Web-Based Parallel ML-EM Reconstruction for SPECT on SMP Clusters *

Werner Backfrieder
Department of Biomedical Engineering and Physics
Vienna University Hospital
L. Boltzmann Institute of Nuclear Medicine
Vienna, Austria

Siegfried Benkner, Gerhard Engelbrecht
Institute for Software Science
University of Vienna
Vienna, Austria

Abstract *In emission tomography the spatial distribution of radioactive tracer is reconstructed from projection data. The whole image volume is simultaneously calculated from the full set of projection values during the iteration steps of fully 3D image reconstruction. In this article we report on the development and implementation of an improved ML-EM algorithm, which implicitly corrects attenuation and scatter and minimizes reconstruction noise. To demonstrate the clinical usefulness results of the algorithm were compared to reconstructions with filtered backprojection. The improved image reconstruction algorithm has been parallelized for efficient execution on SMP clusters by adopting a hybrid parallelization strategy based on a combination of the MPI and OpenMP application programming interfaces. In order to employ the parallel image reconstruction software over the Internet, a Java-based framework is being developed allowing transparent access to remote high-performance computing platforms from clinical environments.*

Keywords: ML-EM reconstruction, attenuation correction, hybrid parallelization, SMP clusters

***Published in:** Proceeding of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences, Las Vegas, Nevada, USA, June 25-28, CSERA Press, 2001. The research was partially supported by the Austrian Science Fund as part of the AURORA project under contract SFB F011.

1 Introduction

In traditional single photon emission computed tomography (SPECT) and positron emission tomography (PET), the spatial distribution of radioactive tracer is reconstructed within transaxial slices through the patient. A stack of consecutive slices forms a 3D image volume. However, data acquisition is a 3D process caused by detector aperture and scatter. Modern iterative reconstruction algorithms provide accurate image generation allowing the correction of most physical effects, but at the expense of high computational resources.

We developed an improved ML-EM algorithm for fully 3D image reconstruction. The algorithm implicitly corrects for scatter and attenuation in 3D. The high computational requirements of the improved method necessitate a parallel implementation. We realized a hybrid parallel implementation on an SMP cluster by combining the MPI and the OpenMP programming paradigms. A web-based framework is being developed to access the reconstruction software over the Internet, allowing on-site clinical use of the system with transparent access to remote parallel computing platforms.

The paper is organized as follows: In Section 2 we describe our improved ML-EM algorithm. In Section 3 the parallelization of the reconstruction algorithm for SMP clusters is discussed followed by experimental performance results in Section 4. In Section 5 we briefly describe the web-based image reconstruction framework. Section 6 concludes the paper.

2 Algorithm

With a SPECT scanner, a detector head measures 2D projections of the patient at equidistant angular positions. In SPECT, data acquisition is no strict 2D process, due to collimator geometry and scatter, photons emitted outside the image plane are registered by the detector row assigned to the plane. In fully 3D image reconstruction a solution for the whole image volume X is found simultaneously by using the whole set of projection values Y. The components x_j of the image vector show the mean activity within the voxel, with $j = 1...J$ and J the number of voxels. The components y_i of the measurement vector are the detected photon counts, with $i = 1...I$, and I the total number of projection values during the whole acquisition. The imaging process is characterized by the following matrix equation

$$Y = A\,X + E \quad , \tag{1}$$

where A is the system matrix, whose elements a_{ij} give the detection probability at detector position i of a photon emitted in voxel j. The system matrix allows the flexible modeling of the scanner geometry, detector efficiency, scatter and attenuation. In typical clinical applications J, the number of unknowns is 128^3 and I, the number of equations is $128^2 \times 120$ with 3^o rotational increment over a range of 360^o. The ill-posedness of A and the error term E, in general, allow no exact solution of Eqn. 1.

