Complexity Analysis of Wavelet Signal Decomposition and Reconstruction

ÇETIN K. KOÇ, Member, IEEE
Oregon State University

GUANRONG CHEN, Senior Member, IEEE
University of Houston

CHARLES K. CHUI, Fellow, IEEE
Texas A&M University

We give certain sequential and parallel algorithms and their computational analysis for signal decomposition and reconstruction based on wavelets. The signal decomposition (respectively, reconstruction) process is separated into two stages: The first is the *preprocessing* stage where certain constants are computed for implementation to prepare for the second stage in which signal *decomposition* (respectively, *reconstruction*) is performed. In the *decomposition* (respectively, *reconstruction*) stage, the input signal is transformed via different methods to compute the output signal without changing the setup initialized in the preprocessing stage. We describe certain sequential algorithms for both the preprocessing and the decomposition (respectively, reconstruction) stages, and parallel algorithms for the latter. The algorithms are finally illustrated for compactly supported spline-wavelets and are analyzed in detail in terms of the required arithmetic operations.

Manuscript received March 11, 1993; revised May 8, 1993.

IEEE Log No. T-AES/30/3/16634.

This work was supported by the U.S. Army Research Office under Grant No. DAAL03-91-G-0106, the President's Research and Scholarship Fund, University of Houston, the U.S. Army Research Office under Grant DAAL03-90-0091, and Texas Higher Education Grants TATP32134 and TARP32135.

Authors' addresses: Ç. K. Koç, Dept. of Electrical and Computer Engineering, Oregon State University, Corvallis, OR 97331; G. Chen, Dept. of Electrical Engineering, University of Houston, Houston, TX 77204; C. K. Chui, Dept. of Mathematics and Dept. of Electrical Engineering, Texas A&M University, College Station, TX 77843.

I. WAVELET PRELIMINARIES

An elegant approach to wavelets is the so-called "multiresolution analysis" introduced in [6 and 7]. Let L^2 denote the space of finite-energy functions in the continuous-time domain $(-\infty,\infty)$. A nested sequence $\{V_k\}$ of closed subspaces of L^2 is said to form a *multiresolution analysis* of L^2, if a function $\phi(x)$ in L^2 exists which satisfies the following properties:

1) the family $\{\phi(x-n) : n = 0,\pm1,\pm2,\ldots\}$ is an unconditional basis of V_0;
2) the union of all the V_ks is dense in L^2;
3) the intersection of all the V_ks is the zero function;
4) $f(x) \in V_k$ if and only if $f(x+(1/2^k)) \in V_k$ for all $k = 0,\pm1,\pm2,\ldots$;
5) $f(x) \in V_k$ if and only if $f(2x) \in V_{k+1}$ for all $k = 0,\pm1,\pm2,\ldots$.

Let W_k be the orthogonal complementary subspace of V_{k+1} relative to V_k, and we will use the notation

$$V_{k+1} = V_k \oplus W_k. \tag{1}$$

Then it is clear that $W_k \perp W_n$ for all $k \neq n$, and the entire space L^2 is an orthogonal sum of the spaces W_k, namely:

$$L^2 = \bigoplus_{k=-\infty}^{\infty} W_k.$$

For the purpose of signal decomposition using wavelets, we first need to compute two sequences, $\{a_n\}$ and $\{b_n\}$, as the coefficients of the following identity:

$$\phi(2x-\ell) = \sum_{n=-\infty}^{\infty} \{a_{\ell-2n}\phi(x-n) + b_{\ell-2n}\psi(x-n)\}$$

where $\ell = 0,\pm1,\pm2,\ldots$. We use the notation

$$\phi_{k,j}(x) = \phi(2^k x - j) \qquad \text{and} \qquad \psi_{k,j}(x) = \psi(2^k x - j)$$

$k, j = 0,\pm1,\pm2,\ldots$, without the normalization constant $2^{k/2}$ (in order to expedite the implementation of the signal decomposition process). Using the two sequences $\{a_n\}$ and $\{b_n\}$, we can decompose any signal $f(x)$ in L^2 as a wavelet series

$$f(x) = \sum_{k=-\infty}^{\infty} g_k(x)$$

$g_k(x) \in W_k$. Let us consider the partial sum

$$f_k(x) := g_{k-1}(x) + g_{k-2}(x) + \cdots$$

in V_k, and write

$$f_k(x) = \sum_{j=-\infty}^{\infty} c_{k,j}\phi_{k,j}(x)$$

$$g_k(x) = \sum_{j=-\infty}^{\infty} d_{k,j}\psi_{k,j}(x)$$

where the subscript k in $c_{k,j}$ and $d_{k,j}$ indicates the "level" of decomposition. To calculate the two sequences $\{c_{k,n}\}$ and $\{d_{k,n}\}$ for any fixed k, the following formulas may be used

$$c_{k,n} = \sum_{j=-\infty}^{\infty} a_{j-2n} c_{k+1,j}$$

$$d_{k,n} = \sum_{j=-\infty}^{\infty} b_{j-2n} c_{k+1,j}.$$

For the purpose of signal reconstruction, we also need two sequences $\{p_n\}$ and $\{q_n\}$ in the identities

$$\phi(x) = \sum_{n=-\infty}^{\infty} p_n \phi(2x-n)$$

$$\psi(x) = \sum_{n=-\infty}^{\infty} q_n \phi(2x-n).$$

In this case, we have

$$c_{k+1,n} = \sum_{j=-\infty}^{\infty} (p_{n-2j} c_{k,j} + q_{n-2j} d_{k,j}).$$

For more details, the reader is referred to [2].

II. IMPLEMENTATION OF WAVELET TRANSFORMATIONS

A practical implementation of wavelet signal decomposition and reconstruction requires careful consideration of the details in the mathematics described above. In order to realize such an implementation, we follow the steps of the decomposition and reconstruction processes and algorithmically specify the operations to be performed. It is also important to analyze these algorithms, i.e., calculating the total number of arithmetic operations required. This is used to estimate the running time of a practical implementation once the cost (time) of an arithmetic operation is known for the computer on which the implementation is realized.

In the following, we introduce sequential and parallel algorithms for wavelet signal decomposition and reconstruction and, in particular, study these algorithms when compactly supported spline wavelets are used. We describe sequential algorithms for both preprocessing and decomposition (respectively, reconstruction) stages, and parallel algorithms for the latter. The sequential algorithms are analyzed in detail in terms of arithmetic operations required.

