Title: Performance Analysis of Large-Scale Applications Based on Wavefront Algorithms

Author(s): Adolfy Hoisie
          Olaf M. Lubeck
          Harvey J. Wasserman

Submitted to: HPCCP/CAS Workshop
              8/25-27/98
              NASA Ames Research Center

Los Alamos NATIONAL LABORATORY
Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantees its technical correctness.

Form 836 (10/96)
DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.
DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.
1. Introduction

Wavefront techniques are used to enable parallelism in algorithms that have recurrences by breaking the computation into segments and pipelining the segments through multiple processors [1]. First described as “hyperplane” methods by Lamport [2], wavefront methods now find application in several important areas including particle physics simulations [3], parallel iterative solvers [4], and parallel solution of triangular systems of linear equations [5-7].

Wavefront computations present interesting implementation and performance modeling challenges on distributed memory machines because they exhibit a subtle balance between processor utilization and communication cost. Optimal task granularity is a function of machine parameters such as raw computational speed, and inter-processor communication latency and bandwidth. Although it is simple to model the computation-only portion of a single wavefront, it is considerably more complicated to model multiple wavefronts existing simultaneously, due to potential overlap of computation and communication and/or overlap of different communication or computation operations individually. Moreover, specific message passing synchronization methods impose constraints that can further limit the available parallelism in the algorithm. A realistic scalability analysis must take into consideration these constraints.

Much of the previous parallel performance modeling of software-pipelined applications has involved algorithms with one-dimensional recurrences and/or one-dimensional processor decompositions [5-7]. A key contribution of this paper is the development of an analytic performance model of wavefront algorithms that have recurrences in multiple dimensions and that have been partitioned and pipelined on multidimensional processor grids. We use a “compact application” called SWEEP3D, a time-independent, Cartesian-grid, single-group, “discrete ordinates” deterministic particle transport code taken from the DOE Accelerated Strategic Computing Initiative (ASCI) workload. Estimates are that deterministic particle transport accounts for 50-80% of the execution time of many realistic simulations on current DOE systems; this percentage may expand on future 100-TFLOPS systems. Thus, an equally-important contribution of this work is the use of our model to explore SWEEP3D scalability and to show the sensitivity of SWEEP3D to per-processor sustained speed, and MPI latency and bandwidth on future-generation systems.

Efforts devoted to improving performance of discrete ordinates particle transport codes extended recently to massively-parallel systems [8-12]. Research has included models of performance as a function of problem and machine size, as well as other characteristics of both the simulation and the computer system under study. In [3] a parallel efficiency formula that considered computation only is presented, while [9] a model specific to CRAY T3D communication is developed. These previous models had limiting assumptions about the computation and/or the target machines.
2. Description of Discrete Ordinates Transport

Although much more complete treatments of discrete ordinates neutron transport have appeared elsewhere [12-15], we include a brief explanation here to make clear the origin of the wavefront process in SWEEP3D. The basis for neutron transport simulation is the time-independent, multigroup, inhomogeneous Boltzmann transport equation, in which the unknown quantity is the flux of particles at the spatial point \( r \) with energy \( E \) traveling in direction \( \Omega \).

Numerical solution involves complete discretization of the multi-dimensional phase space defined by \( r, \Omega, \) and \( E \). Discretization of energy uses a “multigroup” treatment, in which the energy domain is partitioned into subintervals in which the dependence on energy is known. In the discrete ordinates approximation, the angular-direction \( \Omega \) is discretized into a set a quadrature points. This is also referred to as the \( S_N \) method, where (in 1D) \( N \) represents the number of angular ordinates used. The discretization is completed by differencing the spatial domain of the problem on to a grid of cells.

The numerical solution to the transport equation involves an iterative procedure called a “source iteration” [13]. The most time-consuming portion is the “source correction scheme,” which involves a transport sweep through the entire grid-angle space in the direction of particle travel. In Cartesian geometries, each octant of angles has a different sweep direction through the mesh, and all angles in a given octant sweep the same way.

Each interior cell requires in advance the solution of its three upstream neighboring cells — a three-dimensional recursion. This is illustrated in Figure 1 for a 1-D arrangement of cells and in Figure 2 for a 2-D grid.

![Figure 1](image1.png)

Figure 1. Dependences for a 1-D and 2-D (diagonal) Transport Sweep (left, middle). Illustration of the 2-D Domain decomposition on eight processors with 2 k-planes per block. The transport sweep has started at top of the processor in the foreground. Concurrently-computed cells are shaded. (right)

3. Parallelism in Discrete Ordinates Transport

The only inherent parallelism is related to the discretization over angles. However, reflective boundary conditions limit this parallelism to, at most, angles within a single octant. The two-dimensional recurrence may be partially eliminated because solutions for cells within a diagonal are independent of each other (as shown in Figure 1).

