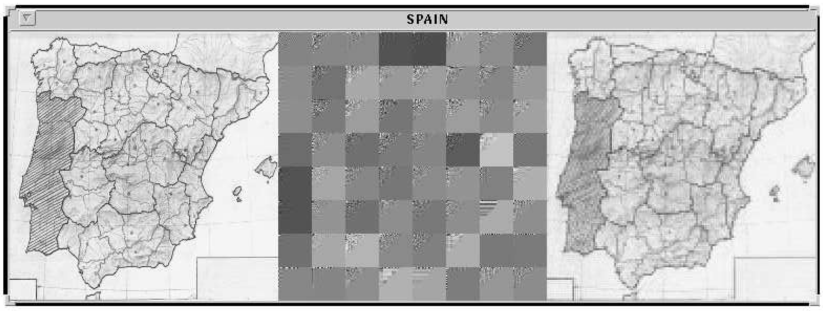
Image Compression Using the Hadamard Transform on Transputer Arrys

V. G. Ruiz, I. García
Depto de Arquitectura de Computadores y Electrónica
Universidad de Almería

31 August 1994

1 Introduction to the image Coding

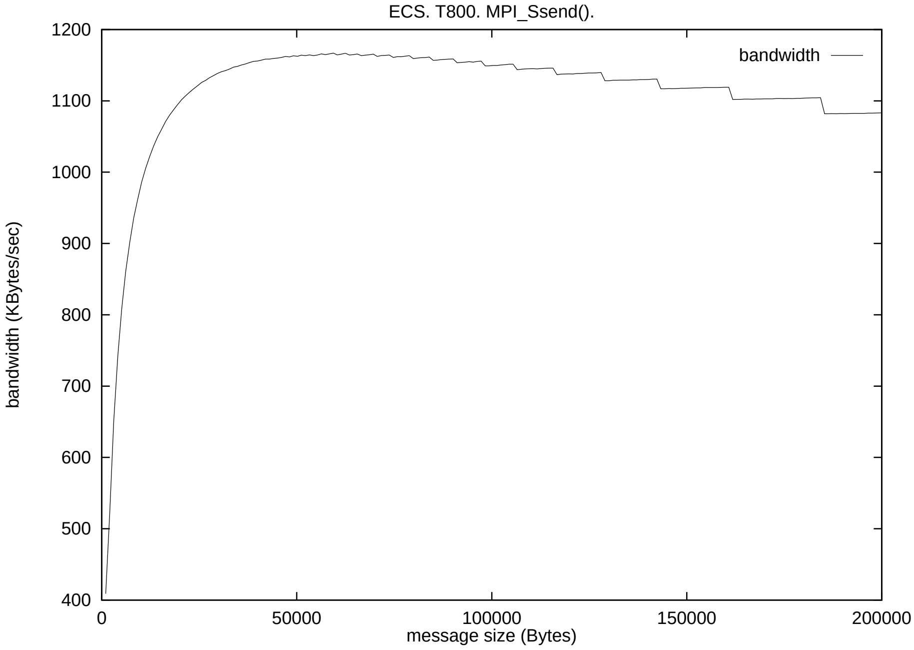
In this project we propose a parallel version of a lossy image compression algorithm [1] based in the fast Hadamard transform [1,2]. This is a reversible, linear transform (such the Fourier transform) and it allows us to map a image into a set of transform coefficients, which are then quantized and coded [2]. A significant number of transform coefficients have small magnitudes and can be coarsely quantized (or discarded entirely) with little image distorsion. For example, in the figure 1(a) we can see a 256x256 8 bits PixMap, in 1(b) the 32x32 Hadamard spectrum transformed coefficients representing the image. Due to redundancy, 80% of the Hadamard coefficients can be erased, and the reconstructed image (SNR = 41.29 dB) is shown in figure 1(c).



2 Transputer Models

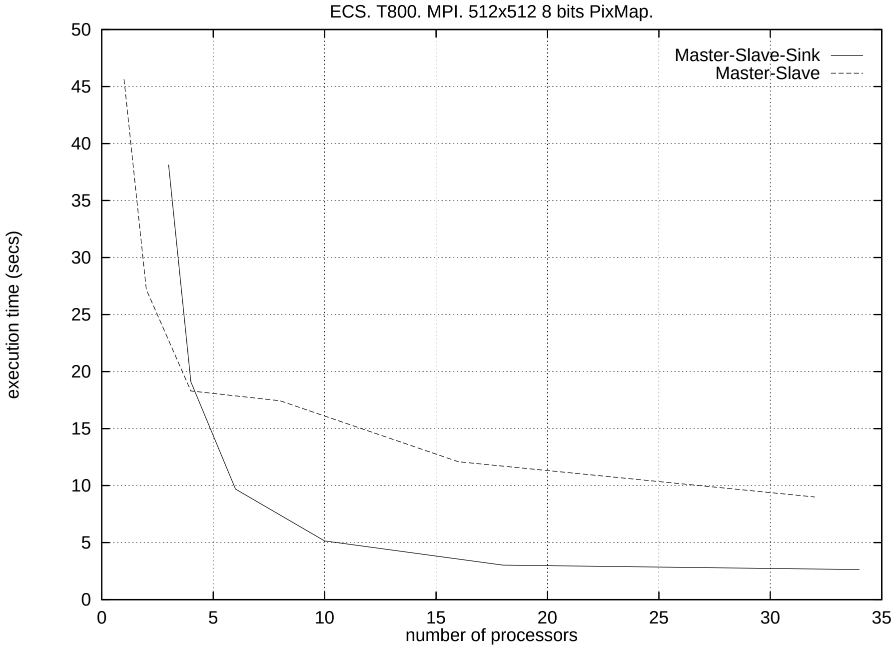
We have tested the parallel operation of this coding algorithm on two different T800 configurations [3]. In the first one, we have used a Master-Slave (MS) model and in the second one, a Master-Slave-Sink model (MSS).

