Zadanie 3

Patryk Lisik

30 Styczeń 2023

Streszczenie

Programista ma napisać kod obliczający wartości funkcji $H(n)$ określonej następującymi równaniami:

$$
\begin{aligned}
H(0) &= 1 \\
H(1) &= 3 \\
H(n) &= [1H(n-1)][2 + H(n-2]
\end{aligned}
$$

Ponieważ jest on zafascynowany pomysłem rekurencji, to natychmiast napisał następujący kod:

```
function H(n)
    if n ⩽ 1 then
        return 2·n + 1
    else
        return (1+h(n-1))·(2+h(n-2))
    end if
end function
```

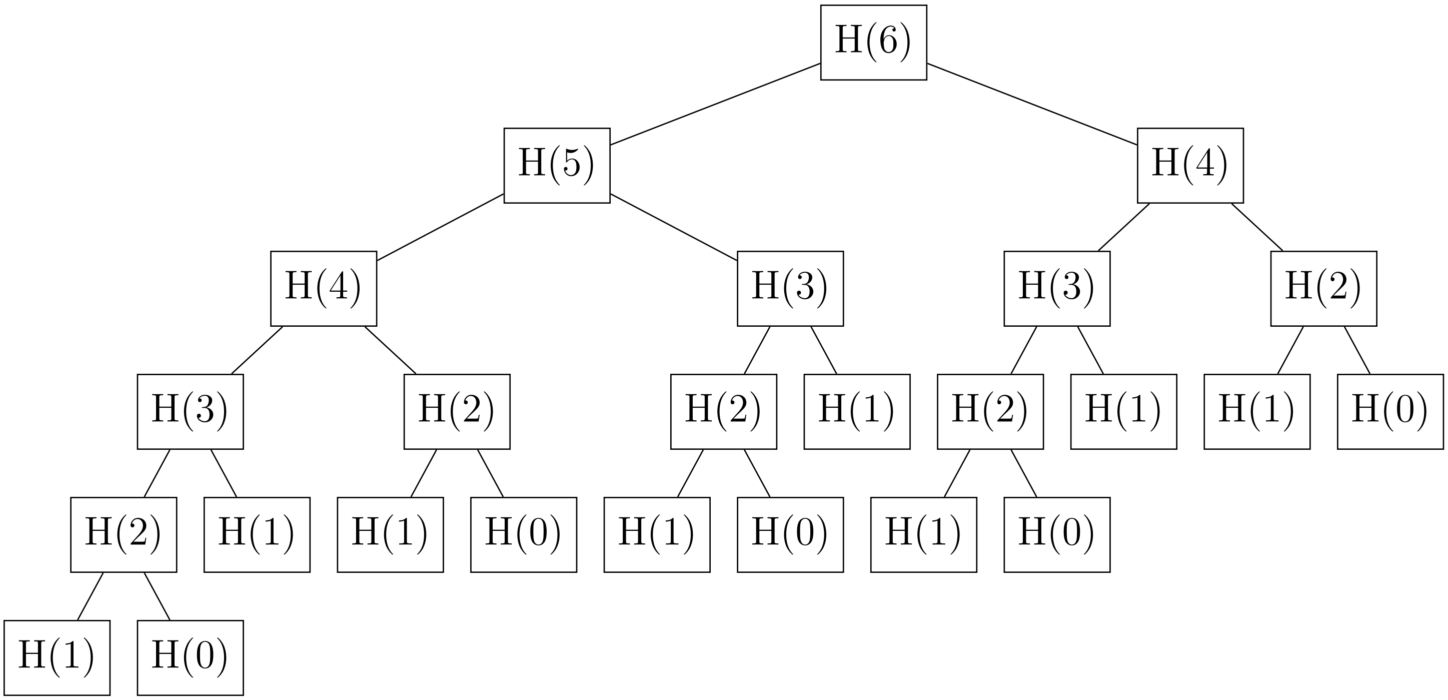
Pomimo swej elegancji i prostoty, algorytm ten jest niewydajny dla dużych wartości n

(a) Zbuduj drzewo wszystkiech wywołań funkcji podczas wykonywania wywołania $H(6)$. Oblicz całkowitą liczbę wywołań funkcji dla $H(6)$.

