SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

LINEÁRNE REKURENTNÉ POSTUPNOSTI
VÝPOČET PERIÓDY RIEŠENIA

Semestrálna práca

Vypracovali: Peter Čuřík, Pavol Sobota

Obsah

1 Formálne definície, výpočet

Úvodom do riešenia problematiky, budú definície operácií, s ktorými pracujeme. Na konkrétnom príklade si všeobecne ukážeme, čo jednotlivé operácie predstavujú.

1.1 Lineárna rekurentná postupnosť

Nech k je kladné celé číslo, a nech a, a_0, ... , a_{k-1} sú prvky dané konečným poľom $\mathbf{F_q}$. Potom postupnosť s_0, s_1, ..., s_{k-1} prvkov z $\mathbf{F_q}$, ktorá spĺňa vzťah:

$$s_{n+k} = a_{k-1}s_{n+k-1} + a_{k-2}s_{n+k-2} + ... + a_0s_n + a \text{ pre } n = 0,1 ...$$

sa nazýva *lineárne rekurentná postupnosť (k-teho rádu)* v $\mathbf{F_q}$. Podmienky s_0, s_1, ..., s_{k-1}, ktoré určujú zvyšok postupnosti unikátne, sa zvyknú nazývať ako *počiatočné podmienky*. Lineárne rekurentný vzťah, v ktorom prvok **a** = 0 nazývame homogénny lineárne rekurentný vzťah; ak **a** != 0 tak hovoríme o nehomogénnom lineárne rekurentnom vzťahu (1).

1.2 Perióda lineárne rekurentnej postupnosti

Nech S je neprázdna množina a nech s_0, s_1, ... je postupnosť prvkov pre S. Ak existuje celé číslo **r** > 0 a n_0 >= 0 také, že $\mathbf{s_{n+r} = s_n}$ pre všetky n >= n_0, tak potom sa postupnosť nazýva *ultimátne periodická* a **r** sa nazýva *perióda* postupnosti. Najmenšie možné číslo medzi všetkými možnými periódami v ultimátne periodickej postupnosti sa nazýva *najmenšia perióda* postupnosti (1).

Lemma:

(i) Každá perióda ultimátne periodickej postupnosti je deliteľná najmenšou periódou.

(ii) Nech $\mathbf{F_q}$ je konečné pole a **k** je kladné celé číslo. Potom každá lineárne rekurentná postupnosť k-teho rádu je ultimátne periodická s najmenšou periódou r spĺňajúcu r <= q^k , a r <= q^k -1 ak je postupnosť homogénna (1).

1.3 Konkrétne príklady

Fibonacciho postupnosť je postupnosť, v ktorej každý prvok je súčtom dvoch predchádzajúcich (2).

Formálny predpis Fibonacciho postupnosti:

$$F_n = F_{n-2} + F_{n-1}$$

Našou úlohou je určiť najmenšiu periódu pre Fibonacciho postupnosť v konečných poliach $\mathbf{F_q}$ pre q = (2,3,5,7).

Postup je nasledovný:
Za počiatočné prvky si dosadíme pre $F_{n-2} = 0$ a pre $F_{n-1} = 1$, potom je Fib. postupnosť v poli:

F(2): 0, 1, 1, **0**, **1**, 1, 0, 1, 1, ...
a perióda r = 3.
F(3): 0, 1, 1, 2, 0, 2, 2, 1, **0**, **1**, 2, 0, 2, 2, 1, 0, 1...
perióda r = 8.
F(5): 0, 1, 1, 2, 3, 0, 3, 3, 1, 4, 0, 4, 4, 3, 2, 0, 2, 2, 4, 1, **0**, **1**, ...
perióda r = 20.
F(7): 0, 1, 1, 2, 3, 5, 1, 6, **0**, **1**, ...
perióda r = 8

1.4 Zhodnotenie

Z uvedených príkladov je zrejmý fakt, že ak vo výpočte prvkov postupnosti natrafíme na sekvenciu prvkov zhodných s prvkami počiatočných podmienok, tak nasledujúce prvky už nebudú unikátne, ale budú sa opakovať. Táto podmienka platí aj vo všetkých ostatných lineárne rekurentných postupnostiach. Pričom číslo označujúce poradie prvku, ktorý sa nachádzal ako posledný pred prvým opakovaním prvkov počiatočných podmienok je zhodné s najmenšou periódou **r**.

