Общая надстрока наименьшей длины

Андрей Осипов

16 декабря 2013 г.

1 Постановка задачи

Дан набор строк $S = \{s_1, \ldots, s_n\}$ над конечным алфавитом. Требуется найти, строку s минимальной длины, содержащую как подстроку каждую строку из данного набора.

Пусть язык L это множество пар вида (S, k) для которых верно, что такая строка s существует, и имеет длину не больше k. Тогда задача разрешения языка L является NP-полной. Доказательство этого факта будет приведено позже. А пока, мы ослабим условие следующим образом: пускай теперь требуется найти такую строку t, что она так же как и s содержит всякую строку из S как подстроку, и при этом $|t| \leq 4 * |s|$

2 Алгоритм

Алгоритм для решения этой задачи на первый взгляд может показаться крайне наивным. Но в дальнейшем выяснится, что этого вполне достаточно для достижения даже такой близкой границы. Более того, на практике полученный ответ разрастается не более чем вдвое.

Итак, без ограничения общности будем считать, что среди строк из S нет таких двух x и y, что x подстрока y. В противном случае от x можно спокойно избавиться.

Определение 1. *Пусть даны строки x и y. Представим их как $pr+ov$ и $ov+su$ соответственно, причем $|pr| > 0$, $|su| > 0$, ov имеет максимальную возможную длину, а опетор (+) - это конкатенация строк. Тогда определим $over(x, y) = ov$, $pref(x, y) = pr$ и $d(x, y) = |pr|$.*

Теперь запустим следующий алгоритм.

1. Если в множестве S осталась ровно одна строка, то выведем её и прекратим работу алгоритма.

2. Иначе, переберём все упорядоченные пары различных строк x и y из S и найдем среди них ту, у которой $|over(x, y)|$ максимален. Если таких несколько можно выбрать любую. Например, лексикографически минимальную.

3. Найдя такую пару, выкинем из S строки x и y, а вместо них положим туда строку $pref(x, y) + y$. И перейдем к первому пункту алгоритма.

3 Доказательство

Давайте посмотрим на то, что на самом деле происходит с исходными строками во время работы алгоритма. Для начала, дадим несколько определений:

Определение 2. *Пусть дана непустая строка s. Тогда $s_{i,j}$, где $1 \leq i \leq j \leq |s|$ - это подстрока s начинающаяся с i-ого символа и заканчивающаяся на j-ом символе строки s включительно.*

Определение 3. *Пусть даны нупустые строки s, t и число pos, тогда:*

$$contains(s, t, pos) = \begin{cases} 1, & \text{если } pos \geq 1, (pos + |t| - 1) \leq |s| \text{ и строка } t \text{ равна } s_{pos, pos+|t|-1} \\ 0, & \text{иначе} \end{cases}$$

Определение 4. *Пусть даны непустые строки s, t, тогда:*

$$index(s, t) = \begin{cases} \text{минимальное число } pos, & \text{такое, что contains(s, t, pos) = 1} \\ 0, & \text{если такого числа не существует} \end{cases}$$

Определение 5. *Пусть* $T = \{t_1, \ldots, t_k\}, k \geq 1$ *- упорядоченное по индексам множество непустых различных строк. Будем считать, что* T *свободно от подстрок. То есть, для любых различных строк* $t_i, t_j \in T$ *верно, что* $index(t_i, t_j) = index(t_j, t_i) = 0$*. Тогда,* $orderedSuperstrings(T)$ *- это множество таких общих надстрок* T*, что* $\forall i, j$ *т.ч.* $1 \leq i, j \leq k$ *верно:* $i \leq j \iff index(t, t_i) \leq index(t, t_j)$*. И* $superstring(T)$ *- это строка из* $orderedSuperstrings(T)$ *наименьшей длины.*

Теперь заметим, что $superstring(T)$ можно построить жадно:

Теорема 1. *Пусть* $T = \{t_1, \ldots, t_k\}, k \geq 1$ *- упорядоченное по индексам множество непустых строк, свободное от подстрок. Тогда,* $superstring(T) = \{pref(t_1, t_2) + pref(t_2, t_3) + \ldots + pref(t_{k-1}, t_k) + t_k\}$.

Доказательство. Докажем индукцией по k. Для $k = 1$ очевидно(сама строка должна лежать в надстроке и она же является своей надстрокой). Пусть теперь для любого $k \leq K$ лемма верна. Тогда покажем, что она верна и для $k = K + 1$.

Докажем от противного. Пусть g строка построенная жадно, а b - кратчайшая, и $b \neq g$. Пусть $begin(s, t) = index(s, t)$ и $end(s, t) = index(s, t) + |t| - 1$. Заметим, что так как T свободно от подстрок, то:

$$\forall i, j \in \{1, \ldots, k\} \text{ верно, что } begin(i) \leq begin(j) \iff end(i) \leq end(j). \tag{1}$$

Заметим также, что g и b заканчиваются на строку t_k. Для g это верно по построению, а для b из оптимальности. Действительно, если $b = pr + t_k + su$, то суффикс su можно спокойно отбросить, так как он ни на что не влияет(t_k - это последняя строчка из T). Теперь рассмотрим начала этих строк:

Лемма 1. *Строки* g *и* b *должны начинаться с* $superstring(\{t_1, \ldots, t_{k-1}\})$.

