Václav Snášel; Jiří Bouchala; Petr Vodstrčil

Kouzlo Fibonacciho kódování

Pokroky matematiky, fyziky a astronomie, Vol. 61 (2016), No. 3, 234–242

Persistent URL: http://dml.cz/dmlcz/145849

Kouzlo Fibonacciho kódování

Václav Snášel, Jiří Bouchala, Petr Vodstrčil, Ostrava

Abstrakt. V článku se budeme zabývat Fibonacciho kódováním, které je díky proměnlivé délce kódu vhodnější (obzvláště pro kódování posloupnosti malých čísel) než např. užití dvojkové soustavy. Ukážeme, jak lze efektivně a bez použití obřích tabulek předzpracovaných dat Fibonacciho kódy dekódovat.

1. Úvod

V tomto článku bychom chtěli prezentovat jeden zajímavý problém, který vznikl v průběhu řešení reálné úlohy [11]. Cílem bylo zrychlení práce s datovými strukturami, které tvoří základ současných moderních databází. Jednou z možností, jak tohoto zrychlení docílit, je snížit pomocí vhodné komprese množství přenášených dat mezi vnější a vnitřní pamětí. Je zřejmé, že pro tento úkol potřebujeme najít algoritmus, který bude umět dekomprimovat tak rychle, aby čas potřebný k načtení a následné dekompresi zkomprimovaných dat byl menší než čas potřebný k přečtení dat nezkomprimovaných.

Pro kompresi posloupnosti malých čísel, které se při práci s různými datovými strukturami často vyskytují (viz např. [2], [3], [9]), se s výhodou používá kódování s proměnlivou délkou kódu [10], tzn. kódování nevyžadující pro jednotlivá čísla předem pevně stanovený počet bitů. Ve svém řešení jsme se inspirovali Fibonacciho kódováním [12].

Pro standardní kompresní algoritmy používající kódování čísel s proměnlivou délkou kódu je dekomprese časově velmi náročná. Vývoj nových metod dekomprese je proto jedním z důležitých témat současné informatiky. První algoritmy rychlé dekomprese pro Fibonacciho kódování byly publikovány v [4], [5], [6]. Tyto algoritmy vyžadují poměrně velké tabulky pro předzpracované výpočty, a jsou tudíž nepoužitelné pro kódování větších čísel.

V následující části ukážeme, co je Fibonacciho kód a jak ho lze využít při kódování posloupnosti přirozených čísel. Dále představíme algoritmus, pomocí něhož lze takovouto zakódovanou posloupnost efektivně dekódovat.

2. Fibonacciho kódování

Uvažujme Fibonacciho posloupnost $\{F_k\}_{k=-2}^{\infty}$ definovanou rekurentně:

$$F_{-2} = 0,\ F_{-1} = 1,\ F_k = F_{k-2} + F_{k-1} \quad \text{pro} \quad k \in \{0, 1, 2, \dots\},$$

tzn.

$$\{F_k\}_{k=-2}^{\infty} = (0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \dots).$$

Prof. RNDr. Václav Snášel, CSc., Doc. RNDr. Jiří Bouchala, Ph.D., Mgr. Petr Vodstrčil, Ph.D., FEI VŠB-TU Ostrava, 17. listopadu 15, 708 33 Ostrava-Poruba, e-mail: vaclav.snasel@vsb.cz, petr.vodstrcil@vsb.cz, jiri.bouchala@vsb.cz

Fibonacciho čísla mají mnoho zajímavých vlastností, kterých lze využít v řadě aplikací. Velmi dobrým přehledem vlastností Fibonacciho čísel je článek [8] a kniha [7].

Fibonacciho kódování je založeno na následující větě (viz [12]).

Věta 1 (**Zeckendorfova**). *Každé přirozené číslo n lze, a to jednoznačně, vyjádřit ve tvaru*

$$n = \sum_{k=0}^{\infty} a_k F_k, \tag{1}$$

kde pro každé $k \in \mathbb{N}_0$ je $a_k \in \{0, 1\}$ a současně $a_k a_{k+1} = 0$ (tzn. že v posloupnosti nul a jedniček $a_0, a_1, a_2, a_3, \ldots$ nejsou dvě jedničky za sebou).

