計算機演習３（講義）　補足　－　２の補数

19970613–14 服部哲弥

質問票３に書いていただいた質問のうち，講義の説明がいちばん不十分だったと思われる２の補数について補足説明します．

２の補数

定義． 講義の通りだが，記号の設定を兼ねて繰り返す．

自然数 M を固定する． $0 < n < 2^M$ を満たす自然数 n の 2 の補数とは自然数 $2^M - n$ のことをいう．

つまり，定義としての 2 の補数は， $0 < n < 2^M$ を満たす自然数から自分自身への写像 $n \mapsto 2^M - n$ のことに過ぎない．

簡単な性質．

- 2 の補数の 2 の補数は元の数に戻る．特に，$0 < n < 2^M$ を満たす自然数とその 2 の補数は一対一に対応する．
- n を M bit の二進数で表示したとき，n の 2 の補数は全ての bit を反転（各 bit の 0 と 1 を入れ替えること）して1 を足し算したものに等しい．
- n を M bit の二進数で表示したとき，n の 2 の補数は右から順に見ていって初めて出た 1 より左にある bit を全て反転したものに等しい．

問． 以上の性質を確かめよ．

意義．定義には大した内容はない． 2 の補数の意義は，負の整数を計算機のデータとして表す有力な方法だという点である．

引き算は符号を変えて足し算をすることだから，負の整数を表す方法によって引き算は足し算に帰着できる．この意味で計算機による引き算の計算にも用いる．

論点 − 符号の表現． 絶対値が $M-1$ bit の（符号付きの）整数 n $(0 \leq |n| < 2^{M-1})$ を表示することを考える．

計算機は 0 と 1 だけでデータを表すから符号も 0 と 1 で表さないといけない．符号の bit を含めて M bit 必要になる．符号 bit は習慣に従って先頭（一番左の桁）に置く．正の数は符号 bit を 0 にとるのが自然である．

例えば，$M = 3$ ならば正の数は $2^{M-1} - 1 = 3$ 以下の自然数である．これらを 3 bit で表すと次表のようになる．

2 bit 自然数	1	2	3
ビットパターン	001	010	011

符号 bit （先頭の 0）がついているので，自然数としては 2 bit だが 3 bit で表示している．

負の数はどうするか？先頭 bit を 1 にすれば正の数でないことは分かるから，残りの $M-1(=2)$ 桁は $|n|$ と一対一対応に対応すれば原理的には何でもよい．あとは単純さ（回路の単純さ）や自然さ（理解しやすさ）に基づいて選ぶ．次の二つの方法を比べよう．

２の補数を使わない方法．負の数 n を表す一つの単純な方法は先頭bit を 1 にして，残り $M-1$ bit は $|n|$ を表すようにすることである．しかし，見かけ上の単純さとは裏腹に，これは 2 の補数を使う方法より複雑で分かりにくい面がある（ということを以下で主張したい）．

２の補数を使う方法．$n<0$ のとき， n を M に対応する $|n|$ の 2 の補数 $2^M-|n|$ で表す．$|n|\leq 2^{M-1}$ ならば$2^M-|n|\geq 2^{M-1}$ となって，先頭 bit （M 桁目，つまり，2^{M-1} の位）は 1 になる．これが計算機における負の整数の標準的な表現である．

二つの方法を $M=3$ の場合に比較すると次表のようになる．

整数	-3	-2	-1	0	1	2	3
２の補数による表示	101	110	111	000	001	010	011
２の補数によらない表示	111	110	101	000	001	010	011

正の数は前の表と同じ．負の数は，２の補数による表示では，例えば -1 は $2^3-1=7=111\ (2)$ となるのに対し，もう一方の表示では先頭 bit を 1 とし，残り 2 bit は $|-1|=1=01\ (2)$ となる．

表現の自由[1]**．**符号をビットパターンでどう表現するかは約束事だから，一対一であれば，原理的にはどんなやり方でもよい．文字のビットパターンによる表現が約束事なのと同じことである．しかし，どういう計算を主に行うかによって，使いやすさや自然さに差がある場合もある．

現実の計算機では，日本語文字（2Byte 文字，全角文字）のビットパターン表現に種々の約束が並立したので，よそのホームページを見たり，他人からメールをもらったりするときに文字化けすることがある．(めったにないのは，ソフトウェアが表現を推測して自動的に変換しているから．１０年前は文字化けするのが普通だった.)