Доказательство. Для строки g - очевидно по построению. Пусть b начинается со строки x из множества $orderedSuperstrings(\{t_1, \ldots, t_{k-1}\})$. Очевидно, что x заканчивается на t_{k-1}. А значит, из утверждения (1) следует, что $|over(x, t_k)| \leq |t_{k-1}|$, так как в противном случае t_{k-1} есть подстрока t_k. Проще говоря, наложение x на t_k зависит только от t_{k-1}. Из чего следует, что в качестве x можно брать любую строку из $orderedSuperstrings(\{t_1, \ldots, t_{k-1}\})$, поскольку все они заканчиваются на t_{k-1}. А значит, наиболее оптимально брать именно $superstring(\{t_1, \ldots, t_{k-1}\})$. □

Если для строки z известны её начало x и конец y, то кратчайшая строка такого вида очевидно строится жадно как $z = pref(x, y) + y$. Из этого утверждения и предыдущего замечания следует, что $b = g$. Но они предполагались различными. Противоречие. □

Итак, для доказательства нам потребуется построить граф $G(S)$, где S - исходное множество строк.

Определение 6. $G(S)$ *- полный ориентированный взвешенный граф* $= (V, E)$*, где* $V = \{1, \ldots, n\}$ *при* $n = |S|$*, а вес ребра* $(i, j) = d(s_i, s_j), \forall i, j \in \{1, \ldots, n\}$.

Определение 7. *Пусть дан граф* $G(S)$*, тогда циклом в этом графе называется последовательность его вершин* $c_1, \ldots, c_p$*, где* $c_i \neq c_j$ $\forall i \neq j$*. При этом, длина цикла* $|c| = p$*, а вес* $w(c) = d(s_{c_1}, s_{c_2}) + d(s_{c_2}, s_{c_3}) + \ldots + d(s_{c_{p-1}}, s_{c_p}) + d(s_{c_p}, s_{c_1})$.

Определение 8. *Разбиением графа* $G(S)$ *на циклы называется такое множество циклов* $C = \{c_1, \ldots, c_m\}$*, что* $c_i \cap c_j = \emptyset$ $\forall i \neq j$ *и* $\bigcup\limits_{i=1}^{m} c_i = \{1, \ldots, n\}$ *при* $n = |S|$*. При этом вес разбиения* $w(C) = \sum\limits_{i=1}^{m} w(c_i)$.

Определение 9. *OPT(S) - это длина кратчайшей общей надстроки множества* S.

Лемма 2. *Пусть* C *- разбиение графа* $G(S)$ *на циклы минимального веса(т.ч.* $w(C)$ *минимально). Тогда* $w(C) \leq$ *OPT(S).*

Доказательство. Для начала покажем, что длина кратчайшего гамильтонового цикла в этом графе $\leq$ OPT(S). Пусть s - кратчайшая надстрока. Найдем порядок первых вхождений s_i в s. Пусть это $s_{k_1}, \ldots, s_{k_n}$. Тогда, $w(k_1, \ldots, k_n) \leq |s| = \mathrm{OPT}(S)$. Очевидно, что гамильтонов цикл - это частный случай разбиения графа на циклы. Значит $w(C) \leq w(k_1, \ldots, k_n) \leq \mathrm{OPT}(S)$. □

Определение 10. *Пусть* c *цикл в графе, тогда* $strings(c)$ *- это множество всех различных циклических сдвигов строки* $pref(c_1, c_2) + pref(c_2, c_3) + \ldots + pref(c_{p-1}, c_p) + pref(c_p, c_1)$*, где* $p = |c|$.

Определение 11. *Пусть* c *цикл в графе, тогда* $shifts(c) = |strings(c)|$.

Лемма 3. *Пусть* C *разбиение графа на циклы минимального веса,* c *- некоторый цикл из этого разбиения. Тогда* $w(c) \leq shifts(c)$.

Доказательство. Пусть $s \in strings(c)$ и $s = t^k$(строка t повторённая k раз), $t = s_{1,shifts(c)}, k = |s|/|t|$. И пусть $w(c) > shifts(c)$. Определим t^∞ как бесконечную строку являющуюся последовательной конкатенацией бесконечного числа строк t. Заметим, что s_i подстрока t^∞ $\forall i \in c$. Причем, $\forall ind \geq 0$ $\exists pos$, т.ч. $ind * |t| + 1 \leq pos \leq (ind + 1) * |t|$ и $contains(t^\infty, s_i, pos) = 1$. Вследсвие чего $index(t, s_i) \neq 0, \forall i \in c$. Значит существует цикл c' веса $|t| = shifts(c) < w(c)$, содержащий все вершины из c. Значит, C не является разбиением графа на циклы минимального веса. Противоречие. □

Теорема 2. *Пусть C разбиение графа на циклы минимального веса, c_1, c_2 - два различных цикла из этого разбиения. s_1 строка из цикла c_1, s_2 строка из цикла c_2. Тогда, $|over(s_1, s_2)| < w(c_1) + w(c_2)$.*

Доказательство. Если $|s_1| < w(c_1)$ $[|s_2| < w(c_2)]$, то $|over(s_1, s_2)| < |s_1|[|s_2|] < w(c_1)[w(c_2)] < w(c_1)+w(c_2)$. Иначе, $|s_1| >= w(c_1)$ и $|s_2| >= w(c_2)$. □

4 NP-полнота

Теорема 3. *the1*