Důkaz. Nejdříve dokažme, a to konstruktivně pomocí tzv. hladového algoritmu, existenci takového vyjádření. Buď $n \in \mathbb{N}$ dáno. Označme $p \in \mathbb{N}_0$ největší index takový, že $F_p \leq n$, a položme $a_p = 1$. Nyní uvažujme nezáporné číslo $n - F_p$. Pokud je toto číslo nulové, položíme $a_q = 0$ pro každé $q \neq p$. Pokud $n - F_p > 0$, lze určit největší index $r \in \mathbb{N}_0$ takový, že $F_r \leq n - F_p$. Všimněme si, že $r < p - 1$ (v opačném případě totiž $F_{p-1} \leq F_r \leq n - F_p$, odkud plyne, že $F_{p+1} = F_{p-1} + F_p \leq n$, a to je ve sporu s volbou p) a položme $a_r = 1$. A situace se opakuje, tentokrát s nezáporným číslem $n - F_p - F_r \ldots$ Tímto postupem získáme po konečně mnoha krocích všechny nenulové (tj. rovné jedné) koeficienty a_k.

Nyní se soustřeďme na důkaz jednoznačnosti takového vyjádření. Nejdříve dokažme, a to indukcí, pomocné lemma.

Lemma 2. *Nechť pro každé $k \in \mathbb{N}_0$ je $a_k \in \{0, 1\}$ a současně $a_k a_{k+1} = 0$. Pak pro každé $m \in \mathbb{N}_0$ platí*

$$\sum_{k=0}^{m} a_k F_k < F_{m+1}.$$

Důkaz lemmatu. Pro $m \in \{0, 1\}$ je tvrzení triviální. Předpokládejme, že dokazované tvrzení platí pro všechna přirozená čísla menší nebo rovná číslu $m \in \mathbb{N}$. Máme dokázat, že

$$\sum_{k=0}^{m+1} a_k F_k < F_{m+2}.$$

Je-li $a_{m+1} = 0$, není co dokazovat, protože pak

$$\sum_{k=0}^{m+1} a_k F_k = \sum_{k=0}^{m} a_k F_k < F_{m+1} < F_{m+2}.$$

Je-li $a_{m+1} = 1$, je nutně $a_m = 0$, a proto

$$\sum_{k=0}^{m+1} a_k F_k = \sum_{k=0}^{m-1} a_k F_k + F_{m+1} < F_m + F_{m+1} = F_{m+2}.$$

Lemma je dokázáno.

Nyní se vraťme k důkazu jednoznačnosti rozkladu (1). Předpokládejme pro důkaz sporem, že existuje číslo $n \in \mathbb{N}$ se dvěma různými rozklady. Odtud snadno plyne, že existují posloupnosti $\{a_k\}$, $\{b_k\}$ a $m \in \mathbb{N}$ takové, že

- $\sum_{k=0}^{m} a_k F_k = \sum_{k=0}^{m} b_k F_k$,
- $\forall k \in \mathbb{N}_0 : \ a_k, b_k \in \{0, 1\}$,
- $\forall k \in \mathbb{N}_0 : \ a_k a_{k+1} = 0 = b_k b_{k+1}$,
- $a_m = 1$, $b_m = 0$.

Odtud a z lemmatu 2 pak vyplývá, že

$$F_m \leq \sum_{k=0}^{m} a_k F_k = \sum_{k=0}^{m} b_k F_k = \sum_{k=0}^{m-1} b_k F_k < F_m,$$

a to je spor. □

Na základě právě dokázané věty můžeme přistoupit k definici Fibonacciho kódu.