The model of parallel computation is the concurrent-read, exclusive-write parallel random access machine (CREW PRAM) in which processors are allowed to simultaneously read from the same memory cell. However, concurrent writing to the same cell is prohibited [1, 5]. The processors can perform simultaneous reads and writes on separate memory cells. As usual, we will count only parallel arithmetic operations and ignore read/write times.

III. ALGORITHMS FOR SIGNAL DECOMPOSITION

As mentioned above, for signal decomposition we need to use the two sequences $\{a_n\}$ and $\{b_n\}$. The calculation for $\{a_n\}$ and $\{b_n\}$ depends on the basic functions $\phi(x)$ and $\psi(x)$. Usually, these two infinite sequences have to be truncated in real-time computations. In this section, we only study the nonsymmetric case where $a_n \neq a_{-n}$ and $b_n \neq b_{-n}$ in general, since the symmetric case requires approximately one half of the arithmetic operations and hence need not be discussed separately. For convenience in the following discussions, we assume that the two finite subsequences $\{a_n\}$ and $\{b_n\}$, with $-L \leq n \leq 0$ for some large positive integer L, have been obtained in the preprocessing stage and hence are available. This loses no generality since the indices of a finite nonsymmetric sequence can always be shifted. Our aim is then to compute the coefficients $\{c_{k,n}\}$ and $\{d_{k,n}\}$, using the available $\{a_n\}$ and $\{b_n\}$ for general wavelet signal decomposition. In Section V, we discuss special methods for calculating the sequences $\{a_n\}$ and $\{b_n\}$ for spline-wavelets.

A. Sequential Computation of Signal Decomposition

Let N be the signal decomposition level that we have chosen. Suppose that we want to decompose a given signal from level N to level $N-M$ for some integer M such that $0 \leq M \leq N$. Starting from level N, using the data $\{c_{N,n}\}$ with $-L_1 \leq n \leq L_2$, where the two positive integers L_1 and L_2 are arbitrarily chosen, and using the given values of $\{a_n\}$ and $\{b_n\}$ with $-L \leq n \leq 0$, we compute $\{d_{k,n}\}$ and $\{c_{N-M,n}\}$ for $k = N-1, N-2, \ldots, N-M$, and for $-L_1 \leq n \leq L_2$.

Algorithm WD: Wavelet Decomposition

Input. Positive integers N, M, L, L_1, L_2. The input data $\{c_{N,n}\}$ for $-L_1 \leq n \leq L_2$. The sequences $\{a_n\}$ and $\{b_n\}$ for $-L \leq n \leq 0$, obtained during the preprocessing stage, are available.

Output. The sequences $\{d_{k,n}\}$ and $\{c_{N-M,n}\}$ for $N-M \leq k \leq N-1$ and $-L_1 \leq n \leq L_2$.

Step 1. Compute $\{c_{k,n/2}\}$ and $\{d_{k,n/2}\}$ for $k = N-1, N-2, \ldots, N-M$, using the recursions

$$c_{k,n/2} = \sum_{j=-L+n}^{n} a_{j-n} c_{k+1,j}$$

$$d_{k,n/2} = \sum_{j=-L+n}^{n} b_{j-n} c_{k+1,j}$$

where n is even and $-L_1 \leq n \leq L_2$.

We remark that due to the downsampling property of the wavelet signal decomposition, we skip the

computation of $c_{k,n/2}$ and $d_{k,n/2}$ for odd integers $n = \pm 1, \pm 3, \ldots$, as indicated in the algorithm. For particular values of k and $n/2$, we can compute $\{c_{k,n/2}\}$ using at most $2(L+1)$ arithmetic operations since it is obtained by computing the inner-product of two vectors of length $L+1$. However, at step k, not all $c_{k,n/2}$ need to be computed because some $c_{k,n/2}$ would be equal to zero. This is due to the fact that as j takes values from $-L+n$ to n, all of the terms $c_{k+1,j}$ in the summation may be equal to zero. Assuming that the input sequences are nonzero ($a_j \neq 0$ for $-L \leq j \leq 0$ and $c_{N,j} \neq 0$ for $-L_1 \leq j \leq L_2$), we proceed to compute $C_{N-1,n/2}$ for $-L_1 \leq n \leq L_2$. As n takes values from $-L_1$ to L_2, we compute $C_{N-1,n/2}$ for only the values of $n/2$ in the range $-\lfloor L_1/2 \rfloor \leq n/2 \leq \lfloor L_2/2 \rfloor$. During the next step we compute $C_{N-2,n/2}$ for $-L_1 \leq n \leq L_2$ using the previously computed $C_{N-1,j}$. Here, noticing the range of j in the summation

$$c_{N-2,n/2} = a_{-L} \cdot c_{N-1,-L+n} + a_{-L+1} \cdot c_{N-1,-L+1+n} + a_{-L+2} \cdot c_{N-1,-L+2+n} + \cdots + a_0 \cdot c_{N-1,n}$$

and taking into account the fact that $C_{N-1,j}$ is zero for $j < -\lfloor L_1/2 \rfloor$ and $j > \lfloor L_2/2 \rfloor$, we conclude that the summation term for $c_{N-2,n/2}$ has a nonzero value if

$$-\left\lfloor \frac{\lfloor L_1/2 \rfloor}{2} \right\rfloor \leq \frac{n}{2} \leq \left\lfloor \frac{L + \lfloor L_2/2 \rfloor}{2} \right\rfloor.$$

Here, we define the sequence of numbers $L_1^{(k)}$ and $L_2^{(k)}$ for $k = N-1, N-2, \ldots, N-M$ such that

$$L_1^{(k)} = \left\lfloor \frac{L_1^{(k+1)}}{2} \right\rfloor \quad \text{with} \quad L_1^{(N-1)} = \lfloor L_1/2 \rfloor, \tag{2}$$

$$L_2^{(k)} = \left\lfloor \frac{L + L_2^{(k+1)}}{2} \right\rfloor \quad \text{with} \quad L_2^{(N-1)} = \lfloor L_2/2 \rfloor. \tag{3}$$

In general, $c_{k,n/2}$ are nonzero and need to be computed for $n/2$ in the range $-L_1^{(k)} \leq n/2 \leq L_2^{(k)}$ as k ranges from $N-1$ to $N-M$. We hence slightly change Step 1 of Algorithm WD and obtain the following efficient sequential algorithm for computing the sequences $\{c_{k,n}\}$ and $\{d_{k,n}\}$.

