

where $u(k)$ is the unit step function. Based on the projection-slice theorem and the derivative property of the Fourier transform, we have

$$\begin{aligned} & \mathcal{F}^{-1}\{\mathcal{F}_\rho(k\vec{\mu})k^{n-1}u(k)\} \\ &= \frac{1}{(i2\pi)^{n-1}}\mathcal{F}^{-1}\{\mathcal{F}_\rho(k\vec{\mu})(i2\pi k)^{n-1}\} * \mathcal{F}^{-1}\{u(k)\} \\ &= \frac{1}{(i2\pi)^{n-1}}\frac{\partial^{n-1}\mathcal{R}_\rho(p,\vec{\mu})}{\partial p^{n-1}} * \left[\frac{1}{2}\delta(p) - \frac{1}{i2\pi p}\right] \\ &= \frac{1}{(i2\pi)^{n-1}}\left[\frac{1}{2}\frac{\partial^{n-1}\mathcal{R}_\rho(p,\vec{\mu})}{\partial p^{n-1}} - \frac{1}{i2\pi}\frac{\partial^{n-1}\mathcal{R}_\rho(p,\vec{\mu})}{\partial p^{n-1}} * \frac{1}{p}\right]. \end{aligned}$$

Substituting this result into (12) with $p = \vec{\mu} \cdot \vec{x}$, (10) immediately follows.

ACKNOWLEDGMENT

The author would like to thank P. C. Lauterbur for helpful discussion.

REFERENCES

- [1] S. R. Deans, *The Radon Transform and Some of Its Applications*. New York: Wiley, 1983.
- [2] F. Natterer, *The Mathematics of Computerized Tomography*. New York: Wiley, 1986.
- [3] D. Ludwig, "The Radon transform on Euclidean space," *Commun. Pure Appl. Math.*, vol. 19, pp. 49–81, 1966.

Parallelized Formulation of the Maximum Likelihood-Expectation Maximization Algorithm for Fine-Grain Message-Passing Architectures

J. L. Cruz-Rivera, E. V. R. Di Bella, D. S. Wills, T. K. Gaylord, and E. N. Glytis

Abstract—Recent architectural and technological advances have led to the feasibility of a new class of massively parallel processing systems based on a fine-grain, message-passing computational model. These machines provide a new alternative for the development of fast, cost-efficient Maximum Likelihood-Expectation Maximization (ML-EM) algorithmic formulations. As an important first step in determining the potential performance benefits to be garnered from such formulations, we have developed an ML-EM algorithm suitable for the high-communications, low-memory (HCLM) execution model supported by this new class of machines. Evaluation of this algorithm indicates a normalized least-square error comparable to, or better than, that obtained via a sequential ray-driven ML-EM formulation and an effective speedup in execution time (as determined via discrete-event simulation of the Pica multiprocessor system currently under development at the Georgia Institute of Technology) of well over two orders of magnitude compared to current ray-driven sequential ML-EM formulations on high-end workstations. Thus, the HCLM algorithmic formulation may provide ML-EM reconstructions within clinical time-frames.

Manuscript received March 23, 1994; revised July 10, 1995. This work was supported by the Joint Services Electronics Program Grant DAAL 04-93-6-0027, Emory/Georgia Tech Biomedical Technology Center Grant B08-321 and a Patricia Roberts-Harris Fellowship (JLCR). The Associate Editor responsible for coordinating the review of this paper and recommending its publication was Y. Censor.

The authors are with the School of Electrical and Computer Engineering and Microelectronics Research Center, Georgia Institute of Technology, Atlanta, Georgia, 30332 USA.

IEEE Log Number 9415793.

I. INTRODUCTION

The Maximum Likelihood-Expectation Maximization (ML-EM) algorithm has been demonstrated to be superior to the Filtered Backprojection (FBP) algorithm for the reconstruction of Positron Emission Tomography (PET) and Single-Photon Emission Computed Tomography (SPECT) medical images, both through simulation studies [1]–[2] and through clinical trials focusing on accurate lesion detection from patient data [3]. Unfortunately, the use of the ML-EM algorithm within the clinical environment has been thwarted by the large computational time and memory requirements involved in its implementation; leaving the FBP algorithm as the method of choice in clinical studies. This situation has led to numerous research efforts directed toward the development of efficient ML-EM algorithmic formulations targeting a variety of computational platforms, with particular effort being geared toward parallel processing systems [4]–[9].