The ML-EM algorithm for emission tomography [1, 2] is an iterative method for finding a feasible solution of Eqn. 1. It takes into account the Poisson statistics of photon emission and detection. During one iteration step, the voxel $x_j^{(n)}$ of the n-th iteration is updated by

$$x_j^{(n+1)} = x_j^{(n)} \sum_i a_{ij} \frac{y_i}{\sum_k a_{ik} x_k^{(n)}} \quad . \tag{2}$$

We developed an improved ML-EM algorithm to correct for attenuation and minimize reconstruction noise, i.e. images are deteriorated by spikes since Eqn. 2 produces a minimum entropy solution towards high iteration numbers. For attenuation correction we introduced a multiplicative factor b_j for each voxel, similar to Chang's attenuation correction [3]

$$x_j^{(n+1)} = x_j^{(n)} b_j \sum_i a_{ij} \frac{y_i}{\sum_k a_{ik} x_k^{(n)}} \quad . \tag{3}$$

The factor b_j is calculated from the linear attenuation coefficients μ_j associated with each voxel in the volume. This simplified method needs a correction vector B of length J, whereas a direct implementation into the system matrix would destroy all symmetries of the matrix used for efficient storage. In the worst case A would have $(I \times J)$ entries. This number is far above the limits of physical memory available.

After each full iteration, i.e. all J voxels of the image volume are updated, the image $X^{(n)}$ is filtered using a $(3 \times 3 \times 3)$ median filter, thus spikes from reconstruction noise are removed. This procedure is similar to the maximum a posteriori (MAP) approach using a median root function as prior information [4]. The system matrix A was designed to correct both for scanner geometry and object independent scatter. The line of response (LOR) of a SPECT scanner is defined by the collimator geometry. It is a truncated cone, its starting width is equal to the detector aperture and broadens with increasing distance from the detector. Scattered photons emitted outside this LOR are accidentally registered by the detector. The LOR was modeled in 3D by a series of 2D Gaussian distributions perpendicular to the direction of projection with distance-dependent full width half maximum (FWHM).

3 Implementation and Parallelization

Most of the elements of matrix A are zero. Assuming a cone like LOR, the non-zero elements within one row of A are $\sqrt{2}J^{(1/3)}/3(3\sigma)^2\pi$, where σ is derived from the FWHM of the most distant point and $\sqrt{2}J^{(1/3)}$ is the flat diagonal of the image volume. Using row column storage reduces memory requirements significantly. By exploiting the block symmetries of A, memory requirements are further reduced. The matrix elements were calculated for one detector row and all angular positions, reducing the matrix size by a factor equal to the number of rows or slices.

The reconstruction algorithm has been initially implemented in C on a SUN UltraSparc10. The computing time required for the reconstruction of a 128^3 image volume is within the range of 90 hours. In order to allow on-site clinical analysis, the computational requirements have to be reduced by a factor of 50 to 80. This is possible only by parallelizing the reconstruction algorithm and running it on a powerful parallel computing platform.

Over the last several years, clusters of shared-memory parallel processors (SMPs) have become the most promising parallel computer architectures. These machines can be characterized as hybrid-parallel systems, combining distributed-memory (DM) and shared-memory (SM) parallel architectures.

In our work, we have been initially targeting an SMP PC cluster, consisting of 6 computer nodes, each equipped with 4 Pentium processors and a large shared memory. The parallelization of the reconstruction algorithm is based on a hybrid parallelization strategy realized in two major development phases. In the first phase, the sequential reconstruction algorithm has been parallelized for a single SMP node utilizing the recently proposed OpenMP [8] API which is a de-facto standard for shared-memory parallel programming. While shared-memory parallelization with OpenMP is well suited for our application, the resulting parallel program can only be executed on a single node of the cluster. Since OpenMP does not support parallelization across the nodes of a cluster, we are utilizing the message-passing interface MPI [7] for realizing distributed-memory parallelization on top of OpenMP.

Shared-memory parallelization of the reconstruction algorithm has been realized by executing the computations described by Eqn.2 in parallel by multiple threads, each executing on its own processor within a node. As a consequence, each processor calculates a part of the weighted sum of corrected projection values along a column of the system matrix A. In the code this is realized by inserting corresponding OpenMP work-sharing directives.

Distributed-memory parallelization is based on the *single-program multiple-data* (SPMD) paradigm. The code is parameterized in such a way that it can be executed by multiple processes, each computing its own part of the image volume. This was implemented by strip-mining the outer loop representing the computations defined by Eqn.2 across multiple processes. Each of these processes employs multiple OpenMP threads as outlined above. In order to minimize communication overheads, most data structures are replicated, i.e. allocated on each processor. At the end of a full iteration cycle, the voxels updated by individual processes are transferred to all other processes in a global communication phase, utilizing MPI collective communication primitives.

