The Cascading Haar Wavelet algorithm for computing the Walsh-Hadamard Transform

Andrew Thompson

***Abstract*—A novel algorithm for computing the Walsh Hadamard Transform (WHT) is proposed which consists entirely of Haar wavelet transforms. It is proved that the algorithm, referred to as the Cascading Haar Wavelet (CHW) algorithm, shares precisely the same serial complexity as the popular divide-and-conquer algorithm for the WHT. A natural parallelization of the algorithm is also proposed which has a number of attractive features.**

***Index Terms*—Walsh-Hadamard Transform, Haar wavelet transform, complexity analysis. EDICS: DSP-FAST.**

I. Introduction

The Walsh-Hadamard Transform (WHT) is a staple of the digital signal processing world, and is used extensively in communication systems, image processing, and in general as a proxy for the Fast Fourier Transform (FFT) [1]. Like the FFT, it is well known that the WHT of a signal of length n, where n is a power of 2, can be computed with $\mathcal{O}(n \log n)$ complexity. There exist well established algorithms for computing the WHT based on divide-and-conquer principles, which exploit the recursive properties of the transform, namely that any WHT of size 2^m can be broken down into two WHTs of size 2^{m-1}. Various orderings of WHT coefficients are possible, most notably natural, dyadic and sequency orderings, and classical WHT algorithms essentially differ depending upon the desired ordering. See [1] for background on the various WHT orderings and their corresponding algorithms. These fundamental algorithms have been known for many years, and more recent work has focused on practical considerations, such as how to incorporate existing algorithms into parallel architectures [8], [3] and FPGAs [2], and how to compute the WHT on sliding windows for the purposes of image filtering [7], [4], [9].

It has been previously noted that there exist interesting relationships between the WHT with dyadic ordering and the oldest and simplest discrete wavelet transform, the Haar wavelet transform. It was observed in [5] that computing the WHT of a signal has a striking interpretation in terms of Haar wavelet coefficients: it is equivalent to applying WHTs of different sizes independently to the coefficients within each scale of the Haar wavelet transform. Based on this observation, the authors propose a *Haar-Walsh* transform, which transforms Haar wavelet coefficients into WHT coefficients, thereby giving an alternative approach to computing the WHT: via a detour into the Haar wavelet domain. This approach was shown to match the complexity of the standard algorithms, and it has the additional appeal of computing the Haar wavelet transform for free in the process. Nonetheless, it appears that the approach never became a popular alternative to the standard WHT algorithms. It is also worth noting that, while a single Haar wavelet transform is computed at the start, the algorithm proceeds using the standard divide-and-conquer approach within each scale of the Haar wavelet transform thereafter.

In this paper, a novel algorithm is proposed for computing the WHT (with coefficients in dyadic order) which is inspired by some of the connections between WHTs and Haar wavelet transforms, but which is fundamentally different from all preceding algorithms. Its marked difference is apparent from the fact that it consists entirely of Haar wavelet transforms. It is shown that the algorithm, which we call the Cascading Haar Wavelet (CHW) algorithm, matches the serial complexity of the standard algorithms for either the natural or dyadic orderings, requiring precisely $n \log_2 n$ addition operations for its computation. Furthermore, a natural parallelization of the algorithm is proposed in which each of the nodes in the parallel architecture performs a single fixed task, namely a Haar wavelet transform of a given size.

II. Relation to prior work

The relationship between the WHT and the Haar wavelet transform has been particularly explored in two papers. In [5], the authors consider a *Haar-Walsh* transform which transforms Haar wavelet coefficients into WHT coefficients (in dyadic order), which they observe to be equivalent to multiplication by the matrix

$$H_m \Psi_m^T = \begin{bmatrix} 1 & & & & \\ & H_0 & & & \\ & & H_1 & & \\ & & & \ddots & \\ & & & & H_{m-1} \end{bmatrix}, \qquad (1)$$

where here H_m denotes the $2^m \times 2^m$ Hadamard matrix with columns in dyadic (Paley) order and Ψ_m denotes the $2^m \times 2^m$ Haar matrix. The Haar-Walsh transform can therefore be computed by taking separate WHTs of the Haar wavelet coefficients at each scale. See also [10] by the current author in which the implications of this decomposition are explored for multilevel compressive sensing.