Sequential WD Algorithm

```
FOR k = N − 1 TO N − M DO
  FOR i = −L_1^(k) TO L_2^(k) DO
    c_{k,i} := 0
    d_{k,i} := 0
    FOR j = −L + 2i TO 2i DO
      c_{k,i} := c_{k,i} + a_{j−2i} · c_{k+1,j}
      d_{k,i} := d_{k,i} + b_{j−2i} · c_{k+1,j}
    END FOR
  END FOR
END FOR
```

THEOREM 1 *Given the sequences $\{a_n\}$ and $\{b_n\}$ with $-L \leq n \leq 0$, the positive integers N, M, L_1, L_2, and the input data $\{c_{N,n}\}$ with $-L_1 \leq n \leq L_2$, the decomposition algorithm requires at most*

$$4(L+1)(L_1 + L_2 + ML - 2L + M)$$

arithmetic operations to compute the output sequences $\{d_{k,n}\}$ and $\{c_{N-M,n}\}$ for $k = N-1, N-2, \ldots, N-M$ and $-L_1 \leq n \leq L_2$.

PROOF. By counting the total number of arithmetic operations in the above code, we find that the computation of $\{c_{k,n}\}$ and $\{d_{k,n}\}$ for $k = N-1, \ldots, N-M$ and for all $-L_1^{(k)} \leq n \leq L_2^{(k)}$ requires at most

$$4(L+1) \sum_{k=N-1}^{N-M} (L_1^{(k)} + L_2^{(k)} + 1)$$

arithmetic operations. Using the definition (2) and the inequality $\lfloor x \rfloor \leq x$ for any $x \geq 0$, we can write

$$L_1^{(N-1)} \leq \frac{L_1}{2}$$

and by iterating it i times, we obtain

$$L_1^{(N-i)} \leq 2^{-i} L_1$$

which implies that

$$\sum_{i=1}^{M} L_1^{(N-i)} \leq \sum_{i=1}^{M} 2^{-i} L_1 = L_1(1 - 2^{-M}) \approx L_1.$$

Similarly, from (3), we write

$$L_2^{(N-2)} = \left\lfloor \frac{L + \lfloor L_2/2 \rfloor}{2} \right\rfloor \leq \frac{L + L_2/2}{2} = \frac{L_2}{4} + \frac{L}{2}.$$

By iterating this inequality $i-1$ times, we obtain

$$L_2^{(N-i)} \leq \frac{L_2}{2^i} + \frac{L}{2^{i-1}} + \cdots + \frac{L}{2} = 2^{-i}(L_2 - 2L) + L$$

and thus

$$\begin{aligned} \sum_{i=1}^{M} L_2^{(N-i)} &\leq \sum_{i=1}^{M} (2^{-i}(L_2 - 2L) + L) \\ &= (1 - 2^{-M})(L_2 - 2L) + ML \\ &\approx L_2 + ML - 2L. \end{aligned}$$

We therefore obtain

$$\sum_{k=N-1}^{N-M} (L_1^{(k)} + L_2^{(k)} + 1) \leq L_1 + L_2 + ML - 2L + M$$

which implies that the total number of arithmetic operations required by the decomposition algorithm is bounded from above by

$$4(L+1)(L_1 + L_2 + ML - 2L + M)$$

as claimed. □

B. Parallel Computation of Signal Decomposition

We now consider parallelization of the wavelet signal decomposition process. We do not attempt to parallelize the preprocessing stage since these routines are to be called once only at the beginning of the wavelet signal decomposition. Thus, in the following, we assume that the values $\{a_i\}$ and $\{b_i\}$ for $-L \le i \le 0$ are already available. The parallelization of Algorithm WD is based on the following observations:

1) Computation of $\{c_{k,I}\}$ requires that $\{c_{k+1,j}\}$, $-L+2I \le j \le 2I$, be given in advance.
2) Computation of $\{c_{k,J}\}$ is independent of the computation of $\{c_{k,I}\}$ for all possible values of I and $J \neq I$.
3) Simultaneous computations of $\{c_{k,I}\}$ and $\{c_{k,J}\}$ may require simultaneous access to the same values in the sequences $\{c_{k+1,j}\}$ with $-L+2I \le j \le 2I$ and $\{c_{k+1,j}\}$ with $-L+2J \le j \le 2J$.
4) The precomputed values $\{a_i\}$ and $\{b_i\}$ for $-L \le i \le 0$, may also be required simultaneously.

Thus, provided that the parallel computing system is equipped with concurrent read capability, we can compute $\{c_{k,I}\}$ and $\{c_{k,J}\}$ in parallel for all possible values of I and J. This means that the second FOR loop (indexed with i) in Sequential WD Algorithm is parallelized. The following parallel algorithm assumes that the CREW PRAM system is equipped with at least $2(L_1^{(k)} + L_2^{(k)} + 1)$ processors during step k. The first half of processors are indexed with $\mathbf{P}_i$ while the second half with $\mathbf{P}'_i$ for $-L_1^{(k)} \le i \le L_2^{(k)}$. Processors $\mathbf{P}_i$ with $-L_1^{(k)} \le i \le L_2^{(k)}$ are assigned for computation of $\{c_{k,i}\}$ while processors $\mathbf{P}'_i$ for $-L_1^{(k)} \le i \le L_2^{(k)}$ compute $\{d_{k,i}\}$ for all possible values of indices $k = N-1,\ldots,N-M$ and $-L_1^{(k)} \le i \le L_2^{(k)}$.

