Math. Maced.
Vol. 7 (2009)
69-80

PARALLEL IMPLEMENTATION OF 2D HAAR TRANSFORM IN NOTHING SHARE ARCHITECTURE

JAUMIN AJDARI

Abstract. Wavelets transform, particularly the Discrete Wavelet Transform (DWT) is an important problem for many applications. This paper describes parallel implementation of the simplest wavelet transform, namely the Haar transform. We have described the mathematical theory of 2D Haar transformation and algorithmic implementation of their computations. From the mathematical theory we have introduced the Haar transform and 2D Haar transformation and we have also shown the relations which we use for their computations. From the algorithmic implementation we have done 2D sequential and parallel implementation and for the parallel implementation we propose several types of parallelization, like the parallelization on mesh and non mesh architecture. Parallel implementations are done based on the nothing share computer system architecture and the MPI programming paradigm. We also analyze their complexity and show that the complexity of parallel implementation depends on the number of processes and on the number of real processing units used for process mapping.

1. Introduction

The wavelet transform presents an important mathematical tool in many disciplines such as signal processing ([1], [2], [3], [4], [5], [6], [7]), solving partial differential equations ([8], [9], [10]), etc. Haar wavelets are a simple example of wavelets. Haar Transform or Haar wavelet transform also presents the simplest example of orthogonal transformation with compact support, being the first known wavelet proposed from Alfred Haar in 1909. In general, the wavelet transform are similar to Fourier Transform, the difference lies in that that the Fourier transform localizes only within the frequency domain whereas the wavelet transform localizes in both, i.e. time and frequency domain. This paper is divided into two essential parts. The first part provides the mathematical definitions and the mathematical aspect of transform computations ([11], [12], [13], [14]), and the second part provides the algorithmic aspect ([15], [16], [17], [18], [19], [20], [21]), in which we analyze some possibilities of parallel computations in calculating the 2D Haar transform. The part dealing with the algorithmic aspect shows the analyses of

2000 *Mathematics Subject Classification.* 68W10.

Key words and phrases. Haar function, Multiresolutional analyses, Discret Haar transform, Parallel transform, MPI.

2D transformations. The complexity of estimation is also given for the proposed algorithm in order to gain an overview of the benefits of parallelization.

2. The math of Haar transform

The wavelet theory gives two functions that have an important role during wavelet analysis, the scaling function called the father wavelet and the other function known as mother wavelet (sometimes detail function). The scaling function is usually denoted with φ, (φ:R $\to$ R) and the mother wavelet with ψ , (ψ:R $\to$ R). The simple wavelet analysis is based on the Haar scaling function and Haar wavelet (detail function) which are defined as follows

$$\varphi(t) = \chi_{[0,1)}(t)$$

and

$$\psi(t) = \chi_{\left[0,\frac{1}{2}\right)}(t) - \chi_{\left[\frac{1}{2},1\right)}(t). \tag{2.1}$$

With the scaling function $\varphi(t)$, by using the dyadic dilation and the right translation by an integer number, so $\varphi_{j,k}(t) = 2^{j/2}\varphi(2^j t - k)$, is defined the Hilbert space denoted as V_j. The set $\{\varphi_{j,k}(t) = 2^{j/2}\varphi(2^j t - k) | k \in \mathrm{Z}\}\}$ represents the basis of space V_j . The space V_j is the space of all step functions at level j:

$$f_j(t) = a_k, \frac{k}{2^j} \leq t < \frac{k+1}{2^j}.$$

Spaces V_j form a multiresolutional analysis in $L^2(\mathrm{R})$ and $V_j \subset V_{j+1}$. From $V_j \subset V_{j+1}$ we define space W_j therefore $V_j \perp W_j$ and $V_{j+1} = V_j \oplus W_j$. It can be shown that the set of functions $\{\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k) | k \in \mathrm{Z}\}\}$ presents an orthonormal basis for space W_j . The space W_j is known as wavelets mother space at level j and complies with $V_{j+1} = V_j \oplus W_j$. Continuing in a recursive way we obtain:

$$V_{j+1} = V_j \oplus W_j = V_0 \oplus W_0 \oplus W_1 \oplus ... \oplus W_j, j \in \mathrm{Z}. \tag{2.2}$$

