Maximum Acyclic Subgraph

Никита Лансков

19 января 2022 г.

Содержание

1 Формулировка задачи распознавания

Задача 1 (Задача MAS)
Дан конечный ориентированный граф $D = (V, A)$ *и константа* $B \in \mathbb{N}$. *Существует ли подмножество* $A' \in A$, *такое, что подграф* $D = (V, A')$ *не содержит циклов и* $|A'| >= B$.

2 Доказательство NP-полноты

Чтобы показать, что задача MAS является NP-полной, требуется:

1. Показать, что $MAS \in NP$

2. Свести к MAS другую известную задачу, чья NP-полнота уже установлена

Определение 1
Задача распознавания P *принадлежит классу* NP *при схеме кодирования* c, *если* $L[P, c] \in NP$

Определение 2
Язык L принадлежит классу NP, если существует НМТ M, распознающая L, и многочлен $p \in \mathbb{Z}[x]$, *такие, что время работы M на любом входе* $x \in \Sigma^*$ *не превосходит* $p(|x|)$

Таким образом, чтобы доказать что задача MAS является NP полной, нам достаточно убедиться в существовании недетерминированной машины тьюринга, которая бы распознавала язык этой задачи.

Действительно, в качестве подсказки достаточно взять набор нулей и единиц длины $|V|$, где каждое значение соответствует конкретной вершине $v \in V$, и единицы стоят на местах вершин, которые входят в максимальный ациклический подграф, а на местах оставшихся вершин - нули. Для этого предлагаю свести к задаче MAS задачу о независимом множестве.

Определение 3
G *- конечный граф.* $W \in V(G)$ *- независимое множество, если*
$\forall u, v \in W (uv \notin E(G))$

Задача 2 (О независимом множестве)
Дан конечный неориентированный граф G *и число* $B \in \mathbb{N}$. *Есть ли в* G *независимое множество размера не менее* B.

Преобразуем неориентированный граф из зачачи о независимом множестве G к ориентированному D следующим образом:

$$V(D) = V(G)$$
$$A(D) = \{\{uv, vu\} \mid \forall uv \in E(G)\}$$

Таким образом, мы строим граф на тех же вершинах, и для каждого ребра исходного графа добавляем две разнонаправленные дуги в наш новый ориентированный граф.

При таком построении, если мы найдем в графе G независимое множество W, то мы также нашли бы максимальный ациклический подграф в D. Это правда, так как добавление любой из оставшихся вершин в подграф появился бы цикл, так как в графе G добавленная вершина была бы связана с одной или несколькими вершинами из независимого множества W.

3 Частные случаи

Определение 4
Степень вершины графа - количество ребер, инцидентных этой вершине.

Определение 5
Регулярный граф степени k - это граф, все вершины которого имеют степень k.

3.1 Регулярный граф степени < 3

Самый простой частный случай - если мы имеем дело с регулярным графом степени меньше 3. Если в таком графе и есть цикл - то это цикл, в который входят все вершины. Чтобы это проверить, достаточно пройтись по всем ребрам, что можно сделать за полиномиальное время.

3.2 Регулярный граф степени 3

Если же регулярный граф имеет степень 3 - то все не так однозначно. В общем случае - задача MAS остается NP трудной для таких графов. В [1] приводится приближенный алгоритм для таких графов, с точностью приближения $\frac{8}{9}$,

Пусть дан регулярный граф степени 3: $G = (V, E)$, для которого мы хотим найти максимальный ациклический подграф $S \subseteq E$. Будем проводить рассуждения в рамках следующих предположений.

Определение 6
Длина цикла - число ребер, входящих в цикл.

Определение 7
Положительная (отрицательная) степень вершины графа - это число всех исходящих (входящих) ребер. Обозначения: $d^+\{v\}, (d^-\{v\})$

Предположение 3.2.0.1
Все вершины в графе G имеют положительную и отрицательную степени не меньше 1 и суммарную степень 3.

Доказательство. Если в G содержится вершина, у которой положительная или отрицательная степень равна нулю, то мы можем сразу включить все смежные с ней ребра в S, так как они будут содержаться в любом максимальном ациклическом подграфе. □

Предположение 3.2.0.2
Граф G не содержит циклов длины 2 и 3.

