ELSEVIER

Future Generation Computer Systems 18 (2001) 97–106

FGCS FUTURE GENERATION COMPUTER SYSTEMS

www.elsevier.com/locate/future

Parallel adaptive wavelet analysis

R. Kutil*, A. Uhl

RIST++ & Department of Scientific Computing, University of Salzburg, Jakob Haringer Str. 2, 5020 Salzburg, Austria

Abstract

Adapted wavelet analysis in the sense of wavelet packet algorithms is a highly relevant procedure in different types of applications, like, e.g. data compression, feature extraction, classification problems, data analysis, numerical mathematics, etc. Given a large or high-dimensional data set the computational demand is too high for interactive or "nearly-interactive" processing. Therefore, parallel processing is one of the possibilities to accelerate the processing speed. In this case, special attention has to be paid towards handling of the large amount of data in addition to the proper organization of the computations. We investigate different data decomposition approaches, border handling techniques and programming paradigms. The memory consuming decomposition into a given arbitrary basis after adaptive basis choice is resolved by a localized decomposition strategy. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Wavelet packets; MIMD algorithms; Best basis algorithm

1. Introduction

Wavelet packets [24] represent a generalization of the method of multiresolution decomposition and comprise the entire family of subband coded (tree) decompositions. Whereas in the wavelet case the decomposition is applied recursively to the coarse scale approximation subband only (leading to the well known (pyramidal) wavelet decomposition tree), in the wavelet packet decomposition the recursive procedure is applied to all subbands (i.e. coarse scale approximation and detailed signals), which leads to a complete wavelet packet tree (i.e. binary tree and quadtree in the 1D and 2D case, respectively) and more flexibility in frequency resolution. See Fig. 1 for the 1D case.

A given data set may be represented by different sets of subbands. There are several possibilities (see below) regarding the determination of subbands suitable for an application. The meaning of *suitable* depends on the type of application, e.g. signal/image compression [26], feature extraction [18], classification algorithms [13,14], telecommunication applications [15], numerical mathematics [16], and many more. The wavelet packet "best basis algorithm" [5] performs an adaptive optimization of the frequency resolution of a complete wavelet packet decomposition tree by selecting the most suitable frequency subbands with respect to an additive cost function. The same algorithm employed with non-additive cost function is denoted "near-best basis algorithm" [20], if the subband structure is restricted to uniform time–frequency resolution, the corresponding algorithm is denoted "best level selection" [4].

In the context of image compression, a more advanced technique is to use a framework that includes both rate and distortion, where the best basis subtree

* Corresponding author. Tel.: +43-662-8044-6303.
E-mail addresses: rkutil@cosy.sbg.ac.at (R. Kutil), uhl@cosy.sbg.ac.at (A. Uhl).

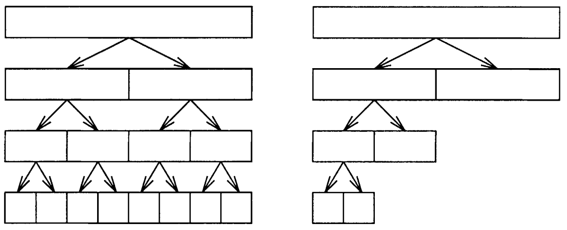


Fig. 1. 1D wavelet packet and pyramidal wavelet decomposition tree.

which minimizes the global distortion for a given coding budget is searched [17]. Other methods use fixed bases of subbands for similar signals (e.g. fingerprints [11]) or search for good representations with genetic optimization methods [1,3].

Whereas, an enormous amount of work has been already done concerning the development and thorough analysis of parallel algorithms for the fast wavelet transform (FWT), only few papers have been devoted to parallel wavelet packet decomposition and its specific features and demands (e.g. approaches for performing the best basis algorithm and the irregular decomposition into such a basis on parallel MIMD [8,10,21,22] and SIMD [2,7] architectures, application of parallel wavelet packet decomposition in numerics [6,16]).

In this work, we discuss and investigate issues related specifically to a parallelization of the adaptive wavelet packet decomposition (i.e. the best basis algorithm) and corresponding programming paradigms. In particular, we treat the questions of data decomposition, boundary handling, and memory consumption. The algorithms are discussed in the context of 3D data as processed for 3D video coding [12] or the analysis of volumetric medical data [23].

2. Sequential algorithm

The FWT uses a quadrature mirror filter (QMF) pair to decompose a signal into a high and a low frequency subband by convolution and downsampling by two. This is repeated for the low frequency subband only in pyramidal decomposition and for all subbands in wavelet packet decomposition. In the 3D case, the decomposition of a subband produces eight new subbands by convolution along each of the three dimensions (see Fig. 2), leading to a full decomposition tree (octtree) which produces 8^n subbands at a decomposition depth of n.

