Temat: Metoda haszowania przy wyszukiwaniu wzorca w tekście (algorytm Karpa-Rabina).

W celu znalezienia wzorca **p** w łańcuchu **s** algorytm naiwny porównuje każdorazowo zawartość okna wzorca ze wzorcem. Prowadzi to do kwadratowej klasy złożoności obliczeniowej $\mathbf{O(n^2)}$. W **algorytmie Karpa-Rabina** postępujemy nieco inaczej. Najpierw odwzorowujemy poszukiwany wzorzec **p** w liczbę całkowitą $\mathbf{H_p}$ za pomocą tzw. funkcji haszującej (ang. hash function). Funkcja haszująca posiada taką własność, iż identyczne łańcuchy znakowe odwzorowuje w identyczne liczby – hasze. Następnie ustawiamy okno wzorca na początku tekstu i haszujemy je przy pomocy tej samej funkcji w liczbę całkowitą $\mathbf{H_s}$. Porównujemy ze sobą liczby $\mathbf{H_p}$ i $\mathbf{H_s}$. Jeśli są równe, to oznacza to, iż wzorzec i jego okno są do siebie podobne z dokładnością do haszów. Aby mieć pewność, iż są identyczne, sprawdzamy je znak po znaku, podobnie jak w algorytmie naiwnym, lecz teraz nie wykonujemy tego każdorazowo, tylko wtedy, gdy jest zgodność haszy. Jeśli hasze nie są zgodne (lub chcemy znaleźć następne wystąpienia wzorca), to okno wzorca przesuwamy o jedną pozycję w obrębie łańcucha **s**, ponownie haszujemy zawartość okna i otrzymaną nową liczbę $\mathbf{H_s}$ porównujemy z haszem wzorca $\mathbf{H_p}$. Zatem cała procedura się powtarza. Gdy okno wzorca wyjdzie poza łańcuch, przeszukiwanie przerywamy.

Funkcja haszująca najczęściej traktuje przetwarzany łańcuch tekstu jak liczbę, zapisaną przy wybranej podstawie (zwykle podstawa jest równa rozmiarowi alfabetu). Kolejne znaki tekstu są traktowane jak cyfry tej liczby. Dodatkowo w celu uniknięcia zbyt dużych wartości haszu stosowana jest arytmetyka modularna – wyniki wszystkich działań są sprowadzane do reszty z dzielenia przez wybrany moduł, który jest liczbą pierwszą. Aby zrozumieć zasadę wyznaczania haszu, przyjrzyjmy się prostemu przykładowi.

Przykład:

Alfabet składa się z trzech liter: **A**, **B** i **C**.

Wyznaczyć hasz dla tekstu: "**CBBAB**".

Jako bazę systemu liczbowego przyjmiemy 3, ponieważ z tylu liter składa się alfabet. Cyfry posiadają wartości: **A**=0, **B**=1 i **C**=2. Jako moduł przyjmiemy liczbę pierwszą 23. Wtedy:

$$\mathbf{H_{[CBBAB]} = (2\times3^4+1\times3^3+1\times3^2+0\times3^1+1\times3^0) \bmod 23}$$
$$\mathbf{H_{[CBBAB]} = (2\times81+1\times27+1\times9+0\times3+1\times1) \bmod 23}$$
$$\mathbf{H_{[CBBAB]} = (162+27+9+0+1) \bmod 23}$$
$$\mathbf{H_{[CBBAB]} = 199 \bmod 23}$$
$$\mathbf{H_{[CBBAB]} = 15}$$

Funkcja haszująca powinna być tak dobrana, aby w prosty sposób można było wyznaczyć hasz okna wzorca po przesunięciu o jedną pozycję w obrębie przeszukiwanego łańcucha **s**. Podana w poprzednim przykładzie funkcja spełnia ten warunek.

