EXTRACTION OF ROOTS BY REPEATED SUBTRACTIONS FOR DIGITAL COMPUTERS*

IWAO SUGAI, I.B.M. Research Center*, Yorktown Heights, N. Y.

This note extends to n-th roots the common desk calculator method of obtaining square roots by subtraction of successive odd integers. The method employs no division and is thus useful for digital computers which either do not have built-in division or for which the divide time is disproportionately great compared to subtract and multiply times. The basic idea is an extension? of the well-known fact that the sum of the first n odd integers is n^2.

If N, a p-decimal place integer, is the square root of X, $(X = N^2)$, then :

$$N = \sum_{i=1}^{i=p} b_i \cdot 10^{p-i}, \qquad \text{where } 0 \leq b_i \leq 9$$

If $a_i = b_i, 10^{p-i}$, one form of grouping X is:

$$1)\quad X = (a_1+a_2+a_3+\ldots\ldots.)^2 = [a_1^2] + [2a_1a_2+a_2^2] + [2(a_1+a_2)a_3+a_3^2] + [2(a_1+a_2+a_3)a_4+a_4^2]$$

If N is the cube root of X, X may be grouped as:

$$2)\quad X = [a_1^3] + [3a_1^2a_2+3a_1a_2^2+a_2^3] + [3(a_1+a_2)^2a_3 + 3(a_1+a_2)a_3^2 + a_3^3] + [3(a_1+a_2+a_3)^2a_4 + 3(a_1+a_2+a_3)a_4^2 + a_4^3] + \ldots\ldots..$$

And in general, if N is the n-th root of X:

$$3)\quad X = (a_1+a_2+a_3+\ldots\ldots..+a_p)^n = \text{“}\sum I : \sum_{i=1}^{i=n} \left[{}_nC_i \left(\sum_{m=1}^{m=k-1} a_m \right)^{n-i} a_k^{\,i} \right]$$

This way of grouping numbers is seen in combinatorial analysis[2, 3, 5]. Equations (1) and (2) show that when the numbers in square brackets are subtracted from X, the following statements can be made:

* Taken from the author's paper of the same name, Report R-640-57, PIB-568, Microwave Research Institute, Polytechnic Institute of Brooklyn, 10 Feb 1958.

I. If there is no remainder after the k-th square-bracketed number is subtracted, the exact square (or cube) root is $\sum_{i=1}^{k} a_i$

II. If there is a positive remainder after the k-th square-bracketed number is subtracted, the approximate square (or cube) root to k decimal places is $\sum_{i=1}^{k} a_i$, with a remainder.

By way of an illustrative example, suppose that the cube root of 9,999,999,999 is required. The actual rocedure is as follows:

Group in units of 3 decimal digits starting from the units position.		$9\ \boxed{999\ 999\ 999}$
Take the first square-bracketed term from Eq. 2 and subtract,		
from the leftmost group, $\Delta[a_1^3]=a_1^3-(a_1-1)^3$, for $a_1=1, 2, 3\ldots$		
This series is 1, 7, 19, 37, 61, 91, 127, 169, 217.		-1
Subtract as many as possible without going negative.		-7
This number becomes b_1 (=2 in this case).		
Add the next group of three digits, shifting the decimal place so that $a_1=20$.		$1\ 999\ \boxed{999\ 999}$
Subtract as many $\Delta[3a_1^2a_2+3a_1a_2^2+a_2^3]$		
as possible for $a_2=1, 2, 3\ldots$ Here	$3\,(20)^2 \longrightarrow 1200+60\cdot 1+1$	$-1\ 261$
$b_2=1$.	$3\,(20)\quad 1200+60\cdot 3+7$	(won't go)
Add the next group of three digits, shifting the decimal place so that		$738\ 999\ \boxed{999}$
$a_1+a_2=210$. Subtract as many		
$\Delta[3(a_1+a_2)^2a_3+3(a_1+a_2)a_3^2+a_3^3]$		
as possible for $a_3=1, 2, 3\ldots$	$132{,}000+630\cdot 1+1$	$-132\ 931$
Here $b_3=5$.	$132{,}000+630\cdot 3+7$	$-134\ 197$
	$132{,}000+630\cdot 5+19$	$-135\ 469$
	$3\,(210)^2 \longrightarrow 132{,}000+630\cdot 7+37$	$-136\ 747$
	$3\,(210)\quad 132{,}000+630\cdot 9+61$	$-138\ 031$
Add the next group of three digits, shifting the decimal		$61\ 624\ 999$
place so that $a_1+a_2+a_3=2150$. Subtract as many		
$\Delta[3(a_1+a_2+a_3)^2a_4+3(a_1+a_2+a_3)a_4^2+a_4^3]$		
as possible for $a_4=1, 2, 3\ldots$	$13{,}867{,}500+6450\cdot 1+1$	$-13\ 873\ 951$
Here $b_4=4$.	$13{,}867{,}500+6450\cdot 3+7$	$-13\ 886\ 857$
	$3\,(2150)^2 \longleftarrow 13{,}867{,}500+6450\cdot 5+19$	$-13\ 899\ 769$
	$3\,(2150)\quad 13{,}867{,}500+6450\cdot 7+37$	$-13\ 912\ 687$
us the approximate 4-digit cube root of 9,999,999,999 is 2154., to an		$6\ 051\ 735$
or of 0.02 per cent.		