Definice 3. Fibonacciho kódem čísla $n \in \mathbb{N}$ rozumíme takovou (jednoznačně určenou) posloupnost nul a jedniček

$$F(n) = a_0 a_1 a_2 \dots a_m$$

ukončenou jedničkou, v níž se neobjeví dvě jedničky vedle sebe a pro niž platí

$$n = \sum_{k=0}^{m} a_k F_k.$$

Je-li $a_0 a_1 a_2 \dots a_m$ Fibonacciho kódem čísla $n \in \mathbb{N}$, budeme psát

$$V(a_0 a_1 a_2 \dots a_m) = n.$$

Podle předchozí definice tedy platí

$$V\big(F(n)\big) = \sum_{k=0}^{m} a_k F_k = n.$$

Příklady.

- $F(2) = 01, \ V(01) = 2$,
- $F(3) = 001, \ V(001) = 3$,
- $F(12) = 10101, \ V(10101) = 12$,
- $F(1591) = 0101001010101010, \ V(0101001010101010) = 1591$.

Skutečnost, že v žádném Fibonacciho kódu se nemohou objevit dvě jedničky za sebou a že každý Fibonacciho kód je ukončen jedničkou, lze využít při kódování posloupnosti přirozených čísel: vložíme jedničku za každé zakódované číslo. Pokud pak při dekódování narazíme na dvě jedničky za sebou, druhá z nich má význam pouze „oddělovače“. Vše se snad vyjasní příkladem:

$$10101101010010101010111011001100110011\ \ldots$$
$$= \underbrace{10101}_{12}1\underbrace{0101001010101011}_{1591}1\underbrace{01}_{2}1\underbrace{01}_{2}1\underbrace{001}_{3}1\underbrace{001}_{3}1\ \ldots\ .$$

3. Efektivní dekódování Fibonacciho kódů

Pro efektivní dekódování by bylo užitečné načítat (a průběžně dekódovat) zakódovanou posloupnost po větších blocích než jen po jednotlivých bitech. Ukážeme si, jak ji lze dekódovat – například – po bytech (čti „bajtech“), tj. po osmi bitech:

$$10101101\|01001010\|10101101\|10110011\|0011\ \ldots\ . \tag{2}$$

Zde bude nutné se vypořádat s problémem, že jednotlivá čísla mohou být v kódu umístěna „napříč“ byty, viz např. číslo 1591 v uvedeném kódu:

$$101011\underbrace{01\|01001010\|10101}_{1591}101\|10110011\|0011\ \ldots\ .$$

Řešení je založeno na následujícím pozorování:

$$V(010100101010101)$$
$$= \underbrace{\underbrace{V(01)}_{2} + \underbrace{V(\overbrace{\mathbf{00}}^{N_2:=2}\,01001010)}_{81}}_{N_1:=2+81} + \underbrace{V(\overbrace{\mathbf{00\ 00000000}}^{N_2:=N_2+8=2+8}10101)}_{1508} = N_1 + 1508 = 1591. \tag{3}$$

Dále ukážeme, jak efektivně počítat hodnoty (o odpovídající počty N_2 nul) „posunutých Fibonacciho kódů“.

Definice 4. Buď $p \in \mathbb{N}$. Posunem Fibonacciho kódu $F(n) = a_0a_1a_2\ldots a_m$ o p pozic rozumíme kód

$$F(n;p) = \underbrace{00\ldots0}_{p\text{-krát}}a_0a_1a_2\ldots a_m.$$

Dále označme

$$F(n;0) = F(n),\ V\big(F(n;-1)\big) = \sum_{k=0}^{m} a_k F_{k-1}.$$

Věta 5 (o hodnotě posunutého kódu). *Nechť $n \in \mathbb{N}$. Pak pro každé $p \in \mathbb{N}_0$ platí*

$$V\big(F(n;p)\big) = F_{p-1} \cdot V\big(F(n)\big) + F_{p-2} \cdot V\big(F(n;-1)\big). \tag{4}$$

