-December-2024]. 2024.