\s many digits of the extracted root may be obtained as one desires, by adding three 0's to the right each additional digit of the cube root, continuing this process. This method has been programmed a drum memory computer (IBM 650) and compared with conventional methods using Newton's mula. It required more memory space, but the execution time was cut on the average by a factor 2 for fixed-point arithmetic.

'erences

Sugai, "On Unequal Partitions of Integers under Certain Imposed Conditions", Report R-566-57, PIB-494, 21 Jun 1957, Microwave Research Institute, Polytechnic Institute of Brooklyn, N. Y.

(2) W. Keister, A. E. Ritchie, S. H. Washburn; "The Design of Switching Circuits", D. Van Nostrand, N. Y. 1950, page 485.

(3) A. D. Booth, "Automatic Digital Calculators," Butterworth's Scientific Pub., London, 1953, page 173.

(4) P. K. Richards, "Arithmetic Operations in Digital Computers," D. Van Nostrand, N. Y. 1955, page 293.

(5) P. A. MacMahon, "Combinatory Analysis," Cambridge University, 1915, page 3.

Editor's Notes:

Although this method is not novel, it has been printed here to summarize for the benefit of a new generation of computer personnel. It should be noted that:

1) This method seems advantageous if only a few significant figures are required. Otherwise the normal method, Log-Multiply-Antilog, is more desirable and faster in particular for higher order roots. These subroutines are normally required for other purposes anyway and space is not lost.
2) One immediately notices many tricky ways of coding this method for a computer, via looping and the use of tables or converting instructions. Note that, as one proceeds, the contribution of the left-hand term becomes proportionately large enough such that it alone might be used within accuracy limits after a certain number of digits are developed.
3) Although the author states that this method used more memory space than other routines, it seems that the converse could well be true if advantage were taken of higher order differences in building up the subtrahend. This appears to be a natural method for a 256 memory machine, if it had good indexing and looping features. Remember that $\Delta^n(X^n) =$ a constant n!

PRELIMINARY REPORT—INTERNATIONAL ALGEBRAIC LANGUAGE

Note:

In the interest of immediate circulation of the results of the ACM-GAMM committee work on an algebraic programming language, this preliminary report is presented. The language described naturally enough represents a compromise, but one based more upon differences of taste than on content or fundamental ideas. Even so, it provides a natural and simple medium for the expression of a large class of algorithms. This report has not been thoroughly examined for errors and inconsistencies. It is anticipated that the committee will prepare a more complete description of the language for publication.

For all scientific purposes, reproduction of this report is explicitly permitted without any charge.

Acknowledgements:

The members of the conference wish to express their appreciation to the Association for Computing Machinery, the "Deutsche Forschungsgemeinshaft" and the Swiss Federal Institute of Technology for substantial help in making this conference and resultant report possible.

A. J. PERLIS
K. SAMELSON
for the ACM-GAMM Committee

PART I—INTRODUCTION

In 1955, as a result of the Darmstadt meeting on electronic computers, the GAMM (association for applied mathematics and mechanics), Germany, set up a committee on programming (Programmierungsausschus). Later a subcommittee began to work on formula translation and on the construction of a translator, and a considerable amount of work was done in this direction.

A conference attended by representatives of the USE, SHARE, and DUO organizations and the Association for Computing Machinery (ACM) was held in Los Angeles on 9 and 10 May 1957 for the purpose