Преобразование алгоритма в терминах нормальной схемы алгоритма Маркова в машину Тьюринга

Алгоритм преобразования

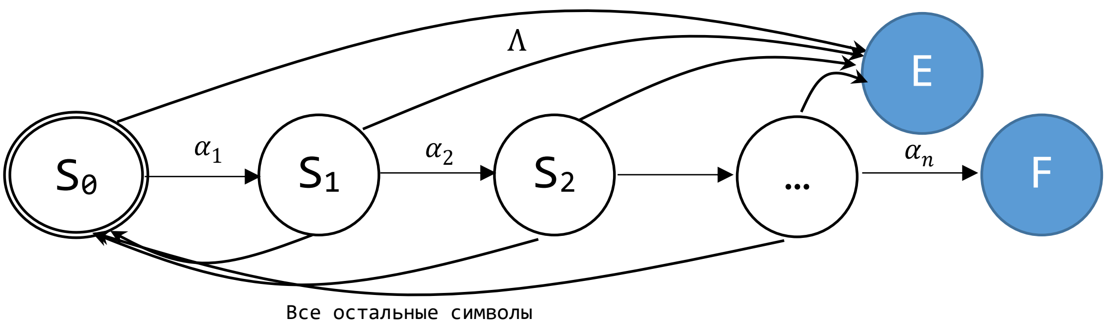
Докажем, что любой алгоритм Маркова можно преобразовать в машину Тьюринга.

Пусть нам дан алгоритм Маркова, определяемый конечным алфавитом Σ, набором преобразований P и конечным преобразованием P_0.

Мы можем построить конечный автомат A, который ищет в строке слово α, а также машину Тьюринга TM, которая заменяет в строке некоторое слово α на некоторое слово β.

Для дальнейших рассуждений введём обозначение q_0 — начальное состояние нашей машины Тьюринга, реализующей данный алгоритм.

Поиск слова



Для слова $\alpha = \alpha_1\alpha_2 \dots \alpha_n$ автомат A примет такой вид:

Его конечное состояние F, означающее, что слово найдено, будет начальным для машины TM, а достижение состояние E будет означать, что автомат дошёл до конца строки, но слова не нашёл (после этого необходимо вернуться в начало строки и перейти на начальное состояние автомата поиска следующего слова.

s_0 α_1 -> s_1 α_1 R

s_1 α_2 -> s_2 α_2 R

…

s_(N-1) α_n -> F α_n E

Также для каждого состояния будет сгенерированы команды:

s Λ -> E Λ L,

отвечающая за переход к состоянию ошибки по достижению конца строки, и набор команд

```
s_i α_j -> s_0 α_j E,
```

где $j \neq i$ для $\forall j \in \Sigma$ отвечающая за возврат в начальное состояние, если попал не тот символ, что ожидался.

Замена слова

В начале работы машина TM будет находиться в конце слова, которое необходимо заменить.

	α_1	α_2	…	α_n	

Поскольку слова α, длина которого n, и β, длина которого m, при построении машины известны, можно рассмотреть три случая:

1) $n = m$

В этом случае нам нужны n команд для замены символов одного слова на другое с конца, где последняя команда также установит состояние машины на q_0 (начальное состояние машины)

2) $n < m$

В этом случае вначале необходимо сдвинуть остаток строки на $(m - n)$ ячеек влево (подробнее о сдвиге ниже), после чего начать запись справа налево нового слова.

3) $n > m$

Необходимо сдвинуть остаток строки на $(m - n)$ ячеек вправо, вернуться в конечную ячейку нового слова и записать его справа налево.

Свдиг «хвоста» строки

Сдвиг остатка строки на ленте машины Тьюринга на n ячеек можно реализовать n повторением сдвига на одну клетку. Рассмотрим сдвиг вправо.

Машина будет находиться в начальном состоянии `q_start` Первой командой машины Тьюринга для свдига строки будет:

```
q_start c -> q_c Λ R,
```

где c — первый символ сдвигаемого «хвоста» строки.

Для каждого символа c алфавита Σ будет сгенерирован набор команд:

```
q_c b -> q_b c R,
```

где b — каждый символ из алфавита $\Sigma - \{c\}$.

Таким образом мы записываем значение символа, которое храним в состоянии машины и запоминаем в состоянии значение, которое было в этой ячейке, сдвигаясь вправо.

Также для каждого символа c алфавита Σ будет сгенерирована команда:

`q_c Λ-> q_end c E,`

где `q_end` — состояние конца строки.

Теперь необходимо вернуться в ячейку, с которой мы начали сдвигать «хвост». Для этого нам понадобится набор команд

`q_end c-> q_end c L,`

для каждого $c \in \Sigma$, что означает, что мы двигаемся влево, пока не встречаем пустой символ:

`q_end Λ-> q_end2 Λ E`

После этого машина приходит в состояние q_end2, означающее конец её работы.

Для сдвига «хвоста» влево все движение влево нужно заменить на движения вправо и наоборот.

Проблема сдвига «хвоста» решена.

