Ako rýchlo násobiť z hlavy a O tom ako počítače rátajú sínus

Peter Csiba, petherz@gmail.com, http://www.csip.sk?p=652

05.06.2011

Contents

1 Úvod

V tomto texte sa zaoberáme metódami ako zefektívniť naše bežné počítacie postupy. V prvej časti si ukážeme zopár trikov, ako sa dá v niektorých prípadoch rýchlejšie z hlavy násobiť. Následne si popíšeme nejaké postupy ako efektívnejšie vykonávať niektoré zložitejšie operácie.

Keďže v dnešnej dobe pre nás väčšinu počítacích úkonov vykonávajú počítače, tak sa v druhej časti budeme zaoberať ako výpočty počítačov zrýchliť. Toto zrýchlenie si demonštrujeme na počítaní hodnôt funkcie $sin(x)$ pomocou algoritmu CORDIC.

2 Ľudia

V nasledujúcom sa budeme zaoberať základnými trikmy ako rýchlejšie násobiť dve čísla z hlavy v niektorých špeciálnych prípadoch. Jediný spôsob ako sa naučiť tieto postupy je, že ich začnete trénovať. Najľahšie sa donútite, ak prestanete používať kalkulačku na príliš triviálne operácie (násobenie trojciferných čísel a menších).

2.1 Násobenie konštantami

Nasledujúce algoritmy sú v angličtine, ktorá je natoľko triviálne, že ju nebudem prekladať.

- 10: triviálne
- 2: triviálne
- 4: 2×2
- 5: Multiply by 10 and divide by 2.
- 6: Sometimes multiplying by 3 and then 2 is easy.
- 9: Multiply by 10 and subtract the original number.
- 11 - sčítavame dve po sebe idúce cifry
- 12: Multiply by 10 and add twice the original number.
- 13: Multiply by 3 and add 10 times original number.
- 14: Multiply by 7 and then multiply by 2
- 15: Multiply by 10 and add 5 times the original number, as above.
- 16: You can double four times, if you want to. Or you can multiply by 8 and then by 2.
- 17: Multiply by 7 and add 10 times original number.
- 18: Multiply by 20 and subtract twice the original number (which is obvious from the first step).
- 19: Multiply by 20 and subtract the original number.
- 24: Multiply by 8 and then multiply by 3.
- 27: Multiply by 30 and subtract 3 times the original number (which is obvious from the first step).
- 45: Multiply by 50 and subtract 5 times the original number (which is obvious from the first step).

- 90: Multiply by 9 (as above) and put a zero on the right.
- 98: Multiply by 100 and subtract twice the original number.
- 99: Multiply by 100 and subtract the original number.

2.2 Násobenie čísel končiacimi 5kou

$$(10a+5)(10b+5) = 100ab + 50(a+b) + 25$$

.

- Ak $a = b$, tak dostávame pomerne známy vzťah $(10a+5)(10a+5) = 100(a(a+1)) + 25$, napríklad $25 \times 25 = 625$.
- Ak $a + b$ je párne, tak je výsledok $100(ab + (a+b)/2) + 25$
- Ak je $(a+b)$ nepárne tak je výsledok $100(ab + (a+b-1)/2) + 75$.

Tento postup je jednoduchý preto, lebo vieme výsledné číslo generovať postupne, a skoro nič si nepotrebujeme pamätať.

2.3 Upravovanie násobeného výrazu

Ak je menší z výrazov delitelný nejaký číslom ktoré je ľahké deliť, tak sa nám často oplatí tohoto deliteľa presunúť do väčšieho čísla.

$625 \times 16 = 1250 \times 8 = 2500 \times 4 = 10000$

2.4 Umocňovanie na druhú

Umocňovať čísla končiace nulou alebo päťkou je triviálne. Ďalej máme vzorce, ktoré nám umožnujú jednoducho vyrátať druhú mocninu čísla, ak vieme druhú mocninu čísla líšiaceho sa o -2, -1, 1 alebo 2. Takže, ak vieme umocňovať čísla končiace 5 a 0, tak pomocou týchto trikov vieme efektívne umocňovať hocijaké celé čísla na druhú.

- $(n-2)^2 = n^2 - 4(n-1)$
- $(n-1)^2 = n^2 - (n + (n-1))$
- $(n+1)^2 = n^2 + (n + (n+1))$
- $(n+2)^2 = n^2 + 4(n+1)$

2.5 Násobenie blízkych čísel

$$(a+b)(a-b) = a^2 - b^2$$

. Ak vieme umocňovať čísla rýchlo, tak sa nám často oplatí použiť takýto postup. Napr. $37 * 43 = 1600 - 9 = 1591$.