Parallel WD Algorithm
FOR $k = N-1$ TO $N-M$ DO
  FOR ALL $i = -L_1^{(k)},\ldots,L_2^{(k)}$ DO IN PARALLEL
    $\mathbf{P}_i$ assigns $c_{k,i} := 0$; $\mathbf{P}'_i$ assigns $d_{k,i} := 0$
    FOR $j = -L+2i$ TO $2i$ DO
      $\mathbf{P}_i$ reads $a_{j-2i}, c_{k+1,j}$; $\mathbf{P}'_i$ reads $b_{j-2i}, c_{k+1,j}$
      $\mathbf{P}_i$ performs $c_{k,i} := c_{k,i} + a_{j-2i} \cdot c_{k+1,j}$;
      $\mathbf{P}'_i$ performs $d_{k,i} := d_{k,i} + b_{j-2i} \cdot c_{k+1,j}$
      $\mathbf{P}_i$ writes $c_{k,j}$; $\mathbf{P}'_i$ writes $d_{k,j}$
    END FOR
  END FOR ALL
END FOR

Since $L_1^{(k)} \le \lfloor L_1/2 \rfloor$ and $L_2^{(k)} \le \lfloor L_2/2 \rfloor$ for $N-M \le k \le N-1$, the number of processors required to compute all $\{c_{k,i}\}$ and $\{d_{k,i}\}$ is equal to

$$p = 2(\lfloor L_1/2 \rfloor + \lfloor L_2/2 \rfloor + 1) \le L_1 + L_2 + 2.$$

THEOREM 2 *Suppose that the sequences $\{a_n\}, \{b_n\}$, where $-L \le n \le 0$, the positive integers N, M, L_1, L_2, and the input data $\{c_{N,n}\}$ with $-L_1 \le n \le L_2$, are given. The output sequences $\{d_{k,n}\}$ and $\{c_{N-M,n}\}$, $k = N-1, N-2, \ldots, N-M$ and $-L_1 \le n \le L_2$, can be computed using at most $2M(L+1)$ parallel arithmetic operations on a CREW PRAM with $p = L_1 + L_2 + 2$ processors.*

If the number of processors p is less than $L_1 + L_2 + 2$, then we partition the inner-product operations into p groups and assign a single processor for each group. This means that p processors will accomplish the same task using at most

$$2M(L+1)\left\lceil \frac{L_1 + L_2 + 2}{p} \right\rceil$$

parallel arithmetic steps.

IV. ALGORITHMS FOR SIGNAL RECONSTRUCTION

In this section, we investigate algorithms for wavelet signal reconstruction. Recall from Section I that to recover a signal from its wavelet decomposition, we can use the formula

$$c_{k+1,n} = \sum_{j=-\infty}^{\infty} (p_{n-2j} c_{k,j} + q_{n-2j} d_{k,j}) \tag{4}$$

with $k = 0, \pm 1, \pm 2, \ldots$, and two precomputed sequences $\{p_n\}$ and $\{q_n\}$ determined by the wavelets that we used. The reader is referred to [2] for more details on the above formula and the two sequences $\{p_n\}$ and $\{q_n\}$.

A. Sequential Computation of Signal Reconstruction

The reconstruction algorithm receives, from the signal decomposition stage, the sequences $\{c_{N-M,i}\}$ and $\{d_{k,i}\}$ with $k = N-M, N-M+1, \ldots, N-1$ and $-L_1 \le i \le L_2$ as inputs. It computes $\{c_{k,i}\}$ for $k = N-M, N-M+1, \ldots, N-1$ and $-L_1 \le i \le L_2$ using the formula

$$c_{k+1,i} = \sum_{j=-L}^{0} (p_{i-2j} c_{k,j} + q_{i-2j} d_{k,j})$$

where $-L_1 \le i \le L_2$, under the assumption that all the information needed from the two coefficient sequences $\{p_i\}$ and $\{q_i\}$ are available. In this general case, we have the following algorithm.

Algorithm WR: Wavelet Reconstruction
Input. Positive integers r, s, N, M, L, L_1, and L_2. The two sequences $\{c_{N-M,i}\}$ and $\{d_{k,i}\}$ where $N-M \le k \le N-1$ and $-L_1 \le i \le L_2$. The precomputed sequences $\{p_i\}$ with $0 \le i \le r$ and $\{q_i\}$ with $0 \le i \le s$.

Output. The sequence $\{c_{N,i}\}$ with $-L_1 \le i \le L_2$.

Step 1. Compute $\{c_{k,i}\}$ for $k = N-M, N-M+1, \ldots, N-1$ using the recursion

$$c_{k+1,i} = \sum_{j=-2L}^{0} (p_{i-j} c_{k,j/2} + q_{i-j} d_{k,j/2})$$

where $-L_1 \le i \le L_2$ and $c_{k,j/2} = d_{k,j/2} = 0$ if j is odd.

Due to the upsampling process of the wavelet signal reconstruction, we use $c_{k,j/2} = d_{k,j/2} = 0$ for odd integers $j = \pm 1, \pm 3, \ldots$, as indicated in the algorithm. Also note that for the mth-order spline-wavelets studied in Section V below, we have $r = m$ and $s = 3m-3$.

THEOREM 3 *Assume that the positive integers r, s, N, M, L, L_1, L_2, the sequences $\{p_i\}$ with $0 \le i \le r$ and $\{q_i\}$ with $0 \le i \le s$, and the input sequences $\{c_{N-M,i}\}$ and $\{d_{k,i}\}$ with $N-M \le k \le N-1$ and $-L_1 \le i \le L_2$, respectively, are all given. The reconstruction algorithm computes the output sequence $\{c_{N,i}\}$, $-L_1 \le i \le L_2$, using at most $4M(L+1)(2L + \max(r,s)+1)$ arithmetic operations.*

PROOF. At step k we compute $c_{k+1,i}$ for a particular value of i, which requires at most $2(L+1)$ multiplications and $2(L+1)$ additions. As k ranges from $N-M$ to $N-1$, we perform at most $4M(L+1)$ arithmetic operations to compute $c_{k+1,i}$ for a particular value of i. The coefficients $\{p_i\}$ and $\{q_i\}$ are non-zero for $0 \le i \le r$ and $0 \le i \le s$, respectively. Assuming that the input sequences $\{c_{N-M,i}\}$ and $\{d_{k,i}\}$ are non-zero for $N-M \le k \le N-1$ and $-L_1 \le i \le L_2$, we conclude that the summation for $c_{k+1,i}$ has the terms p_{i-j} and q_{i-j}, and thus $c_{k+1,i}$ needs to be computed if

$$0 \le i-j \le r, \qquad \text{or}$$

$$0 \le i-j \le s$$

or equivalently, if $j \le i \le \max(r,s)+j$. Hence, as j ranges from $-2L$ to 0, we need to compute $\{c_{k+1,i}\}$ for $-2L \le i \le \max(r,s)$. It thus follows that the total number of $c_{k+1,i}$ to be computed is equal to $2L + \max(r,s)+1$. This means that we perform at most $4(L+1)(2L+\max(r,s)+1)$ arithmetic operations to compute all $\{c_{k+1,i}\}$ for $N-M \le i \le N-1$. □

