given targets of $TP_{max}$. It should be noted that our comparisons are thus different from most previous comparisons in which networks with the same replication and expansion factors were compared (e.g. comparing a K-RDN to a K-RDN [I]), in which it was always implied that the crossbar switch is more complex.

For all networks under consideration, Fig. 2 shows $C_m$ as a function of the target $TP_{max}$ for $N = 128$. We notice that expansion costs the least $C_m$ for a wide range of target $TP_{max}$. We also notice that for target values of $TP_{max}$ greater than 0.56 (0.61), a CS requires lower $C_m$ than both RDN and DDN (EDN). When the target $TP_{max}$ is less than 0.52, replication requires lower $C_m$ compared to dilation. Fig. 3 shows $C_m$ against $\log N$ for target $TP_{max} = 0.61$. We notice that for $N > 128$, an EDN requires the lowest $C_m$. Finally, the RDN and the DDN keep interchanging their positions while being close to each other.

Summary: Based on extensive results obtained by the approach described in this Letter, the findings, applicable to unbuffered interconnection networks for which one packet only can be accepted at any output in each cycle, are summarised as follows:

(i) Among delta-based networks, the EDN has the lowest number of crosspoints, almost always.

(ii) For target values of the maximum throughput close to the maximum achievable throughput, the CS requires the lowest number of crosspoints for small to moderate network sizes.

(iii) For a given network size, replication outperforms dilation up to a certain target maximum throughput, above which they seem to have comparable complexities.

References


Mapping iterative networks to parallel lookahead circuits

J.-C. Shyur and T.-M. Parng

Indexing term: Finite state machines

Finite state machines have sequential iterative network implementations. The authors show that by the proposed algorithm and Boolean matrix operations, nonlinear parts of recurrence sequences can be eliminated. Linear mapping is thus obtained, which by use of the prefix technique, results in parallel lookahead circuits.

Introduction: Finite state machines can be implemented in iterative networks [I]. However, with parallel input and output, there are signals propagating along the networks, resulting in the worst case $O(N)$ time. However, if the network is associative, by using the prefix technique [2], a product circuit can be achieved with depth exactly $\log N$ and size bounded by $4N$. For simplicity, such circuits are called parallel lookahead circuits [3,4].

In this Letter, we present a mapping from iterative networks to parallel lookahead circuits. We first show how linear recurrence sequences can be mapped to parallel lookahead circuits, and then give an algorithm to transform the nonlinear parts in recurrence sequences into linear parts. The resultant recurrence sequences are shown to be associative in the domain of the propagating signals, and parallel lookahead circuits can be realised.

Linear recurrence sequence: Let the $j$th input and propagating signals of an iterative array be vectors $A^j$ and $Y^j$. The recurrence relation of $Y$ in the form of sum-of-products is

$$y_j' = \sum (\prod \phi_{ij} \prod y_{i-1}^j) + \sum \prod \phi_{ip}$$

for $j = 1, 2, ..., n$, where $n$ is the dimension of $Y$, $\Pi$ and $\Sigma$ are the AND and OR operations.

A special case of eqn. 1 is that each $IY^j$ contains only one component from $Y^j$; eqn. 1 can thus be written in linear form:

$$y_j' = \sum_k e_{jk}(A^j)y_{k-1}^j + c_j(A^j)$$

where $e_{jk}(A^j)$ and $c_j(A^j)$ are Boolean expressions from $A^j$. It then forms a linear mapping by augmenting $Y^j$ by a '1':

$$(Y^j)' = (Y^j)^* 1$$

where $E^j(A^j)$ is the augmented matrix of $(e_j(A^j))$ by $(c_j(A^j))$:

$$E^j(A^j) = \begin{pmatrix} E(A^j) & C(A^j) \\ 0 & 1 \end{pmatrix}$$

We note that Boolean matrix multiplication is implied in eqn. 3 which can be further expanded as

$$(Y^j)' = E^j(A^j)E^{j-1}(A^j) \cdots E^0(A^j)(Y^0)'$$

where $Y^0$ represents the initial values propagated to the first cell of the iterative network. Without proof that Boolean matrix multiplication is associative, eqn. 5 becomes

$$(Y^j)' = (E^j(A^j)E^{j-1}(A^j) \cdots E^0(A^j))(Y^0)'$$

where the term $E^j(A^j)E^{j-1}(A^j) \cdots E^0(A^j)$ can be parallel-computed by a product circuit, forming the following prefix problem.