Доказательство. Если мы имеем дело с неориентированными циклами, то мы можем договориться для каждого такого цикла включать в S все ребра цикла, при этом сами циклы удалить из рассмотрения. Удаление цикла длины три не добавит новых циклов, так как мы работаем в рамках предположения 3.2.0.1. В случае ориентированных циклов длины 2 и 3, нам достаточно не включать в S какое-то одно ребро цикла. □

Определение 8
α-ребро - ребро (i,j), такое, что

$$d^-\{i\} = 2, d^+\{i\} = 1 \qquad d^-\{j\} = 1, d^+\{j\} = 2$$

По лемме 2.1 из [1], если регулярный граф степени 3 не содержит α-ребер, то все циклы в нем не содержат общих ребер. Это утрверждение легко доказывается от обратного. Если мы рассматриваем два цикла в регулярном графе степени три без α-ребер, то возможны два случая: они пересекаются по одному ребру или содержат несколько общих ребер. В случае пересечерния по одному ребру очевидно, что это ребро пересечения обязано быть α-ребром. Если же пересечение состоит из нескольких ребер, то среди этих ребер всегда найдется α-ребро. Это легко проверить следующим алгоритмом. Предположим, что у нас направление движения по общему пути для двух циклов - сверху вниз (для определенности). Мы будем пробовать привести такой пример графа, в котором этот самый общий путь не содержал бы α-ребер. Зная, что все вершины у нас степени 3, и так как это часть цикла, попробуем добавить ребра к вершинам которые находятся внутри пути так, чтобы не появилось α-ребер. Будем идти сверху вниз. При таком подходе мы чередуем добавление ребер в разных направлениях, при этом мы никак не можем избежать появления аьфа-ребра среди общих ребер циклов.

Если в графе нет α-ребер, то мы можем найти максимальный ациклический подграф за полиномиальное время следующим алгоритмом. Мы просто находим цикл в графе, выбрасываем произвольное ребро из этого цикла, остальные добавляем в максимальный ациклический подграф. Также после этого мы стягиваем соответствующие ребра в оставшемся графе таким образом, чтобы 3.2.0.1 и 3.2.0.2 оставались истинными.

Если в графе есть α-ребра, то делаем следующее.

1. Находим α-ребро e в графе
2. Удаляем e Добавляем все $E(e)$ и вершины с нулевой положительной/отрицательной степенью в решение.
3. стягиваем вершины у которых $d^+\{v\} = 1$, $d^-\{v\} = 1$

Под стягиванием вершин в данном случае подразумевается следующее. если у вершины i одно входящее и одно исходящее ребро, то мы выбрасываем эту вершину из рассмотрения вместе с инцидентными ребрами, при этом соединяем начало входящего ребра с концом исходящего ребра новым ребром напрямую.

Понятно, что последний приведенный алгоритм будет давать апроксимацию $\dfrac{8}{9}$ в регулярном графе степени 3. Если мы нашли α-ребро e для которого $|\mathrm{C(e)}| = 9$ то мы решаем эту компоненту точно, а если нет - то с точностью $8/9$, так как для каждого альфа-ребра на удаление одного ребра мы добавляем в наше решение как минимум 8 ребер.

В [1] также показано, что если более аккуратно выбирать α-ребра, то мы можем получить алгоритм с точностью $\dfrac{11}{12}$.

3.3 Другие частные случаи

В [2] показано, что если регулярный граф степени 3 также является планарным, то MAS можно решить за полиномиальное время. Также показано, что если планарный граф имеет максимальную степень вершин больше 3, то задача вновь становится NP-полной.

4 Точный экспоненциальный алгоритм

4.1 Описание алгоритма

Рассмотрим теперь точный экспоненциальный алгоритм, для этого вначале приведем несколько определений.

Покажем как построить алгоритм для двойственной к MAS задаче:

Задача 3 (FAS)
Дан конечный ориентированный граф $G = (V, A)$. Нужно найти такое подмножество $F \subseteq A$, такое чтобы G - F Был бы ациклическим графом и при этом $|F|$ была минимальной из всех возможных.