In the two dimensional case, the basis of space V_j^2, ($V_j^2 = V_j \otimes V_j$), is contracted as tensor product of all possible basis functions $\varphi_{j,k}(t)$ and $\psi_{j,k}(t)$. Spaces V_j^2 form the multiresolutinal space in $L^2\left(\mathrm{R}^2\right)$. Let φ be the scaling function and ψ corresponding detail function (mother wavelet) that generate an orthonormal base into $L^2(\mathrm{R})$. The set of functions

$$\varphi_{k,l}^j(t_1,t_2) = \varphi_{j,k}(t_1)\varphi_{j,l}(t_2), \psi_{j,k,l}^{01}(t_1,t_2) = \varphi_{j,k}(t_1)\psi_{j,l}(t_2)$$
$$\psi_{j,k,l}^{10}(t_1,t_2) = \psi_{j,k}(t_1)\varphi_{j,l}(t_2), \psi_{j,k,l}^{11}(t_1,t_2) = \psi_{j,k}(t_1)\psi_{j,l}(t_2)$$

is an orthonormal basis in space $L^2(\mathrm{R}^2)$ ([9], [10]). According to the hierarchy of spaces $V_j \subset V_{j+1}$ and $W_j \subset V_{j+1}$, we know that there are sequences $\{p_k\}_{k\in\mathrm{Z}}, \{q_k\}_{k\in\mathrm{Z}} \in \ell^2(\mathrm{Z})$ that satisfy

$$\varphi(t) = \sum_k p_k \varphi(2t-k)$$

and

$$\psi(t) = \sum_k q_k \varphi(2t-k). \tag{2.3}$$

From (2.3) and the definition of $\varphi(t)$ and $\psi(t)$ we have solution $p_0 = p_1 = 1$, $q_0 = -q_1 = 1$, $p_k = q_k = 0$, $k \in \mathrm{Z}\backslash\{0,1\}$ and the last relations are

$$\varphi(t) = \varphi(2t) + \varphi(2t-1)$$

and

$$\psi(t) = \varphi(2t) - \varphi(2t-1). \tag{2.4}$$

Let $f_j(t_1,t_2) \in V_j^2$ is orthogonal projection of function $f$in V_j^2, there exist $\{a_{k,l}^j\}_{k,l\in\mathrm{Z}} \in \ell^2(\mathrm{Z}^2)$ and

$$f_j(t_1,t_2) = \sum_{k,l} a_{k,l}^j \varphi(2^j t_1 - k)\varphi(2^j t_2 - l).$$

The last expression can be put in the following form:

$$\begin{aligned} f_j(t_1,t_2) = &\sum_{k,l} a_{2k,2l}^j \varphi(2^j t_1 - 2k)\varphi(2^j t_2 - 2l) + \\ &+ \sum_{k,l} a_{2k+1,2l}^j \varphi(2^j t_1 - 2k - 1)\varphi(2^j t_2 - 2l) + \\ &+ \sum_{k,l} a_{2k,2l+1}^j \varphi(2^j t_1 - 2k)\varphi(2^j t_2 - 2l - 1) + \\ &+ \sum_{k,l} a_{2k+1,2l+1}^j \varphi(2^j t_1 - 2k - 1)\varphi(2^j t_2 - 2l - 1). \end{aligned}$$

We use (2.4) and after the reordering we obtain

$$\begin{aligned} f_j(t_1,t_2) = &\sum_{k,l} a_{k,l}^{j-1} \varphi(2^{j-1} t_1 - k)\varphi(2^{j-1} t_2 - l) + \\ &+ \sum_{k,l} d_{k,l}^{1,j-1} \varphi(2^{j-1} t_1 - k)\psi(2^{j-1} t_2 - l) + \\ &+ \sum_{k,l} d_{k,l}^{2,j-1} \psi(2^{j-1} t_1 - k)\varphi(2^{j-1} t_2 - l) + \\ &+ \sum_{k,l} d_{k,l}^{3,j-1} \psi(2^{j-1} t_1 - k)\psi(2^{j-1} t_2 - l) \end{aligned}$$

So, we obtained $f_j(t_1,t_2) = f_{j-1}(t_1,t_2) + g_{j-1}^1(t_1,t_2) + g_{j-1}^2(t_1,t_2) + g_{j-1}^3(t_1,t_2)$, where $f_{j-1}(t_1,t_2) \in V_{j-1}^2$, $g_{j-1}^1(t_1,t_2) \in W_{j-1}^{01}$, $g_{j-1}^2(t_1,t_2) \in W_{j-1}^{10}$ and $g_{j-1}^3(t_1,t_2) \in W_{j-1}^{11}$. We mark

