1970

Efficient tree-listing algorithm

Chan, Shu-Gar; Chang, Wen-Tao

masking costs and adding to the flexibility of the silicon slices. These 2-layer techniques are already in use for other custom-designed integrated circuits. Alternatively, single generalised cells could be interconnected by beam leads on a silica substrate.

Multipliers have been proposed[3-5] which use the cell shown in Fig. 1 in which the sum output S and carry P are obtained from the binary sum of the inputs B, AD and C.

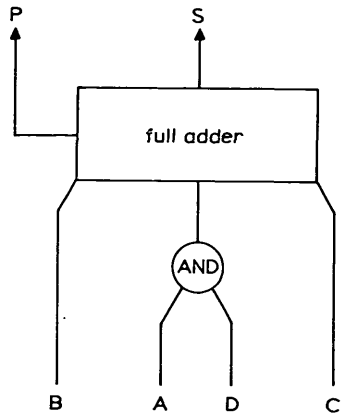

Fig. 1

Guild[6] has proposed a square-rooter, the unit cell of which is shown in Fig. 2. This cell is, effectively, a subtractor, but operates by complementing. Although it is only suggested that this can be used for producing square roots, it is clearly possible to use such a cell as the basis of a general dividing array. In Fig. 2, S is the difference output and P is the borrow out from the summed inputs B, $\bar{A}E+A\bar{E}$ and C.

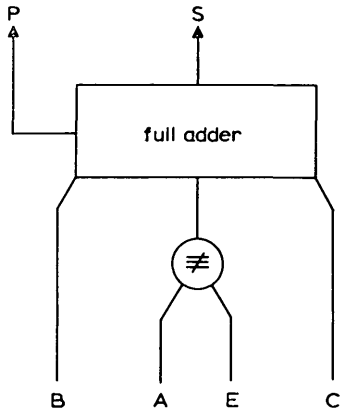

Fig. 2

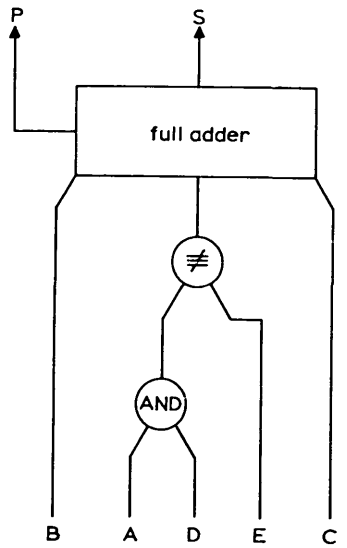

Fig. 3

The proposed generalised cell is shown in Fig. 3. This cell can be used as a parallel adder/subtractor using complementing for subtraction, and as a cellular full multiplier, divider and square-rooter. By putting $E = 0$ the cell acts as an adder/multiplier and by putting $E = 1$ the cell can be used as a subtractor/divider.

With a typical gated full adder* these generalised cells of Fig. 3 can be implemented using two invertors and two 2-input NAND gates. The complete cell requires seven connections (plus two power-supply leads), so that a dual cell could be produced in a standard 16-lead dual-inline package. The cell represents about 16 NAND gates, so that an array of 100 cells would seem possible as a target for large-scale integration.

G. WHITE

Muirhead Ltd.
Beckenham, Kent, England

1st April 1970

References

1 DEAN, K. J.: 'Logical circuits for use in iterative arrays', *Electron. Lett.*, 1968, **4**, pp. 81–82
2 AVIZIENIS, A., and TUNG, C.: 'Design of combinational arithmetic nets', 1st annual IEEE computer conference, 1967, pp. 25–28
3 BURTON, D. P., and NOAKS, D. R.: 'High-speed iterative multiplier', *Electron. Lett.*, 1968, **4**, p. 262
4 DE MORI, R.: 'Suggestion for an i.c. fast parallel multiplier', *ibid.*, 1969, **5**, pp. 50–51
5 DEAN, K. J.: 'Versatile multiplier arrays', *ibid.*, 1968, **4**, pp. 333–334
6 GUILD, H. H.: 'Cellular logical array for nonrestoring square-root extraction', *ibid.*, 1970, **6**, pp. 66–67

* Mullard FJH191

COMPUTER-PROGRAM DESCRIPTION

EFFICIENT TREE-LISTING ALGORITHM†

Indexing terms: Network topology, Trees

An algorithm, based on the T-triangle method, is given for the generation of all the trees in a nonoriented connected graph. The efficiency of the algorithm is verified by computer results.