Diagonal concurrency can also be the basis for implementation of a transport sweep using a decomposition of the mesh into subdomains using message passing to communicate the boundaries [12], shown in Figure 1. The transport sweep is performed subdomain by subdomain in a given angular direction. Each processor’s exterior surfaces are computed by, and received in a message from, “up-
stream'' processors owning the subdomains sharing these surfaces.

However, as pointed out in [9] and [3], the dimensionality of the $S_N$ parallelism is always one order lower than the spatial dimensionality.

Parallel efficiency would be limited if each processor computed its entire local domain before communicating information to its neighbors. A strategy in which blocks of planes in one direction (k) and angles are pipelined through this 2-D processor array improves the efficiency, as shown in Figure 1. Varying the block sizes changes the balance between parallel utilization and communication time.

4. A Performance Model for Parallel Wavefronts

This section describes a performance model of a message passing implementation of SWEEP3D. Our model uses a pipelined wavefront as the basic abstraction and predicts the execution time of the transport sweep as a function of primary computation and communication parameters. We use a two-parameter (latency/bandwidth) linear model for communication performance, which is equivalent to the LogGP model [16]. The demonstrations are sketched only due to space limitations in this abstract.

4.1 Pipelined Wavefront Abstraction

An abstraction of the SWEEP3D algorithm partitioned for message passing on a 2-D processor domain (ij plane) is described in Figure 2. The inner-loop body of this algorithm describes a wavefront calculation with recurrences in two dimensions. Multiple waves initiated by the octant, angle-block and k-block loops are pipelined one after another as shown in Figure 3, in which two inner loop bodies (or “sweeps”) are executing on a $P_x$ by $P_y$ processor grid. Using this abstraction, we can build a model of execution time for the transport sweep. The number of steps required to execute a computation of $N_{sweep}$ wavefronts, each with a pipeline length of $N_s$ stages and a repetition delay of $d$ is given by equation (1).

$$\text{Steps} = N_s + d(N_{sweep} - 1),$$

(1)

The pipeline consists of both computation and communication stages. The number of stages of each kind and the repetition delay per wavefront need to be determined as a function of the number of processors and shape of the processor grid.

FOR EACH OCTANT DO
  FOR EACH ANGLE-BLOCK IN OCTANT DO
    FOR EACH K-BLOCK DO
      IF (NEIGHBOR_ON_EAST) RECEIVE FROM EAST (BOUNDARY DATA)
      IF (NEIGHBOR_ON_NORTH) RECEIVE FROM NORTH (BOUNDARY DATA)
      COMPUTE_MESH (EVERY I,J DIAGONAL; EVERY K IN K-BLOCK;
                  EVERY ANGLE IN ANGLE-BLOCK)
      IF (NEIGHBOR_ON_WEST) SEND TO WEST (BOUNDARY DATA)
      IF (NEIGHBOR_ON_SOUTH) SEND TO SOUTH (BOUNDARY DATA)
    END FOR
  END FOR
END FOR

Figure 2. Pseudo Code for the wavefront Algorithm

Figure 3. Communication (left) and Computation Pipelines.
4.2 Computation Stages

Figure 3 shows that the number of computation stages is simply the number of diagonals in the grid. A different number of processors is employed at each stage but all stages take the same amount of time since processors on a diagonal are executing concurrently. Equation (2) gives the number of computation steps in the pipeline,

$$N_{s,\text{comp}} = P_x + P_y - 1$$  \hspace{1cm} (2)  \\

$$T_{cpu} = \left( \frac{N_x}{P_x} + \frac{N_y}{P_y} + \frac{N_z}{K_b} + \frac{N_d}{A_b} \right) \frac{N_{flops}}{R_{flops}}$$  \hspace{1cm} (3)

and Equation 3 gives the cost of each step with $N_x$, $N_y$, and $N_z$ being the number of grid points in each direction; $K_b$ is the size of the k-plane block; $A_b$ is the size of the angular block; $N_{flops}$ is the number of floating-point operations per gridpoint; and $R_{flops}$ is a characteristic floating-point rate for the processor. The next sweep can begin as soon as the first processor completes its computation so the repetition delay, $d_{\text{comp}}$, is 1 computational step (i.e., the time for completing one diagonal in the sweep).

4.3 Communication Stages

In Figure 6 edges labeled with the same number are executed simultaneously and the graph shows that it takes 12 steps to complete one communication sweep on a 4 x 4 processor grid. One can generalize the number of stages to a grid of $P_x$ by $P_y$:

$$N_{s,\text{comm}} = 2(P_y - 1) + 2(P_x - 1)$$  \hspace{1cm} (4)  \\

$$T_{msg} = t_0 + \frac{N_{msg}}{B}$$  \hspace{1cm} (5)

The cost of any single communication stage is the time of a one-way, nearest neighbor communication given (5). Latency ($t_0$) and bandwidth ($B$), are defined above.