(b) Niech a_n będzie całkowitą liczba wywołań funkcji przy obliczaniu H(n). Napisz i rozwiąż związek rekurencyjny spełniany przez a_n.

(c) Przepisz kod dla $H(n)$ tak, aby był on iteracyjny i wydajny.

a) Drzewo wywołań



Rysunek 1: Drzewo wywołań funkcji H

Jak widać na rysunku 1 ilość wywołań dla $H(6) = 25$

b) Związek rekurencyjny ilości wywołań

Ilość wywołań można zdefiniować jako sumę ilości wywołań na poprzednich poziomach dodać obecny poziom. Jest to rekurencja liniowa niejednorodna drugiego rzędu.

$$a_0 = 1$$
$$a_1 = 1$$
$$a_n = a_{n-2} + a_{n-1} + 1$$

Rozwiążmy rekurencję liniową jednorodną $a_n^o = a_{n-2} + a_{n-1}$. Równanie charakterystyczne

$$r^2 = r + 1$$
$$r_1 = \frac{1+\sqrt{5}}{2}$$
$$r_2 = \frac{1-\sqrt{5}}{2}$$

Rekurencja a_n^o ma rozwiązanie postaci

$$a_n^o = \alpha_1 r_1^n + \alpha_2 r_2^n$$

Znajdujemy szczególne rozwiązanie dla rekurencji niejednorodnej.

$$\begin{aligned}
&b_n = b_{n-2} + b_{n-1} + 1 \\
&b_n = \beta_0 \\
&\beta_0 = \beta_0 + \beta_0 + 1 \\
&\beta_0 = -1
\end{aligned}$$

Sumujemy ogólne rozwiązanie rekurencji jednorodnej a_n^o i rozwiązanie szczeólne części niejednorodnej.

$$a_n = \alpha_1 \left(\frac{1+\sqrt{5}}{2}\right)^n + \alpha_2 \left(\frac{1-\sqrt{5}}{2}\right)^n - 1$$

Podstawiamy znane stałe i obliczamy α_1 i α_2

$$\begin{cases} a_0 = 1 \\ a_1 = 1 \end{cases}$$

$$\begin{cases} a_0 = \alpha_1 r_1^0 + \alpha_2 r_2^0 - 1 = \alpha_1 + \alpha_2 - 1 \implies \alpha_2 = -\alpha_1 + 2 \\ a_1 = \alpha_1 r_1^1 + \alpha_2 r_2^1 - 1 = \alpha_1 \frac{1+\sqrt{5}}{2} + \alpha_2 \frac{1-\sqrt{5}}{2} - 1 \end{cases}$$

$$1 = \alpha_1 \frac{1+\sqrt{5}}{2} + (-\alpha_1 + 2)\frac{1-\sqrt{5}}{2} - 1$$

$$2 = \alpha_1 \frac{1+\sqrt{5}}{2} + \alpha_1 \frac{\sqrt{5}-1}{2} + 1 - \sqrt{5}$$

$$1 = \alpha_1 (\frac{1+\sqrt{5}}{2} + \frac{\sqrt{5}-1}{2}) - \sqrt{5}$$

$$1 = \alpha_1 \sqrt{5} - \sqrt{5}$$

$$\alpha_1 = \frac{1+\sqrt{5}}{\sqrt{5}} = \frac{5+\sqrt{5}}{5}$$

$$\begin{cases} \alpha_1 = \frac{\sqrt{5}+5}{5} \\ \alpha_2 = -\alpha_1 + 2 = \frac{5-\sqrt{5}}{5} \end{cases}$$

Finalnie

$$a_n = \frac{\sqrt{5}+5}{5} \left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{5-\sqrt{5}}{5} \left(\frac{1-\sqrt{5}}{2}\right)^n - 1$$

c) Implementacja iteracyjna

Iteracyjna implementacja funkcji H w języku Python 3

```python
def H(x):
    if x == 0:
        return 1
    if x == 1:
        return 3
    h_0 = 1
    h_1 = 3
    h=None
    for _ in range(x-1):
        h = (1+h_1)*(2+h_0)
        h_0 = h_1
        h_1 = h
    return h
```