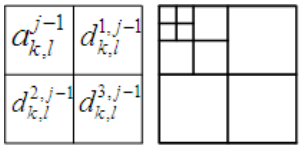
$$a_{k,l}^{j-1} = \frac{1}{2}\left(\frac{a_{2k,2l}^j + a_{2k,2l+1}^j}{2} + \frac{a_{2k+1,2l}^j + a_{2k+1,2l+1}^j}{2}\right),$$

$$d_{k,l}^{1,j-1} = \frac{1}{2}\left(\frac{a_{2k,2l}^j - a_{2k,2l+1}^j}{2} + \frac{a_{2k+1,2l}^j - a_{2k+1,2l+1}^j}{2}\right),$$

$$d_{k,l}^{2,j-1} = \frac{1}{2}\left(\frac{a_{2k,2l}^j + a_{2k,2l+1}^j}{2} - \frac{a_{2k+1,2l}^j + a_{2k+1,2l+1}^j}{2}\right),$$

$$d_{k,l}^{3,j-1} = \frac{1}{2}\left(\frac{a_{2k,2l}^{j} - a_{2k,2l+1}^{j}}{2} - \frac{a_{2k+1,2l}^{j} - a_{2k+1,2l+1}^{j}}{2}\right). \tag{2.5}$$

and these relations give us the two dimensional Haar transform. Coefficients $a_{k,l}^{j-1}$ are the scaling coefficients and the coefficients $d_{k,l}^{1,j-1}$, $d_{k,l}^{2,j-1}$ and $d_{k,l}^{3,j-1}$ are the detail coefficients. If the two dimensional function is considered as a matrix, than after the transformation we have a situation as shown in the illustration on the left, whereas by recursion the transformation can continue until the sub matrix $a_{k,l}^{j-1}$ remains with one element only (illustration on the right).



3. The sequential implementation of the discrete Haar transform (DHT)

Let the two dimensional signal (given function by their coefficients) be $\{a(k,l),\ k = 0,1,...,2^m-1,\ l = 0,1,...,2^n-1,\ m,n \in \mathrm{N}\}$. From (5) and the separability of the two dimensional Haar basis functions, we notice that the two dimensional transform can be realized as a combination of two one dimensional transforms. If the signal is considered as a matrix, than the two dimensional transform will be carried out in two steps:

(1) 1D transform by rows of the matrix and
(2) 1D transform by columns of the transformed matrix (transformed into step i)).

For 1D transform we can use any sequential algorithm ([15], [16], [17], [18], [19]), because all algorithms are similar and have the same complexity which is $O(N)$, where N is the number of discrete points of function (usually it is the discrete signal, while N represents the discrete points of the signal).

A calculation of an element consists of one addition and one multiplication, therefore in the resolution j, $j = 0,1,...,n-1$, $n = \log_2 N$ (we put equal number of rows and columns, N=M) we will have$2N \cdot \frac{N}{2^j}$ for the transformation of rows and $2N \cdot \frac{N}{2^j}$ for the transformation of columns. The complexity of the algorithm of the 2D Haar transform is

$$T(N) = \sum_{j=0}^{n-1} 4N \cdot \frac{N}{2^j} = O(N^2).$$

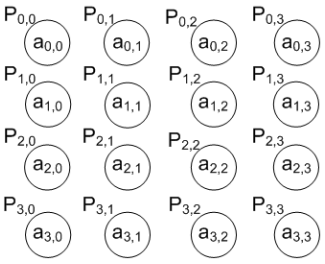
4. Parallel implementation of the Discrete Haar Transform

For the parallel implementation of the Haar Transform we will examine the nothing share architecture, so that each processing element has its processing unit and working memory. Processing elements will communicate between them through the communication channels, respectively through a private computer

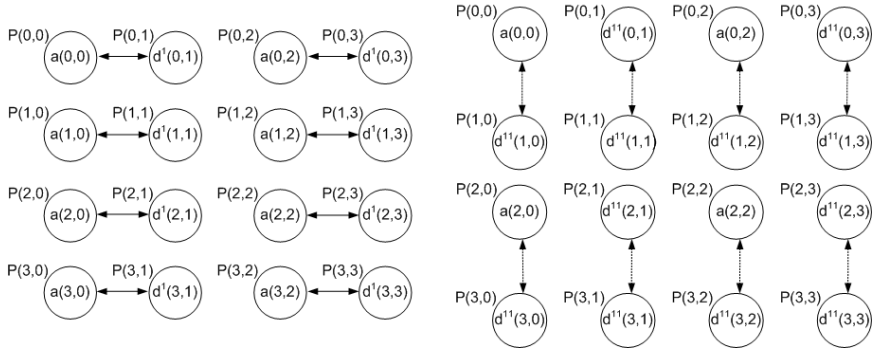
network. The communication will be done through sending and receiving data (messages – data packs) from one to another processing element. From the technical aspect, the implementation will be done for the cluster platform of the work stations, and for the realization of the parallelization we use the MPI paradigm.

