Flyttalsprecision

Johannes Stenbäck

March 29, 2022

Sammanfattning

I denna avhandling behandlas de grundläggande aritmetiska operationerna addition, subtraktion, multiplikation och division för flyttal av arbiträr storlek i GNU MPFR biblioteket. Inledningsvis sammanfattas IEEE-754 standarden och binära representationen av flyttal. Algoritmerna för att utföra flyttalsaritmetik förklaras stegvist och med exempel. Koncepten förlängs sedan till fllyttal av arbiträr precision. I avhandlingen analyseras hur biblioteket komponerar flyttal av arbiträr storlek med hjälp av lemmar, och hur aritmetiska operationer sedan utförs på dessa flyttal.

1 Inledning

Flyttalsaritmetik är en systematisk approximering av reell aritmetik.[1] Flyttal kan endast representera en ändlig delmängd reella tal. Standarden för flyttal i datorsystem definieras av IEEE-754. Standarden anger format för flyttal och hur räkneoperationer bör utföras på dem i datorsystem. Standarden kan tillämpas i hårdvara, mjukvara eller en kombination av både hård- och mjukvara. De flesta hårdvaruarkitekturer idag implementerar stöd för högst 64-bitars dubbelflyttal. I denna avhandling presenteras mjukvarubiblioteket GNU MPFR för flyttal av arbiträr storlek. Syftet med avhandlingen är att analysera hur flyttal av arbiträr kan byggas upp med hjälp av lemmar, och hur aritmetiska operationer utförs på dessa flyttal.

2 IEEE-754

IEEE-754 standarden grundar sig på att uttrycka reella tal i form av

$$(-1)^s * b^e * m$$

Där

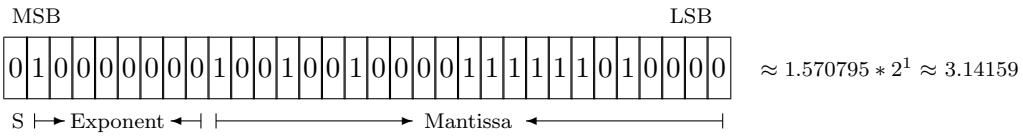
- s = tecken. 0 är positivt, 1 är negativt
- b = talbas. Baserna 2 och 10 stöds.
- e = exponent för talbasen.
- m = mantissa.

I enlighet med GNU biblioteket MPFR (del 3) behandlas enbart flyttal med basen två i denna avhandling. Flyttal representeras binärt genom att inkoda tecken, exponent och mantissa. Storleken på exponenten och mantissan varierar beroende på flyttalets storlek. Fördelningen mellan exponentens och mantissans storlek bestämmer vilka reella tal kan uttryckas som flyttal. Format för de grundläggande flyttalen i talbas två hittas i Tabell 1.

Bitar totalt	Tecken	Exponent	Mantissa
32	1	8	24
64	1	11	53
128	1	15	113

Table 1: *[1] Format för de grundläggande flyttalen i talbas två enligt IEEE-754 specifikationen.*

Flyttal vars mest signifikanta bit är en etta kallas normaliserade flyttal. Denna etta är implicit och behöver ej inkodas binärt. Såvida kan 24 bitars precision sparas med 23 bitars mantissa i minnet. I flyttal med arbiträr precision är det inte lika viktigt att spara en bit i minnet och då kan ettan inkodas binärt.[4] För att möjliggöra negativa exponenter i flyttal, subtraheras 2^{n-1} från exponenten där n är exponentens storlek i bitar. Hälften av exponentens värden är då negativa och hälften positiva. En godtagbar siffras binära uppställning visas i Figur 1.