We want to estimate how suitable specific parts of the decomposition tree are with respect to a target application. Within the best basis algorithm, this is done by evaluating a cost function. A classical example for compression purposes is the entropy $H(X) = -\sum_j p_j \log p_j$, where $p_j = |x_j|^2/\|X\|^2$ and $X = (x_0, x_1, x_2, \dots)$ are the coefficients in the subband. An additive analogue is $\lambda(X) = -\sum_j |x_j|^2 \log |x_j|^2$ by the use of which $\sum_i \lambda(X_i)$ can be compared efficiently with $\lambda(X)$ if X is decomposed into $X_1, \dots, X_8$.

Classically, in order to determine the best basis (i.e. the optimal subband structure with respect to the cost function in use), one has to perform a complete decomposition and to compute for each subband on each decomposition level its *cost function value*. Subsequently, the cost of each subband is compared with the cost of its *children subbands* traversing the tree in a bottom-up manner in order to determine the subbands with optimal cost function values. This is a recursive procedure, leading to a decomposition tree, that stops branching where the cost cannot be further minimized. After that, the subbands that minimize the *overall cost* have to be selected by either accessing subband data kept in memory or performing a second

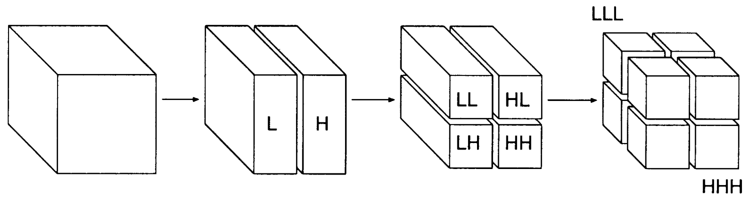


Fig. 2. 3D filtering with a QMF pair.

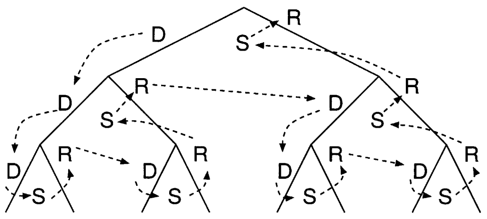


Fig. 3. Best basis decomposition without a second run and minimized memory demand. The letters used in the figure imply the following: D — Backup subband and Decompose; S — Sum the cost of subbands; R — Restore subband if decomposition does not reduce the cost.

decomposition run according to the best basis tree. Note that the first strategy has high memory requirements and poses corresponding difficulties on distributed memory systems whereas the second strategy trades off computational complexity for memory demand and results in a potentially highly irregular second decomposition run which is hard to parallelize especially on massively parallel systems [22].

Therefore, we present a strategy to avoid the second run by backing up and restoring subbands and by traversing the decomposition tree in depth first order (see Fig. 3). Pseudo code 1 (Fig. 4) implements this strategy as a recursive procedure. Within this procedure, first the cost function is evaluated for the current subband. Subsequently, the subband is backed up before it is transformed into eight new subbands by use of the QMF pair. After that, each new subband is treated the same way and we receive a cost function value for each subband which is the lowest cost that can be reached by the right choice of the decomposition subtree. These values are summed up and compared to the value determined in the beginning of the procedure. If this sum is not less, then the subband is restored. The lower value is returned.

3. Parallel algorithm

3.1. Message passing approach

For a message passing parallelization, computations need to be distributed in a coarse grained way. The data are simply split into several parts along one or more axes and distributed among the processor elements (PE). But at a certain decomposition depth, subbands get too small to be processed efficiently in parallel since single subbands are spread across all PE. This stage is of course dependent on the number of PE employed and is denoted “redistribution level” [8,22]. At this stage, data has to be redistributed so that entire subbands are located on single PE which enables to perform the remaining computations without any additional communication. Therefore, the concept of the sequential algorithm cannot be completely adapted to the parallel algorithm.

We choose a mixed approach: as visualized in Fig. 5, in a first phase data are classically distributed (e.g. in equally sized blocks) and completely decomposed up to the redistribution level. Subsequently, data is redistributed (see redistribution level in Fig. 5) and decomposed using the sequential algorithm from Section 2 (including the best basis selection for these

```
SequDecompose (Subband S) return Cost :=
    C = Cost (S)
    -- end of recursion:
    if DecompositionLevel (S) = MaxLevel then return C
    Backup S
    Filter S to get Subbands S[1] .. S[8]
    -- decompose the subbands recursively:
    D = 0
    for i = 1 .. 8
        D = D + SequDecompose (S[i])
    -- decide whether the best basis tree branches:
    if   C < D
    then Restore S; return C
    else return D
```

Fig. 4. Pseudo code 1: sequential algorithm.

