ВикипедиЯ

Алгоритм Кнута — Морриса — Пратта

Материал из Википедии — свободной энциклопедии

Алгоритм Кнута — Морриса — Пратта (КМП-алгоритм) — эффективный алгоритм, осуществляющий поиск подстроки в строке. Время работы алгоритма линейно зависит от объёма входных данных, то есть разработать асимптотически более эффективный алгоритм невозможно.

Алгоритм был разработан Д. Кнутом и В. Праттом и, независимо от них, Д. Моррисом[1]. Результаты своей работы они опубликовали совместно в 1977 году[2].

Содержание

Постановка задачи
Идея
Описание алгоритма и оценка времени работы
См. также
Примечания
Ссылки

Постановка задачи

Даны образец (строка) S и строка T. Требуется определить индекс, начиная с которого образец S содержится в строке T. Если S не содержится в T — вернуть индекс, который не может быть интерпретирован как позиция в строке (например, отрицательное число). При необходимости отслеживать каждое вхождение образца в текст имеет смысл завести дополнительную функцию, вызываемую при каждом обнаружении образца.

Идея

Алгоритм Ахо — Корасик также позволяет искать одну строку за линейное время. Но слабое место этого алгоритма — конечный автомат, который в явном виде строится за $O(|needle|\cdot|\Sigma|)$ операций и требует столько же памяти.

Если искать всего одну строку, каждое состояние будет иметь только один «прямой» переход. Побочные же переходы будем вычислять динамически, никак их не кэшируя.

```
если haystack[i] = needle[state]
   то state = state + 1
   иначе state = побочный_переход(state, haystack[i])
```

Легко заметить, что суффиксные ссылки алгоритма Ахо — Корасик представляют собой префикс-функцию искомого шаблона.

Описание алгоритма и оценка времени работы

Рассмотрим сравнение строк на позиции i, где образец $S[0, m-1]$ сопоставляется с частью текста $T[i, i+m-1]$. Предположим, что первое несовпадение произошло между $T[i+j]$ и $S[j]$, где $1 < j < m$. Тогда $T[i, i+j-1] = S[0, j-1] = P$ и $a = T[i+j] \neq S[j] = b$.

При сдвиге вполне можно ожидать, что префикс (начальные символы) образца S сойдется с каким-нибудь суффиксом (конечные символы) текста P. Длина наиболее длинного префикса, являющегося одновременно суффиксом, есть значение префикс-функции от строки S для индекса j.

Это приводит нас к следующему алгоритму: пусть $\pi[j]$ — значение префикс-функции от строки $S[0, m-1]$ для индекса j. Тогда после сдвига мы можем возобновить сравнения с места $T[i+j]$ и $S[\pi[j]]$ без потери возможного местонахождения образца. Можно показать, что таблица π может быть вычислена (амортизационно) за $\Theta(m)$ сравнений перед началом поиска. А поскольку строка T будет пройдена ровно один раз, суммарное время работы алгоритма будет равно $\Theta(m+n)$, где n — длина текста T.

См. также

- Z-функция
- Алгоритм Бойера — Мура

Примечания

1. *Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К.* Алгоритмы: построение и анализ = Introduction to Algorithms / Под ред. И. В. Красикова. — 2-е изд. — М.: Вильямс, 2005. — 1296 с. — ISBN 5-8459-0857-4.
2. Donald Knuth; James H. Morris, Jr, Vaughan Pratt (1977). “Fast pattern matching in strings” (http://citeseer.ist.psu.edu/context/23820/0). *SIAM Journal on Computing*. **6** (2): 323—350. DOI:10.1137/0206024 (https://doi.org/10.1137%2F0206024).

Ссылки

- Алгоритм Кнута-Морриса-Пратта (http://algolist.manual.ru/search/esearch/kmp.php) на сайте Algolist, перевод работы Thierry Lecroq, Christian Charras, Knuth-Morris-Pratt algorithm (http://www-igm.univ-mlv.fr/~lecroq/string/node8.html) // Цикл лекций Exact String Matching Algorithms, Université de Rouen, 1997