The parallel implementations that have been put forth to date differ not only in structure, but also in the data-sets targeted. Coarse-grain implementations of the ML-EM have typically targeted the large data-sets encountered in PET (e.g., 256×256 , 120 views) [4]–[6]. Fine-grain ML-EM algorithmic formulations, on the other hand, have particularly targeted the smaller SPECT data-sets (e.g., 64×64 , 96 views), due to the low-memory per node present in commercially available massively parallel systems [7]–[9]. If large data-sets are targeted (e.g., PET), a major factor influencing the cost/performance metric is the computation/communication ratio achieved by the algorithmic formulation. In coarse-grain systems this ratio is maximized through the use of large local processor memories and through the development of efficient data and task partitioning schemes [6]; unfortunately, the large imbalance between storage and computational resources in these machines effectively constrains the overall cost/performance levels that can be achieved, as a large portion of the hardware cost is attributed to the sequentially accessed memory resource and performance is limited by the inability to exploit very fine-levels of parallelism. In fine-grain systems, on the other hand, cost/performance is limited by the small amount of memory available per processing node, by limited interprocessor communication throughput, and by the SIMD (single instruction-multiple data) computational model commonly employed. In these systems high performance levels can be obtained for small data-sets [7]–[8], but scaling the algorithmic implementations without an ensuing scaling of the machine resources does not provide added parallelism (e.g., *virtual processors* could be employed, but this would not permit additional parallelism to be exploited). Thus, while very good performance levels can be achieved through both fine-grain and coarse-grain parallel implementations, the execution model employed by current systems constrains cost/performance. This situation can be potentially improved through the use of a new class of parallel architectures that are based on a fine-grain, message-passing MIMD (multiple instruction-multiple data) computational model.

Fine-grain, message-passing systems target a high-communications, low-memory (HCLM) execution model. The driving force of these systems is the desire to exploit higher levels of parallelism than are currently possible from fine-grain SIMD systems, while minimizing the memory and communication costs associated with coarse-grain MIMD systems. The architectural and technological issues that enable this type of system to be implemented are advances in system architectures (low latency routing protocols, primitive architectural mechanisms for high-throughput operation,

and latency-hiding mechanisms) and advances in electronic and optoelectronic interconnection and packaging technologies (high-density, high-bandwidth interconnection networks). Example fine-grain, message-passing systems include the MIT J-Machine [10] (for which a prototype exists) and the Pica multiprocessor system [11] (currently under development at the Georgia Institute of Technology).

The purpose of this paper is to assess the potential performance benefits that can be obtained through HCLM ML-EM algorithmic formulations. In Section II, the iterative ML-EM update equation is recast into an operator-based representation that is more suitable to a communications-intensive algorithmic implementation. The actual parallelization of the operator-based equation for PET data-sets (with particular attention given to task and data partitioning) is discussed in Section III and the computational performance of the algorithm, as determined via a set of simulation tools developed for the Pica multiprocessor, is discussed in Section IV. A summary is presented in Section V.

II. OPERATOR-BASED ML-EM ALGORITHM

The ML-EM algorithm attempts to reconstruct the radionuclide density distribution of a patient from projection data produced by the decay of an injected radiotracer. The algorithm iteratively maximizes the expected value of the logarithm of the likelihood function [1]. The ML-EM update equation can be written as

$$\lambda_{k+1}(b) = \frac{\lambda_k(b)}{\xi(b)} \cdot \sum_{t=1}^T \frac{n(t)p(b,t)}{\sum_{b'=1}^B \lambda_k(b')p(b',t)} = \lambda_k(b) \cdot \beta(b) \quad (1)$$

where $\lambda_k(b)$ is the k^{th} estimate of the radionuclide density function at box b (where a box corresponds to the discretization of the object being imaged or equivalently a pixel in the image space), $n(t)$ is the number of photon counts at tube t (where a tube corresponds to the line-of-flight between two oppositely located detectors in a PET configuration), and $p(b,t)$ is the probability that a photon emitted at box b is detected by tube t [1]. B and T correspond to the total number of boxes and tubes in the configuration. The $\xi(b)$ array is a scaling factor array that remains constant over the iterations and the $\beta(b)$ array is the *update factors* array, which captures the summations and scaling factors in (1).