The parallelized source program is compiled with an OpenMP C compiler and linked with the MPI library. The resulting program is executed on an SMP cluster as an MPI program by a set of parallel processes, each running on a separate node of the cluster within its own local address space. Due to the use of OpenMP, each of these processes employs a set of threads (usually according to the number of processors available within a node) which execute concurrently in the shared address space of a node.

The reconstruction algorithm has been implemented in such a way that it is portable across distributed-memory architec-

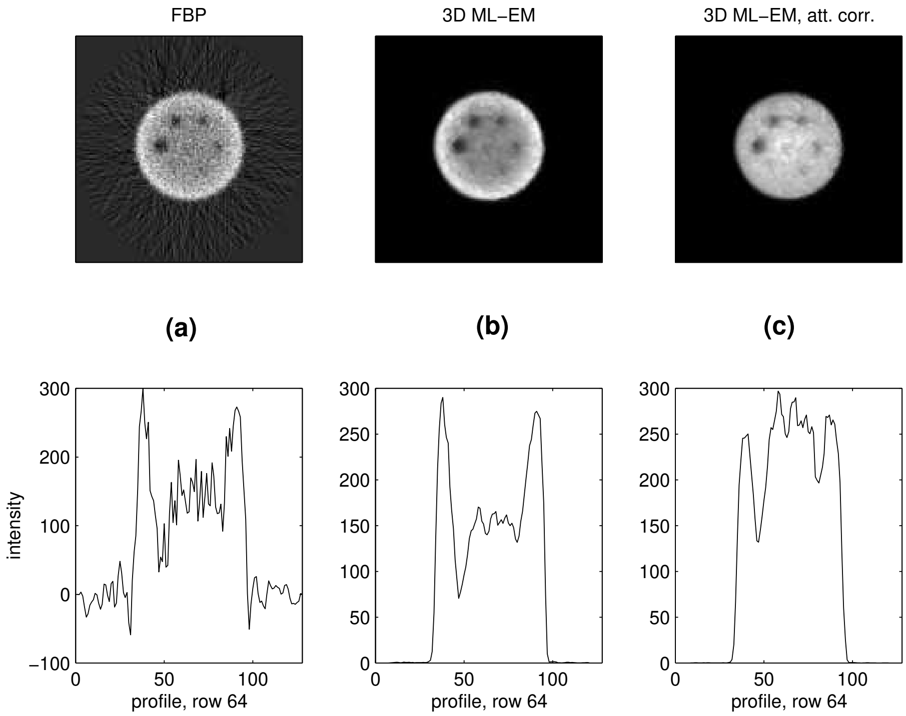

Figure 1: Jaszczak phantom: A reconstructed slice in the region with 5 cold spheres is shown together with an intensity plot along row 64 for (a) filtered backprojection, (b) 3D ML-EM without attenuation correction, and (c) attenuation corrected 3D ML-EM.

tures, shared-memory architectures and clusters.

On a hybrid-parallel architecture the combination of MPI and OpenMP offers greater flexibility with respect to memory management than an approach based on MPI only [5]. Moreover, due to the shared memory available within nodes, the number of communications can be reduced while message sizes can be increased. In a pure MPI implementation of our algorithm, not only more messages but also much more memory is required. On the other hand, by using OpenMP, certain synchronization overheads may be introduced whenever multiple threads update data structures allocated in the shared memory of a node. Early experimental results comparing the hybrid-parallel implementation of the reconstruction algorithm with a pure message passing version are presented in the next section.

4 Experimental Results

Data from a clinical standard phantom were reconstructed using the fully 3D algorithm for testing both the spatial resolution and the accuracy of attenuation correction. The efficiency of the parallel implementation was evaluated on an SMP cluster using several combinations of MPI and OpenMP.