Понятно, что если мы знаем решение для задачи FAS, то мы легко найдем и сам максимальный ациклический подграф в исходном графе, как $G\left[A - F\right]$

Также введем функцию, которая упорядочивает вершины в графе:

Определение 9
$\pi : V \to 1, ..., |V|$ - функция, которая задает некоторую нумерацию на множестве вершин V. При этом, $\forall a = (i, j) \in A$:

$$\pi(i) < \pi(j) \Rightarrow a - \textit{прямая}$$
$$\pi(i) > \pi(j) \Rightarrow a - \textit{обратная}$$

С учетом функции π можно немного переформулировать задачу FAS следующим образом:

Задача 4 (FAS)
Дан ориентированный граф $G(V, A)$. Найти такую перестановку π, что $\sum\limits_{((u,v)\in A \ \& \ \pi(u)>\pi(v))} 1$ - минимальна. Иными словами, требуется найти такую перестановку, в которой количество обратных дуг минимально.

Под оптимальной перестановкой будем понимать такую перестановку, в которой число обратных дуг минимально.

Определение 10
$X(S)$ - число обратных дуг в оптимальной перестановке для индуцированного графа $G[S]$, $S \subseteq V$.

Также $X(S)$ можно представить в виде следующей рекурсивной формулы:

$$X(S) = \min_{u \in S} \left\{ X(S-u) + \sum_{((u,v) \in A \ \& \ v \in (S-u))} \right\} \tag{1}$$

Покажем корректность рекурсивной формулы (1). Важно понимать, что при каждом увеличении размерности, к примеру при переходе от $S-u$ к S, мы при каждом рассмотрении новой вершины u присваиваем есть новый, самый большой номер в нумерации вершин. Соответственно, все дуги, которые приходят из вершин множества $S-u$ в вершину u будут являться обратными по определению. Ну и чтобы найти минимальное количество обратных дуг мы ищем минимум суммы минимального числа обратных дуг для множества без рассматриваемой вершины и числа всех дуг, которые приходят в рассматриваемую вершину из вершин $S-u$.

Перейдем теперь непосредственно к алгоритму [3]. Ключевой структурой данных, которую мы будем использовать, будет массив Y размерности $2^n \times 2$. В ходе работы алгоритма будем рассматривать подмножества множества вершин $S \subseteq V$, и $\forall S$:

$$Y[S,1] = X(S)$$
$$Y[S,2] = \{v | v \in V : X(S) \ is \ minimized \ in \ eq. \ (1)\}$$

Таким образом в результате работы алгоритма мы получим:

$$Y[V,1] - \textit{число обратных ребер в оптимальной перестановке для } G$$
$$\bigcup_{S \subseteq V} Y[S,2][1] - \textit{множество вершин, которые инцидентны обратным дугам}$$
$$\textit{в минимальном по размеру множестве } F$$

4.2 Оценка сложности

Теорема 4.2.1
$G(V,E)$ - ориентированный граф, $|V| = n$, $|A| = m$. Тогда размер множества обратных дуг может быть найден за время $O^*(2^n)$ и с использованием $O^*(2^n)$ памяти.

Доказательство. В алгоритме мы для каждого подмножества S отрабатываем некоторое количество вершин за время $O^*(n)$.

Algorithm 1: FAS

Input: A directed graph G
Output: Size of a minimum feedback set
Let Y be a $2^n \times 2$ array indexed from 0 to $2^n - 1$
Initialize $Y[S,1] = \infty$, $Y[S,2] = 0$ for all subsets $S \subseteq V$ *and* $S \neq \varnothing$
Initialize $Y[\varnothing, 1] = Y[\varnothing, 2] = 0$
for $S \subseteq V$*enumerated in increasing order of cardinality* **do**
  for $u \in V - S$ **do**
    $P = Y[S,1] + \sum_{((u,v) \in A \ \& \ v \in (S-u))} 1$
    if $P = Y[S \cup \{u\}]$ **then**
      $Y[S \cup \{u\}, 2] = Y[S \cup \{u\}, 2] \cup \{u\}$
    if $P < Y[S \cup \{u\}]$ **then**
      $Y[S \cup \{u\}, 1] = P$
      $Y[S \cup \{u\}, 2] = u$
return Y[V, 1]

5 Полиномиальный приближенный алгоритм

Теперь рассмотрим несколько алгоритмов, приведенных в [4]