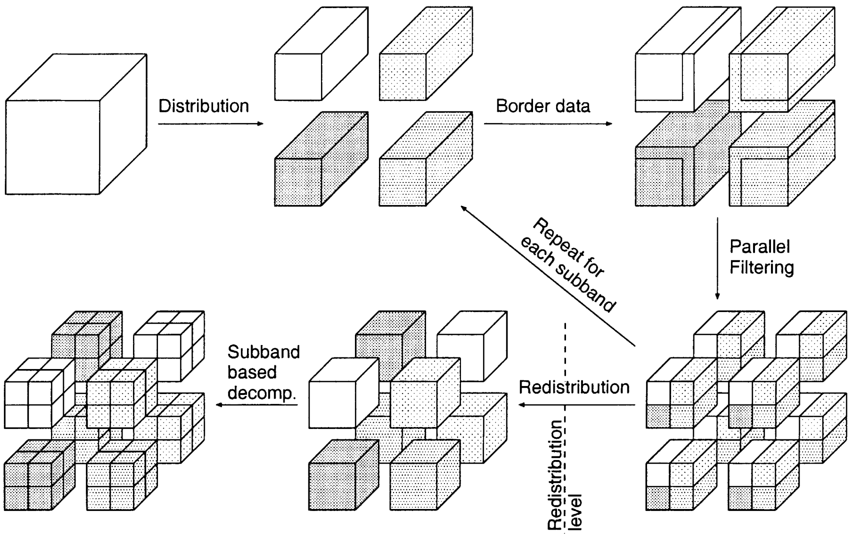


Fig. 5. Parallel decomposition (without best basis selection) — different colours symbolize different PE.

subbands). In the third phase (where the final subband structure is generated), we have to cope with the possibility, that a subband at a level below the redistribution level is required, so these data have to be regenerated somehow, as it was not backed up for memory reasons. We choose the approach to recompose this data from higher level subbands, because subbands are more likely to be decomposed at these levels.

Pseudo code 2 (Fig. 6) shows how this can be implemented. In a first phase, the data is distributed among the node PE and decomposed completely up to the redistribution level without considering the best basis tree. In the second phase, the data is redistributed and subbands can be decomposed using the sequential algorithm (SequDecompose), because entire subbands are located on single PE. This includes the decomposition into the best basis (sub) tree for these subbands. After that (third phase), the cost function values and subtree information are exchanged and the best basis tree is finally determined by "connecting" the subtrees together. If the best basis tree contains a branch, that does not grow up to the redistribution level, the

```
-- Phase 1: split decomposition
for t = 1 .. NumberOfNodes
    Send Part of data to Node t
for l = 1 .. RedistributionLevel
    for all subbands S at level l
        Exchange border data
        Filter S into 8 new subbands
-- Phase 2: subband based decomposition
Redistribute subbands to responsible node
for all subbands S at RedistributionLevel
    SequDecompose (S)
-- Phase 3: partial reverse second run
Exchange subtree information
Determine part of best basis tree below RedistributionLevel
Recompose subbands below RedistributionLevel if necessary
```

Fig. 6. Pseudo code 2: parallel algorithm.

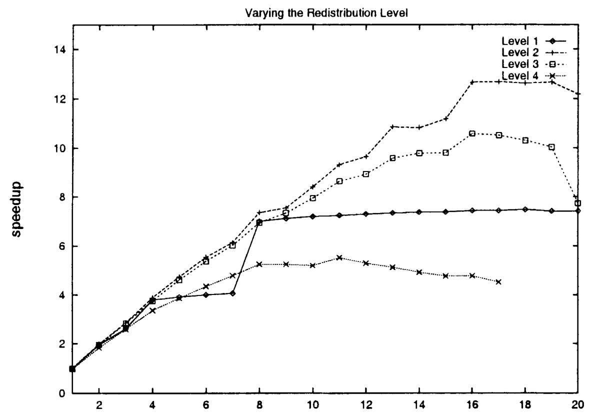
corresponding subband has to be recomposed, because it was not backed up in the first phase. This is a process similar to the first phase and does not lead to a load balancing problem, because single subbands are distributed among all PE below the redistribution level.

The choice of the location of the redistribution level is obviously very important with respect to performance — the fewer decomposition levels are performed before redistribution, the less boundary treatment operations are necessary (see below). However, if redistribution is performed too early, a severe load balancing problem may arise (especially if low dimensional data are processed on massively parallel systems). Note again that in the s-dimensional case, we result in 2^{sn} subbands at decomposition level n. In order to guarantee a nicely distributed load, the number of available subbands should be significantly larger as compared to the number of PE. Consequently, there is a trade-off between optimally balanced load and performing the redistribution as early as possible.