Note that if only geometric factors are considered, the $p(b,t)$ values in (1) correspond to the length of intersection between a ray (within a tube) and a box. This leads to the recognition that (assuming an ideal psf and neglecting random coincidences and scatter) the ML-EM algorithm can be reduced to a series of simple summation (*projection*) and replication (*backprojection*) operations, provided the λ and β arrays are rotated so that they are aligned with the projection bins. Thus, we recast (1) into the operator-based equation

$$\lambda_{k+1} = \frac{\lambda_k}{\xi} \cdot \sum_{\theta} \left(R_{-\theta} \left\{ B_t \left(\frac{n_{t\theta}}{P_t \{ R_{\theta}(\lambda_k) \}} \right) \right\} \right) = \lambda_k \cdot \beta \quad (2)$$

where we have introduced three vector operators: R_{θ} , B_t , and P_t . R_{θ} is a rotation operator that rotates an $N \times N$ array by the specified angle θ , B_t is a backprojection operator which replicates each element of a column vector of length N across the corresponding row of an $N \times N$ array, and P_t is a projection operator that calculates the ray-sum of an $N \times N$ array on a row-by-row basis.

The operator-based representation of the ML-EM update equation is highly suitable for fine-grain parallelization, as not only are the memory requirements of the ML-EM reduced by the absorption of the $p(b,t)$ values by the rotation operation, but the highly modularized representation of (2), leads to self-contained operations that can be parallelized at a very fine level of granularity. The feasibility

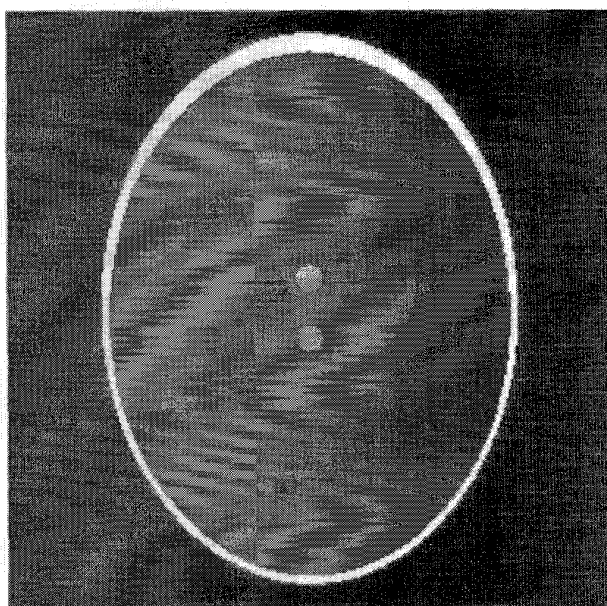
of such a fine-grained implementation of the ML-EM, however, is highly dependent on the accuracy of the rotation operator. While good rotations can be obtained through a variety of interpolation algorithms [12], the need to balance processor and communication throughputs under the targeted HCLM execution model favors a rotation scheme where each pixel in the nonrotated image is treated as being composed of a block of U sub-pixels, each with value equal to $1/U$ of the original. The upsampled image can then be rotated directly (using nearest-neighbor interpolation in the calculation of rotated coordinates) and then down-sampled by simply summing the $\sqrt{U} \times \sqrt{U}$ sub-pixel blocks that constitute a single pixel of the image. The higher the upsample value, the higher the quality of the rotation operation and hence, the higher the quality of image reconstruction.

The operator-based ($U = 4$) ML-EM reconstruction of a noiseless 256×256 Shepp-Logan head phantom from analytically generated sinogram data (120 projections over 180°) is shown in Fig. 1. The normalized least-square error of the reconstructed image is plotted in Fig. 2, where the operator-based ML-EM formulation is compared to a sequential ray-driven ML-EM implementation and to reconstruction via the FBP algorithm. Note that even for the low upsample value $U = 4$, the operator-based algorithm yields an error comparable to that attained via the ray-driven implementation. The qualitative error difference between the two algorithms is negligible.

