

KSI 2014/2015

Úloha 1-1: Karlík si chce hrát!

Jan Horáček

Gymnázium, Brno, Vídeňská 47; jan.horacek@seznam.cz

1. listopadu 2014

1 Úvod

Ve svém řešení jsem vyšel z metody binárního vyhledávání - metody půlení intervalů - kterou jsem přizpůsobil situaci, kdy není možné průběžně ukládat levé a pravé hranice intervalu.

Z principu metody půlení intervalů bude výsledný algoritmus dosahovat časové složitosti $O(\log n)$, kde n je velikost Karlíkova intervalu.

Nyní podrobně popíši algoritmus, který jsem pro řešení zvolil.

2 Popis algoritmu

Tělo funkce *hadej* začíná jednoduchou podmínkou řešící první hádání - v případě, že jsem ještě nehádal, tipnu si číslo uprostřed hádaného intervalu. To plyne z metody půlení intervalů.

Ve všech dalších krocích stojím před problémem získání mezí intervalu pomysleného binárního vyhledávání, protože si tyto meze nikam nemohu ukládat. Při řešení tohoto problému jsem vyšel z přesné hodnoty argumentu *minuleHadani*.

2.1 Určení mezí

Hodnota argumentu *minuleHadani* v sobě skrývá, na kolik částí jsem si interval již rozdělil. Po zakreslení několika modelových situací jsem pro relativní šířku nového intervalu ($\frac{1}{n}$, kde n je velikost intervalu), ve kterém se chystám hádat, odvodil vztah 1.

$$sirka_noveho_intervalu = \frac{NSD(do - od, minuleHadani - od)}{do - od} \quad (1)$$

kde $NSD(a, b)$ je funkce vracející největší společný dělitel čísel a a b .

Po prvním hádání je výsledek vztahu 1 číslo $\frac{1}{2}$ (tedy budu hádat v intervalu poloviční velikost, než je původní Karlíkův interval), v další iteraci $\frac{1}{4}$ (tedy budu hádat v intervalu čtvrtinové velikost, než je původní Karlíkův interval) atd. Z této hodnoty jsem odvodil nový tip: hodnotu *sirka_noveho_intervalu* vydělím číslem 2 (tím se dostanu do relativního středu nového intervalu) a násobím velikostí intervalu (tím získám tzv. absolutní posun vůči - v programu proměnná *bound*). Absolutní posun přičtu (resp. odečtu) k (*od*) *minuleHadani* (podle toho, jestli je Karlíkova hodnota větší, nebo menší, než

mnou tipovaná), tím získám absolutní pozici středu nového intervalu. Tento "tip" uložím do proměnné *ret*.

Získal jsem tak číslo, které je přesně v polovině nového intervalu i bez explicitní znalosti hranic tohoto intervalu.

2.2 Ukončovací podmínka

Nepříjemnou vlastností tohoto algoritmu je to, že pracuje s racionálními čísly, ale Karlíkem tipované číslo je zaručeně číslo celé (alespoň doufám - v zadání se to explicitně nepíše...). Klíčovou vlastností algoritmu je tedy schopnost rozhodnout se, kdy aktuálně tipované číslo po zaokrouhlení nutně musí být číslo Karlíkovo. Zaokrouhlovat v každém kroku si nemůžeme dovolit, protože bychom ztratili přesnost, na základě které počítáme *sirka_noveho_intervalu*.

Ze vztahů pro součet nekonečné geometrické posloupnosti (naše *sirka_noveho_intervalu* nabývá v extrémní situaci - když je hledaná hodnota úplně na kraji Karlíkova intervalu - právě hodnot geometrické posloupnosti) a ostatně i z úvah o algoritmu u číselné osy plyne, že v každé iteraci (resp. volání funkce *hlelej*) se hledaná hodnota může nacházet maximálně ve dvojnásobku *sirka_noveho_intervalu* ve směru, o kterém nám Karlík prohlásil, že se tam nachází.

Podle směru tedy vypočteme hodnotu proměnné *potential*, což je minimální (resp. maximální) číslo, kterého může Karlíkova hodnota nabývat. Určení této hodnoty plyne z metody půlení intervalů, resp. z toho, jakou část intervalu jsem již ořízl.

Pokud je poslední tipovaná hodnota po zaokrouhlení rovna maximální odchýlené hodnotě (proměnná *potential*) po zaokrouhlení, lze s jistotou říci, že hádaná hodnota nikdy nedosáhne jiné celočíselné hodnoty, než je hodnota *minuleHadani*. Algoritmus si je tedy v tomto momentě jistý, že ví, co Karlík tipoval, a může si dovolit ztratit přesnost racionální čísel. Vráti tedy zaokrouhlenou vstupní pozici, která je Karlíkem tipovanou hodnotou.

3 Závěr

V Karlíkových testovacích podmínkách, kde je 9000 čísel, dosahuje výše zmíněný algoritmus výsledku průměrně v 15 krocích, nejvýše pak v krocích 16. To je docela slušná časová složitost vzhledem k tomu, že $\log_2(9000) \doteq 13.1$.