The possibility of recursively decomposing Hadamard matrices using (1) appears to be spotted in the concluding remarks

A. Thompson is with the Mathematical Institute, University of Oxford, UK.

of [6], and indeed it is possible to derive the CHW algorithm from repeated application of (1). Closely related though the ideas in [6] are, to the author's best knowledge there is no mention in the literature of a WHT consisting entirely of Haar wavelet transforms, nor any statement of the decomposition result given in Theorem 1.

III. Description of the Algorithm

Given $m \geq 0$, the $2^m \times 2^m$ Hadamard matrix with columns in dyadic (Paley) order [5], [1], H_m, is defined by the recursion

$$H_0 := 1; \quad H_{m+1} = \frac{1}{\sqrt{2}} \begin{bmatrix} H_m \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ H_m \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \text{ for } m \geq 0, \tag{2}$$

where $\otimes$ denotes the Kronecker product. Given $m \geq 0$, the $2^m \times 2^m$ Haar matrix [1], Ψ_m, may be defined by the recursion

$$\Psi_0 := 1; \quad \Psi_{m+1} = \frac{1}{\sqrt{2}} \begin{bmatrix} \Psi_m \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ I_m \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \text{ for } m \geq 0, \tag{3}$$

where we write I_m for the $2^m \times 2^m$ identity matrix.

The CHW algorithm is based on a particular decomposition of a Hadamard matrix in terms of Haar wavelet transform matrices. We use the notation $\prod_{r=1}^{p} M_r = M_p \cdots M_2 M_1$ for a p-fold matrix product.

Theorem 1:

$$H_m = \left\{ \prod_{r=1}^{m-1} I_{r-1} \otimes \begin{bmatrix} I_{m-r} & 0 \\ 0 & \Psi_{m-r} \end{bmatrix} \right\} \Psi_m, \text{ for } m \geq 1. \tag{4}$$

Proof: We proceed by induction. The result holds trivially for $m = 1$. Assume (4) holds for $m-1$. Then

$$\begin{aligned}
&\sqrt{2} \cdot \left\{ \prod_{r=1}^{m-1} I_{r-1} \otimes \begin{bmatrix} I_{m-r} & 0 \\ 0 & \Psi_{m-r} \end{bmatrix} \right\} \Psi_m \\
&= \left\{ \prod_{r=1}^{m-1} I_{r-1} \otimes \begin{bmatrix} I_{m-r} & 0 \\ 0 & \Psi_{m-r} \end{bmatrix} \right\} \begin{bmatrix} \Psi_m \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ I_m \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \\
&= \left\{ \prod_{r=2}^{m-1} I_{r-1} \otimes \begin{bmatrix} I_{m-r} & 0 \\ 0 & \Psi_{m-r} \end{bmatrix} \right\} \cdot \begin{bmatrix} I_{m-1} & 0 \\ 0 & \Psi_{m-1} \end{bmatrix} \begin{bmatrix} \Psi_{m-1} \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ I_{m-1} \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \\
&= \left\{ \prod_{r=2}^{m-1} I_{r-1} \otimes \begin{bmatrix} I_{m-r} & 0 \\ 0 & \Psi_{m-r} \end{bmatrix} \right\} \begin{bmatrix} \Psi_{m-1} \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ \Psi_{m-1} \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \\
&= I_1 \otimes \left\{ \prod_{r=2}^{m-1} I_{r-2} \otimes \begin{bmatrix} I_{m-r} & 0 \\ 0 & \Psi_{m-r} \end{bmatrix} \right\} \cdot \begin{bmatrix} \Psi_{m-1} \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ \Psi_{m-1} \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \\
&= \begin{bmatrix} \prod_{r=1}^{m-2} I_{r-1} \otimes \begin{bmatrix} I_{m-1-r} & 0 \\ 0 & \Psi_{m-1-r} \end{bmatrix} \Psi_{m-1} \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ \prod_{r=1}^{m-2} I_{r-1} \otimes \begin{bmatrix} I_{m-1-r} & 0 \\ 0 & \Psi_{m-1-r} \end{bmatrix} \Psi_{m-1} \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix}
\end{aligned}$$