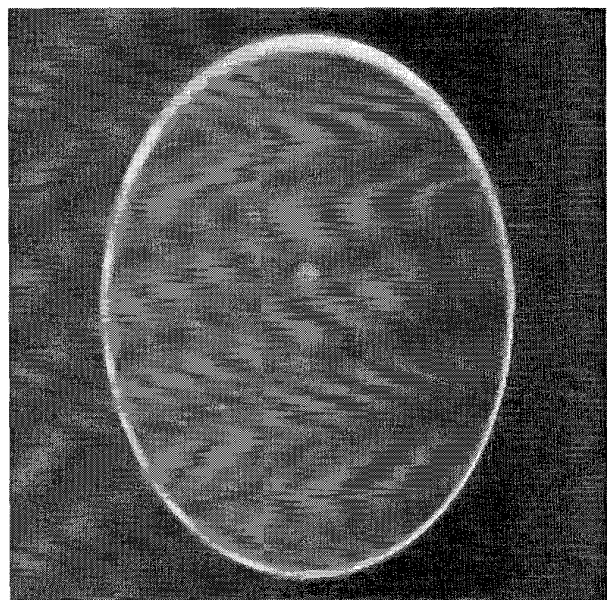
III. PARALLELIZATION

The operator-based ML-EM equation exhibits various levels of potential parallelism. At the highest level we can exploit parallelism by reconstructing multiple slices of a PET data-set simultaneously. Exploiting parallelism at the slice level requires that the algorithmic formulation be distributed among a minimal number of processing elements (PE's) while preserving a close to optimal balance in the utilization of memory, communication, and computational resources, as this would allow the largest possible number of slices to be reconstructed on a given machine size. The backprojection and projection operations can also be parallelized by pipelining the processing across view angles. This can be performed by dividing the task of evaluating (2) among two different sets of PE's, appropriately denoted as the Projection PE's (P-PE's) and Backprojection PE's (B-PE's), and letting them collaborate in time-offset fashion, as will be discussed. Finally, the rotation operation can be parallelized by distributing the λ and β arrays across various processors and having them cooperate in the physical rotation of the array through message-passing. We seek to exploit all these levels of parallelism in our HCLM formulation. To guide the development of the formulation (task and data partitioning) we focus on a typical PET data-set of 21 slices of 256×256 pixels with data collected over 120 views. Before proceeding, it is important to note that our algorithmic formulation differs from that of previous coarse-grain implementations in that the main objective is to balance the granularity of tasks with the memory and communication capabilities of the system, rather than to minimize the latency associated with individual messages. The design also differs from previous fine-grain SIMD implementations in that the message-passing MIMD computational model employed allows us to exploit finer levels of parallelism, while operating on larger data-sets.

The parallelization of (2) can be performed as follows. Since the projection and backprojection operations work on individual rows of appropriately aligned (rotated) arrays, the most natural way to distribute the data arrays among PE's is in a row-wise manner (an alternative block-wise partitioning scheme is discussed later). In order to pipeline the projection and backprojection operations and to subscribe to the low-memory per node constraint, data allocation is such that the P-PE's are allocated one row each of the 256×256 λ_k



(a)



(b)

Fig. 1. (a) Ideal Shepp-Logan head phantom and (b) reconstructed phantom after 32 iterations with operator-based ML-EM formulation ($U = 4$).

and $R_\theta\{\lambda_k\}$ arrays and the *sine* and *cosine* values for the 120 view angles to be processed (for a total of 752 words), while the B-PE's are allocated one row each of the 256×256 β and ξ arrays, the *sine* and *cosine* values for the 120 view angles, and 120 $n_{t\theta}$ values (for a total of 872 words). Each P-PE is responsible for reconstructing one row of the image array, λ , while each B-PE is responsible for computing one row of the update array, β . The two sets of PE's collaborate in a pipelined fashion by having the P-PE's rotate the λ_k array to a particular view angle, $R_\theta\{\lambda_k\}$, compute the projections of the rotated array, $P_t\{R_\theta(\lambda_k)\}$, and then send these values to the