Przykład:

Tekst ma postać "**CBBABB**". Wyznaczyliśmy hasz pierwszych pięciu liter, czyli $\mathbf{H_{[CBBAB]}} = 15$. Teraz przesuwamy okno o jedną pozycję w prawo. Okno zawiera tekst "**BBABB**". Wyznaczymy $\mathbf{H_{[BBABB]}}$ na podstawie poprzednio wyznaczonego haszu $\mathbf{H_{[CBBAB]}}$.

Najpierw pozbywamy się z haszu najstarszej cyfry, czyli **C** = 2. W tym celu wyznaczamy mnożnik $d = 3^4 \bmod 23 = 12$. Od haszu $\mathbf{H_{[CBBAB]}}$ odejmujemy modularnie iloczyn **d** i cyfry **C:**

$$\mathbf{H = (H_{[CBBAB]}-d\times2) \bmod 23}$$
$$\mathbf{H = (15-12\times2) \bmod 23}$$
$$\mathbf{H = (-9) \bmod 23}$$
$$\mathbf{H = 14}$$

Teraz otrzymany hasz **H** mnożymy modularnie przez 3 – spowoduje to przesunięcie w nim wszystkich cyfr o jedną pozycję w lewo:

$$\mathbf{H = (H \times 3) \bmod 23}$$
$$\mathbf{H = (14 \times 3) \bmod 23}$$
$$\mathbf{H = 42 \bmod 23}$$
$$\mathbf{H = 19}$$

Na koniec do haszu **H** dodajemy modularnie nową cyfrę **B**:

$$\mathbf{H = (H+1) \bmod 23}$$
$$\mathbf{H = (19+1) \bmod 23}$$
$$\mathbf{H = 20 \bmod 23}$$
$$\mathbf{H = 20}$$

Sprawdźmy, czy otrzymaliśmy $\mathbf{H_{[BBABB]}}$:

$$\mathbf{H_{[BBABB]} = (1 \times 3^4 + 1 \times 3^3 + 0 \times 3^2 + 1 \times 3^1 + 1 \times 3^0) \bmod 23}$$
$$\mathbf{H_{[BBABB]} = (81+27+0+3+1) \bmod 23}$$
$$\mathbf{H_{[BBABB]} = 112 \bmod 23}$$
$$\mathbf{H_{[BBABB]} = 20 = H}$$

Zwróć uwagę, iż wyznaczenie haszu okna po przesunięciu wymaga stałej liczby operacji, zatem jest wykonywane w czasie stałym **O(1)**.

Moduł powinien być względnie dużą liczbą, aby ograniczyć liczbę fałszywych zgodności haszów wzorca i jego okna.

Typowo algorytm Karpa-Rabina pracuje w liniowej klasie złożoności obliczeniowej **O(m+n)**, zatem dla długich tekstów ma on zdecydowaną przewagę nad algorytmem naiwnym.

Algorytm Karpa-Rabina wyszukiwania wzorca

Wejście:

s – przeszukiwany łańcuch.

p – poszukiwany wzorzec.

Wyjście:

Kolejne pozycje wystąpień wzorca w łańcuchu.

Elementy pomocnicze:

i : położenie okna wzorca **p** w przeszukiwanym łańcuchu **s**; **i** ∈ **N**.

m : długość wzorca **p**; **m** ∈ **N**.

n : długość łańcucha **s**; **n** ∈ **N**.

$\mathbf{H_p}$: hasz wzorca**.**

$\mathbf{H_s}$: hasz okna wzorca.

h(**x**) : wybrana funkcja haszująca.