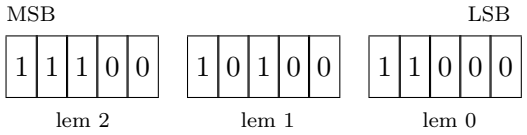


Figur 1: *[7] Approximering av π till femte decimalen sparat som ett 32 bitars flyttal. Mantissans mest signifikanta bit är implicit ett och visas inte i figuren.*

Tal som är för stora för att representeras med exponentens maximum värde överflöder till $+\infty$ eller $-\infty$. Tal som är för små för att representeras med exponentens minimum värde behandlas som subnormala flyttal. Subnormala flyttal innebär att den mest signifikanta biten är noll. Detta möjliggör gradvist underflöde. Tal som är mindre än de subnormala talen avrundas till +0 eller -0. Subnormala tal är inte nödvändiga i arbiträr precision, eftersom exponenten kan praktiskt taget vara obegränsat stor.[4]
Ett annat specialfall är NaN (eng. Not A Number) som används för att representera oinitierade flyttal eller resultatet från ogiltiga operationer.[4]

3 MPFR

De flesta hårdvaruarkitekturer idag har inbyggt stöd för högst 64-bitars flyttal. För att använda 128-bitars flyttal eller större, behövs ett mjukvarugränssnitt. GNU MPFR är ett mjukvarubibliotek och en generisk tillämpning av IEEE-754 standarden. Biblioteket möjliggör flyttalsaritmetik med arbiträr precision och korrekt avrundning.[2] Precisionen begränsas i praktiken enbart av tillgänligt minne på datorsytemet. Precisionen är konfigurerbar och tillåter även storlekar som avviker från 2^n mönstret. MPFR använder talbasen två. Biblioteket avviker en del från IEEE-754 standarden. Subnormala flyttal implementeras inte och mantissans mest signifikanta bit är explicit, det vill säga den inkodas binärt.
Som underlag för MPFR ligger lågnivå kategorin MPN i biblioteket GNU MP. MPN bildar stortal av arbiträr storlek genom att dela upp dem i en array av lemmar. Lemmar är teckenlösa heltalsrepresentationer. Storleken på lemmar definieras av GMP_NUMB_BITS som vanligtvis är 32 eller 64-bitar.[2] Eftersom flyttalens precision p kan avvika från att vara en multipel av lemmarnas storlek, så sparas precisionen skilt. De lägsta bitarna utanför p är icke-signifikanta och sätts till noll.[2] Figur 2 illustrerar hur en mantissa kunde delas upp i lemmar och hur icke-signifikanta bitar nollas.



Figur 2: *Exempel på hur MPFR skulle dela upp 12-bitars mantissan [111001010011] i lemmar på en fem bitars dator. De tre lägsta bitarna i "lem 0" används inte och har nollats.*

I MPFR är exponenten i flyttal 32 eller 64-bitars stor. Eftersom exponenten kan bli mycket stor behövs inte subnormala tal implementeras. MPFR använder dock inte exponentens alla bitar, utan exponentens minimum värde MPFR_EMIN_MIN $= 1 - 2^{n-1}$ och maximum värde MPFR_EMAX_MAX $= 2^{n-1} - 1$ där n anger bitstorleken. Detta gör det möjligt att representera summan av två andra exponenter utan att summan överflöder. Således förenklas vissa räkneoperationer som multiplikation.[6]

3.1 Aritmetiska operationer

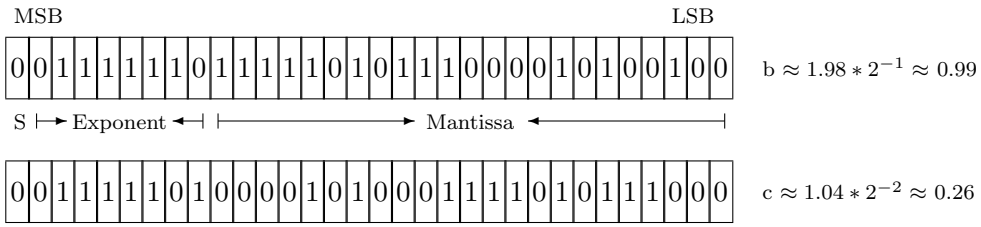
I denna del av avhandlingen kommer de grundläggande aritmetiska operationerna addition, subtraktion, multiplikation och division att behandlas. Innehållet grundar sig dels på artiklarna [5] och [6] av Vincent Lefèvre, dels på [3] MPFR version 4.2.0 källkod. Som notation för aritmetiska operationer används a, b och c där a avser resultat medan b och c avser operander. MPFR tillåter operanderna b och c ha olika precision. Resultatet a kan även efterfrågas i en precision som avviker från b och/eller c precision.[5] För att möjliggöra detta krävs strikta regler för exakt avrundning som leder till ökad komplexitet. MPFR biblioteket utför dessutom en del avancerade optimeringar vid avrudning av a som är utanför denna avhandlingens ramar. Syntaxen för aritmetiska funktioner i MPFR är

```
int mpfr_func(a, b, c, rnd_mode)
```