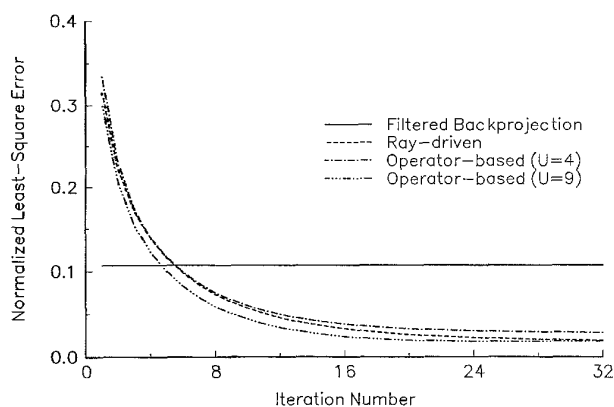


Fig. 2. Normalized least-squares error function for implementations of the filtered backprojection (FBP) algorithm, a sequential ray-driven ML-EM formulation, and the operator-based ML-EM formulation across upsample values.

B-PE's in charge of the same row of the β array as the P-PE's row of the λ array. The B-PE's, which are provided with the necessary $n_{t\theta}$ values, take the received projection values as arguments in computing this angle's contribution to the update factors array, β , backproject the $n_{t\theta}/P_t\{R_\theta(\lambda_k)\}$ value across a temporary array, β' , and then rotate this array to the negative of the view angle being processed before adding it to the β array. Once all views have been processed, the B-PE's divide the β values by the corresponding ξ values and send the scaled β values to the appropriate P-PE's so that the latter may update their assigned λ_k values, producing λ_{k+1} .

The row-wise task and data partitioning scheme results in very low-memory per node requirements and a high utilization of computational and communication resources. Computational and communication resources are particularly stressed by the rotation operation, since the projection and backprojection tasks are simple once the appropriate arrays are aligned to the particular view angle being processed (a result of row-wise data partitioning). The rotation operation is based on axis transformations over the λ (P-PE) or β (B-PE) arrays (computation) and on remote-memory accesses of pixels from the nonrotated arrays (communication).

While the row-wise partitioning scheme provides a very elegant parallelized implementation, the overall efficiency with which processing resources are used is limited by load imbalances. That is, under this partitioning scheme, the amount of work and the number of pixel requests and replies a given PE will generate depends not only on the view angle being processed, but also on the row of the array assigned to the PE. PE's in charge of rows in the middle of the array will have a substantially larger amount of work to perform than those in charge of the outer rows of the array, since only a fraction of the pixels near the edges fall within the circular field of view. This problem can be alleviated by employing an alternative *block-wise* partitioning scheme, where each PE is assigned an $N^t \times N^t$ sub-block of the array to be rotated, rather than an entire row of the array. In this scheme, each P-PE is responsible for calculating a partial projection value for each of its N^t assigned rows. In a pipelined fashion, each B-PE is then responsible for gathering the partial projections and adding them together before proceeding with scaling the assigned detector-count values. The resulting number of active pixels (i.e., pixels within the circular field of view) assigned to each PE under both the row-wise and block-wise partitioning schemes is shown in Fig. 3. Note that while the block-wise partitioning scheme serves to better balance the computational load for each view angle, it does not minimize load imbalances experienced during any one