Algorithm 2: Простейший приближенный алгоритм для MAS

Input: G(V, A) - ориентированный граф, некоторая нумерация π вершин в нем
Output: $A' \subseteq A$ - подмножество множества дуг, такое что индуцированный граф $G'(V, A')$ является ациклическим
$A_1 = \{(i,j) \in A | \pi(i) < \pi(j)\}$
$A_2 = \{(i,j) \in A | \pi(i) > \pi(j)\}$
if $|A_1| > |A_2|$ **then**
  return A_1
return A_2

В простейшем алгоритме мы разбиваем множество дуг на две части: Часть с прямыми ребрами и часть с обратными ребрами. Соответственно, и A_1 и A_2 - являются ацикличными по построению, и при этом $\max\{|A_1|, |A_2|\} \geqslant |A|/2$

Теперь рассмотрим ряд менее тривиальных алгоритмов, а также построим оценки.

Определение 11
Будем считать, что у нашего невзвешенного графа веса всех ребер одинаковы и равны 1. Тогда $\forall i \in V$:

$$w_i^{in}(S) = \sum_{j \in S} w_{ji} - \textit{число входящих дуг в вершину } i$$
$$w_i^{out}(S) = \sum_{j \in S} w_{ij} - \textit{число исходящих дуг из вершины } i$$

Определение 12
С учетом (11) будем называть ***весом графа*** $G(V, A)$ *число дуг* $|A|$.

Определение 13
Будем говорить, что перестановка π *индуцирует подграф* $G' = (V, A')$, *и* $A' = \{(i,j) \in A | \pi(i) < \pi(j)\}$

В следующем алгоритме мы также на выходе получаем перестановку, которая индуци рует ациклический подграф с не менее чем половиной ребер от их общего числа в исходном графе, так как это свойство сохраняется на каждом шаге цикла.

Algorithm 3: Базовый алгоритм

Input: G(V, A) - ориентированный граф
Output: Перестановка π, которая индуцирует ациклический подграф.

$S = V, l = 1, u = n$
while $u \geqslant l$ **do**
  Выбираем $i \in S$
  $S = S - \{i\}$.
  if $w_i^{in}(S) \leqslant w_i^{out}(S)$ **then**
    $\pi(i) = l$
    $l = l + 1$
  else
    $\pi(i) = u$
    $u = u - 1$
return π

Теперь, воспользовавшись базовым алгоритмом 3, построим вероятностный алгоритм. Будем считать, что исходный граф не содержит циклов размера 2.

Algorithm 4: Вероятностный алгоритм

Input: G(V, A) - ориентированный граф

Output: Перестановка π, которая индуцирует ациклический подграф.

Разбиваем V на две части V_1, V_2, добавляя каждую вершину в каждую из частей с вероятностью 0.5.

for $r \in \{1, 2\}$ **do**

  $A_r = \{(i,j)|i, j \in V_r\}$

  π_r - перестановка вершин из V_r

  Применяем алгорим 3 к (V_r, A_r), вершины выбираем в порядке возрастания индексов.

if $|A_{(\pi_1,\pi_2)}| > |A_{(\pi_2,\pi_1)}|$ **then**

  return (π_1, π_2)

else

  return (π_2, π_1)

Теорема 5.0.1

Пусть APX - число дуг в решении, полученном при помощи алгоритма 4. Тогда

$$APX = \left(0.5 + \Omega(\frac{1}{\sqrt{d_{max}}})\right)|A|$$

Доказательство. $i \in V_r$.

Определим ряд значений:

$$\begin{aligned} D_i^{in} &= |(j,i) \in A : j > i| \\ D_i^{out} &= |(i,j) \in A : j > i| \\ d_i^{in} &= |(j,i) : j \in V_r, j > i| \\ d_i^{out} &= |(i,j) : j \in V_r, j > i| \end{aligned}$$

d_i^{in} - биномиально распределенная случайная величина с параметрами $(0.5, D_i^{in})$, так как для каждой дуги инцидентной с i вероятность того, что другой ее конец содержится также в V_r равна 0.5 Аналогично, d_i^{out} - биномиально распределенная случайная величина с параметрами $(0.5, D_i^{out})$.