Построение машины Тьюринга для замены одного преобразования нормального алгоритма Маркова

Каждое преобразование нормального алгоритма — это замена одной подстроки на другую. Эта задача разбивается на поиск подстроки и на её замену, реализация которых описана выше. Соединим эти два процесса.

Итак, нам необходимо заменить слово $\alpha = \alpha_1\alpha_2 \dots \alpha_n$ на слово $\beta = \beta_1\beta_2 \dots \beta_m$.

Начальное состояние нашей машины обозначим `prod1_s_0`. Построим машину, описанную в разделе «Поиск слова»:

`prod1_s_0` α_1 `->` `prod1_s_1` α_1 `R`

`prod1_s_1` α_2 `->` `prod1_s_2` α_2 `R`

…

`prod1_s_(n-1)` α_n `->` `prod1_found` α_n `E`

$\forall \mathrm{prod1}\_s \in S, \forall \alpha_i, \alpha_j \in \Sigma, j \neq i$, где S — набор состояний машины поиска.

prod1_s Λ -> prod1_error Λ L,

prod1_s_i α_j -> prod1_s_0 α_j E,

Нам понадобятся также обработка состояния ошибки prod1_error:

prod1_error Λ -> prod2_s_0 Λ R

$\forall \alpha \in \Sigma$:

prod1_error α -> prod1_error α L,

где prod2_s_0 — начальное состояние машины поиска следующего преобразования алгоритма Маркова.

Теперь необходимо запустить замену слова α на слово β. Мы находимся в ячейке с символов α_n в состоянии prod1_found.

Если длина второго слова больше первого на $(m - n) = d$ символов, то необходимо d раз сдвинуть «хвост» строки вправо. Построим машину Тьюринга, сдвигающую «хвост», с начальным состоянием prod1_found.

Первая команда нашей машины будет:

prod1_found α_n -> prod1_shift1_c Λ R,

где $c = \alpha_n$.

Затем будет набор команд:

prod1_shift1_c b -> prod1_shift1_b c R,

Где b — каждый символ из алфавита алгоритма.

Последней командой будет

prod1_shift1_c Λ-> prod1_shift1_end c E,

где prod1_shift1_end — состояние конца строки.

Теперь необходимо вернуться в ячейку, с которой мы начали сдвигать «хвост». Для этого нам понадобится набор команд

prod1_shift1_end c-> prod1_shift1_end c L,

для каждого $c \in \Sigma$, что означает, что мы двигаемся влево, пока не встречаем пустой символ:

prod1_shift1_end Λ-> prod1_shift1_end2 Λ E

После этого машина приходит в состояние q_end2, означающее конец её работы.

В случае, если необходимо сдвинуть «хвост» более, чем на одну позицию, переходим на начальное состояние аналогичной машины `prod1_shift2_start`.

`prod1_shift2_start c -> prod1_shift2_c Λ`

Далее аналогично.

Для последнего сдвига состояние `prod1_shiftL_end2` будет начальным для машины, записывающей слово β:

`prod1_shiftL_end2 Λ -> prod1_write_m` β_m `L`

`prod1_write_m * -> prod1_write_(m-1)` β_{m-1} `L,`

где * — любой символ (значит, таких команд будет столько же, сколько символов в алфавите).

Машина закончит свою работу в состоянии `prod1_write_1`. Вернём каретку на начало строки.

`prod1_write_1 * -> prod1_write_1 * L`,

где * — любой символ (значит, таких команд будет столько же, сколько символов в алфавите).

Если преобразование алгоритма Маркова, которое мы разбираем, не конечное, установим начальное состояние машины.

`prod1_write_1 Λ -> prod1_s_0 Λ R`

Если преобразование конечное («с точкой»), установим конечное состояние машины:

`prod1_write_1 Λ -> q_end Λ R`

Построение машины Тьюринга, выполняющей нормальный алгоритм Маркова

Чтобы получить конечную машину Тьюринга, выполняющую заданный алгоритм, необходимо каждое преобразование алгоритма Маркова «разобрать» на машину Тьюринга для поиска и замены слов данного преобразования.

Начальным состоянием всей машины будет `prod1_s_0` — начальное состояние машины поиска слова из первого преобразования. Переход к последующему преобразованию будет происходить через состояние `prodX_error`, означающему, что искомое слово в строке не найдено, и нужно искать следующее.

Таким образом, можно построить такую машину Тьюринга TM, что $TM = \{Q, \Sigma, E, q_0, F\}$, где Q — конечный набор состояний машины, складываемый из состояний всех автоматов поиска и машин замены слова, Σ — алфавит данного алгоритма Маркова, E — конечный набор преобразований машины Тьюринга, складывающийся из описанных выше преобразований для каждого преобразования нормального алгоритма, q_0 — начальное состояние машины (`prod1_s_0`), F — набор финальных состояний машины Тьюринга.

Литература

1. CHEN Yuanmi “Markov Algorithm”, 2007 (https://www.irif.fr/~carton/Enseignement/Complexite/ENS/Redaction/2007-2008/yuanmi.chen.pdf)