The memory requirement of the above algorithm for a node PE is therefore $((t/\#PE) + (2^r - 1)(f - 2))xy$ floating point numbers for the first phase (t,x and y are the data sizes along the corresponding axes, f the filter length and r the redistribution level) plus $txy8^{-r}(1+b)$ for the second phase (where b stands for the memory needed to back up the subbands before decomposition) which should be rather small for $r > 1$. In this case, $b = 1 + \frac{1}{8} + \frac{1}{8^2} + \cdots \approx \frac{8}{7}$. If we performed a second run, backing up the subbands would not be necessary, therefore $b = 0$. The conventional algorithm (storing all subbands) would imply $b = d - r$, where d is the maximum decomposition depth.

Boundary problem. A classical problem in parallel wavelet and wavelet packet algorithms is the question of how to handle the exchange of the necessary border data. Recall that the need for border data located on adjacent PE is caused by the nature of the QMF process which involves several neighbouring data points in order to compute a single transform coefficient. In the literature, two approaches for the boundary problems have been discussed and compared [19,25]. During the *data swapping* method, each PE calculates only its own data and exchanges these results with the appropriate neighbour PE in order to get the necessary border data for the next decomposition level. Employing *redundant data calculation* each PE computes also necessary redundant border data in order to avoid additional communication to obtain this data. It was shown that the decision, which method to apply, is very important for certain wavelet algorithms (see, e.g. [9] for the á trous algorithm and Fig. 9 for 3D pyramidal fast wavelet decomposition). Therefore, we investigate its influence on the wavelet packet decomposition.

Multidimensional data splitting. There are several possibilities to distribute higher dimensional data in



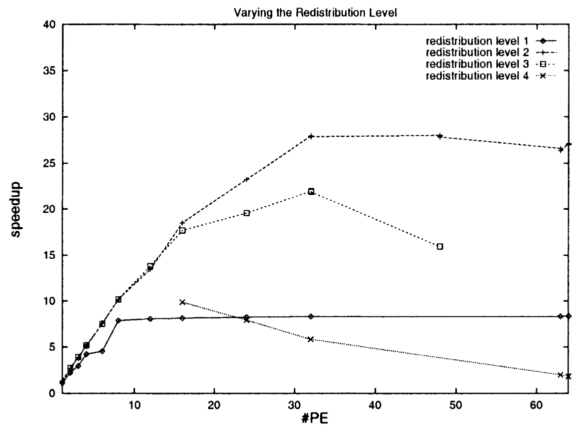


Fig. 7. Speedup comparison for different redistribution levels. (a) SGI Powerchallenge, (b) Cray T3E.

the start phase, because they can be split along several axes. It can be shown that by splitting the data along several coordinate axes it is possible to reduce the amount of border data that has to be exchanged, but it increases the communication complexity (since the number of adjacent PE is higher).

3.2. Shared memory approach

Data parallel programming on a shared memory architecture is easily achieved by transforming a sequential algorithm into a parallel one by simply identifying areas which are suitable to be run in parallel, i.e. in which no data dependencies exist and different iterations access different data. Subsequently, local and shared variables need to be declared and parallel compiler directives are inserted.

Here we choose parallelization of a decomposition step along one axis of the 3D data. Because one has to take care, that parallel regions are not too short with regard to execution time, we have to change this strategy at a certain decomposition depth. This leads to a simulation of the message passing algorithm (data redistribution).

4. Experimental results

We conduct experiments on an SGI Powerchallenge GR (at RIST++, Salzburg University) with 20 MIPS R10000 processors and 2.5 GB memory and on a Cray T3E with 512 compute nodes type 1200. The size of the video data is 128×128 pixels in the spatial domain, combined to a 3D data block consisting of 512 frames. QMFs with 8 coefficients are used. The PVM version employed on the SGI Powerchallenge is a special shared memory variant for SGI systems. On the Cray T3E MPI is used.

Fig. 7 shows a speedup comparison for subband based redistribution. One can see clearly that a too low redistribution level limits the number of subbands that are redistributed, leaving some PE without work (load balancing problem). A too high redistribution level increases the communication complexity of the redistribution. This shows that redistribution based on subbands is necessary. Note that, if best basis computations are omitted (full decomposition), the speedup behaviour is the same, because the computational demand for each subband is increased in the same way over all decomposition levels.