Důkaz. Důkaz provedeme indukcí podle p. Nejdříve si všimněme, že tvrzení platí pro $p = 0$:

$$\begin{aligned} V\big(F(n;0)\big) = V\big(F(n)\big) &= 1 \cdot V\big(F(n)\big) + 0 \cdot V\big(F(n;-1)\big) \\ &= F_{-1} \cdot V\big(F(n)\big) + F_{-2} \cdot V\big(F(n;-1)\big) \end{aligned}$$

a pro $p = 1$:

$$\begin{aligned} V\big(F(n;1)\big) = \sum_{k=0}^{m} a_k F_{k+1} &= \sum_{k=0}^{m} a_k \big(F_k + F_{k-1}\big) = \sum_{k=0}^{m} a_k F_k + \sum_{k=0}^{m} a_k F_{k-1} \\ &= V\big(F(n)\big) + V\big(F(n;-1)\big) = 1 \cdot V\big(F(n)\big) + 1 \cdot V\big(F(n;-1)\big) \\ &= F_0 \cdot V\big(F(n)\big) + F_{-1} \cdot V\big(F(n;-1)\big). \end{aligned}$$

Nyní předpokládejme, že $p \geq 2$ a že pro každé $q \in \mathbb{N}_0,\ q < p$, platí

$$V\big(F(n;q)\big) = F_{q-1} \cdot V\big(F(n)\big) + F_{q-2} \cdot V\big(F(n;-1)\big). \tag{5}$$

Pak

$$\begin{aligned} V\big(F(n;p)\big) = \sum_{k=0}^{m} a_k F_{k+p} &= \sum_{k=0}^{m} a_k \big(F_{k+p-1} + F_{k+p-2}\big) \\ = \sum_{k=0}^{m} a_k F_{k+p-1} + \sum_{k=0}^{m} a_k F_{k+p-2} &= V\big(F(n;p-1)\big) + V\big(F(n;p-2)\big) \\ = F_{p-2} \cdot V\big(F(n)\big) + F_{p-3} \cdot V\big(F(n;-1)\big) + F_{p-3} \cdot V\big(F(n)\big) + F_{p-4} \cdot V\big(F(n;-1)\big) \\ = \big(F_{p-2} + F_{p-3}\big) \cdot V\big(F(n)\big) + \big(F_{p-3} + F_{p-4}\big) \cdot V\big(F(n;-1)\big) \\ = F_{p-1} \cdot V\big(F(n)\big) + F_{p-2} \cdot V\big(F(n;-1)\big). \end{aligned}$$

□

Nyní se vraťme k úkolu dekódovat danou posloupnost po bytech. Právě načtenému bytu vždy přiřadíme osmici čísel $M = [M_1, M_2, \ldots, M_8]$ takovou, že

- $M_1, M_8 \in \{0,1\}$ jsou hodnoty 1. a 8. bitu,
- $M_2, \ldots, M_5$ odpovídají dekódovaným číslům nacházejícím se v daném bytu, tj. odděleným druhou jedničkou za sebou (pokud takovýchto čísel není v daném bytu plný počet, doplníme sérii nulami),
- M_6 je částečně dekódované číslo (tzn. v daném bytu neukončené dvěma jedničkami), které je zapsané pomocí M_7 bitů.

Aby nedošlo k nedorozumění, uveďme několik příkladů:

- $\underbrace{10101}_{M_2=12}1\,\underbrace{01}_{M_6=2} \longrightarrow M = [M_1, M_2, \ldots, M_8] = [1, 12, 0, 0, 0, 2, 2, 1],$
- $\underbrace{01001010}_{M_6=31} \longrightarrow M = [M_1, M_2, \ldots, M_8] = [0, 0, 0, 0, 0, 31, 8, 0],$
- $\underbrace{01}_{M_2=2}1\,\underbrace{001}_{M_3=3}1\,\underbrace{0}_{M_6=0} \longrightarrow M = [M_1, M_2, \ldots, M_8] = [0, 2, 3, 0, 0, 0, 1, 0],$
- $11111111 \longrightarrow M = [M_1, M_2, \ldots, M_8] = [1, 1, 1, 1, 1, 0, 0, 1],$
- $00000000 \longrightarrow M = [M_1, M_2, \ldots, M_8] = [0, 0, 0, 0, 0, 0, 8, 0].$