Regarding parallel implementation of 2D transform we will discuss two cases depending on the process architecture [23]. The first case is the process organization in mesh architecture and the second in non mesh architecture.

4.1. **Parallel implementation in mesh architecture.** Let p be the number of processes and suppose that $\sqrt{p}$ is a natural number. We organize the processes as a two dimensional matrix with these dimensions $\sqrt{p}x\sqrt{p}$ and we identify the processes through the coordinates. First, we analyze the situation when the number of processes is the same with the overall number of values that have to transform, so, $p = N^2$. We map the values as one value one process, so in the process $P(i, j)$ we map the value $a(i, j)$ and i represents the rows, while j the columns, *i, j=0,1,. . . ,N-1.* One transformation will be carried out through two steps, i.e. first step 1D transformation by rows and the second step 1D transformation by columns. The identification of the processes that need to communicate will be carried out through coordinates (indexes) and j will determine the first step communications whereas i will determine the second step communications. The communication will include processes that differ in one bit only (from left) and other bits on the left have zero. The transformation into one resolution contains one communication and one computation in each step. The computation will be performed by the processes involved in communication. In the first step of the resolution, the communication will go through processes in which the index j differs for one bit (from the left) and other bits on the left are zero. The process P(i,j) will calculate $a(i, j) \leftarrow \frac{a(i,j)+a(i,j+1)}{2}$ if the identifying bit is zero, otherwise it will calculate $a(i, j) \leftarrow \frac{a(i,j)-a(i,j+1)}{2}$. In the second step (transformation by columns), the processes in which i differs for one bit the other bits on the left are zero and the process P(i,j) will calculate $a(i, j) \leftarrow \frac{a(i,j)+a(i+1,j)}{2}$ if the identifying bit is zero, otherwise it will calculate $a(i, j) \leftarrow \frac{a(i,j)-a(i+1,j)}{2}$. The overall transform (transformation in all resolutions) will be done for $\log_2 N$ steps (until only one value remains). The given explanation will be illustrated for the case *p=N=16.* Initially, (after the mapping) we have:

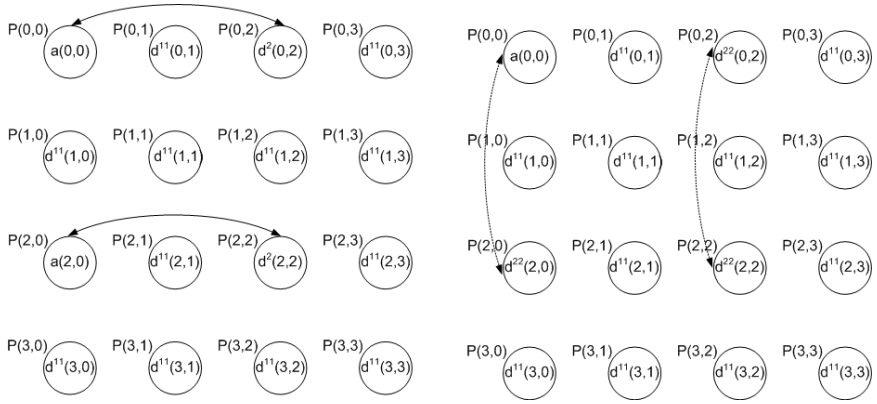


The complete transformation will be done in two steps $k = \log_2 N = 2$. In the first transformation (first resolution) we have



In the first step we communicate the processes $P(i,0) \leftrightarrow P(i,1)$ and $P(i,2) \leftrightarrow P(i,3)$ for $i = 0,1,2,3$ and each process performs one computation. In the second step, the processes that communicate are $P(0,j) \leftrightarrow P(1,j)$ and $P(2,j) \leftrightarrow P(3,j)$ for $j = 0,1,2,3$ and here, each process performs one computation also.

In the second transformation (resolution zero) we have



Now, in the first step the communication will go through $P(i,0) \leftrightarrow P(i,2)$ for $i = 0,2$ and the processes will perform one computation each. In the second step the communication will go through $P(0,j) \leftrightarrow P(2,j)$ for j=0,2 and each process will also perform one computation.