4.1 Physical Phantom Data

Spatial resolution and attenuation correction were tested using projection data from a Jaszczak phantom (Data Spectrum, Corp., Chapel Hill, NC), filled with 400 MBq Tc^{99m}. Emission and transmission data were acquired on a Millenium VG (GE Medical Systems, Milwaukee, WI). Acquisition parameters were: 128x128 matrix, 3.2x3.2 mm^2 pixel size, 3^o rotation increment over 360^o for SPECT. The attenuation maps were acquired on co-registered

128x128 matrices using 140 kV voltage and 1.6 mA anode current. Data were reconstructed using the fully 3D algorithm described in this paper, both with and without attenuation correction. Reconstruction was stopped after 30 iterations. Results were compared with standard filtered backprojection (FBP) using a Hamming window. Figure 1 shows a slice of the image volume with a set of 5 perspex spheres with decreasing diameter. These spheres indicate cold lesions in a hot area. The same slice is shown for FBP (a), 3D ML-EM without attenuation correction (b), and attenuation corrected 3D ML-EM (c). An intensity plot along row 64 of each image is shown, respectively. 3D ML-EM shows higher signal-to-noise ratio compared to FBP. Furthermore no overshoots are in the image areas with high contrast. 3D ML-EM without attenuation correction has strong intensity defects in the central region of the phantom, whereas the attenuation corrected image shows a homogeneous intensity distribution in the hot regions. 3D ML-EM proved to be an accurate method for image reconstruction in SPECT.

4.2 Implementation on an SMP Cluster

In this section we discuss performance experiments with the initial version of our parallel image reconstruction code. The experiments were performed on a Beowulf-type PC cluster consisting of 6 nodes connected by Fast Ethernet. Each node is equipped with 4 Pentium III Xeon processors (700 MHz) and 4 GB of memory. For the compilation of the MPI/OpenMP source program the C compiler from Portland Group Inc. was used together with the MPI library mpich 1.2.2.

We have evaluated a pure OpenMP version on a single node of the cluster, and a hybrid MPI/OpenMP version and a pure MPI message-passing version on up to 6 nodes. Most performance experiments were performed using a small image volume of only 16^3 voxels. Larger image volumes have been evaluated as well, however not yet for all possible combinations of MPI processes and OpenMP threads.

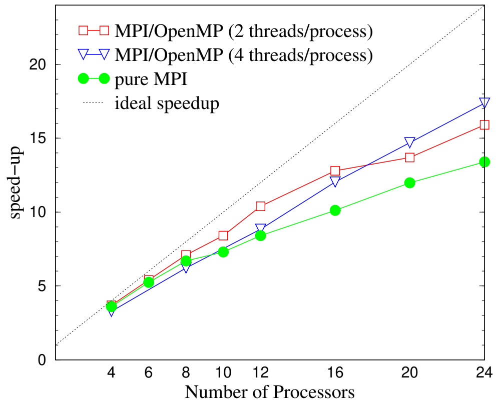

Figure 2: Experimental comparison of a pure MPI version to hybrid-parallel MPI/OpenMP versions with 2 and 4 threads per MPI process, respectively.

On a single node of the cluster the pure OpenMP version achieved a speed-up of 3.2 for the small image volume, and a speed-up of up-to 3.6 for the large image size (128x128x128). Slightly better speed-ups were observed with the pure MPI version on a single node which mainly seems to be due to mpich's optimized inter-process communication (via shared memory) within a node. All speed-up numbers have been obtained with respect to the sequential code executed on a single processor of the cluster.

In Figure 2 three different code variants are compared. The pure MPI version did not utilize any OpenMP multi-threading features. This version is compared to hybrid-parallel versions which employed 2 and 4 threads within each MPI process, respectively. The hybrid-parallel versions have been executed in such a way that each thread is always running on its own processor. On two nodes (8 processors), all three versions give approximately the same speed-ups. On more than two-nodes, the pure MPI-version is outperformed by the MPI/OpenMP versions. One reason for this behavior is that the MPI version requires more memory because replicated data structures are allocated within each process, while in the MPI/OpenMP version all threads have access to the data allocated within their parent processes. Another reason for the worse scaling of the pure MPI version is the larger communication overhead. As can be observed from Figure 2, the performance of the MPI/OpenMP version where only 2 threads were generated within each MPI process decreases when more than one MPI process is run on a node, i.e. if more than 12 processors are used. If all 24 processors of the cluster are utilized, a significant performance difference between the pure MPI version and the MPI/OpenMP versions can be seen. However, this gap is becoming smaller for larger image volumes because of an increased computation-to-communication ratio.