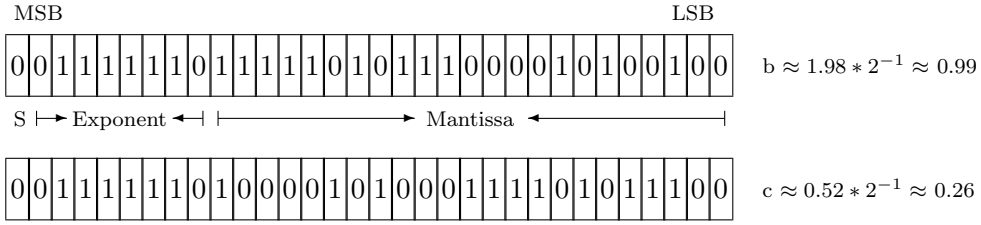
där a är en ut-parameter för funktionens resultat, b och c är in-parametrar för operanderna och rnd_mode är in-parameter för avrundningsläge. Funktionens returvärde är ett ternärt heltal som anger åt vilket håll a har avrundats från det exakta värdet. Ett positivt returvärde betyder att a är större än det exakta värdet. Returvärdet noll betyder att a är exakt och ett negativt returvärde betyder att a är mindre än det exakta värdet.

3.2 Algoritm

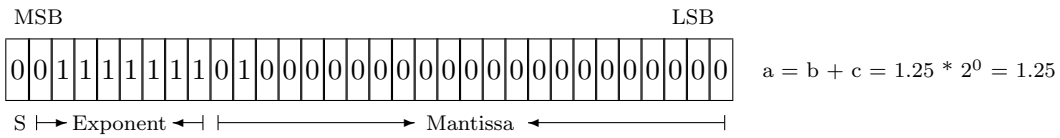
Addition och subtraktion med flyttal grundar sig på att ändra b och c binära uppställning så att de kan uttryckas med samma exponent. Detta görs genom att först ordna om b och c så att $e_b \geq e_c$ där e anger exponenten för flyttalet. Sedan beräknas exponentdifferansen $d = e_b - e_c$.[5] Mantissan m_c antar nya värdet $\frac{m_c}{2^d}$, eftersom att c då kan uttryckas med samma exponent som b. Denna divison kan utföras genom att bitförskjuta m_c d steg lägre. För att beräkna resultatet a utförs aritmetiska operationen på mantissan m_b och den förskjutna mantissan m_c. Exponenten e_a antar värdet e_b. Om additionsoperation leder till en överbliven minnessiffra bör summan normaliseras genom att m_a bitförskjuts ett steg lägre och e_a inkrementeras med ett.[5] Om a är denormaliserat efter subtraktion, förskjuts m_a mot högsta biten och exponenten sänks. Efter räknaoperationen avrundas m_a till precision p och de lägsta bitarna i m_a utanför p sätts till noll. I Figur 3, 4 och 5 visas ett exempel för algoritmen med addition.