しかし，符号（負の整数）の表現は２の補数を使う方法に統一されている．それほど自然で単純であることが以下のように分かる．

例 − 加法．負の数（符号）の表示法が違えば，加法（足し算）をビットパターンに対する論理演算として記述する仕方が変わる． $M=3$ のときに具体的に加法の表 $(m,n)\mapsto m+n$ を書いてみる．

$m \mid n$	-2	-1	1	2	3
-3		(-4)	-2	-1	0
-2	(-4)	-3	-1	0	1
-1		-2	0	1	2
1			2	3	

表を見やすくするために，以下の理由で本質的でない部分を省略した．

- $m>n$ の場合は m と n の交換で $m<n$ の場合に帰着できる．

[1] この題は単なるだじゃれ.

- $|m+n| \geq 2^M$ だと，結果を符号付きで M bit で表せない（これを overflow error 桁あふれの間違い，と呼ぶ ）．但し， 2 の補数を用いる方法では,「おまけ」で $-2^{M-1} \mapsto 2^M - 2^{M-1} = 2^{M-1} = 100\cdots00\ (2)$ が表示できるので， 本当は $m+n=-2^M$ は overflow にならない.「2 の補数を使わない方法」は -4 は overflow になる．表では括弧で表している．
- $m=0$ または $n=0$ のときは $m+n$ は自明.

表の中の空欄や，表に現れていない行や列は以上のどれかの理由である．

加法の表を二つの方法それぞれを用いて，ビットパターンで書き直す．整数とビットパターンの対応（符号付き二進数表示として）は先ほどの表を参照しながら置き換えていけばよい.

2 の補数による表示

$m \mid n$	110	111	001	010	011
101		100	110	111	000
110	100	101	111	000	001
111		110	000	001	010
001			010	011	

2 の補数によらない表示

$m \mid n$	110	101	001	010	011
111		OV	110	101	000
110	OV	111	101	000	001
101		110	000	001	010
001			010	011	

表の中の OV の欄は 2 の補数による表示では 3 bit で表現できるが 2 の補数によらない表示では 4 bit 必要なので 3 bit では overflow する．

2 の補数表示の利点． 加法の表を翻訳して作ったことを忘れて，ビットパターンによる加法の表をそのまま眺めると， 2 の補数による表示で書いた加法の表は,

> 3 bit の自然数（正の数）どうしの加法の表であると思ったのと一致する．

実際，2 の補数による表示の表を（符号付き整数ではなく）3 bit 自然数の表だと思って，逆翻訳（解読，この場合は意図的な誤読）すると，例えば $110\ (2) = 6\ (10)$, $101\ (2) = 5\ (10)$ などとなるので,

$m \mid n$	6	7	1	2	3
5		4	6	7	0
6	4	5	7	0	1
7		6	0	1	2
1			2	3	

となる．一見無意味に見えるかもしれないが，よく考えると，例えば，$7+5=12=8+4$ なので，**普通に足し算を行った後，答が 8 を越えたら 8 を引く**とこの表を再現できることが分かる（各自確かめよ）．8 (10) = 1000 (2) なので「8 を越えたら 8 を引く」とは，二進法の「下から 4 bit 目を無視する」ことと同値である．だから，数字を 3 bit で表すことにしておいて**4 bit 目以上のくり上がりを無視すればよい**（論理演算や論理回路としては無視するのは自明な操作）．

先頭 bit が符号を表すことやそのビットパターンが表す数の正負による場合分けは必要なく，講義で既にやった自然数どうしの加法の論理回路（全加算器 full-adder）を $M=3$ bit として**そのまま**使えば，符号付きの整数の加法の回路になっていることを意味する．

> 加法の表を 2 の補数による表示で表すと，符号ビットであるかどうかやその数の正負を気にすることなく，あたかも符号ビットまで含めて自然数の M 桁の二進数表示であるかのように扱って足し算を行えばよい． 桁あふれ overflow が起きても，高位の bit を単に無視して（あふれるにまかせて）答を求めれば，答が M bit で表せる符号付き整数 ($-2^{M-1} \le m+n < 2^{M-1}$) に収まるならばこの操作で正しい答を得る．

このようなすっきりした性質は 2 の補数を使わない方法の表からは得られない．

種明かし．最後に種明かしをしておく．2 の補数を用いて負の整数を表すと加法がビットパターンに対する演算として負の数まで含めて統一的に表せる．その理由は数学的には単純である．2 の補数で負の数を表す方法は

> $n<0$ のとき， n を M に対応する $|n|$ の 2 の補数 $2^M-|n|$ で表す

ということであった．言い換えると $n<0$ のとき，n を 2^M+n で表すということである．他方，$n \ge 0$ のときは n のままであった．これらは

> 整数 n を 2^M を法として M bit 自然数 $0, 1, \cdots, 2^M-1$ で表す

ことに他ならない．

加法が自然に表せるのは自然な写像 $\phi:\ \mathbf{Z} \to \mathbf{Z}/\mathbf{2^M Z}$ が準同型写像であることの当然の結果だったのだ！