Lista kroków:

K01: **m** ← |**p**| **; obliczamy liczbę znaków wzorca**

K02: **n** ← |**s**| **; oraz liczbę znaków łańcucha**

K03: $\mathbf{H_p}$ ← **h**(**p**) **; obliczamy hasz wzorca**

K04: $\mathbf{H_s}$ ← **h**(**s**[0:**m**]) **; obliczamy hasz okna**

K05: **i** ← 0 **; ustawiamy pozycję okna**

K06: **Jeśli** $\mathbf{H_s \neq H_p}$, **; sprawdzamy równość haszy**
to idź do kroku K08

K07: **Jeśli p** = **s**[**i**:**i**+**m**], **; znaleziono wzorzec p w s**
to przetwarzaj i **; na pozycji i-tej**

K08: **i** ← **i**+1 **; przesuń okno wzorca o jedną pozycję**
; w prawo w łańcuchu s

K09: **Jeśli i** = **n**-**m**, **; koniec łańcucha?**
to zakończ

K10: **Oblicz nowe H_s** **; wyznaczamy hasz przesuniętego okna,**
dla **s** [**i**:**i**+**m**] **; wykorzystując poprzedni hasz**
K11: **Idź** do kroku K06 **; kontynuujemy pętlę**

PROGRAM – WSTĘP

Program generuje 79 znakowy łańcuch zbudowany z pseudolosowych kombinacji liter **A**, **B** i **C** oraz 4 znakowy wzorzec wg tego samego schematu. Funkcja haszująca ma postać:
$h(\mathbf{s}) = \mathbf{c}\times3^3+\mathbf{c}\times3^2+\mathbf{c}\times3^1+\mathbf{c}\times3^0$,
gdzie **c** oznacza cyfrę trójkową uzyskaną z liter łańcucha przez odjęcie od ich kodu liczby 65.
Ponieważ 4 cyfrowa liczba trójkowa posiada największą wartość $2222_{(3)} = 80_{(10)}$ i mieści się bez problemu w 32 bitowym typie całkowitym, zrezygnowaliśmy z arytmetyki modulo (nie musimy wykonywać dzielenia modulo dla tak krótkiego wzorca i ograniczonego alfabetu do 3 liter).

KOD PROGRAMU

ZADANIA DO WYKONANIA

Zadanie 1

Napisz program, który znajdzie wszystkie wystąpienia wzorca w tekście składającym się z liter: X, Y, Z. Program automatycznie wygeneruje tekst o długości maksymalnie 15 znaków oraz wzorzec o długości maksymalnie 3 znaków z podanego zbioru liter. Algorytm powinien korzystać z haszowania.

Zadanie 2

Napisz program, który znajdzie wszystkie wystąpienia wzorca w tekście składającym się z liter: Q, R, S, T. Program automatycznie wygeneruje tekst o długości maksymalnie 30 znaków oraz wzorzec o długości maksymalnie 2 znaków z podanego zbioru liter. Algorytm powinien korzystać z haszowania.

Zadanie 3

Napisz program, który znajdzie wszystkie wystąpienia wzorca w tekście składającym się z liter: U, V, W, X. Program automatycznie wygeneruje tekst o długości maksymalnie 100 znaków oraz wzorzec o długości maksymalnie 4 znaków z podanego zbioru liter. Algorytm powinien korzystać z haszowania.

Zadanie 5*

Napisz program, który znajdzie wszystkie wystąpienia wzorca w tekście składającym się z liter: M, N, O, P. Program automatycznie wygeneruje tekst o długości maksymalnie 25 znaków oraz wzorzec o długości maksymalnie 3 znaków z podanego zbioru liter. Hashe wzorca i fragmentów tekstu powinny być obliczane z użyciem dzielenia modulo przez liczbę pierwszą 31.

Zadanie 6*

Napisz program, który znajdzie wszystkie wystąpienia wzorca w tekście składającym się z liter: A, B, C, D, E. Program automatycznie wygeneruje tekst o długości maksymalnie 50 znaków oraz wzorzec o długości maksymalnie 4 znaków z podanego zbioru liter. Hashe wzorca i fragmentów tekstu powinny być obliczane z użyciem dzielenia modulo przez liczbę pierwszą 37.

* - zadania dodatkowe