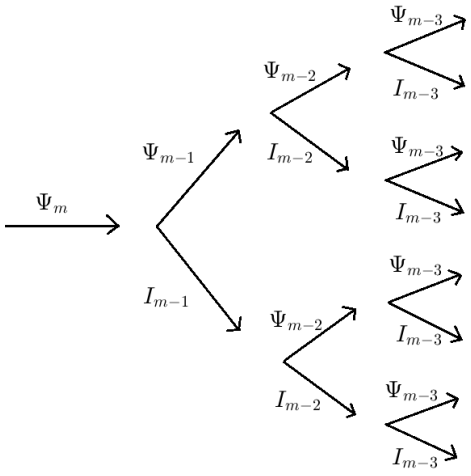

Fig. 1. A conceptual illustration of the CHW algorithm.

which, by the inductive hypothesis, is equal to

$$\begin{aligned}
&= \begin{bmatrix} H_{m-1}\Psi_{m-1}^T\Psi_{m-1} \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ H_{m-1}\Psi_{m-1}^T\Psi_{m-1} \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix} \\
&= \begin{bmatrix} H_m \otimes \begin{pmatrix} 1 & 1 \end{pmatrix} \\ H_m \otimes \begin{pmatrix} 1 & -1 \end{pmatrix} \end{bmatrix}, \\
&= \sqrt{2} \cdot H_m.
\end{aligned}$$

■

Expanding the product in (4), we have

$$H_m = \left\{ \begin{bmatrix} I_1 & & & & & & \\ & \Psi_1 & & & & & \\ & & I_1 & & & & \\ & & & \Psi_1 & & & \\ & & & & \ddots & & \\ & & & & & I_1 & \\ & & & & & & \Psi_1 \end{bmatrix} \cdots \right.$$

$$\left. \cdots \begin{bmatrix} I_{m-2} & & & \\ & \Psi_{m-2} & & \\ & & I_{m-2} & \\ & & & \Psi_{m-2} \end{bmatrix} \begin{bmatrix} I_{m-1} & \\ & \Psi_{m-1} \end{bmatrix} \right\} \Psi_m,$$

which shows that the WHT can be computed by first computing the Haar wavelet transform, and then employing a divide-and-conquer approach also consisting of Haar wavelet transforms, as illustrated in Figure 1.

The identity transforms are included to conceptually illustrate the decomposition formula (4). However, in practice only the Haar wavelet transforms need to be performed. See Section V and Figure 2 for a practical illustration of the implementation of the algorithm. The complexity of the CHW algorithm is analyzed in Section IV, where it is shown that the algorithm requires $n \log_2 n$ summations, where $n = 2^m$: exactly the same as the standard WHT algorithms [1].

IV. Complexity analysis

We have proposed a method for computing the WHT which is built up entirely of Haar wavelet transforms. Analysing its complexity therefore requires a complexity result for the Haar wavelet transform.

Lemma 1 ([11, Section 7.3.3]): The Haar wavelet transform corresponding to multiplication by Ψ_m can be computed in $2(2^m - 1)$ operations.

Equipped with this result, we can determine the complexity of the CHW algorithm.

Theorem 2: The CHW algorithm can be implemented in $n \log_2 n$ operations.

Proof: From Figure 1 we see that the CHW algorithm requires a single Haar wavelet transform of size 2^m, and 2^{m-1-r} Haar wavelet transforms of size 2^r, for $r = 1, 2, \ldots, m-1$. By Lemma 1, the total number of operations is therefore

$$\begin{aligned} & 2(2^m - 1) + \sum_{r=1}^{m-1} \left\{ 2^{m-1-r} \cdot 2(2^r - 1) \right\} \\ = \; & 2^{m+1} - 2 + \sum_{r=1}^{m-1} 2^m - \sum_{r=1}^{m-1} 2^{m-r} \\ = \; & 2^{m+1} - 2 + 2^m(m-1) - 2(2^{m-1} - 1), \end{aligned}$$

which simplifies to $m \cdot 2^m = n \log_2 n$. ∎

We have shown that the CHW algorithm has precisely the same serial complexity as the popular divide-and-conquer algorithms for the WHT. In the next section, we propose a natural way of parallelizing the CHW.