Communication optimizations and a more detailed performance evaluation including experiments with different load-balancing strategies at the OpenMP level are the subject of ongoing work. Moreover, the reconstruction code will be ported to other parallel computing platforms, including a heterogeneous SMP cluster (2-processor nodes and 4-processor nodes), an SGI Origin 2000, and an NEC SX-5 multinode vector supercomputer. In order to efficiently utilize the impressive computing power offered by vector supercomputers such as the SX-5, minor modifications of our implementation are required for exploiting all three levels of parallelism (i.e. distributed-memory, shared-memory, and vector parallelism).

5 Web-Based Framework

Since usually parallel computing platforms are not available on-site in a clinical environment, we are developing a web-based framework for image reconstruction. This framework is based on a client-server architecture implemented with Java technology. The client part consists of a graphical user interface (GUI) realized as a Java applet which can be deployed with a standard web browser. The server software is implemented as a Java application which realizes the client-server communication based on Java's Remote Method Invocation (RMI) [9] mechanism.

In a typical use-case scenario of the system, projection data are displayed in the browser. The operator selects the slices to be reconstructed and various reconstruction parameters. The raw image data is transferred to the remote parallel machine by means of the Java RMI mechanism. Upon receipt of the raw image data on the server-side, the reconstruction algorithm is launched. Reconstructed images are transferred back to the client-side and displayed in the browser applet. For compatibility reasons, conversion to the medical standard image format DICOM is provided.

The framework has been designed in such a way that various parallel computing platforms may be supported seamlessly. Currently we are including an SGI Origin 2000 shared-memory parallel computer as well as an NEC SX-5 vector supercomputer into the reconstruction framework. For the future we plan to integrate the system with a medical image data base.

6 Conclusions

In this paper we have described a fully 3D reconstruction algorithm which realizes scatter and attenuation correction. Attenuation correction is object-dependent, based on multiplicative correction of voxels retaining the symmetry of the system matrix. Due to the high computational requirements of the reconstruction process, we have implemented a parallel version of the algorithm on SMP clusters utilizing a hybrid parallelization strategy combining MPI and OpenMP. Early experimental results indicate the potential of reducing image reconstruction time through parallelization to an extent where routine patient examination becomes feasible. With the implementation on an SMP cluster, as described in section 4.2, reconstruction times of 2.4 hours could be achieved for an 128^3 image volume passing through 30 iterations. Further speed-ups of the algorithm with a factor of 30 can be achieved by implementing ordered subsets [6].

By employing a web-based framework providing transparent access to a remote parallel computer over the Internet, highly sophisticated 3D image reconstruction will be available for on-site clinical use.

Acknowledgements
Authors would like to thank the Clinic of Nuclear Medicine of the Vienna University Hospital for providing the equipment for measurements, Ing. Peter Schaffarich for measuring SPECT phantom data and Dr. Willy Weisz form the Institute for Software Technology of the University of Vienna for valuable discussions on the SMP cluster.

References

[1] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. Roy. Statist. Soc. B, vol. 39, pp. 1-38, 1977.

[2] L. A. Shepp and Y. Vardi, Maximum likelihood reconstruction for emission tomography, IEEE Trans. Med. Imaging, vol. 1, no. 2, pp. 113-122, 1982.

[3] L. T. Chang, A method for attenuation correction in radionuclide computed tomography, IEEE Trans. Nuclear Sci., vol. 25, pp. 638-642, 1978.

[4] S. Alenius, U. Ruotsalainen, Bayesian image reconstruction for emission tomography based on median root prior, Eur. J. Nuc. Med., vol. 24, no. 3, pp. 258-265, March 1997.

[5] D. S. Henty. Performance of Hybrid Message-Passing and Shared-Memory Parallelism for Discrete Element Modeling. In *Proceedings of SC 2000*, Dallas, November 2000.

[6] H. M. Hudson, R. S. Larkin, Accelerated image reconstruction using ordered subsets of projection data, IEEE Trans. Med. Im., vol. 13, pp. 601-609, 1994.

[7] Message Passing Interface Forum. MPI: A Message-Passing Interface Standard. Vers. 1.1, June 1995. MPI-2: Extensions to the Message-Passing Interface, 1997.

[8] The OpenMP Forum. OpenMP C and C++ Application Program Interface. Version 1.0, October 1998. `http://www.openmp.org`.

[9] JavaTM Remote Method Invocation Specification. Sun Microsystems Inc., v1.3.0, Dec. 1999. `http://java.sun.com/j2se/1.3/docs/guide/rmi/spec/rmiTOC.html`