The order of the transformed elements is not the natural transform order. In order to achieve the regular order we must move the rows and columns. The movement will be done same as the movement of the elements in the case of the computation of the 1D transform, e.g. the case of the two dimensional signal $2x2$ the column *0* of the transform will be the column *0* from the result, the column *1* of the transformation will be the column 2 of the result $(1+5 = (101)_2 \rightarrow (110)_2 \rightarrow (10)_2 = 2)$, the column *2* of the transformation will be the column 1 of the result and the column 3 of the transformation will be the column 3 of the result. The rows can be moved in the same way.

As a result, we see that the total transformation will be done for $\log_2 \sqrt{p} = \log_2 N$ steps and each step consists of two sub steps (transformation by rows and

transformation by columns). In each sub step we have a communication (value exchange between two processes) and a calculation consisting of a single addition and a single multiplication. The complexity of the algorithm is

$$T(N) = 2(C_{comm} + 2C_{comp}) \log_2 N$$

$$T(N) = C \cdot \log_2 N = O(\log_2 N)$$

If $N^2 > p$, meaning the number of elements to be transformed are higher than the number of processes. Let's suppose that we have p square of a natural number and $\frac{N}{\sqrt{p}}$ is an even number. We divide the matrix $N\,x\,N$ into sub matrixes $\frac{N}{\sqrt{p}}\ x\ \frac{N}{\sqrt{p}}$ and one of these sub matrixes we map into a process. Initially (after the mapping) each process will perform the 2D sequential transformation for its part, so as a result each process will obtain one element which will be put in the further transformation. The following transformation is carried out by the algorithm in a case when the number of processes is equal to the number of points being transformed. We turn the transformed result into a matrix $N\,x\,N$ and we move the columns and rows (according to the above given explanation). The complexity of this algorithm consists of two parts, the part which complies with the 2D sequential transformation $O\left(\frac{N^2}{p}\right)$ and the part which complies with the transformation where in one process we have one mapped value $O(\log_2 p)$. The complexity is $T(N,p) = O\left(\frac{N^2}{p}\right) + O(\log_2 p)$. Taking into account that $p << N^2$, the complexity of this algorithm can be noted as:

$$T(N,p) = O\left(\frac{N^2}{p}\right).$$

4.2. **Parallel implementation in non mesh architecture.** Further on, we will analyse another possibility of the parallelism of 2D Haar Transformations which will be based on the mapping of rows into processes. Let us suppose that we have the matrix $N\,x\,N$, N exponent of number *2*, of the values needed to be transformed. We will discuss two situations, the situation when the number of processes is the same with the number of rows and the situation when the number of processes is smaller than the number of rows.

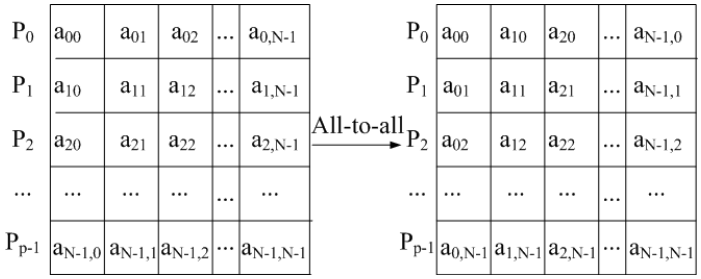
The case p=N, we map a row from the matrix into one process. The transformation in a resolution will be carried out through a number of steps. In the first step each process will perform a 1D transformation for the values of its own row, the matrix transpose of values will be performed in the second step, and in the third step each process will again perform the 1D transformation in its own row. The result of the transformation will be the transposition of the obtained matrix after the third step.

For the 1D transformation in the first and third step we will use the sequential algorithm based on the lifting scheme, which after one transform, the sequence of the transformed values divides into the scaling coefficient part (the first half of the sequence) and the part of detail coefficients (the second half), e.g. let $\{a_i\}_{i=0,1,...,n}$,

$$i = 0, 1, ..., \tfrac{n}{2} - 1,\ a_i = \tfrac{a_{2i} + a_{2i+1}}{2} \text{ and } d_i = a_i - a_{2i+1}$$

and after the transformation we have $\{a_i\}_{i=0,1,\ldots,\frac{n}{2}-1}$,$\{d_i\}_{i=0,1,\ldots,\frac{n}{2}-1}$.

In the second step we use the All-to-all personalize operation (All-to-all MPI routine) for communication [22], which acts in a way that sends the element j of the row i as the element i of the row j,



In the third step each process will again perform a 1D transformation for its own row and in order to attain the transformation we perform an all-to-all operation.