Since topological methods in network analysis require the determination of all the trees of the network graph,[1] the computation time needed by a topologically based computer program depends heavily on the efficiency of the tree-finding subroutine. Many tree-finding methods have been proposed, as mentioned in References 2 and 3. However, programs available to the authors indicate that an efficient algorithm is still to be desired.

Definition: Let the vertices of a graph G_0 be numbered from 1 to v. We can define that a T-triangle T_0 of G_0 is a triangular array, of which the entry $t_{i,j}$ in row i and column j represents the set of edges incident to both vertex $i+1$ and vertex j. If we let S_k be the set of edges incident to vertex k, then $t_{i,j}$ is given by

$$t_{i,j} = \{S_{i+1} \cap S_j\} \qquad \begin{matrix} i = 1, 2, \ldots, v-1 \\ j \leqslant i \end{matrix}$$

where the symbol $\cap$ denotes the intersection of the sets S_{i+1} and S_j.

T-transformation: The T-transformation is a process for transforming a T-triangle T_0 into a new T-triangle $T_{0,r}$ with respect to column r of T_0 according to the following steps:

(*a*) Replace each entry $t_{i,j}$ by $t_{i+1,j}$, where $v-1 > i \geqslant r-1$ and $r > j \geqslant 1$.

(*b*) Replace each entry $t_{i,j}$ by $t_{i+1,j+1}$, where $v-1 > i \geqslant r$ and $v-1 > j \geqslant r$, and simultaneously replace each entry $t_{v-1,j}$ by $t_{j,r}$ for $j \geqslant r$.

The following properties of the T-triangle can be derived:

Property 1: The collection of the edges, taking one from each

† The program listing and accompanying documentation are held in the IEE Program Library, IEE, Savoy Place, London WC2R 0BL, England. Copies are available on application and on payment of a charge. Please quote Program CP 53

row of a T-triangle at a time, is a complete set of branches of a tree.

Property 2: Let T_1 be a T-triangle obtained from T_0 by applying the T-transformation and T_1' be the T-triangle obtained by replacing the entry $t_{v-1,1}$ of T_1 with the entry $t_{1,1}$ of T_0. Then the graphs G_0 and G_1' corresponding to T_0 and T_1', respectively, are within a 2-isomorphism.

Algorithm:

(i) From the original T-triangle T_0, apply the T-transformation to obtain a new T-triangle $T_{0,r}$ with respect to each column r for $2 \leqslant r \leqslant v-1$, provided that

(*a*) $\sum_i t_{i,r} \neq 0$

(*b*) $\sum_{i=r}^{v-1} \sum_{j=1}^{r-1} t_{i,j} \neq 0$

(ii) Save all the newly generated T-triangles in step 1 except those in which $\sum_j t_{k,j} = 0$ for any row k.

(iii) List all the trees from each T-triangle according to property 1.

(iv) Repeat steps 1–3 successively on each T-triangle in (ii) provided that the number of T-transformations performed on any T-triangle with respect to any column r is not greater than $(v-r)$.

This algorithm has been tested on the IBM 360 computer and found to be very efficient. The temporary computer storage is kept at a minimum since the maximum number of T-triangles in storage at any time is $v-1$, where v is the number of vertices of the graph.