B. Parallel Computation of Signal Reconstruction

We should point out that the above algorithm WR can be parallelized. The parallelization of this algorithm is very similar to that of the decomposition algorithm. We assign $p = 2L + \max(r,s) + 1$ processors for computing $\{c_{k+1,i}\}$ at step k, such that processor $\mathbf{P}_i$ computes $c_{k+1,i}$ for a particular value of i.

```
FOR k = N − M TO N − 1 DO
  FOR ALL i = −2L,...,max(r,s) DO IN
  PARALLEL
    P_i assigns c_{k+1,i} := 0
    FOR j = −L TO 0 DO
      P_i reads p_{i−2j}, q_{i−2j}, c_{k,j}, d_{k,j}
      P_i performs c_{k+1,i} := p_{i−2j} · c_{k,j} + q_{i−2j} · d_{k,j}
      P_i writes c_{k+1,i}
    END FOR
  END FOR ALL
END FOR
```

THEOREM 4 *Suppose that the positive integers r, s, N, M, L, L_1, L_2, the sequences $\{p_i\}$ with $0 \le i \le r$ and $\{q_i\}$ with $0 \le i \le s$, and the input sequences $\{c_{N-M,i}\}$ and $\{d_{k,i}\}$ with $N-M \le k \le N-1$ and $-L_1 \le i \le L_2$, respectively, are all given. The parallel reconstruction algorithm computes the output sequence $\{c_{N,i}\}$ for $-L_1 \le i \le L_2$ using at most $4M(L+1)$ parallel arithmetic operations on a CREW PRAM with $p = 2L + \max(r,s) + 1$ processors.*

V. SPLINE-WAVELET COMPUTATIONS

In the previous sections, we have discussed in detail certain sequential and parallel algorithms and their computations for general wavelet signal decomposition and reconstruction. In this section, we specialize such analysis to spline wavelets.

Let us use the mth-order cardinal B-spline $N_m(x)$ as the low-pass window function $\phi(x)$. Then the corresponding compactly supported spline-wavelet $\psi = \psi_m$, which has a minimum support and can be used to generate all wavelets in W_k, $k = 0, \pm 1, \pm 2, \ldots$. More precisely, $\psi_m(x)$ is given by

$$\psi_m(x) = \sum_{j=0}^{3m-2} \left\{ \frac{(-1)^j}{2^{m-1}} \sum_{\ell=0}^{m} \binom{m}{\ell} N_{2m}(j+1-\ell) \right\} \times N_m(2x-j) \tag{5}$$

(see [2, ch. 6] for more details). Note, in particular, that the supports of N_m and ψ_m are given by

$$\operatorname{supp} N_m = [0, m] \qquad \text{and}$$

$$\operatorname{supp} \psi_m = [0, 2m-1].$$

For these spline-wavelets, the two sequences $\{a_n\}$ and $\{b_n\}$ are uniquely determined by the identities

$$N_m(2x-\ell) = \sum_{n=-\infty}^{\infty} \{a_{\ell-2n} N_m(x-n) + b_{\ell-2n} \psi_m(x-n)\}$$

where $\ell = 0, \pm 1, \pm 2, \ldots$. To compute $\{a_n\}$ and $\{b_n\}$, let $\Pi_{2m-3}(z)$ be the Euler-Frobenius polynomial of degree

$2m-3$ defined inductively by

$$\begin{cases} \Pi_0(z) = 1, \\ \Pi_k(z) = (1+(k-1)z)\Pi_{k-1}(z) + z(1-z)\dfrac{d}{dz}\Pi_{k-1}(z), \\ \qquad\qquad k = 1,2,\ldots. \end{cases} \tag{6}$$

Set

$$P(z) = 2^{-m+1}(1+z)^m$$

$$Q(z) = \frac{2^{-m+1}}{(2m-1)!}(1-z)^m \Pi_{2m-3}(-z)$$

and consider the linear system

$$\begin{bmatrix} P(z) & Q(z) \\ P(-z) & Q(-z) \end{bmatrix} \begin{bmatrix} G(z) \\ H(z) \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

for the two rational polynomials $G(z)$ and $H(z)$. Then $G(z)$ and $H(z)$ are given by

$$G(z) = \frac{1}{2^m}(1+z)^m \frac{\Pi_{2m-3}(z)}{z\Pi_{2m-3}(z^2)}$$

and

$$H(z) = -\frac{(2m-1)!}{2^m}(1-z)^m \frac{1}{z\Pi_{2m-3}(z^2)}.$$

Expanding

$$G(z) = \sum_{k=-\infty}^{\infty} a_k z^k \tag{7}$$

and

$$H(z) = \sum_{k=-\infty}^{\infty} b_k z^k \tag{8}$$

we obtain $\{a_n\}$ and $\{b_n\}$ from the coefficients of the two series.

A. Preprocessing for Spline-Wavelet Signal Decomposition

In this subsection, we discuss the special case of signal decomposition using spline-wavelets. In [3], real-time algorithms using the spline-wavelet ψ_m were derived for decomposing as well as reconstructing a given signal $f(x)$, $x \in (-\infty,\infty)$. For this purpose, the decomposition sequences $\{a_n\}$ and $\{b_n\}$ are computed by using the Euler-Frobenius polynomial $\Pi_{2m-3}(z)$ of degree $2m-3$ as discussed previously. We find it useful to separate the signal decomposition process into two stages. During the *preprocessing stage*, we compute constants which are to be repeatedly used during the *decomposition stage*, where the output data (signal) are produced for each set of the input data (signal). We will only describe and analyze the algorithms for the preprocessing stage since the reconstruction stage has been discussed above for the general case.

For two given positive integers m and L, where m is the order the cardinal B-spline used to generate the wavelet, the preprocessing stage is to compute the coefficient sequences $\{a_i\},\{b_i\}$ given in (7) and (8) for $-L \le i \le 0$. Once these coefficients are computed, they are used repeatedly during the decomposition stage which has been described in detail above.