2 Algoritmizácia všeobecného výpočtu

V nasledujúcich kapitolách vysvetlíme, aké algoritmy sme vytvorili pre výpočet periódy postupnosti. Výpočet otestujeme na konkrétnom príklade.

2.1 Pole, predpis

Predmetom našej práce je nájsť najmenšiu periódu **r** pre akýkoľvek homogénnu lineárne rekurentnú postupnosť v poli **F(2)**. Táto postupnosť môže nadobudnúť tvar:

$$s_{n+k} = a_{k-1}s_{n+k-1} + a_{k-2}s_{n+k-2} + \ldots + a_0 s_n + a \text{ pre } n = 0,1\ldots \text{ a } a = \{0,1\}$$

To znamená, že koeficienty **a** môžu nadobúdať hodnoty 0 a 1. Hodnota **k** môže byť lubovoľné celé číslo a pravá strana predpisu postupnosti môže nadobúdať ľubovolný počet prvkov.

Pre každú postupnosť vieme tento predpis extrahovať zo zápisu, aby nám určoval, súčet ktorý predošlých hodnôt tvorí momentálne číslo.

Napríklad pre postupnosť $s_{n+4} = s_{n+2} + s_n$ bude tento predpis **δ** tvorený číslami [-2,-4].

2.2 Počiatočné podmienky

Pre každú postupnosť k-teho rádu potrebujeme vytvoriť 0 ... k - 1 počiatočných podmienok, aj keď nie sú v prvotnom predpise postupnosti. Je to z toho dôvodu, aby sme vedeli vytvoriť všetky po sebe nasledujúce prvky postupnosti. Prvotné podmienky nemusia byť vždy definované ako napríklad vo Fibonacciho postupnosti, a tak ich musíme vytvoriť ako všetky kombinácie hodnôt pre prvky predpisu.

Riešením pre tento problém je vytvorenie matice **A**, ktorej riadky hodnoty pre všetky prvky predpisu postupnosti s_{n+k-1} , s_{n+k-2} , … s_n. Stĺpce budú tvorené všetkými kombináciami týchto hodnôt c_1, c_2, … c_{2^k}. V praxi to môže pre postupnosť $s_{n+3} = s_{n+1} + s_n$ vyzerať nasledovne.

$$A = \begin{matrix} S_0 \\ S_1 \\ S_2 \end{matrix} \begin{matrix} 0\,1\,0\,1\,0\,1\,0\,1 \\ 0\,0\,1\,1\,0\,0\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1 \end{matrix}$$

2.3 Výpočet periódy

Ak máme vytvorený predpis **δ** a maticu **A**, náš výpočet vyzerá nasledovne. Začneme prvkom s_{n+k}, pre ktorého vyčíslenie sú dostatočné počiatočné podmienky. V matici **A** vytvoríme nový riadok pre s_{n+k} a naplníme ho súčtom predchádzajúcich hodnôt v každom stĺpci. Predchádzajúce hodnoty sú určené na základe predpisu **δ**. Tento krok opakujeme pre každý nasledujúci prvok postupnosti, až dokým sa nám v matici **A** nevytvorí sekvencia riadkov, ktoré sú zhodné s prvotnými podmienkami. Potom poradie riadku, po ktorom sa táto sekvencia vytvorila bude tvoriť najmenšiu periódu **r**. Pre príklad z predošlej kapitoly bude výpočet vyzerať nasledovne.

$$\begin{matrix} S_3 \\ S_4 \\ S_5 \end{matrix} \begin{matrix} 0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,1\,1\,1\,1\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,1 \end{matrix}$$

A=	S_6	0 1 0 1 1 0 1 0	perióda **r** = 7
	S_7	**0 1 0 1 0 1 0 1**	
	S_8	**0 0 1 1 0 0 1 1**	
	S_9	**0 0 0 0 1 1 1 1**	

3 Programová implementácia

V nasledujúcich kapitolách sú uvedené pseudo-algoritmické vyjadrenia pre každý algoritmus predstavujúci konkrétnu výpočtovú funkcionalitu v našom implementovanom systéme. Systém je v originálnej implementácii vytvorený v jazyku Java Script, pre webové spracovanie.

3.1 Získanie predpisu δ

Predpokladáme, že užívateľ zadal rekurentnú postupnosť v tvare $s_k = s_{k-a} + s_{k-b}$ ... Čísla k, k-a, k-b ... sme extrahovali a vložili do premennej **input**.