Před zpracováním právě načteného bytu potřebujeme znát z předchozího dekódování tři čísla (viz (3)): částečně přečtené číslo N_1, posun N_2 a číslo $N_3 \in \{0,1\}$, které udává hodnotu posledního bitu z předchozího bytu (a pomocí kterého budeme moci rozhodnout, zda případná jednička na začátku právě čteného bytu není jen dříve zmíněný „oddělovač“). Zpracováním (dekódováním) právě načteného bytu pak rozumíme (případné) doplnění již dekódovaných čísel $x_1, x_2, x_3, \ldots, x_k$ a předefinování trojice $N = [N_1, N_2, N_3]$. Schematicky lze algoritmus znázornit takto:

$$\boxed{\begin{matrix} x_1, x_2, x_3, \ \ldots, x_k \\ N = [N_1, N_2, N_3] \end{matrix}} \quad \begin{matrix} [M_1, M_2, \ldots, M_8] \\ \rightsquigarrow \end{matrix} \quad \boxed{\begin{matrix} x_1, x_2, x_3, \ \ldots, x_k, \ \ldots \\ N = [N_1, N_2, N_3] \end{matrix}} \quad \rightsquigarrow \ \ldots$$

Chceme-li popsat jeden krok tohoto algoritmu, musíme rozlišit několik případů:

1. $\boxed{N_3 \cdot M_1 = 0, \ M_2 = 0}$

   V takovém případě seznam $x_1, x_2, x_3, \ldots, x_k$ ničím nedoplňujeme a N změníme na

$$N = \begin{cases} [N_1 + V\big(F(M_6; N_2)\big), N_2 + M_7, M_8] \text{ , je-li } M_6 \neq 0, \\ [N_1, N_2 + M_7, M_8] \text{ , je-li } M_6 = 0. \end{cases}$$

2. $\boxed{N_3 \cdot M_1 = 0,\ M_2 \neq 0}$

   Nyní seznam $x_1, x_2, x_3, \ \ldots, x_k$ doplníme o číslo $N_1 + V\big(F(M_2; N_2)\big)$ a o ta z čísel M_3, M_4, M_5, která jsou nenulová. N změníme na

$$N = [M_6, M_7, M_8].$$

3. $\boxed{N_3 \cdot M_1 = 1}$

   Zde budeme pracovat místo s osmicí $M = [M_1, M_2, \ldots, M_8]$ odpovídající bytu

$$1b_2b_3 \ldots b_8\,, \text{ kde } b_i \in \{0, 1\},$$

   s osmicí čísel

$$\widetilde{M} = [\widetilde{M}_1, \widetilde{M}_2, \ldots, \widetilde{M}_8]$$

   odpovídající bytu

$$b_2b_3 \ldots b_80.$$

   Seznam $x_1, x_2, x_3, \ \ldots, x_k$ pak doplníme o číslo N_1 a o ta z čísel $\widetilde{M}_2, \widetilde{M}_3, \widetilde{M}_4$, která jsou nenulová, a N změníme na

$$N = [\widetilde{M}_6, \widetilde{M}_7 - 1, b_8].$$

Ilustrujme právě popsaný algoritmus na konkrétním příkladě.

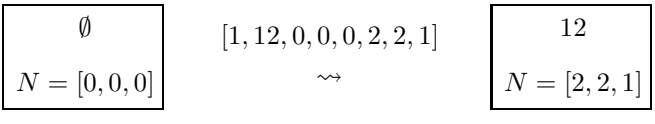
Příklad. Ukažme, jak probíhá dekódování po bytech kódu

$$10101101\big\|01001010\big\|10101101\big\|10110011\big\|0011\ \ldots$$

Na začátku je seznam již dekódovaných čísel prázdný a $N = [0, 0, 0]$. Postupně budeme načítat byte po bytu, doplňovat dekódovaný seznam a aktualizovat trojici N.