The first step is to compute the coefficients of the polynomial $\Pi_{2m-3}(z)$ for the given integer m. We first consider the general Euler-Frobenius polynomial $\Pi_k(z)$, which by definition is a degree-k polynomial in z. Let

$$\Pi_k(z) = \sum_{i=0}^{k} \alpha_{k,i} z^i. \tag{9}$$

Then using the recursion (6), we can easily prove that 1) $\alpha_{k,0} = \alpha_{k,k} = 1$, and 2) $\alpha_{k,i} = \alpha_{k,k-i}$ for $0 \le i \le k$. Thus, we only need to compute $\alpha_{k,i}$ for $i = 1,2,\ldots,\lfloor k/2 \rfloor$. Using the recursion (6) and the above definition (9), we obtain

$$\alpha_{k,i} = (1+i)\alpha_{k-1,i} + (k-i+1)\alpha_{k-1,i-1} \tag{10}$$

for $k \ge 2$ and $1 \le i \le k-1$. The following algorithm computes the sequences $\{a_i\},\{b_i\}$ for $-L \le i \le 0$ using the recursion (10) with straight polynomial multiplication and division operations:

Algorithm PSWD: Preprocessing for Spline-Wavelet Decomposition

Input. Positive integers m and L.

Output. The sequences $\{a_i\},\{b_i\}$ with $-L \le i \le 0$.

Step 1. Set $\alpha_{0,0} = \alpha_{1,0} = \alpha_{1,1} = 1$. For $k = 2,3,4,\ldots,2m-3$ and $i = 0,1,2,\ldots,k$, compute $\{\alpha_{k,i}\}$:

1) Set $\alpha_{k,0} = \alpha_{k,k} = 1$.

2) Compute $\alpha_{k,i} = (1+i)\alpha_{k-1,i} + (k-i+1)\alpha_{k-1,i-1}$ for $i = 1,2,3,\ldots,\lfloor k/2 \rfloor$.

3) Set $\alpha_{k,i} = \alpha_{k,k-i}$ for $i = \lfloor k/2 \rfloor + 1, \lfloor k/2 \rfloor + 2,\ldots,k-1$.

Step 2. Compute the product

$$\sum_{i=0}^{3m-3} \beta_i z^i = 2^{-m}(1+z)^m \cdot \Pi_{2m-3}(z)$$

$$= \left[\sum_{i=0}^{m} 2^{-m} C(m,i) z^i\right] \cdot \left[\sum_{i=0}^{2m-3} \alpha_{2m-3,i} z^i\right]$$

where $C(m,i)$ is the number of i combinations of m objects.

Step 3. Let θ_i be the coefficient of z^i in the $(4m-5)$-degree polynomial $z\Pi_{2m-3}(z^2)$. Thus

$$\theta_i = \begin{cases} 0 & \text{for } i \text{ is even} \\ \alpha_{(i-1)/2} & \text{for } i \text{ is odd} \end{cases}$$

for $0 \le i \le 4m-5$. Perform the following polynomial division in order to obtain $G(z)$ as a truncated power series in z, and thus compute the coefficients $\{a_i\}$ for

$-L \le i \le 0$:

$$G(z) = \frac{\sum_{i=0}^{3m-3} \beta_i z^i}{\sum_{i=0}^{4m-5} \theta_i z^i} \approx \sum_{i=-L}^{0} a_i z^i.$$

Step 4. Let

$$-2^{-m}(2m-1)!(1-z)^m = \sum_{i=0}^{m} \gamma_i z^i$$

which implies that $\gamma_i = -2^{-m}(2m-1)!(-1)^m C(m,i)$ for $0 \le i \le m$. Perform the following polynomial division in order to obtain $H(z)$ as a truncated power series in z, and thus compute the coefficients $\{b_i\}$ for $-L \le i \le 0$:

$$H(z) = \frac{\sum_{i=0}^{m} \gamma_i z^i}{\sum_{i=0}^{4m-5} \theta_i z^i} \approx \sum_{i=-L}^{0} b_i z^i.$$

THEOREM 5 *Given positive integers m and L, the computation of $\{a_i\}, \{b_i\}$, $-L \le i \le 0$, requires*

$$16mL - 23m^2 + 84m + 6\lfloor m/2 \rfloor + 4\lfloor \log_2 m \rfloor - 14L - 61$$

arithmetic operations.

PROOF. At Step 1, we compute $\{\alpha_{k,i}\}$ for $k = 2,3,\ldots,2m-3$ and $i = 1,2,\ldots,\lfloor k/2 \rfloor$ via the recursion (10) using a total of

$$\sum_{k=2}^{2m-3} \sum_{i=1}^{\lfloor k/2 \rfloor} 5 = 5(m-1)(m-2)$$

arithmetic operations. The remaining terms are computed using assignment operations.

At Step 2, we first compute $\{C(m,i)\}$, $i = 0,1,2,\ldots,m$. Since $C(m,0) = 1$ and $C(m,1) = m$, and also

$$C(m,i) = \frac{m-i+1}{i} \cdot C(m,i-1)$$

we can compute $\{C(m,i)\}$ for $2 \le i \le \lfloor m/2 \rfloor$, using $4(\lfloor m/2 \rfloor - 1)$ arithmetic operations. Note that $\{C(m,i)\}$, $i = \lfloor m/2 \rfloor + 1, \lfloor m/2 \rfloor + 2,\ldots,m$, can be computed by assignment operations using the rule $C(m,i) = C(m,m-i)$. The computation of 2^{-m}, on the other hand, requires at most $2\lfloor \log_2 m \rfloor$ multiplications using the binary method [4]. We then compute $\{2^{-m}C(m,i)\}$ for $1 \le i \le \lfloor m/2 \rfloor$ using $\lfloor m/2 \rfloor$ multiplications. The remaining terms, $\{2^{-m}C(m,i)\}$ for $i = 0$ and $\lfloor m/2 \rfloor \le i \le m$, can be computed with assignment operations. Thus, a total of $5\lfloor m/2 \rfloor + 2\lfloor \log_2 m \rfloor - 4$ arithmetic operations is needed to compute $\{2^{-m}C(m,i)\}$ for all $0 \le i \le m$.