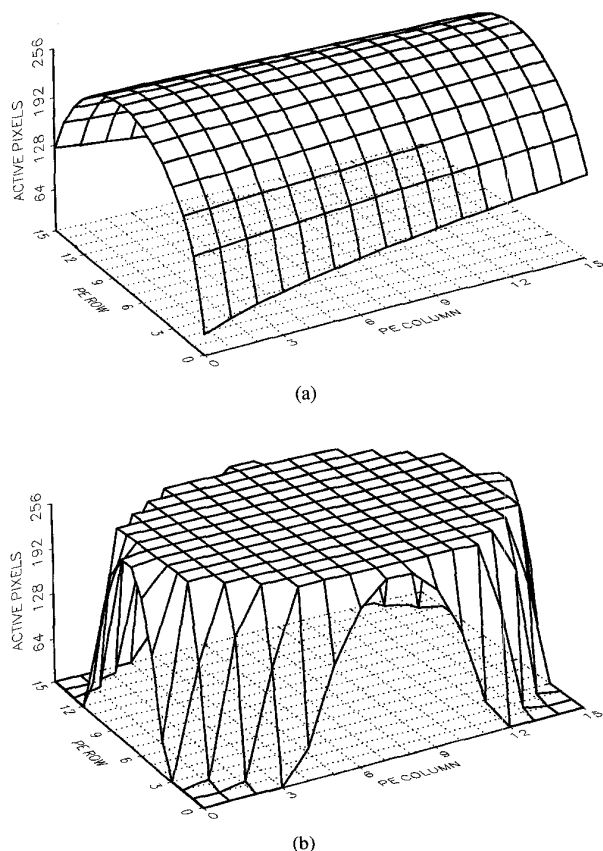


Fig. 3. Active pixels assigned per processing elements: (a) row-wise partitioning, and (b) block-wise partitioning.

particular view angle. However, balancing the amount of processing each PE performs across a *particular* view angle would require a less regular data partitioning strategy that would ultimately decrease performance, since extra synchronization events would arise in the computation of the projection and backprojection operations.

IV. SIMULATED MACHINE PERFORMANCE EVALUATION

To evaluate the performance benefits of the HCLM parallelized implementation of the operator-based ML-EM algorithmic formulation, a version of the algorithm was coded using the instruction set architecture (ISA) of the Pica multiprocessor. The Pica multiprocessor is to consist of 4096 PE's, each employing a custom Reduced Instruction Set Computer (RISC) architecture and 4Kwords of local memory (divided into 256 contexts of 16 words each). The complexity of each PE is such that multiple PE's may be implemented on a single chip (four in the initial prototype). The RISC node architecture is designed to provide low-cost synchronization, storage management, task-management, and latency-hiding mechanisms suitable to the HCLM execution model employed. High-bandwidth interprocessor communications are provided by a through-wafer optical interconnection scheme that permits an elegant implementation of the offset-cube interconnection topology [13]. The optoelectronic network is expected to provide an aggregate I/O bandwidth of 800 Gb/s, while allowing the 4096 PE system to be implemented in a cube 10 cm on a side.

Performance measures for the Pica implementation were obtained via discrete-event simulation and metering tools that allow an arbitrarily configured Pica system to be emulated on a single-processor high-

performance workstation and permit a detailed monitoring of the state of the simulated system. Among the metrics and statistics monitored are task run and wait times, dynamic context storage life-spans, message lengths, message life-spans, number of messages, context cache hit rates, and dynamic instruction utilization. The simulator effectively models a Pica node via its ISA, while it models the interactions between nodes by time-stamping all messages injected into the network and delaying their appearance at a destination queue by a time corresponding to the appropriate topological distance function of the network configuration employed. The performance results obtained through simulation can be deemed as representative of the results expected from the actual physical machine provided the message traffic generated during parallel program execution is shown to be below network saturation. Therefore, to validate our results, the message trace file generated by the instruction-level simulator was fed to a flow-control-digit level simulator of Pica's offset-cube network. The results indicated that the normalized latency (i.e., the ratio of the number of cycles it takes a message to reach its destination under the loaded and unloaded network conditions) for both partitioning schemes is near unity, indicating that the message traffic produced during execution is indeed well below network saturation.

The overall performance of the HCLM ML-EM formulation was evaluated for the reconstruction of a 256×256 PET image from sinogram data acquired across 120 angles; an upsampling value of $U = 4$ was used. A summary of the results for the row-wise and block-wise partitioning schemes is provided in Table I. While the average number of tasks performed per PE (and consequently the average number of messages generated per PE) is very similar for both partitioning schemes ($\sim 213\,000$ tasks), the more load-balanced block-wise partitioning implementation yielded a higher concurrency level and lower execution time. The block-wise partitioning scheme gave a 15% higher concurrency and 35.4% lower execution time than the row-wise partitioning scheme. The low average length of the generated messages (2.50 words) and the low average task run time (~ 52 cycles) are significant, since high message traffic for small, ephemeral tasks is the hallmark of fine-grain, message-passing architectures tailored for a HCLM execution model. For both partitioning schemes an average of ~ 40 contexts were dynamically allocated during program execution (about 15% of the total number of contexts available per PE). The low number of context allocations indicates that the memory available per PE did not constrain the algorithmic implementation. The average message injection rate and the average message length translate to an average requirement of 60 Mb/s (row-wise partitioning) and 90 Mb/s (block-wise partitioning) of I/O bandwidth per PE. During the 90° rotation, however, the average I/O requirements increase to 115 Mb/s and 160 Mb/s for the row-wise and block-wise partitioning schemes, respectively, since all pixels map within the square processing array, resulting in a higher number of messages being injected during the rotational stage, while the number of execution cycles increases only slightly as compared to other angles. (Note that for each angle a PE must calculate the rotated coordinates for all the pixels in its domain before it can determine whether or not the pixel falls within the circular field of view). The average latency (and normalized latency) experienced by any particular message in traveling from its source to its destination was $\sim 6\%$ longer for the row-wise partitioning case; a result of the more uniformly distributed traffic patterns achieved via block-wise partitioning.