Не умаляя общности, предположим что $D_i^{in} \geqslant D_i^{out}$. Для $a \geqslant 0$:

$$P \equiv Pr(|d_i^{in} - d_i^{out}| \geqslant a) \geqslant Pr(d_i^{in} - d_i^{out} \geqslant a)$$

Согласно нашему предположению о том, что исходный граф не содержит циклов длины 2, d_i^{in} и d_i^{out} - независимые случайные величины. Тогда:

$$P \geqslant Pr(X_1 - X_2 \geqslant a) \geqslant Pr(X_1 \geqslant 0.5D_i^{in} + a, X_2 \leqslant 0.5D_i^{in}) = 0.5Pr(X_1 \geqslant 0.5D_i^{in} + a)$$

Где первое неравенство следует из предположения, что $D_i^{in} \geqslant D_i^{out}$. Установим $a = \frac{1}{2}(D_i^{in})^{1/2}$ - стандартное отклонение для X_1. Тогда получим:

$$Pr(|d_i^{in} - d_i^{out}| \geqslant a) \geqslant \beta, \beta > 0$$

Обозначим $D_i = D_i^{in} + D_i^{out}$, тогда

$$D_i^{in} \geqslant D_i/2$$
$$a = \Omega(\sqrt{D_i}) = \Omega(D_i/\sqrt{D_i}) = \Omega(D_i/\sqrt{d_{max}})$$

На каждом шаге цикла мы присваиваем вершину i на следующую верхнюю или нижнюю позицию, в зависимости от знака выражения $d_i^{in} - d_i^{out}$. Суммарное число дуг, индуцированных перестановкой π_r:

$$\sum_{i \in V_r} \max\{d_i^{in}, d_i^{out}\} = \sum_{i \in V_r} \left(\frac{d_i^{in} + d_i^{out}}{2} + \frac{1}{2}|d_i^{in} - d_i^{out}| \right) = 0.5|A_r| + 0.5 \sum_{i \in V_r} |d_i^{in} - d_i^{out}|$$

Получаем, что число вершин, индуцированное π_r, равно:

$$APX_r = 0.5|A_r| + \Omega \left(\sum_i \frac{D_i}{\sqrt{d_{max}}} \right)$$

Так как $\sum_i D_i = |A|$, получаем:

$$APX_1 + APX_2 = 0.5(|A_1| + |A_2|) + \Omega(\frac{|A|}{\sqrt{d_{max}}})$$

$\square$

В [4] также показано что алгоритм 4 будет работать за $O(d_{max}^3 + |A|)$.

Также, в качестве конструктивного алгоритма в [4] рассматривается применение локального поиска к множеству соседних перестановок V_π, которое определяется следующим образом:

$$\pi' \in V_\pi \Leftrightarrow \forall j, k : j < k,$$
$$\pi' = (\pi_1, ..., \pi_{j-1}, \pi_k, \pi_{j+1}, ..., \pi_{k-1}, \pi_j, \pi_{k+1}, ...)$$

6 Направления для дополнительных исследований

Вот несколько статей, рассмотрение которых может быть интересно для дальнейших исследований:

- В [5] приводится эффективный вероятностный алгоритм, а также нижняя оценка для MAS.
- В [6] приводится сведение задачи 3-SAT к MAS, а также несколько примеров построения релаксационных алгоритмов.
- В [7] исследуются (1, n) графы как более общий случай 3 регулярных графов для нахождения эффективных точных алгоритмов.

Список литературы

[1] Alantha Newman. The maximum acyclic subgraph problem and degree-3 graphs. pages 147–158, 01 2001.

[2] Mourad Baïou and Francisco Barahona. Maximum weighted induced bipartite subgraphs and acyclic subgraphs of planar cubic graphs. *SIAM Journal on Discrete Mathematics*, 30:1290–1301, 01 2016.

[3] Venkatesh Raman and Saket Saurabh. Improved fixed parameter tractable algorithms for two "edge" problems: Maxcut and maxdag. *Information Processing Letters*, 104:65–72, 10 2007.

[4] Refael Hassin and Shlomi Rubinstein. Approximations for the maximum acyclic subgraph problem. *Information Processing Letters*, 51:133–140, 08 1994.

[5] Moses Charikar, Konstantin Makarychev, and Yury Makarychev. On the advantage over random for maximum acyclic subgraph. *Electronic Colloquium on Computational Complexity (ECCC)*, 14, 10 2007.

[6] Alantha Newman. Approximating the maximum acyclic subgraph. 05 2014.

[7] Henning Fernau and Daniel Binkele-Raible. Exact algorithms for maximum acyclic subgraph on a superclass of cubic graphs. pages 144–156, 02 2008.