TLM Modeling Using Distributed Computing

P. J. Parsons, S. R. Jaques, S. H. Pulko, and F. A. Rabhi

Abstract—**Distributed computing techniques offer the potential to significantly reduce the run time of transmission-line-matrix (TLM) calculations. This letter describes the implementation of a TLM model of a two-channel waveguide distributed across a network of workstations using parallel virtual machine (PVM). The methods for distributing the TLM matrix across the workstations and the effect on performance of different approaches are described and discussed.**

I. Introduction

The transmission-line-matrix (TLM) method has become a widely used tool for electromagnetic analysis [1], [2]. Distributed computing provides the potential for significantly reducing the run time of TLM computations as the structure of the TLM algorithm is well suited to this approach. A two-channel waveguide excited by a TE profile is modeled.

TLM has previously been implemented on massively parallel SIMD machines [3], [4]. Access to such computers is not always readily available, however networks of workstations are now commonplace and provide a readily available source of interconnected processors. Software exists for harnessing such resources, of which parallel virtual machine (PVM) [5] is one example. PVM has been used previously to implement finite-difference models [6]. Here, we use PVM to distribute TLM computations across a network of workstations, focusing on issues relating to performance rather than on performance itself. In the conclusion, we derive some general guidelines for the implementation of TLM routines on parallel architectures that are applicable to a range of TLM problems.

II. Design

TLM algorithms are iterative and operate on a fixed data space. Communication between the nodes used is regular and takes the form of passing a single pulse value between nearest neighbors in each direction during every iteration, as illustrated in Fig. 1. The figure also shows three possible approaches to distributing the matrix over a number of processors, labeled arrangements A, B, and C. This structure makes distributed implementation particularly attractive. However, there are factors that should be considered when determining the best data distribution to achieve a highly efficient implementation on a given system: the number of messages and the number of TLM pulses communicated in a single iteration of the model.

The number of messages and data contained within each message can be used to calculate the total number of TLM pulses transferred between processes per iteration. For arrangement A, 10 messages each containing 12 pulses, a total of 120 pulses, are transferred. Arrangement B generates 10 messages of six pulses, a total of 60. Arrangement C gives eight messages of three pulses and six messages of four pulses, a total of 48. The computational cost of initiating a message in PVM is significant, and so it is not obvious which arrangement is preferable. However, it can be said that since A and B involve the same number of messages but B transfers fewer pulses than A, B will be more efficient for this array. If the number of processors is now increased to 72, each node in the TLM array will now reside on its own processor. Communication will now involve 252 messages each of one pulse. Although nodal calculations will all be performed in parallel, this arrangement produces a large number of messages and a large volume of data. The processors will therefore spend more of their time communicating and consequently less time performing calculations than in the six-processor implementation.

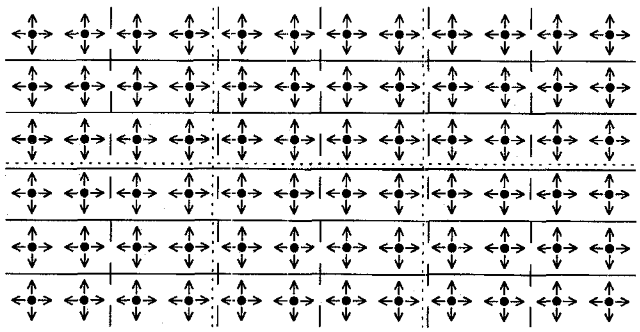

Fig. 1. Example TLM matrix. • TLM node. → Direction of exchanged TLM pulse. ——— Arrangement A (six rows). – – – – Arrangement B (six columns) · · · ·. Arrangement C (two rows by three columns).

III. Implementation

A matrix of shunt nodes was used of size 210 $\times$100 nodes. Pulses incident at the external side boundaries of the waveguide are reflected with opposite sign and at the end boundaries are absorbed. Contiguous areas of the matrix are allocated to individual processors. A master process automatically allocates areas of the matrix to the processors. The master process is also responsible for initiating processes, collecting and collating results, and producing an output file. The slave processes are each responsible for working on the area of the model allocated to them by the master. Each slave calculates its position in the overall model and which processes it has to communicate with. It communicates data directly with adjacent slaves. The slaves are also responsible for generating the required output data for the area of the model on which they are working and sending this data back to the master. The network used comprised

Manuscript received October 10, 1995. P. J. Parsons and S. R. Jaques were supported by Research Studentships from the Engineering and Physical Sciences Research Council of Great Britain.

The authors are with the Departments of Computer Science and Electronic Engineering, University of Hull, Hull HU6 7RX, U.K.

Publisher Item Identifier S 1051-8207(96)02061-2.