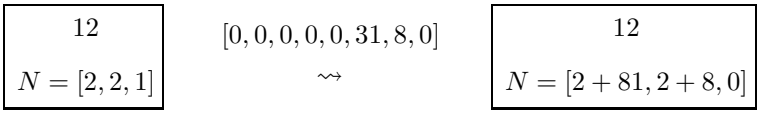
1. byte

$$\mathbf{10101101}\big\|01001010\big\|10101101\big\|10110011\big\|0011\ \ldots$$



2. byte

$$10101101\big\|\mathbf{01001010}\big\|10101101\big\|10110011\big\|0011\ \ldots$$



3. byte

$$10101101\|01001010\|\mathbf{10101101}\|10110011\|0011\ \ldots$$

12 $N=[83,10,0]$	$[1,12,0,0,0,2,2,1]$ $\rightsquigarrow$	$12,83+1508$ $N=[2,2,1]$

4. byte

$$10101101\|01001010\|10101101\|\mathbf{10110011}\|0011\ \ldots$$

$12,1591$ $N=[2,2,1]$	10110011 $\downarrow$ 01100110 $\widetilde{M}=[0,2,3,0,0,0,1,0]$ $\rightsquigarrow$ $\ldots$	$12,1591,2,2,3$ $N=[0,0,1]$

Uvědomme si, že pro praktickou realizaci popsaného algoritmu potřebujeme znát pouze Fibonacciho čísla, hodnoty $V\big(F(n;-1)\big)$ pro každé přirozené číslo n, jehož Fibonacciho kód má nejvýše 8 bitů (tzn. pro každé $n \leq 54$), a pro každý z 256 různých bytů konkrétní instrukce, co je třeba na základě výše popsaného algoritmu provést (např. pro byte 00101101 instrukce zní: doplň sérii dekódovaných čísel o $N_1 + V\big(F(11;N_2)\big)$ a změň N na $N=[2,2,1]$). Tyto informace si lze předpřipravit v tabulce.

Zájemce o podrobnější informace odkazujeme na článek [11], kde lze (mimo jiné) najít výsledky experimentů prováděných na náhodných i reálných datech. Tyto experimenty ukazují, že dekódování Fibonacciho kódu pomocí námi popsaného algoritmu je výrazně efektivnější než dekódování konvenční, tj. po jednotlivých bitech.

Poděkování. Tato práce byla podpořena grantem SGS SP2016/108, VŠB-Technická univerzita Ostrava.

L i t e r a t u r a

[1] Apostolico, A., Fraenkel, A.: *Robust transmission of unbounded strings using Fibonacci representations*. IEEE Trans. Inform. Theory *33* (1987), 238–245.

[2] Bayer, R., McCreight, E.: *Organization and maintenance of large ordered indexes*. Acta Inform. *1* (1972), 173–189.

[3] Guttman, A.: *R-trees: a dynamic index structure for spatial searching*. Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, Annual Meeting, ACM Press, Boston, 1984, 47–57.

[4] Klein, S. T.: *Fast decoding of Fibonacci encoded texts*. Proceedings of the International Data Compression Conference, DCC 2007, IEEE Computer Society, 2007, 388.

[5] Klein, S. T., Ben-Nissan, M. K.: *Using Fibonacci compression codes as alternatives to dense codes*. Proceedings of the International Data Compression Conference, DCC 2008, IEEE Computer Society, 2008, 472–481.

[6] Klein, S. T., Ben-Nissan, M. K.: *On the usefulness of Fibonacci compression codes.* The Computer Journal *53 (6)* (2010), 701–716.

[7] Koshy, T.: *Fibonacci and Lucas numbers with applications.* John Wiley, 2001.

[8] Křížek, M., Luca F., Somer, L.: *Aritmetické vlastnosti Fibonacciových čísel.* PMFA *50 (2)* (2005), 127–140.

[9] Salomon, D.: *Data compression: the complete reference.* 3rd ed., Springer-Verlag, New York, 2004.

[10] Salomon, D.: *Variable-length codes for data compression.* Springer-Verlag, 2007.

[11] Walder, J., Krátký, M., Bača, R., Platoš, J., Snášel, V.: *Fast decoding algorithms for variable-lengths codes.* Inf. Sci. *183 (1)* (2012), 66–91.

[12] Zeckendorf, E.: *Représentation des nombres naturels par une somme de nombres de Fibonacci ou de nombres de Lucas.* Bull. Soc. Roy. Sci. Liège *41* (1972), 179–182.