Prefix problem formulation: It is revealed by eqn. 6 that the prefix technique can be applied to solve the prefix problem [2] defined as follows:

Let $*$ be an associative operation on a domain $D$. The prefix problem is to compute, for given $x_i, i = 1, ..., N$, of the products $x_1 * x_2 * \ldots * x_k, 1 \leq k \leq N$.

Define the Boolean matrix multiplication as operation $*$ and $E(A^j)$ as $x_1, i = 1, ..., N$, we have a prefix problem that computes $E^j(A^j)E^{j-1}(A^j) \cdots E^0(A^j), 1 \leq k \leq N$.

Transform algorithm: Finally we are left with the problem of eliminating the nonlinear parts in eqn. 1 such that the recurrence
sequences can be described by eqn. 2. The algorithm introduces new variables to $Y$ to substitute for inversion or product terms, and decomposes them by recurrence relations of the substituted terms.

Algorithm 1:

(a) Restate eqn. 1 as

$$y_j^n = \sum \prod a_i^n y_i^{n-1} + c_j(A')$$

for $j = 1, \ldots, n$

(b) While there is an inversion signal $y_i^n$ or a product term $Pi y_i^n$ composed of more than one component from $Y^n$,

(i) Add a new component $y_{n+1} = y_i^n$ or $Pi y_i^n$.

(ii) Replace $y_i^n$ or $Pi y_i^n$ by $y_{n+1}$.

(iii) Decompose $y_{n+1}$ by substituting recurrence relations of $y_i^n$ or every $y_j$, such that $y_{n+1}$ can be expressed by a recurrence relation of components of $Y^n$:

$$y_{n+1} = \sum \prod a_i^n y_i^{n-1} + c_{n+1}(A')$$

(c) Describe components of $Y$ as a linear form like in eqns. 2.

Examples: We use two examples to illustrate the mapping: the parity checker and the comparator. Let $p'$ be the parity of $a^t$ to $a$, that is $p' = 1$ if there are odd numbers of $1$s in $[a^t, \ldots, a^t]$. We have the recurrence relation with $p' = 0$:

$$p^n = a^n p^{n-1} + a^n p^{n-1}$$

where the inversion $p'$ can be expanded by algorithm 1:

$$p^n = a^n p^{n-1} + a^n p^{n-1}$$

Finally the linear recurrence sequence can be obtained as follows:

$$\begin{pmatrix} p^n \\ p' \end{pmatrix} = \begin{pmatrix} a^n & a^n \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p^{n-1} \\ p^{n-1} \end{pmatrix}$$

For comparators, we use $g^t$ and $l$ to denote that the number $x^t$ is greater/less than the number $y^t$ ... $y'$, with initially $g^t = p' = 0$. The recurrence relation is

$$g^n = g^{n+1} + g^{n+1} + \bar{p} + g^n y^n$$

where the nonlinear term $\bar{g}$, $l$ can be substituted by $e'$ and expanded:

$$e^n = g^n = e^{n+1} (2x^t y + x^t y')$$

resulting in the linear recurrence sequence

$$\begin{pmatrix} g^n \\ l \\ e' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 2x^t y & 0 & 0 \\ 0 & 1 & 2x^t y & 0 \\ 0 & 0 & 2x^t y + x^t y' & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} g^{n+1} \\ \bar{p}^{n+1} \\ e^{n+1} \\ 1 \end{pmatrix}$$

Summary: We have presented an approach to mapping iterative networks to parallel lookahead circuits. By using the proposed algorithm, nonlinear parts in the recurrence sequences can be eliminated and substituted by linear parts. The resultant linear recurrence sequences are then shown to be expressible by augmented Boolean matrices and matrix multiplications. As the multiplications are associative operations, the linear recurrence sequences can be implemented by parallel lookahead circuits by using the prefix technique. We conclude that all iterative networks can be transformed to parallel lookahead circuits.