Algoritmus:

```
k = input[0]
FOR i : 1 … LENGTH OF input
     ADD input[i] – input[0] → predpis
```

3.2 Generovanie prvotných podmienok

Na základe získaného rádu **k** z predošlého kroku vytvoríme maticu **A** a naplníme ju všetkými kombináciami vstupov.

Algoritmus:

```
CREATE A[k][2^k]
j ← 0
FOR i : 0 … k-1
     WHILE j < 2^k
          FOR b : 0 ... 1
               FOR c : 0 ... 2^i
                    A[i][j] ← b
```

```
                    INCREMENT j
          j ← 0
```

Pre zaujímavosť:

(i) Veľkosť rádu **k** pre postupnosť ovplyvňuje zložitosť a rýchlosť algoritmu. Algoritmus má výpočtovú zložitosť $O(k*2^k)$.

(ii) Ak by sme pracovali v inom poli ako F(2), algoritmus by mal zložitosť $O(k*q^k)$ pre **F(q)**.

3.3 Kontrola matíc zhodných s prvotnými podmienkami

Kontrola má za úlohu zistiť, či sa vo výpočte prvkov postupnosti nevyskytla postupnosť zhodná s prvotnými podmienkami. Vtedy môžme výpočet ukončiť a vyjadriť číslo **r** ako najmenšiu periódu. Kontrola je volaná vždy po vypočítaní i-teho riadku.

```
FUNCTION INTEGER kontrolaRiadkov (číslo riadku)
     i ← číslo riadku
     IF A[i] EQUALS A[k-1] THEN
          state ← TRUE
     for a : 2 … k+1
          IF A[i+1-a] NOT EQUALS A[k-a]
               state ← FALSE
     IF state EQUALS TRUE THEN
          r = i + 1 – a
          RETURN r
```

3.4 Hlavný výpočet

Úlohou tohto algoritmu je cyklické počítanie riadkov predstavujúcich prvky postupnosti v matici **A** na základe predpisu postupnosti. Algoritmus vykonáva tento výpočet, dovtedy pokým sa nenájde v postupnosti perióda. Na zistenie, či sme našli periódu používame algoritmos z predošlej kapitoly.

```
     r ← 0 (r : perióda)
     i ← k (k : rád postupnosti)

     WHILE r EQUALS 0
          FOR j : 0 … 2^k-1
```

```
        sucet ← 0
        FOR c : 0 … LENGTH OF predpis – 1
              sucet ← sucet + A[i + predpis[c]][j]
sucet ← sucet MOD 2
        ADD sucet → A[i][j]
r ← kontrolaRiadkov(i)
INCREMENT i
```

3.5 Obmedzenia impementácie

Z dôvodu pretečenia zásobníka alebo dlhého výpočtového času sme obmedzili rád postupností na najviac **10**. V teoretickom stroji by bol výpočet periódy za použitia týchto algoritmov možný pre akýkoľvek rád postupnosti. Užívateľ zadáva ako vstup koeficienty k pre postupnosť. Ich zadanie musí byť **zostupne** od najväčšej po najmenšiu, inak program nie je spustiteľný.

Záver

Našou úlohou bolo vytvoriť systém schopný vypočítať najmenšiu periódu pre lineárne rekurentné postupnosti v poli F(2). Problematiku sme preskúmali z teoretickej perspektívy a taktiež sme naštudovali metodiku výpočtu. Pre výpočet periódy postupnosti akéhokoľvek tvaru sme navrhli algoritmy. Jednotlivé algoritmy sme implementovali v jazyku Java Script a vytvorili stránku (3) s aplikáciou vykonávajúcu tento výpočet.

Odkazy

1. **LIDL, R. a NIEDERREITER, H.** *Introduction to finite fields and their applications.* s.l. : Cambridge University Press, 1986. ISBN 0-521-30706-6.

2. **Alan Koch, Chrissy Franzel, Chuya Guo, Rose Psalmond, Shan Shan, Hilary Tobiasz, Meina Zhou.** *Periods of sequences given by linear recurrence relations mod p.* [Online] 17. 9 2013. http://facultyweb.kennesaw.edu/lritter/Koch_talkSPSU2013.pdf.

3. **Čuřík, Peter a Sobota, Pavol.** Algoritmus nájdenia periódy v konečnom poli F(2). [Online] https://ral.netlify.app/.