Figur 3: *Normaliserad representation av flyttalen* $b \approx 0.99$ *och* $c \approx 0.26$*. Mantissornas högsta bit är implicit ett och visas inte i figuren.*



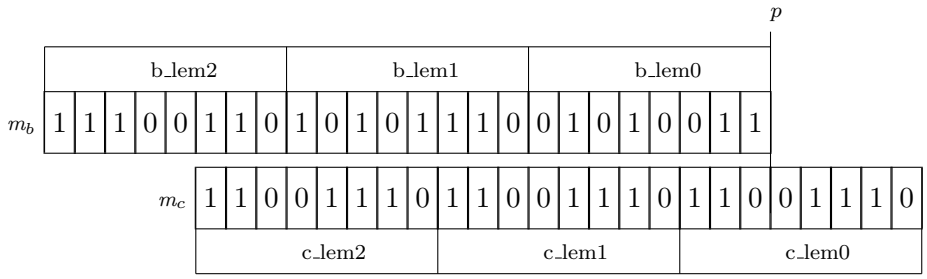
Figur 4: *Flyttalet c representerat med samma exponent som flyttalet b.* m_c *implicita etta är nu inkodad binärt eftersom bitarna har förskjutits ett steg lägre. Flyttalet c har fortfarande samma värde som i Figur 3.*



Figur 5: *Summan av additionen b + c som ger det exakta värdet 1.25.*

3.3 Implementering

MPFR innehåller en generisk implementering för varje aritmetisk funktion samt flera optimerade implementeringar. Optimeringarna baserar sig på att olika precisioner hanteras skilt. Här diskuteras huvudsakligen koncept som går att tillämpa generellt. För att tillämpa flyttalsaritmetik med lemmar krävs flera delsteg. I implementeringarna förskjuts inte mantissan m_c d steg lägre, utan istället utförs bitvisa och aritmetiska operationer på lemmarna skilt. Antalet lemmar i slutliga mantissan m_a bestäms av $l = \frac{p}{GMP\_NUMB\_BITS}$ avrundat uppåt till närmaste heltal. Från l och d härleds även antalet m_c högsta lemmar som behövs för räkneoperationen. Dessa lemmar kopieras till m_a med offset d. Sedan kopieras högst l stycken m_b lemmar till m_a utan offset. Figur 6 föreställer hur m_b och m_c arrangeras före addition eller subtraktion.



Figur 6: *Föreställning hur m_b och m_c arrangeras vid kopiering till m_a. I detta exempel är GMP_NUMB_BITS = 8, d = 5 och p = 24. Notera att i denna illustration har ingen aritmetisk operation ännu utförts på lemmarna.*

När lemmarna i m_b och m_c kopieras till m_a bör räkneoperationen utföras på alla bitar som överlappar. De generiska funktionerna mpfr_add1 och mpfr_sub1 delegerar kopieringen och räkneoperationerna till MPN funktionen mpn_add_n respektive mpn_sub_n. De optimerade funktionerna komponerar istället resultatet a själva genom att utföra bitförskjutningar, bitmaskeringar och räkneoperationer på lemmarna skilt. Optimeringarna grundar sig på att skilt beakta specifika satser av precision p och exponentdifferansen d.

3.4 Multiplikation och division

I detta kapitel kommer multiplikation och division att behandlas.

References

[1] *IEEE Standard for Floating-Point Arithmetic, Standard 754-2019 (Revision of IEEE 754-2008)*

[2] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélissier, P. Zimmerman (2007) *MPFR: A Multiple-Precision Binary Floating-Point Library With Correct Rounding*

[3] G. Hanrot et al. *The MPFR library* http://www.mpfr.org/. Version 4.2.0.

[4] R. P. Brent, P. Zimmerman (2010) *Modern Computer Arithmetic*

[5] V. Lefèvre (2007) *The Generic Multiple-Precision Floating-Point Addition With Exact Rounding (as in the MPFR Library)*

[6] V. Lefèvre (2007) *Correctly rounded arbitrary-precision floating-point summation*

[7] user4686 Stackexchange https://tex.stackexchange.com/a/163339 *The binary representation of the value 3.14159 stored as float.*