Within Fig. 8, several runs of a decomposition with different choices of the redistribution level are visualized with a monitoring tool. The lowest horizontal line symbolizes the progression of the host PE in time, the other represent the node PE. The bold black parts of these time lines represent calculation phases. Crossed lines (almost vertical) symbolize the transmission of data from one PE to the other. Time is measured in seconds.

Figs. 9 and 10 show that whereas for pyramidal wavelet decomposition the choice of the border handling strategy does affect the performance of the algorithm significantly, it turns out to be of almost no relevance for wavelet packet decomposition ("n steps with redundant data" means that the first n steps of the decomposition do not have to exchange border data). The reason is that in the wavelet case the entire processing time is dominated by the first decomposition level (where the choice of border handling strategy is crucial) whereas in the wavelet packet case all decomposition levels require roughly the same computational amount. The computations above the redistribution level are not affected by any border problem at all which explains the different behaviour. For the same reason, the use of different techniques for multidimensional data splitting shows little influence on the speedup behaviour of the algorithm.

Finally, we compare the message passing approach with a version of shared memory programming (see Fig. 11). A straightforward approach (denoted "PowerC without redistribution" in the plot) where simply the loop corresponding to the dimension of the data block is parallelized does not even show any speedup across the entire range of processors. The reason for this phenomenon is that the high number of parallel regions (corresponding to the number of subbands) cause a significant parallelization overhead. Only by simulating the message passing approach to some extent (which is very expensive in terms of implementation effort and therefore the advantage of shared memory programming is lost) we reach some speedup but still significantly below the message passing implementation. This effect is caused by the fact that the computations are performed on a large amount of data, causing higher access times.

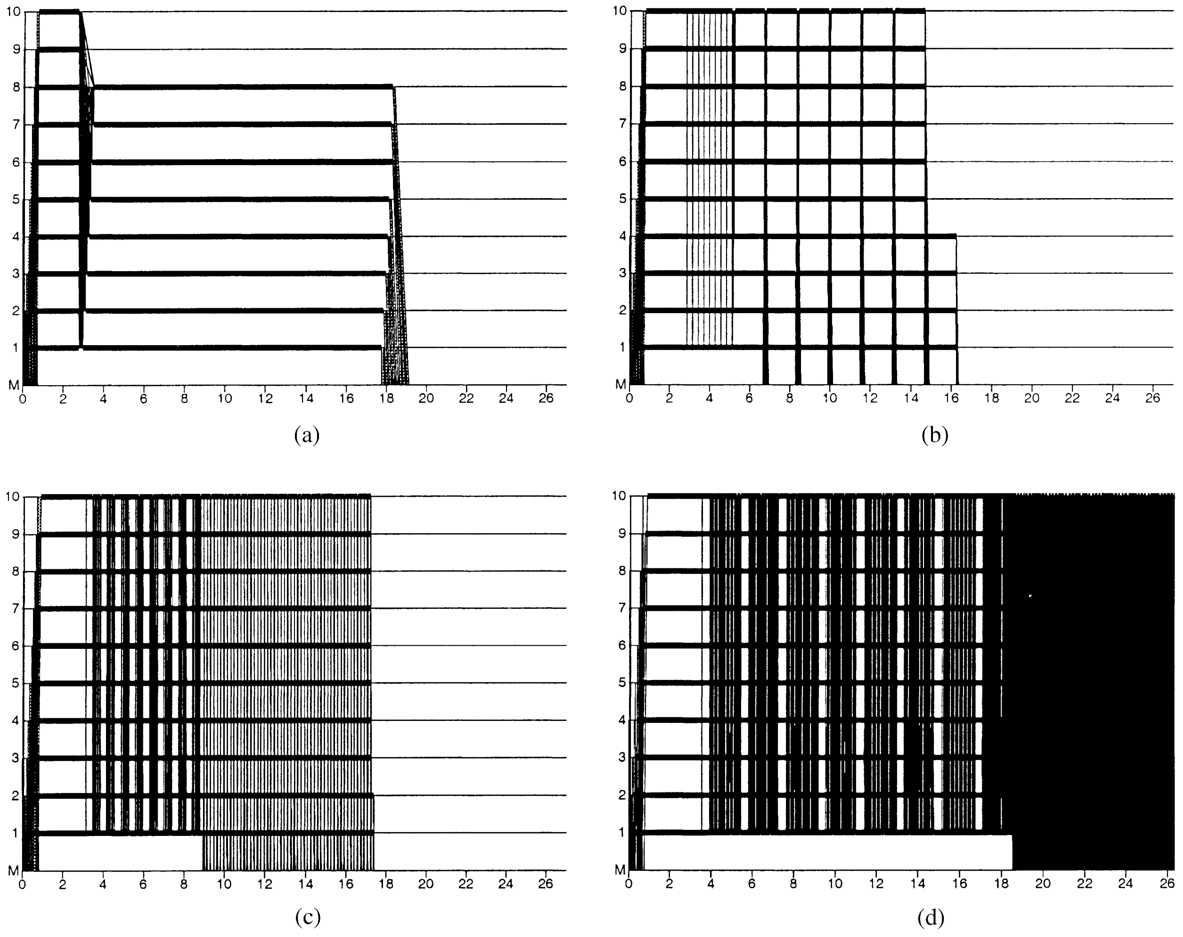


Fig. 8. Effect of subband based redistribution on execution. (a) Level 1, (b) level 2, (c) level 3 and (d) level 4.