Existuje druhá možnosť, ak súčet posledných cifier čísla je 10 (je to špeciálny prípad posledného).

$$(10a+b)(10(a+k) + (10-b)) = 100a(a+k+1) + 10bk + b(10-b)$$

. Čo vyzerá ako pomerne zložitý vzorec, ale je jednoducho zapamätateľný. Vynásobíme prvé časti čísel (jedno posunuté o 1) na posledné dve miesta pripéšeme súčin posledných cifier a ešte pripočítame $10bk$ ktoré je v prípade $k = 0$ ozaj triviálne. Takže napríklad pre $112 \times 118 = 11 \times 12 \times 100 + 2 \times 8 = 13216$.

2.6 Rozklad na prvočísla

Bohužiaľ* sa to vo všeobecnosti nedá robiť jednoduchšie ako rozoberaním všetkých možností. Môžeme si ale mierne uľahčiť prácu. Ak postupne hľadáme čo najmenšie prvočíselné delitele, stačí nám vyskúšať prvočísla po $\sqrt{(}N)$ kde N je číslo, ktoré sme chceli na začiatku rozložiť. Pri tomto sa nám hodí vedieť rýchlo modulovať (klasické kritériá na 2,3,5,11) a vedieť rýchlo deliť. Ja používam postup, kde sa snažím dostať na posledné miesto nulovú cifru a potom si to predelí 10 (to môžeme robiť, lebo 10 je nesúdeliteľné so všetkými netriviálnymi prvočíslami).
*Skôr našťastie. Keby sme vedeli rýchlo faktorizovať (rozkladať na prvočísla), tak by väčšina dnešného šifrovania bola nepoužiteľná. Najpoužívanejší algoritmus RSA je práve založený na rozklade na prvočísla.

2.7 Záver

Na mnoho vzorcov vie človek prísť sám. Niekoho môže táto tematika zaujať a iných nie. Napríklad ja si tak zvyknem rekreačne rátať z hlavy, keď sa naozaj nudím. Zlepšuje to matematickú predstavivosť a je možné potom ohurovať ostatných ľudí.

Uvediem ešte zopár výdod. Naučíte sa lepšie odhadovať výsledky, resp. znižujete svoju omyslnosť pri numerických výpočtoch. Zovšeobecnením: ak si v niečom neveríte alebo sa v tom často mýlite, je fajn si to vyskúšať robiť bez toho, aby ste sa na to pozerali (písanie na klávesnici, geometria alebo roznásobovanie výrazov naslepo).

Ja som si napríklad dokázal vynásobiť dve trojciferné čísla kým som sa rozprával. Mojou úchylkou je rozkladať aktuálny čas na prvočísla (a nie som jediný). Ráno je to ľahšie, ako pred spaním :)

3 Počítače

3.1 Štruktúra počítača - rýchle operácie ktoré nám poskytuje

Architekúra počítača je viacúrovňová. Ak chceme dosiahnúť čo najlepšie rýchlosti, tak sa snažíme mať našu pracovnú množinu dát a inštrukcií priamo v procesore (ešte lepšie je, ak je náš program hardwerovo implementovaný). Základná dátová jednotka počítača je register. Vieme si v ňom uložiť jedno slovo (zopár bitov, väčšinou 32).

Dobrými aplikáciami na simulovanie numerických výpočtov sú Excel (pre začiatočníkov úplne stačí) a potom nástroje ako Mathematica, Maple alebo GNU Octave.

Reprezentácia neceločíselných hodnôt v počítači Keby sme len celočíselné hodnoty prenásobili konštantou, tak by sme mali rovnomerne rozložené neceločíselné hodnoty $(0.001, 0.002, \ldots)$. Okrem toho by sme si nevedeli pamätať väčšie hodnoty. Preto reprezentujeme neceločíselné čísla ako $\pm m2^{exp}$.

- Znamienko (1 bit) - exponenciálna funkcia je kladná na celkom obore, preto si potrebujeme pamätať aj znamienko.
- Normalizovaná mantisa (23 bitov) - m. Z istých dôvodov vždy začína jednotkou.
- Exponent (8 bitov) - exp.
- Reprezentácia nuly

Vo v3eovecnosti sa používa sústava $M(\beta, t, L, U)$ kde β je základ sústavy, t je počet znakov mantisy, L je dolná hranica exponentu a U je horná hranica exporentu. Napríklad bežne sa používa v počítačoch $M(2, 23, -126, 127)$.

Základné inštrukcie pre register Sú približne zoradené podľa rýchlosti. Pričom rádové skoky (veľké rozdieli) sú medzi SHIFT a Priradiť hodnotu, XOR a Sčítanie, Sčítanie a Násobenie, Modulovanie a Umocňovanie.

- SHIFT -
- Priradiť hodnotu -
- AND -
- OR -
- XOR -
- Sčítanie -
- Násobenie -
- Delenie -
- Modulovanie -
- Umocňovanie -
- Operácie s konštantami - (rýchlosť závisí od operácie).

3.2 CORDIC - ako vyrátať sínus uhla

COordinate **R**otation **DI**gital **C**omputer (z roku 1960). Ide o algoritmus navrhnutý pre počítače na výpočet *sin* ľubovolného uhla. Samozrejme, stačia nám uhly $< 0, \pi/4 >$, z ktorých si vieme dopočítať všetky ostatné.