The simulated performance results can be used to project the time it would take the physical Pica multiprocessor to reconstruct the clinically-sized PET data-set studied above. Using the fact that up to 8 slices of the data-set can be simultaneously reconstructed (Pica consists of 4096 PE's, and only 512 are required in the reconstruction

TABLE I
SIMULATED PERFORMANCE METRICS (256 × 256
IMAGE, 120 VIEWS, 1 SLICE, 1 ITERATION)

	Row-wise Partitioning	Block-wise Partitioning
Execution Cycles	21,065,915	13,599,260
Concurrency	72.3%	90.9%
Avg. number of tasks per PE	213,428	212,928
Avg. run time per task (cycles)	53.2	52.8
Avg. dynamic memory allocation (contexts)	39.06	41.56
Avg. injection rate per PE (messages/cycle)	0.0101	0.0157
Avg. message length (words)	2.50	2.50
Avg. message latency (cycles)	13.5	12.7
Avg. network normalized latency	1.21	1.12

of one slice), the processing of 32 iterations of the $256 \times 256 \times 120$ views \times 21 slice PET data-set is expected to take approximately 23 seconds if block-wise partitioning is used, or 44 seconds if row-wise partitioning is employed (based on 50 MIPS computational throughput of each PE). These execution times translate to a total speedup of over two orders of magnitude (645 for block-wise partitioning) compared to a workstation-based ray-driven ML-EM formulation. The extrapolation to the processing of multiple slices is valid since our algorithmic formulation maps to two 256-PE processing layers, permitting multiple slices to be reconstructed by simply mapping them to other pairs of processing layers. Since no link resources are shared between the processors evaluating different slices, no penalty is paid in terms of network loading. The extension of the HCLM ML-EM formulation to other upsample values and/or image sizes follows readily, with varying impact on the overall performance that can be achieved. A detailed discussion of these issues and possible extensions of the algorithm to encompass SPECT data-sets can be found in [14].

V. SUMMARY

We have presented and evaluated a highly-parallelized version of the ML-EM algorithm suitable for emerging fine-grain, message-passing multiprocessors. The algorithmic formulation (under two different task and data partitioning schemes) exploits three different levels of parallelism: multiple slices being reconstructed simultaneously, projection and backprojection operations being pipelined, and the rotation operation being cast in a very fine level of parallelism. Simulation results obtained via an instruction-level simulator for a representative system (the Pica multiprocessor) reveal that PET image reconstructions for 21 slices of a 256×120 sinogram data-set can be obtained in less than a minute, which represents over two orders of magnitude speedup compared to current sequential implementations of a ray-driven ML-EM algorithm. Thus, the HCLM ML-EM formulation presented and the simulations performed indicate that fine-grain, message-passing MIMD architectures provide an excellent platform for the incorporation of ML-EM reconstructions into the clinical environment.

REFERENCES

- [1] L. A. Shepp and Y. Vardi, "Maximum likelihood reconstruction for emission tomography," *IEEE Trans. Med. Imag.*, vol. 1, pp. 113–121, Oct. 1982.
- [2] J. Llacer, E. Veklerov, K. J. Coakley, E. J. Hoffman, and J. Nuñez, "Statistical analysis of maximum likelihood estimator images of human brain FDG PET studies," *IEEE Trans. Med. Imaging*, vol. 12, pp. 215–231, June 1993.