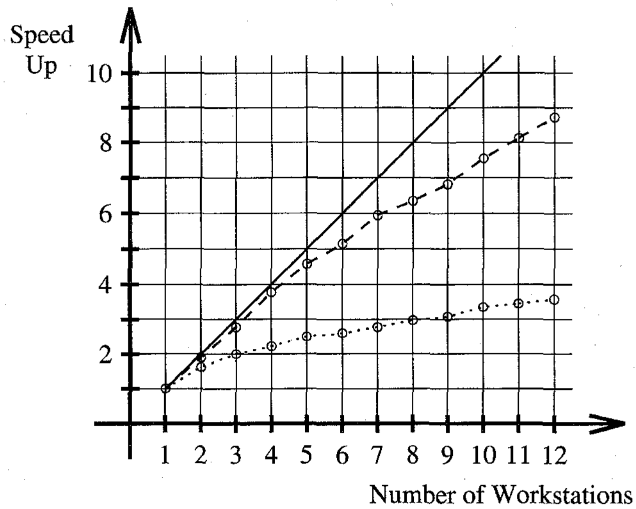

Fig. 2. Speed up attained by dividing along and across the matrix. ——— Linear speed up. ··· Division parallel to long side (Configuration A). – – – Division parallel to short side (Configuration B).

TABLE I
PERFORMANCE FOR DISTRIBUTIONS OVER 12 WORKSTATIONS

Distribution	Messages	Volume of Data	Speed Up
12x1	22	4620	3.58
6x2	32	2300	4.19
4x3	34	1660	5.09
3x4	34	1440	6.00
2x6	32	1420	6.54
1x12	22	2200	8.72

Sun SPARCstation SLC workstations connected by a single ethernet. Each run was compiled using the same compiler and options and the same network, with no other loading on the workstations.

IV. Results

The graph in Fig. 2 shows the speed up when the data space is distributed using arrangements corresponding to A and B in Fig. 1, employing from 1–12 processors. The speed up represents the actual performance improvement, calculated by dividing the time to run the model on a single processor by the run time for a particular distribution.

For each additional processor in configuration A, two additional messages of size 210 pulses are created per iteration. In configuration B, each additional processor corresponds to two additional messages of 100 pulses. Since the computing power used in communication varies with the size of messages as well as the number of messages, it is anticipated that configuration B should be the more advantageous. This is clearly in agreement with the results observed.

Table I presents the results of an investigation into different distributions over 12 processors. The distributions indicate the number of divisions parallel to the long and short sides of the waveguide, respectively; 2×6 indicates two processors parallel to the long side (210 nodes) by six processors parallel to the short side (100 nodes). The results show that the number of messages and the volume of data both affect the speed up attained. It can be clearly seen from these results that the argument derived in Section II (in distributions generating the same number of messages, the one producing the greater volume of data will be less efficient) is satisfied in all cases. In this example, the first three distributions in the table could automatically be discarded as strategies before any implementation is undertaken.

V. Summary and Conclusion

It can be seen from the results that distributing a TLM calculation across a network of workstations can significantly reduce run time. It can also be seen that the distribution of the matrix across the workstations has a significant effect on performance. The example used in this letter illustrates the interdependency of some of the factors that should be considered when determining an efficient distribution for a given model.

Both the generation of a message and the transference of a piece of data require computing time, thus the larger the number of messages and the larger the volume of data, the less efficient the implementation. As initiating messages takes more time than passing a single item of data, a reduction in the number of items tends to be more advantageous than the same reduction in the number of pieces of data transferred; for the case described, the ratio is approximately 100 to 1. However, the relative importance of the number of messages and the volume of data is system dependent. In this example, communication is not point to point, but via a shared network. Here, as the amount of communication increases there is more contention for the network, resulting in the workstations having to wait longer to complete their communications and lowering the efficiency further.

Although the factors determining the efficiency of any particular distribution of a matrix are complex, we comment on the ease with which the number of messages and the volume of data can be derived from the processor network and the TLM matrix size, and how this information can be used to eliminate certain distributions in advance of any trial implementation. Work to derive a metric to allow the prediction of performance for parallel TLM implementations is currently being undertaken; the number of factors involved in such a calculation places it beyond the scope of this initial study.

References

[1] W. J. Hoefer, "The transmission line matrix (TLM) method," in *Numerical Techniques for Microwave and Millimeter Wave Passive Structures*, J. Itoh, Ed. New York: Wiley, 1989, ch. 8.

[2] C. Christopolous, "The transmission line modeling (TLM) method," in *Electromagnetic Wave Theory*. Piscataway, NJ: IEEE Press, 1995.

[3] P. P. M. So, C. Eswarappa, and W. J. R. Hoefer, "Parallel and distributed TLM computation with signal processing for electromagnetic field modeling," *Int. J. Numerical Modeling: Electronic Networks, Devices and Fields,* vol. 8, pp. 169–185, 1995.

[4] C. C. Tan and V. F. Fusco, "TLM modeling using an SIMD computer," *Int. J. Numerical Modeling: Electronic Networks, Devices and Fields,* vol. 6, pp. 299–304, 1993.

[5] A. Geist, A. Begulin, J. Dongarra, W. Jiang, R. Manchek, and V. Sunderam, "PVM 3 user's guide and reference manual," Oak Ridge National Laboratory, May 1994.

[6] V. Varadarajan and R. Mittra, "Finite-difference time-domain (FDTD) analysis using distributed computing," *IEEE Microwave Guided Wave Lett.,* vol. 4, no. 5, pp. 144–145, May 1994.