The total transformation will be done for $\log_2 N$ steps, where in each step we will perform the described sub steps. The complexity of this algorithm will have two 1D transformations of the N elements per one resolution and two All-to-all operations. The complexity of one All-to-all operation is $O(N^2)$ when each process has N elements, than the complexity of the algorithm is:

$$T(N,p) = 2(O(N^2) + O(N)) \log_2 N = O(N^2 \log_2 N), N = p.$$

The generalization of the algorithm is the case when the number p of the processes is smaller than the number of the rows N, $p<N$. In one process we map $\frac{N}{p}$ rows and we suppose that $\frac{N}{p}$ is an even number, therefore in one process we map an even number of rows. Each process will perform the 2D transformation within the part of the matrix that it owns (the matrix is taken in dimensions $\frac{N}{p}\, x\, N$) according to the sequential algorithm. For the transformation conducted through rows and columns we will use the 1D sequential algorithm based on the lifting scheme:

$$a_i = \tfrac{a_{2i}+a_{2i+1}}{2} \text{ and } d_i = a_i - a_{2i+1}.$$

The sequential part will finish in $\log_2 \frac{N}{p}$ steps, at the end in each process the sequence of p elements will remain for further transformation. The next transformation follows according to the algorithm where one row is mapped into one process.

After the complete transformation is done the rows will be moved in a way that in the beginning we take the first rows of each process, then the second rows, the third and so we continue with the last rows.

The complexity of this algorithm also consists of two parts, the sequential and the parallel part. In the sequential part each process will calculate $\log_2 \frac{N}{p}$ resolutions of the part $\frac{N}{p}\, x\, N$ of the matrix. Since for one computation we have two additions and one multiplication, for each row at the step k we have $3\frac{N}{2^k}$ operations for the transformation of the row and $3\frac{N}{2^k p}$ for the transformation of the

column. Therefore

$$T_s(N,p) = \frac{N}{p} \cdot 3 \cdot \frac{N}{p} \sum_{k=1}^{\log_2 \frac{N}{p}} \frac{1}{2^k} = 3\frac{N}{p}\left(N + \frac{N}{p} - p - 1\right)$$

and after $p<<N$, we have $T_s(N,p) = O\left(\frac{N^2}{p}\right)$. After $\log_2 \frac{N}{p}$ steps we have a situation when in each process we have one row from p elements and the further transformation will be carried out according to the algorithm when in one process we map one row. The total complexity of the algorithm is:

$$T(N,p) = T_s(N,p) + T_p(p) = O\left(\frac{N^2}{p}\right) + O(p^2 \log_2 p) = O\left(\frac{N^2}{p}\right).$$

Because of $p<<N$, for overall complexity we can take $T(N,p) = O\left(\frac{N^2}{p}\right)$.

5. Experimentation

Experimentation of the algorithms is done in the cluster of the PCs (seeucluster.seeu.edu.mk/ganglia/), composed by eight PCs, set up in Red Hat Enterprise Linux 4 and clustered through Oscar 4.2.

The methodology of experimentation was as follows: because the described algorithm has intensive communications we decided to analyse two separate time measurements, general communication time and general computation time. The total execution time is the sum of those times. Experimentation is done for different number of processes p=4, 16, 64, 256 (the number of processes is taken to be a power of number two because of mesh architecture) and for each number of processes p the dimensions of 2D signals 64x64, 128x128, 256x256, 512x512, 1024x1024, 2048x2048, 4096x4096 (real numbers randomly generated). For each case, the program is executed many times (minimum 10 times) and for the result is taken the maximum time obtained during the executions. The results are shown in the following tables

Table 1. Communication time

Commun.	p \ n x n	64 x64	128 x 128	256 x256	512x512	1024x1024	2048x2048	4096x4096
Block mapping (Mesh)	2x2	0.000399	0.00038	0.000403	0.0004199	0.000463	0.000504	
	4x4	0.0009569	0.0008868	0.0008548	0.0009201	0.001062	0.0010601	0.001072
	8x8	0.0031	0.0031983	0.003385	0.0032289	0.003225	0.003432	0.003443
	16x16	0.0143129	0.0150378	0.0150936	0.0149741	0.015763	0.0153411	0.0153754
Rows mapping (No Mesh)	4	0.0006762	0.0007611	0.0007201	0.0007822	0.000792	0.0007931	
	16	0.0083382	0.0083733	0.008439	0.0083822	0.008488	0.0086689	0.0085839
	64	0.236737	0.233936	0.235308	0.235744	0.23187	0.23433	0.224406