SHU-GAR CHAN
WEN-TAO CHANG

Naval Postgraduate School
Monterey, Calif. 93940, USA

23rd February 1970

References

1 CHAN, S. P., and CHAN, S. G.: 'Modification of topological formulas', *IEEE Trans.*, 1968, **CT-15**, pp. 84–86
2 CHEN, W. K.: 'Generation of trees by algebraic methods', *Electron. Lett.*, 1968, **4**, pp. 456–457
3 CHEN, W. K.: 'Computer generation of trees and co-trees in a cascade of multiterminal methods', *IEEE Trans.*, 1969, **CT-16**, pp. 518–526

Program details

(*a*) Language: FORTRAN IV

(*b*) Number of variables: 16

(*c*) Special word-length requirements: none

(*d*) Number of statements: 387

Performance guide

(*a*) Computer used: IBM 360/67—OS system

(*b*) Core size required: 100 000 bytes

(*c*) Input medium: punched cards

(*d*) Output medium: line printer

(*e*) Work or data files needed: none

(*f*) Time taken to run submitted problem: 3 s for a problem with 16 807 trees

(*g*) Limitations: nonoriented graph with maximum 15 vertices and 300 edges, containing no self-loops

(*h*) Additional relevant information: user may increase or decrease the dimensions of variables according to the size of his computer and the size of the network graph

DELTA MODULATION WITH SLOPE-OVERLOAD PREDICTION

Indexing terms: Delta modulation, Filtering and prediction theory

An improved delta-modulation system is described which uses a delay line to obtain information about the 'future' behaviour of the input signal. Conditions of potential slope overload are detected before the signal is encoded and the encoding pattern altered accordingly. Computer simulation results show that signal/noise ratios can be 3 dB better than for simple delta modulation.

Delta modulation[1,2] is one way in which analogue signals may be converted to binary pulses. The pulses carry the message information corresponding to the slope of the analogue signal, and at the receiver these pulses are integrated and filtered to recover the original waveform.

The part of the system of Fig. 1 inside the broken line is a simple delta modulator. When the input signal to the subtractor is greater than the voltage on capacitor C the quantiser gives a positive output. This is clocked to the line by the bistable and charges C through R. The charging of C will continue until the voltage across C exceeds the voltage of the input signal, when the quantiser output becomes zero. Provided that the clock frequency is sufficiently high, the voltage across C closely follows the input signal.

The noise associated with delta modulation falls into two categories: granular noise and slope-overload noise. Granular noise exists because, at clock instants, the decoded signal can only exist as multiples of the delta step d volts assuming that the RC time constant is such that the voltage across C changes linearly. Slope-overload noise occurs when the slope of the

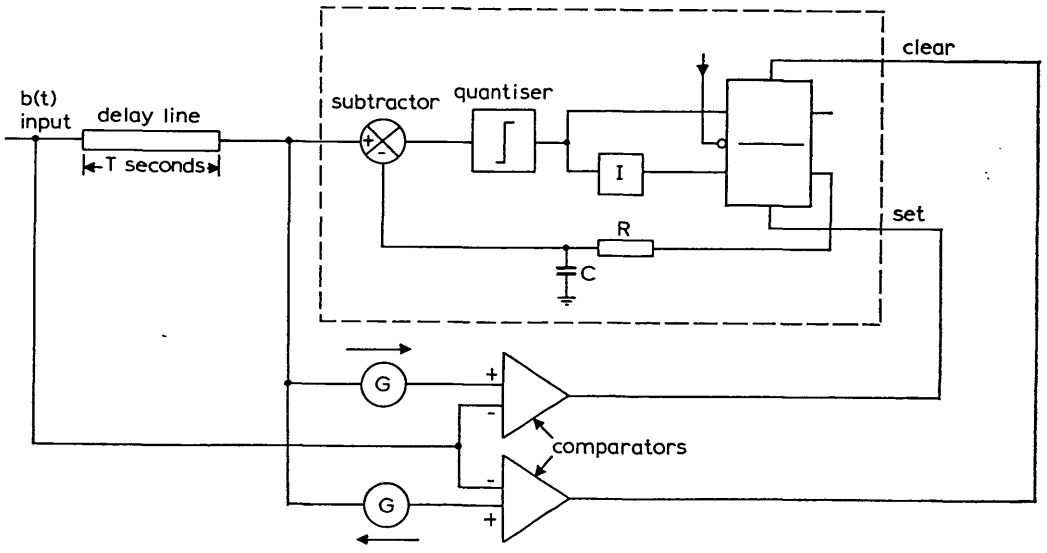

Fig. 1 *Overload predictive delta modulator*