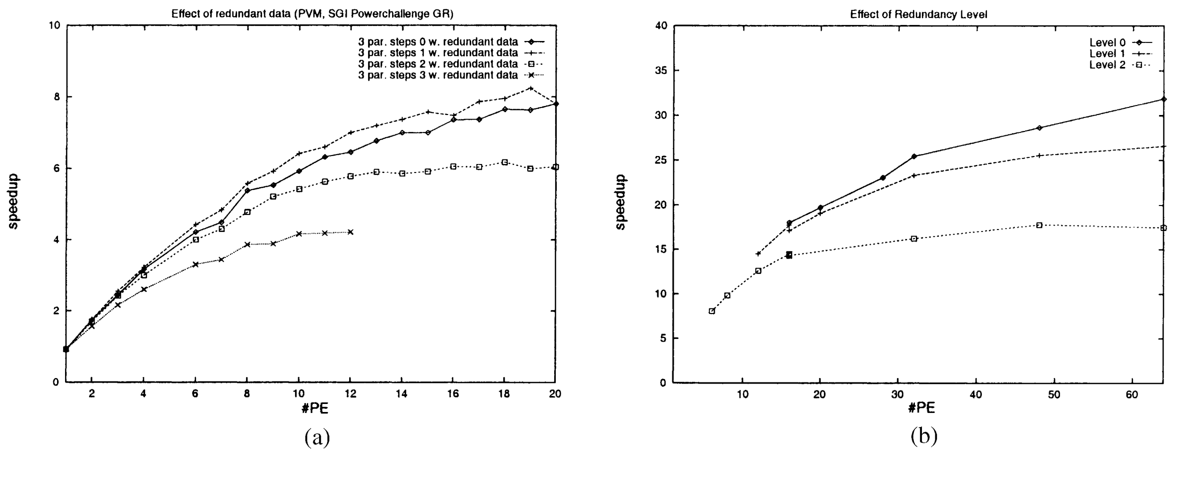
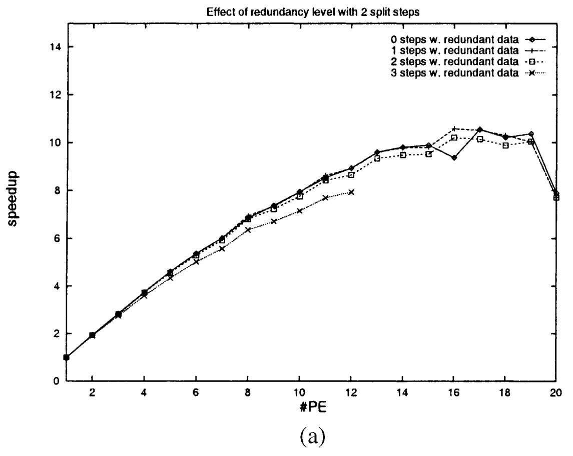


Fig. 9. Effect of different data decomposition techniques for pyramidal wavelet decomposition. (a) SGI Powerchallenge, (b) Cray T3E.



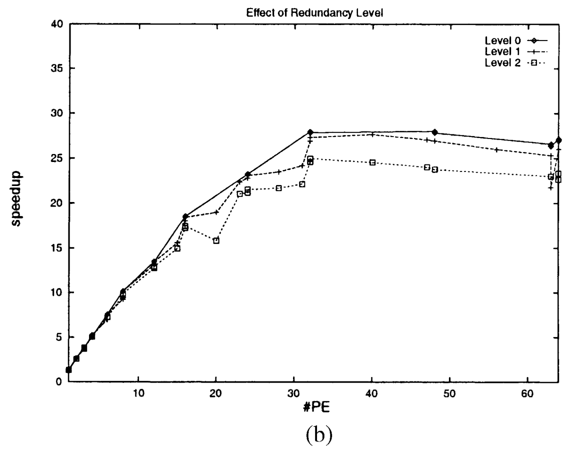


Fig. 10. Effect of different data decomposition techniques for wavelet packet decomposition. (a) SGI Powerchallenge, (b) Cray T3E.