Table 2. Computation time

Comput.	p \ n x n	64 x64	128 x 128	256 x256	512x512	1024x1024	2048x2048	4096x4096
One process		0.0002141	0.000873	0.0035961	0.0148921	0.0589749	0.24613	0.980677
Block mapping (Mesh)	2x2	0.0001479	0.0005229	0.001358	0.0063251	0.0850721	0.3747	
	4x4	0.0001278	0.0002218	0.000594	0.0016781	0.0077579	0.0806887	0.37683
	8x8	0.0001771	0.000404	0.000752	0.0029749	0.008339	0.042731	0.500538
	16x16	0.0001852	0.0006619	0.001209	0.0024991	0.0115592	0.0293169	0.115725
Rows mapping (No Mesh)	4	7.203E-05	0.0002449	0.00092	0.005003	0.045833	0.323172	
	16	9.289E-05	0.0001127	0.0003311	0.0013476	0.0102358	0.117933	0.385616
	64	0.0001234	0.0072529	0.0099601	0.0118338	0.0223034	0.038637	0.783476

6. Conclusion

The Haar transformation, and more specifically, the two dimensional Haar transformations are important transformations. According to their computing nature, the Haar transformations are not transformations of many calculations, but in the case of the two dimensional transformations the number of the values needed to be transformed increases and as a result the number of operations increases as well. In general, the complexity of computing the 2D Haar transformations is $O(N^2)$. We have analysed the possibility of parallelism of the Haar transformations in a platform where each processing element has its own local memory and we discussed two cases, the first case when in one process we could map only by one value and the case when in one process we could map by more values. And we obtained the results $O(\log_2 N)$ during the mesh architecture and the mapping of one value in one process, $O(p^2 \log_2 p)$ in the case of mapping one row in one process, $O\left(\frac{N^2}{p}\right)$ in cases of mapping more rows in one process, as well as the case when one sub matrix is mapped into one process. In all assessments of the complexity it is notable that the dominant factor is the part of the complexity that occurs from the local transformations (which are carried out by sequential algorithms), whereas the communication between processes, almost in all cases, occurs in some communications with the help of a one value message between the processes. Therefore, the common conclusion is that the complexity depends from the number of processes and with the increase of the number of processes (meaning the increase of the physical number of the processing elements) the time of the completion of transformations decreases.

Regarding the experimentations we can conclude as follows:

(1) In both cases (mesh implementation and row implementation) by increasing the number of processes we have increase of communications time and decrease of computations time. This is because by the increase of number of processes, the data have to distribute to more processes. This affects the processes to perform more communications. In terms of calculations, a small quantity of data into process reduces the number of calculations.
(2) In both cases increasing the size (dimensions) of two dimensional signal (i.e. increase the number of elements for transform) affects to increase of the communications time and computations time. So, if the numbers of processes is unchanged and we increase the amount of data than an increase of communications time is coming as a result of increase of the amount of data to be transferred.

From both cases we see that the parallel implementations suffer from communication between processes and intensive communications are a key factor. Regarding on the general results (computations + communications) implementation in mesh architecture shows something better performance.

The general conclusion is that the overall performances of the parallel implementation depend on the communications performances between processes.

References

[1] D. L. Donoho, M. E. Raimondo, "A Fast Wavelet Algorithm for Image Deblurring", Australian and country-regionplace New Zealand Industrial and Applied Mathematics Journal, ANZIAM (E) 2005, pp. C29-C46

[2] Oivind Ryan, "Applications of the wavelet transform in image processing", Sponsored by the Norwegian Research Council, project nr. 160130/V30, 2004

[3] P. Raviraj, M. Y. Sanavullah, "The Modified 2D-Haar Wavelet Transformation in Image Compression", Middle-East Journal of Scientific Research 2 (2): 73-78, 2007, ISSN 1990-9233

[4] Hanghang Tong, Mingjing Li, Hongjian Zhang, Changshui Zhang, "Blur detection for digital images using wavelet transform", In In Proceedings of IEEE International Conference on Multimedia & Expo (2004), pp. 17-20.