Myšlienka algoritmu je založená na tom, že pozície na jednotkovej kružnici vieme písať ako $(cosx, sinx)$. Navyše rotácia o uhol sa dá jednoducho zapísať maticami. Určený uhol potom binárne* vyhľadávame s maticami otáčania pre každý uhol $\pm arctan(\pm 2^{-n})$ (nie úplne binárne). Tieto matice sme si schopný čiastočne predrátať a tým zefektívniť celkový výpočet. Okrem tohto si ukážeme ešte ďalšie modifikácie ktoré nám algoritmus zrýchľujú (hráme sa aj na konštanty).
*Myšlienka binárneho vyhľadávania sa najlepšie demonštruje na nasledujúcej hre: jeden z hráčov si myslí číslo a druhý háda. Pričom ten ktorý si číslo myslel hovorí viac alebo menej. Ideálna stratégia je tipnúť polovicu z možného rozsahu. Napríklad si myslí číslo od 1 po 1000 tak tipneme 500. On povie menej, tak tipneme 125. A tak ďalej.

Algoritmus Máme matice rotácie v upravenom tvare pre uhly $\pm arctan(\pm 2^{-n})$. Upravený tvar spočíva v tom, že sme vytiahli konštanty pred maticu a vnútri sme nechali len čísla tvaru 1 a $\pm 2^k$. Tým pádom vieme násobenie matíc robiť len za pomoci binárneho shiftu a násobenia konštantou, čo sú rýchlejšie operácie ako pôvodne.

1. (Init) Začneme s vektorom (1,0) a načítame si konštanty pred maticami.
2. (Hľadanie) Začali sme s uhlom 0. Opakujeme nasledujúci krok:
   (a) Ak sme dosť blízko hľadaného uhla, skončíme.
   (b) Inak vynásobíme náš aktuálny vektor takou maticou z našej množiny, aby sa výsledný uhol čo najmenej líšil od cieľového.
3. Sínus cieľového uhla je v druhej zložke výsledného vektora. Ako vedľajší produkt sme dostali cosínus cieľového uhla.

k	$\frac{1}{\sqrt{1+2^{-k}}}$	$arctan(2^k)$
0	0,707106781	45
1	0,447213595	26,56505118
2	0,242535625	14,03624347
3	0,124034735	7,125016349
4	0,062378286	3,576334375
5	0,031234752	1,789910608
6	0,015623093	0,89517371
7	0,007812262	0,447614171
8	0,00390622	0,2238105
9	0,001953121	0,111905677
10	0,000976562	0,055952892

Keďže používame uhly v tvare $arctan(\pm 2^{-n})$ tak si treba uvedomiť, či ich jednotlivým naščítaním vieme dostať ľubovolný uhol. To nie je na prvý pohľad zrejmé. Na intervale $x \in < 0, 1 >$ ale platí, že $arctan(x) \doteq \pi/4x$ a teda zmena uhla je podobná, ako keby sme použili binárne vyhľadávanie. (Ak nedelíme intervaly presne na polovice alo trochu väčšei časti, tak aj tak dospejeme k cieľu.)

3.3 Taylorove polynómy

Taylorov rozvoj funkcie. Používa sa, keď je aproximovaná funkcia dostatočne hľadká (má napr. dosť veľa derivácií). Vieme určiť jeho chybu v závislosti od vzdialenosti bodu v ktorom rozvoj robíme. Používa sa v okoliach bodu v ktorom vieme rozvoj robiť. Napr. pre sínus je to v okolí $< -0,1; 0,1 >$.

Špeciálne pre sínus v bode 0 je Taylorov rozvoj nasledovný:

$$sin(x) = \frac{xcos(0)}{1!} - \frac{x^3cos(0)}{3!} + \frac{x^5cos(0)}{5!} - \frac{x^7cos(0)}{7!} + O(x^9)$$

4 Záver

Hlavná výhoda tohto typu matematiky je tá, že sa jej výsledky dajú rýchlo overiť. A keď nami popísané algoritmy naozaj fungujú, tak nám to prináša podobný pocit blaženosti ako keď vykonáme fyzickú prácu ktorá má viditeľné výsledky.

V prvej časti sme videli, ako dokážeme preprogramovať svoju myseľ tak, aby sme dokázali rýchlejšie násobiť. Nové metódy treba pochopiť, potom si ich párkrát vyskúšať a ak sa dá, tak ich posúvať ďalším ľuďom ďalej, lebo platí, že najlepšie sa naučíte ak to niekoho učíte.

Dúfam, že táto prednáška oslovila čitateľov a prezentovala numerickú matematiku v kladnom zmysle. Akýkoľvek feedback posielajte prosím vás emailom (adresa na začiatku), alebo do komentárov na mojej homepage (adresa na začiatku).