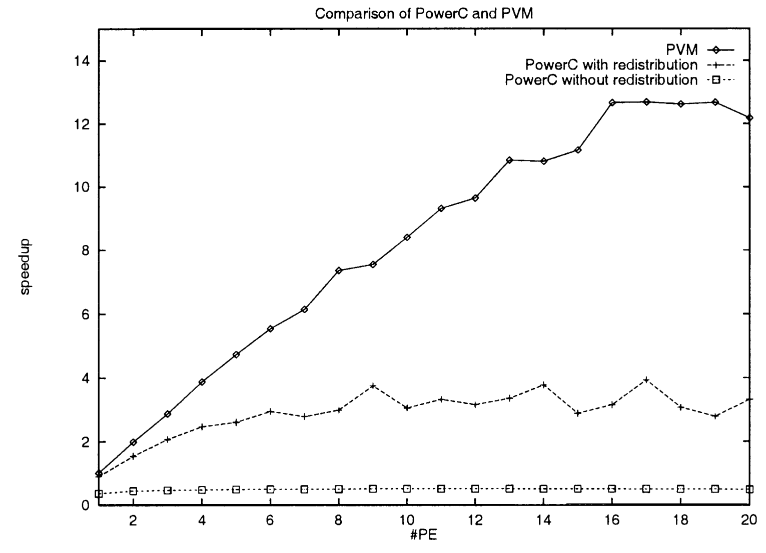


Fig. 11. Message passing vs. data parallel.

5. Conclusion

In this work, we have discussed several issues important for efficient wavelet packet best basis algorithms on parallel architectures. A localized decomposition strategy has been developed which can avoid the need for a second run of the algorithm in memory critical environments and corresponding load balancing problems. Additionally, the experiments show clearly that in contrast to the pyramidal wavelet transform the border data problem can be neglected. Furthermore, we have found that a subband based data redistribution is necessary to achieve satisfying speedup behaviour. The shared memory programming approach is not able to compete against the message passing implementation on a multiprocessor.

Acknowledgements

We want to thank the NIC Jülich for providing access to its Cray T3E systems. R. Kutil's work was partially supported by the Austrian Science Fund FWF, Project No. P11045-ÖMA.

References

[1] A. Bruckmann, Th. Schell, A. Uhl, Evolving subband structures for wavelet packet based image compression using genetic algorithms with non-additive cost functions, in: Proceedings of the International Conference on Wavelets and Multiscale Methods, IWC'98, Tangier, 1998, INRIA, Rocquencourt, April 1998, 4 pp.

[2] S. Chawla, D.M. Healy, Fast parallel implementation of the wavelet-packet best-basis algorithm on the MP-2 for real-time MRI, in: A.F. Laine, M.A. Unser, A. Aldroubi (Eds.), Wavelet Applications in Signal and Imaging Processing IV, SPIE Proceedings, Vol. 2825, 1996, pp. 409–421.

[3] C.H. Chu, Genetic algorithm search of multiresolution tree with applications in data compression, in: H.H. Szu (Ed.), Wavelet Applications, SPIE Proceedings, Vol. 2242, 1994, pp. 950–958.

[4] R.R. Coifman, Y. Meyer, S. Quake, M.V. Wickerhauser, Signal processing and compression with wavelet packets, in: Y. Meyer, S. Roques (Eds.), Progress in Wavelet Analysis and Applications, Proceedings of the International Conference on Wavelets and Applications, Toulouse, 1992, Editions Frontières, Dreux, 1993, pp. 77–93.

[5] R.R. Coifman, M.V. Wickerhauser, Entropy based methods for best basis selection, IEEE Trans. Inform. Theory 38 (2) (1992) 719–746.

[6] S. Corsaro, L. D'Amore, A. Murli, On the parallel implementation of the fast wavelet packet transform on MIMD distributed memory environments, in: P. Zinterhof, M. Vajtersic, A. Uhl (Eds.), Parallel computation, Proceedings of ACPC'99, Lecture Notes on Computer Science, Vol. 1557, Springer, Berlin, 1999, pp. 357–366.

[7] M. Feil, A. Uhl, Wavelet packet decomposition and best basis selection on massively parallel SIMD arrays, in: Proceedings of the International Conference on Wavelets and Multiscale Methods, IWC'98, Tangier, INRIA, Rocquencourt, April 1998, 4 pp.

[8] M. Feil, A. Uhl, Algorithms and programming paradigms for 2D wavelet packet decomposition on multicomputers and multiprocessors, in: P. Zinterhof, M. Vajtersic, A. Uhl (Eds.), Parallel computation, Proceedings of ACPC'99, Lecture Notes on Computer Science, Vol. 1557, Springer, Berlin, 1999. pp. 367–376.