We now use the coefficients $\{\alpha_{2m-3,i}\}$ for $0 \le i \le 2m-3$ obtained at Step 1 in order to compute the coefficients β_i for $0 \le i \le 3m-3$. Here we note that the product of two polynomials of degree s and t can be computed using the "Cauchy product" rule [4]. To obtain the product polynomial

$$\sum_{i=0}^{s+t} r_i z^i = \left[\sum_{i=0}^{s} p_i z^i\right] \cdot \left[\sum_{i=0}^{t} q_i z^i\right],$$

we compute

$$r_i = p_0 q_i + p_1 q_{i-1} + p_2 q_{i-2} + \cdots + p_i q_0$$

for $i = 0,1,2,\ldots,s+t$, in which $p_i = 0$ with $i > s$ and $q_i = 0$ with $i > t$. This computation requires $st + s + t + 1$ multiplications and st additions, thus, a total of $2st + s + t + 1$ arithmetic operations. Applying this procedure at Step 2, we find that $2m(2m-3) + m + 2m - 3 + 1 = 4m^2 - 3m - 2$ arithmetic operations are required to compute the product $2^{-m}(1+z)^m \cdot \Pi_{2m-2}(z)$. Therefore, Step 2 requires a total of

$$4m^2 - 3m + 5\lfloor m/2 \rfloor + 2\lfloor \log_2 m \rfloor - 6$$

arithmetic operations.

In order to perform the polynomial division operation required at Step 3, we use the synthetic division algorithm [4]. For this purpose, we define the following vectors of length $4m - 4 + L$:

$$\mathcal{B} = \left(\underbrace{0,0,\ldots,0}_{m-2+L \text{ zeros}}, \beta_0, \beta_1, \ldots, \beta_{3m-4}, \beta_{3m-3}\right)$$

$$\Theta_k = \left(\underbrace{0,0,\ldots,0}_{L-k \text{ zeros}}, \theta_0, \theta_1, \ldots, \theta_{4m-6}, \theta_{4m-5}, \underbrace{0,0,\ldots,0}_{k \text{ zeros}}\right).$$

Let $\mathcal{B}[i]$ and $\Theta_k[i]$ be the ith element of the vectors $\mathcal{B}$ and Θ_k, respectively. The vectors $\mathcal{B}$ and Θ_k represent the polynomials

$$\sum_{i=0}^{4m-5} \mathcal{B}[i] z^{i-(m-2)-L} \quad \text{and} \quad \sum_{i=0}^{4m-5} \Theta_k[i] z^{i-k-L}$$

respectively. The polynomial division operation can be performed by executing the following code.

Algorithm PDO: Polynomial Division Operation

Step 1. Set the vectors $\mathcal{B}$ and Θ_0 defined as above.

Step 2. Compute $\{a_i\}$ for $i = -(m-2), -(m-1), -m, \ldots, -L$:

1) Compute

$$a_i = \frac{\mathcal{B}[4m-5+L+m-2+i]}{\Theta_{m-2+i}[4m-5+L+m-2+i]}.$$

2) Update $\mathcal{B} = \mathcal{B} - a_i \cdot \Theta_{m-2+i}$.

3) Obtain $\Theta_{m-2+i-1}$ by shifting Θ_{m-2+i} to the left.

A close inspection shows that the synthetic division algorithm computes $\{a_i\}$ for $-(m-2) \le i \le -L$ using

$$[L-(m-2)+1][1+2(4m-4)] = 8mL - 8m^2 + 31m - 7L - 21$$

arithmetic operations.

At Step 4, we first compute γ_i for $0 \le i \le m$. The computation of $-(-1)^m 2^{-m}(2m-1)!$ requires $2\lfloor \log_2 m \rfloor + 2m - 1$ multiplications. Since $\{C(m,i)\}$ for $0 \le i \le m$ was already computed at Step 2, we proceed to compute $\{\gamma_i\}$ for $2 \le i \le \lfloor m/2 \rfloor$ using $\lfloor m/2 \rfloor - 1$ multiplications, and the remaining terms using only assignment operations. Thus, the computation of $\{\gamma_i\}$ for $0 \le i \le m$ requires

$$2m + \lfloor m/2 \rfloor + 2\lfloor \log_2 m \rfloor - 2$$

arithmetic operations. In order to compute $\{b_i\}$ for $-L \le i \le 0$, we divide a polynomial of degree m by a polynomial of degree $4m-5$. Using a code similar to the one given above, we compute $\{b_i\}$, $-(3m-5) \le i \le -L$, using

$$[L-(3m-5)+1][1+2(4m-4)]$$
$$= 8mL - 24m^2 + 69m - 7L - 42$$

arithmetic operations. Thus, Step 4 requires a total of

$$8mL - 24m^2 + 71m + \lfloor m/2 \rfloor - 7L + 2\lfloor \log_2 m \rfloor - 44$$

arithmetic operations. Adding these numbers together, we arrive at the stated result. □

B. Preprocessing for Spline-Wavelet Signal Reconstruction

In this subsection, we illustrate the preprocessing of signal reconstruction, where we compute two coefficient sequencies $\{p_n\}$ and $\{q_n\}$, which need to be repeatedly used in the reconstruction stage, by examining in detail the spline-wavelet case.

Let $P(z)$ and $Q(z)$ be polynomials of degree m and $3m-3$, respectively, in the form

$$P(z) = 2^{-m+1}(1+z)^m = \sum_{i=0}^{m} p_i z^i \tag{11}$$

$$Q(z) = \frac{2^{-m+1}}{(2m-1)!}(1-z)^m \Pi_{2m-3}(-z)$$
$$= \sum_{i=0}^{3m-3} q_i z^i. \tag{12}$$

During the preprocessing stage, we compute the coefficients of the above polynomials, i.e., $\{p_i\}$ with $0 \le i \le m$ and $\{q_i\}$ with $0 \le i \le 3m-3$.

Algorithm PSWR: Preprocessing for Spline-Wavelet Reconstruction

Input. Positive integer m.

Output. Sequences $\{p_i\}$ with $0 \le i \le m$ and $\{q_i\}$ with $0 \le i \le 3m-3$.

Step 1. Compute $\{p_i\}$ for $0 \le i \le m$ using $p_i = 2^{-m+1}C(m,i)$.

Step 2. Set $\alpha_{0,0} = \alpha_{1,0} = \alpha_{1,1} = 1$. For $k = 2,3,4,\ldots,2m-3$ and $i = 0,1,2,\ldots,k$, compute $\{\alpha_{k,i}\}$:

1) Set $\alpha_{k,0} = \alpha_{k,k} = 1$.

2) Compute $\alpha_{k,i} = (1+i)\alpha_{k-1,i} + (k-i+1)\alpha_{k-1,i-1}$, $i = 1,2,3,\ldots,\lfloor k/2 \rfloor$.