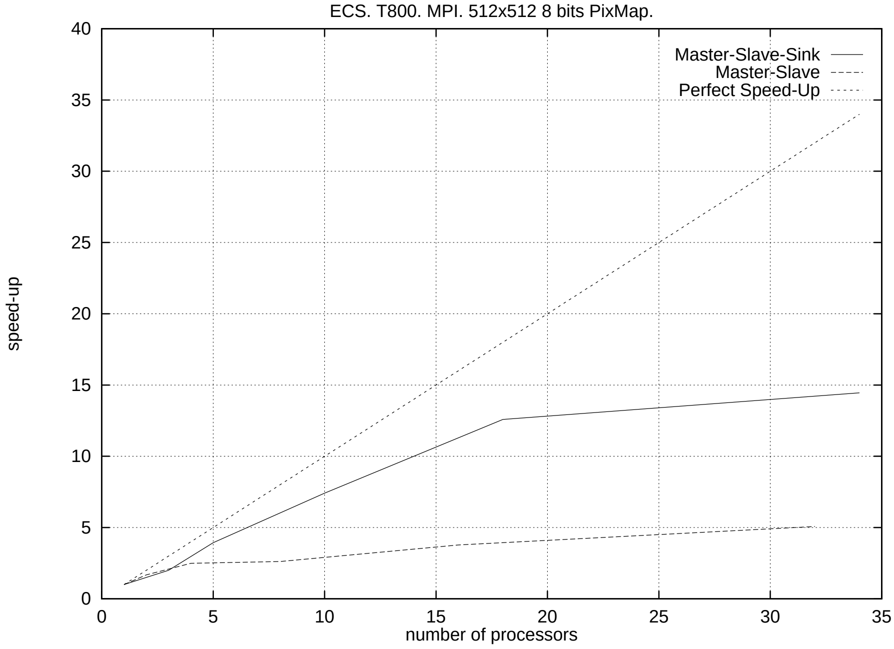
Every program is written in a single C program using the new MPI standard [4] (running over CHIMP [5]). The machine used is ECS with the transputer T800. Figure 3 shown the message bandwidth required for interfacing with the transputer array. In our application the image size (512 x 512 pixels) and the number of processors used fix the message size near to 32 KBytes.



3 Results

The main problem found is the bottle neck in the processor which read and write the IO device, because the computation time of the algorithm is the same order than the communication time, independently of the image size (we work with constant subimage of 8x8 pixels). For this reason, if we duplicate this processor (such in the MSS model), we can double the performance. In the figure 4 we show the absolute execution time for processing one image frame. In the figure 5 the speed-up achieved in the two models is plotted for different numbers of processing elements in the transputer surface. All the timings are taken without read and write the image in the hard disc,because the IO transfer rate is very low in our particular implementation.


ECS. T800. MPI. 512x512 8 bits PixMap.
execution time (secs)
number of processors
Master-Slave-Sink
Master-Slave
0
5
10
15
20
25
30
35
40
45
50


ECS. T800. MPI. 512x512 8 bits PixMap.
speed-up
number of processors
Master-Slave-Sink
Master-Slave
Perfect Speed-Up
0
5
10
15
20
25
30
35
40

4 Discussion

If we compare against other work [6], we see that the performance reached in out application is less. We can explain this reduction in two ways:

1. The CCITT recommendation H.261 algorithm is based in the DCT. This transform works with real coefficients while the Hadamard transform operates with integer coefficients. For example, Inmos claim peak performance of 4.3 MFLOPS and 30 MIPS for the IMS T800. For this reason, the computation time is very small in our application, and the processors spend the most part of the time in the interprocessor communications. This factor is very important to calculate the speed-up.

2. Elliott et al. use the Tiny, a message passing system which supports such a model of communications and was developed at the University of Edinburgh on the ECS. We use MPI, running on the CHIMP. The MPI is a high level and hardware independent tool. In occam we can build the physical configuration of processor to minimize the communication time, while in our application this is not possible. We work with a processors farm and there is no restrictions like the number of physical links, etc. We think that these characteristics of the MPI thus affect to our performance.

5 Conclusion

We see in the figure 5 that 16 processors achieve a speed-up of 12 times which indicate this is a good size of transputer array to use for the image coding problem. Larger arrays only achieve minimal reduction in execution time and hence offer much lower speed-up.

Larger arrays could only be used for the image coding problem if the interprocessor communications is improved beyond the T800 capabilities. Interprocessor communications is the restricting factor in the application considered.

6 References

[1] Rafael C. Gonzalez and Paul Wintz.
Digital Image Processing.
Addison-Wesley Publising Company, 1988.

[2] R. J. Clarke.
transform Coding of Images.
Academic Press, 1985.

[3] INMOS LTD.
Transputer Reference Manual.
Prentice Hall, UK, 1988.

[4] MPI. A Standard Message Passing.
Message Passing Interface Forum.
Document for a Standard Message-Passing
Interface.
Technical Report CS-93-214, University of Tennessee,
November 1993.

[5] R. Alasdair A. Bruce, James G. Mills, and A. Gordon Smith.
CHIMP version 2.0 Interface.
Technical Report EPCC-KTP-CHIMP-V2-IFACE,
Edinburgh Parallel Computing Centre,
University of Edinburgh, February 1994.

[6] J.A, Elliott, C. Cubiss, P.M. Grant, J.T.E. McDonnell.
Real-time simulation of videophone image coding algorithms on
reconfigurable multicomputers.
IEE Proceedings-E, Vol. 139, No. 3, MAY 1992.