V. A proposal for a parallel implementation

Given the WHT's importance in signal processing, it is not surprising that there already exists a body of work addressing the question of how to efficiently parallelize it; see [8] for an example. In this section, we make the observation that there is a very natural way to parallelize the CHW algorithm which possesses a number of attractive features.

In the CHW algorithm, a signal of length 2^m is cascaded through a succession of Haar wavelet transforms. It is possible therefore to consider a parallel architecture in which each of $m-1$ nodes is devoted to the task of performing Haar wavelet transforms of a certain size. A scheduling chart illustrating this procedure for $m = 4$ is shown in Figure 2. In this case, we have three nodes, each devoted to the task of performing the Haar wavelet transforms Ψ_1, Ψ_2 and Ψ_3. A full Haar wavelet transform Ψ_4 must first be performed (by one of the three nodes, or by an extra one), and thereafter each node is occupied for approximately half of the total running time. The output is the WHT coefficients in dyadic order.

Note the attractive properties of this scheme: each node need only be programmed to perform a single task, and communication of the output from any given node follows fixed and straightforward rules. The algorithm could therefore

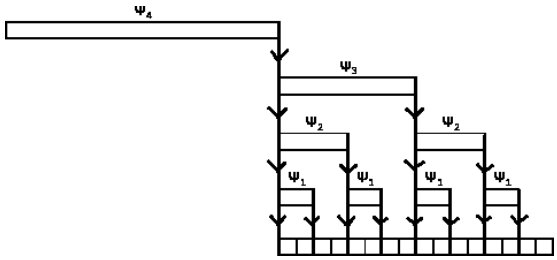

Fig. 2. An illustration of the proposed parallel implementation of the CHW algorithm.

be implemented using a circuit in which individual processors are programmed to perform a fixed task, always receiving input from and distributing output to the same processors. By contrast, parallel implementations of traditional WHT algorithms typically require careful readjustment of shared memory allocation [3]. Another attractive property of the scheme is that, since output is cascaded from one node to another, synchronization of the nodes occurs automatically; each node performs its computation when it has received all necessary input. The algorithm can therefore be implemented by an asynchronous circuit (one which is not governed by a global clock signal).

VI. Concluding remarks

A novel Cascading Haar Wavelet (CHW) algorithm has been proposed for computing the WHT. A parallelization scheme has also been proposed, and it remains to comprehensively understand the practical implementation advantages that the CHW might have over other approaches to parallelization of the WHT.

References

[1] Agaian, S., Sarukhanyan, H., Egiazarian, K. and Astola, J. *Hadamard Transforms*. SPIE Press, 2011.

[2] Amira, A. and Chandrasekaran, S. *Power Modeling and Efficient FPGA Implementation of FHT for Signal Processing*. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 15(3), pp. 286–295, 2007.

[3] Andrade, J., Falcao, G. and Silva, V. *Optimized Fast Walsh-Hadamard Transform on GPUs for Non-Binary LDPC Decoding*. Parallel Computing 40(9), pp. 449–453, 2014.

[4] Ben-Artzi, H., Hel-Or, Y. and Hel-Or, H. *The gray-code filter kernels*. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(3), pp. 382–393, 2007.

[5] Falkowski, B. and Rahardja, S. *Walsh-like functions and their relations*. IEEE Proceedings on Vision, Image and Signal Processing 143(5), pp. 279–284, 1996.

[6] Fino, B. *Relations Between Haar and Walsh/Hadamard Transforms*. Proceedings of the IEEE 60(5), pp. 647–648, 1972.

[7] Hel-Or, Y. and Hel-Or, H. *Real-Time Pattern Matching Using Projection Kernels*. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(9), pp. 1430–1435, 2005.

[8] Ososanya, E., Chen, J. and Poo, R. *Performance evaluation of parallel fast Walsh transform algorithms on a shared-memory multiprocessor computer*. Southeastern Symposium on System Theory, pp.562–566, March 1994.

[9] Ouyang, W. and Cham, W. *Fast algorithm for Walsh-Hadamard transform on sliding windows*. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(1), pp. 165–171, 2010.

[10] Thompson, A. and Calderbank, R. *Compressive imaging using fast transform coding*. SPIE 9992, Emerging Imaging and Sensing Technologies, September 2016.

[11] Wang, R. *Introduction to Orthogonal Transforms with Applications in Data Processing and Analysis*. Cambridge, 2012.