3) Set $\alpha_{k,i} = \alpha_{k,k-i}$, $i = \lfloor k/2 \rfloor + 1, \lfloor k/2 \rfloor + 2, \ldots, k-1$.

Step 3. Let

$$\Pi_{2m-3}(-z) = \sum_{i=0}^{2m-3} \beta_i z^i$$

$$\frac{2^{-m}}{(2m-1)!}(1-z)^m = \sum_{i=0}^{m} \gamma_i z^i$$

or equivalently,

$$\beta_i = (-1)^i \alpha_i \qquad \text{for} \quad 0 \le i \le 2m-3$$

$$\gamma_i = \frac{2^{-m}}{(2m-1)!}(-1)^m C(m,i) \qquad \text{for} \quad 0 \le i \le m$$

where $C(m,i)$ is the number of i combinations of m objects. Compute the product

$$\sum_{i=0}^{3m-3} q_i z^i = \left[\sum_{i=0}^{m} \beta_i z^i\right] \cdot \left[\sum_{i=0}^{2m-3} \gamma_i z^i\right]$$

using the Cauchy product rule.

THEOREM 6 *For a given positive integer m, the computation of $\{p_i\}$ with $0 \le i \le m$ and $\{q_i\}$ with $0 \le i \le 3m-3$ requires*

$$9m^2 - 16m + 6\lfloor m/2 \rfloor + 2\lfloor \log_2 m \rfloor + 2\lfloor \log_2 (m-1) \rfloor + 3$$

arithmetic operations.

PROOF. We will use the results of the analysis given in the Proof of Theorem 1 since similar computations are performed. At Step 1, the computation of $p_i = 2^{-m+1}C(m,i)$ for $0 \le i \le m$ requires $5\lfloor m/2 \rfloor + 2\lfloor \log_2 (m-1) \rfloor - 4$ multiplications. Step 2 requires $5(m-1)(m-2)$ arithmetic operations, as was shown. At Step 3, we compute β_i for $0 \le i \le 2m-3$ using assignment operations. In order to compute γ_i, we first compute $(-1)^m 2^{-m}$ and $(2m-1)!$ using $2\lfloor \log_2 m \rfloor$ and $2m-2$ multiplications, respectively. We then perform a single division to obtain $(-1)^m 2^{-m}/(2m-1)!$. Using the already computed values $\{C(m,i)\}$, $0 \le i \le m$, we compute $\{\gamma_i\}$ for $1 \le i \le \lfloor m/2 \rfloor$ with $\lfloor m/2 \rfloor$ multiplications. The remaining $\{\gamma_i\}$ are computed using assignment operations. The multiplication of a degree m polynomial by a degree $2m-3$ polynomial, on the other hand, requires $2m(2m-3) + m + 2m - 3 + 1 = 4m^2 - 3m - 2$ arithmetic operations. Adding these numbers together yields the stated result. □

REFERENCES

[1] Akl, S. G. (1989) *The Design and Analysis of Parallel Algorithms.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[2] Chui, C. K. (1992)
An Introduction to Wavelets.
Boston, MA: Academic Press, 1992.

[3] Chui, C. K., and Wang, J. Z. (1992)
On compactly supported spline wavelets and a duality principle.
Transactions of American Mathematical Society, 330 (1992), 903–916.

[4] Knuth, D. E. (1981)
The Art of Computer Programming, Vol. 2: *Seminumerical Algorithms.*
Reading, MA: Addison-Wesley, 1981.

[5] Lakshmivarahan, S., and Dhall, S. K. (1990)
Analysis and Design of Parallel Algorithms.
New York: McGraw-Hill, 1990.

[6] Mallat, S. (1988)
Multiresolution representations and wavelets.
Ph.D. dissertation, University of Pennsylvania, 1988.

[7] Meyer, Y. (1986)
Ondelettes et functions splines.
Seminaire Equations aux Derivees Partielles, Ecole Polytechnique, Paris, France, 1986.

Çetin Kaya Koç (M'88) received his B.S. degree in 1980, summa cum laude, and M.S. degree in 1982, both in electrical engineering from Istanbul Technical University, and his M.S. and Ph.D. degrees in 1985 and 1988, respectively, in electrical and computer engineering from the University of California at Santa Barbara.

Since September 1992, he has been an Assistant Professor in the Department of Electrical and Computer Engineering at Oregon State University. From June 1988 to September 1992, he was an Assistant Professor at the University of Houston. He was a United Nations' Fellow and a Visiting Professor at Istanbul Technical University during the summer of 1992. He is also working as an industrial consultant in the area of data and communication security. His research interests include parallel computation, scientific computing, computer arithmetic, computer algebra, and cryptology.

Dr. Koç is member of IEEE (Computer Society), ACM, SIAM, AMS, MAA, and IACR.

Guanrong Chen (M'89—SM'92) received the M.S. degree in computer science from Sun Yat-sen (Zhongshan) University, China in 1981, and the Ph.D. degree in mathematics from Texas A&M University, College Station, in 1987.

After working as a postdoctorate fellow at the Center for Approximation Theory, Texas A&M University for a few months, he joined Rice University, Houston, Texas, where he held the positions of Senior Research Associate and Visiting Assistant Professor in both the Department of Electrical and Computer Engineering and the Department of Mathematical Sciences from 1987 to 1990. He joined the Department of Electrical Engineering, University of Houston, Texas, as an Assistant Professor in July 1990. His research interests include spline approximation theory with its engineering applications, nonlinear systems dynamics and control, robotics, real-time estimation and filtering, and numerical computation.

Charles K. Chui (M'79—SM'89—F'93) was born on May 7, 1940, and obtained his B.S., M.S., and Ph.D. degrees in 1962, 1963, and 1967, respectively, all from the University of Wisconsin, Madison.

After teaching at SUNY, Buffalo, for three years, he moved to College Station, Texas, and has been on the faculty of Texas A&M University ever since, as Associate Professor (1970–1974), Professor (1974–1989), and Distinguished Professor since 1989. He has joint appointments in the Departments of Mathematics, Electrical Engineering, and Statistics and is the Director of the Center for Approximation Theory. His current research interests include multivariate splines, wavelets, signal processing, and systems theory.

Dr. Chui's academic activities include being editor of eight journals, two book series, and organizer of several international and domestic conferences.