- [3] J. Llacer, E. Veklerov, L. R. Baxter, S. T. Grafton, L. K. Griffeth, R. A. Hawkins, C. K. Hoh, J. C. Mazziotta, E. J. Hoffman, and C. E. Metz, "Results of a clinical receiver operating characteristic study comparing filtered backprojection and maximum likelihood estimator images in FDG PET studies," *J. Nucl. Med.*, vol. 34, pp. 1198–1203, July 1993.
- [4] K. Rajan, L. M. Patnaik, and J. Ramakrishna, "High-speed computation of the EM algorithm for PET image reconstruction," *IEEE Trans. Nuclear Sci.*, vol. 41, pp. 1721–1728, Oct. 1994.
- [5] M. S. Atkins, D. Murray, and R. Harrop, "Use of transputers in a 3D Positron Emission Tomograph," *IEEE Trans. Med. Imag.*, vol. 10, pp. 276–83, Sept. 1991.
- [6] C. M. Chen, S.-Y. Lee, and Z. H. Cho, "Parallelization of the EM algorithm for 3-D PET image reconstruction," *IEEE Trans. Med. Imag.*, vol. 10, pp. 513–522, Dec. 1991.
- [7] A. W. McCarthy and M. I. Miller, "Maximum likelihood SPECT in clinical computation times using mesh-connected parallel computers," *IEEE Trans. Med. Imag.*, vol. 10, pp. 426–436, Sept. 1991.
- [8] C. S. Butler and M. I. Miller, "Maximum a posteriori estimation for SPECT using regularization techniques on massively parallel computers," *IEEE Trans. Med. Imag.*, vol. 12, pp. 84–89, Mar. 1993.
- [9] C. S. Butler, M. I. Miller, T. R. Miller, and J. W. Wallis, "Massively parallel computers for 3D single-photon-emission computed tomography," *Phys. Med. Biol.*, vol. 39, pp. 575–582, Mar. 1994.
- [10] W. J. Dally, J. A. S. Fiske, J. S. Keen, R. A. Lethin, M. D. Noakes, P. R. Nuth, R. E. Davison, and G. A. Fyler, "The Message-Driven Processor: A multicomputer processor node with efficient mechanisms," *IEEE Micro*, vol. 12, pp. 23–39, Apr. 1992.
- [11] D. S. Wills, W. S. Lacy, C. Camperi-Ginestet, B. Buchanan, H. H. Cat, S. Wilkinson, M. Lee, N. M. Jokerst, and M. A. Brooke, "A three-dimensional high-throughput architecture using through-wafer optical interconnect," *J. Lightwave Technol.*, vol. 13, pp. 1085–1092, June 1995.
- [12] E. V. R. Di Bella, A. B. Barclay, R. L. Eisner, and R. W. Schafer, "Comparison of rotation-based projectors for iterative reconstruction algorithms," submitted to *IEEE Nucl. Sci. Symp. and Med. Imag. Conf. 1995*.
- [13] W. S. Lacy, J. L. Cruz-Rivera, and D. S. Wills, "A scalable optical interconnection network for ultra-compact 3D computing systems," submitted to *IEEE Trans. Comput.*, Oct. 1995.
- [14] J. L. Cruz-Rivera, E. V. R. Di Bella, D. S. Wills, T. K. Gaylord and E. N. Glytsis, "Parallelized formulation of the Maximum Likelihood-Expectation Maximization algorithm for fine-grain, message-passing architectures," VLSI Architectures Group Tech. Rep. 95-1, 1995.

On the Convergence of an EM-type Algorithm for Penalized Likelihood Estimation in Emission Tomography

Alvaro R. De Pierro

Abstract— Recently, we proposed an extension of the expectation maximization (EM) algorithm that was able to handle regularization terms in a natural way. Although very general, convergence proofs were not valid for many possibly useful regularizations. We present here a simple convergence result that is valid assuming only continuous differentiability of the penalty term and can be also extended to other methods for penalized likelihood estimation in tomography.

Manuscript received March 13, 1995; revised September 25, 1995. This work was partially supported by CNPq Grant 301699/81 and FAPESP Grant 94/0012-6. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was Y. Censor.

The author is with the Applied Mathematics Department, State University of Campinas, CP 6065, CEP 13081-970, Campinas, SP, Brazil.
IEEE Log Number 9416008.