[5] Leila Fallah Araghi, Mohammad Reza Arvan, "An Implementation Image Edge and Feature Detection Using Neural Network", Proceedings of the International MultiConference of Engineers and Computer Scientists 2009 Vol I, IMECS 2009, March 18 - 20, 2009, Hong Kong

[6] Kamrul Hasan Talukder and Koichi Harada, "Haar Wavelet Based Approach for Image Compression and Quality Assessment of Compressed Image", IAENG International Journal of Applied Mathematics, 36:1, IJAM_36_1_9, Advance online publication: 1 February 2007

[7] Anuj Bhardwaj, Rashid Ali, " Image Compression Using Modified Fast Haar Wavelet Transform", World Applied Sciences Journal 7 (5):647-653, 2009, ISSN 1818-4952

[8] Masaaki Ohkita, Yasuhiro Kobayashi and Michio Inoue, "Application of the Haar functions to solution of differential equations", Mathematics and Computers in Simulation, Volume 25, Issue 1, February 1983, Pages 31-38

[9] Phang Chang, Phang Piau, "Haar Wavelet Matrices Designation in Numerical Solution of Ordinary Differential Equations", IAENG International Journal of Applied Mathematics, 38:3, IJAM_38_3_11, Advance online publication: 21 August 2008

[10] F. Soesianto, Kaes Ismael Ibraheem, "On Wavelet-Based Algorithm for Solving Parabolic Differencial Equations", Proceedings Komputer dan system Intelejen (KOMMIT 2002), Auditorium Universitas Gunadarma, Jakarta, 2002

[11] I. Daubechies, "Ten Lectures on Wavelets", CBMS-NSF Conference on Wavelets and Applications 1990, Published by country-regionplaceSIAM: Society for Industrial and Applied Mathematics, 1992

[12] B. Zhechev, "Null Spaces and Haar Transform", CompSysTech'07, ©2007 ACM, 2007

[13] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation", IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. II, No. 7. July 1989

[14] placeI. Laszlo, F. Shipp, S. P. Kozaitis, "Construction of Wavelets and Applications", Journal of Universal Computer Science, vol. 12, no. 9 (2006), pp. 1278-1291

[15] J. A. R. Macias, A. G. Exposito, "Efficient Computation of the Running Discrete Haar Transform", IEEE Transaction on Power Delivery, Vol. 21, No. 1, January 2006, pp. 504-505

[16] P. Chang, P. Piau, "Modified Fast and Exact Algorithm for Fast Haar Transform", Proceedings of World Academy of Science, Engineering And Technology, Vol. 26 December 2007,pp 509-512

[17] S. M. Salih, N. Uzunoglu, L. A. Kadhim, L. A. E. Anzu, "A proposed model for MC-CDMA based in-place wavelet transform, Proceedings of the 3rd international conference on Mobile multimedia communications 2007, Nafpaktos, Greece", ACM International Conference Proceeding Series; Vol. 329

[18] P.S.Jamkhandi, A.Mukherjee, K.Mukherjee and R.Franceschini, "Novel Hardware-Software Architecture for the Recursive Merge Filtering Algorithm", Eighth ACM/SIGDA International Symposium on Field Programmable Gate Arrays, 2000

[19] Jaumin Ajdari, "Parallel Hypercube Implementation of Haar Transform," csie, vol. 3, pp.321-324, 2009 WRI World Congress on Computer Science and Information Engineering, 2009

[20] Annika Schiller, Godehard Sutmann, Liang Yang "Parallel 2d-Wavelet Transform on the Cell/B.E. for Fast Calculation of Coulomb Potentials", CBSB 08 (From Computational Biophysics to Systems Biology), 19–21 May 2008, Julich, Germany

[21] Piyush Jamkhandi, Amar Mukherjee, Kunal Mukherjee, Robert Franceschini, "Parallel Hardware-Software Architecture for computation of Discrete Wavelet Transform using the Recursive Merge Filtering algorithm", Proceedings of the 15 IPDPS 2000 Workshops on Parallel and Distributed Processing, pp: 250 – 256, ISBN:3-540-67442-X

[22] Luiz Angelo Steffenel, Maxime Martinasso and Denis Trystram, "Assessing Contention Effects on MPI_Alltoall Communications", Proceedings of the 2nd International Conference on Grid and Pervasive Computing - GPC 2007 - (2007) - http://hal.archives-ouvertes.fr/hal-00136204/en/

[23] Jaumin Ajdari, "1D and 2D Haar Transform and Parallel Implementation in Nothing Share Architecture", International Journal of Pure and Applied Mathematics IJPAM, ISSN 1311-8080, Volume 64, No. 4, pp. 491 - 516, December 2010, Sofia, Bulgaria

Information and Communication Technology - Research and Training Centre, South-East European University, Tetovo, Macedonia

E-mail address: j.ajdari@seeu.edu.mk