[9] M. Feil, A. Uhl, Real-time image analysis using wavelets: the "à trous" algorithm on MIMD architectures, in: D. Sinha (Ed.), Real-time Imaging IV, SPIE Proceedings, Vol. 3645, 1999, pp. 56–65.

[10] E. Goirand, M.V. Wickerhauser, M. Farge, A parallel two-dimensional wavelet packet transform and some applications in computing and compression analysis, in: R. Motard, B. Joseph (Eds.), Applications of Wavelet Transforms in Chemical Engineering, Kluwer Academic Publishers, Dordrecht, 1995, pp. 275–319.

[11] T. Hopper, Compression of gray-scale fingerprint images, in: H.H. Szu (Ed.), Wavelet Applications, SPIE Proceedings, Vol. 2242, 1994, pp. 180–187.

[12] W.L. Hsu, H. Derin, Video compression using adaptive wavelet packets and DPCM, in: H.H. Li, S. Sun, H. Derin (Eds.), Video Data Compression for Multimedia Computing, Kluwer Academic Publishers, Dordrecht, 1997, pp. 55–94.

[13] A. Laine, J. Fan, Texture classification by wavelet packet signatures, IEEE Trans. Pattern Anal. Machine Intell. 11 (15) (1993) 1186–1191.

[14] R.E. Learned, A.S. Willsky, A wavelet packet approach to transient signal classification, Appl. Comput. Harmonic Anal. 2 (1995) 265–278.

[15] A.R. Lindsey, J.C. Dill, Digital transceiver implementation for wavelet packet modulation, in: H.H. Szu (Ed.), Wavelet Applications V, SPIE Proceedings, Vol. 3391, 1998, pp. 255–264.

[16] L. Bacchelli Montefusco, Semi-orthogonal wavelet packet bases for parallel least-squares approximation, J. Comput. Appl. Math. 73 (1996) 191–208.

[17] K. Ramchandran, M. Vetterli, Best wavelet packet bases in a rate-distortion sense, IEEE Trans. Image Process. 2 (2) (1993) 160–175.

[18] N. Saito, R.R. Coifman, Local discriminant bases, in: A.F. Laine, M.A. Unser (Eds.), Wavelet Applications in Signal and Image Processing II, SPIE Proceedings, Vol. 2303, 1994, pp. 2–14.

[19] S. Sullivan, Vector and parallel implementations of the wavelet transform, Technical Report, Center for Supercomputing Research and Development, University of Illinois, Urbana, IL, 1991.

[20] C. Taswell, Satisficing search algorithms for selecting near-best bases in adaptive tree-structured wavelet transforms, IEEE Trans. Signal Process. 44 (10) (1996) 2423–2438.

[21] A. Uhl, Adapted wavelet analysis on moderate parallel distributed memory MIMD architectures, in: A. Ferreira, J. Rolim (Eds.), Parallel Algorithms for Irregularly Structured Problems, Lecture Notes in Computer Science, Vol. 980, Springer, Berlin, 1995, pp. 275–284.

[22] A. Uhl, Wavelet packet best basis selection on moderate parallel MIMD architectures, Parallel Comput. 22 (1) (1996) 149–158.

[23] J. Wang, H.K. Huang, Medical image compression by using three-dimensional wavelet transformation, IEEE Trans. Medical Imaging 15 (4) (1996) 547–554.

[24] M.V. Wickerhauser, Adapted Wavelet Analysis from Theory to Software, A.K. Peters, Wellesley, MA, 1994.

[25] M.-L. Woo, Parallel discrete wavelet transform on the Paragon MIMD machine, in: R.S. Schreiber, et al. (Ed.), Proceedings

of the Seventh SIAM Conference on Parallel Processing for Scientific Computing, 1995, pp. 3–8.
[26] Z. Xiong, K. Ramchandran, M.T. Orchard, Wavelet packet image coding using space–frequency quantization, IEEE Trans. Image Process. 7 (6) (1998) 892–898.

R. Kutil received the BS (Computer Science) degree from the University of Salzburg and is currently working as a Research Assistant at the Department of Scientific Computing. His research interests include wavelets and parallel processing.

A. Uhl received the BS and MS degrees (both in Mathematics) from the University of Salzburg and he completed his PhD on Applied Mathematics at the same University. Currently, he is an Assistant Professor at the Department of Scientific Computing and at the Research Institute for Software Technology. His research interests include wavelets, image processing (with emphasis on adaptive image compression), number theoretical methods in numerical analysis, and parallel processing.