The repetition delay for the communication pipeline, $d_{\text{comm}}$, is 4 because a message sent from the top-left processor (processor 0) to its east neighbor (processor 1) on the second sweep cannot be initiated until processor 1 completes its communication with its south neighbor from the first sweep (Figure 3).

4.4 Combining Computation and Communication Stages

We can summarize the discussion so far in two equations that give the separate contributions of computation and communication:

$$T_{comp} = [(P_x + P_y - 1) + (N_{\text{sweep}} - 1)] \ast T_{cpu}$$  \hspace{1cm} (6)  \\

$$T_{comm} = [2(P_x + P_y - 2) + 4(N_{\text{sweep}} - 1)] \ast T_{msg}$$  \hspace{1cm} (7)

The major remaining question is whether the separate contributions, $T_{comp}$ and $T_{comm}$, can be summed to derive the total time. We have demonstrated that the total time is the sum of eqns. (6) and (7), where $T_{cpu}$ is given by eqn. (3) and $T_{msg}$ is given by eqn. (5). The proof is not presented in this abstract.

5. Validation of the Model

The model was validated with performance data from SWEEP3D on three different machines (SGI Origin 2000, IBM SP2 and Cray T3E), with up to 500 processors, over the entire range of the various model parameters. Inspection of eqns. (6) and (7) leads to identification of the following validation regimes: $N_{\text{sweep}} = 1$: This case validates the number of pipeline stages in $T_{comp}$ and $T_{comm}$, as functions of $(P_x + P_y)$.

$N_{\text{sweep}} \sim (P_x + P_y)$: Validation of a case where the contributions of the $(P_x + P_y)$ and $N_{\text{sweep}}$ are comparable.
This case validates the repetition rate of the pipeline.

For each of these three cases, we analyze problem sizes chosen in such a way as to make:

\[ T^{\text{comp}} >> T^{\text{comm}}, \quad (\text{validate eqn. (6) only}) \]

\[ T^{\text{comp}} = 0; \quad (\text{validate eqn. (7) only}) \]

\[ T^{\text{comp}} \sim T^{\text{comm}}, \quad (\text{validate the sum of eqns. (6) and (7)}) \]

The agreement of the model with the measured data is very good in all cases (not presented here).


ASCI is targeting a 100-TFLOPS system in the year 2004, with a workload defined by specific engineering needs. In this section we apply our model to predict the machine parameters under which the runtime goal might be met. We assume a 100-Tflops-peak system composed of about 20,000 processors (based on an extrapolation of Moore's law).

Three sources of difficulty with such a prognosis are (1) making reasonable estimates of machine performance parameters for future systems; (2) managing the SWEEP3D parameter space (i.e., block sizes); and (3) estimating what problem sizes will be important. We handle the first by studying a range of values covering both conservative and optimistic changes in technology. We handle the second by reporting results that correspond to the shortest execution time (i.e., we use block sizes that minimize runtime). We handle the third as follows. For particle transport, one ASCI target problem involves \( O(10^5) \) mesh points, 30 energy groups, \( O(10^4) \) time steps, and a runtime goal of about 30 hours. On 20,000 processors the resulting subgrid size is approximately \( 6 \times 6 \times 1000 \). In a different ASCI scenario, particle transport problem size is determined by external factors. Based on [17], such computations will involve smaller grid sizes (20 million cells) on the full machine. The 20 million-cell problem would utilize a \( 2 \times 2 \times 250 \) subgrid.

6.1. The 1 billion-cell problem

Plots showing dependence of runtime with sustained processor speed and MPI latency are shown in Figure 4 for several k-plane block sizes, using optimal values for the angle-block size. Table 1 collects some of the modeled runtime data for a few important points: sustained processor speeds of 10% and 50% of peak, and MPI latencies of 0.1, 1, and 10 microseconds. Our model shows that the dependence on bandwidth is small, and as such no sensitivity plot based on ranges for bandwidth is presented. All results assume 400 Mbytes/s MPI bandwidth [18].

We note that runtime under the most optimistic technological estimates in Table 1 is larger than the 30-hour goal by a factor of two. The runtime goal could be met if, with these values of processor speed and MPI latency, we used what we believe to be an unrealistically high bandwidth value of 4 GBytes/s.

Assuming a more realistic sustained processor speed of 10% of peak, Table 1 shows that we miss the goal by about a factor of six even when using 0.1 \( \mu \text{s} \) MPI latency. With the same assumption for processor speed, but with a more conservative value for latency (1 \( \mu \text{s} \)), the model predicts that we are a factor of 6.6 off. Our results show that the best way to decrease runtime is to achieve better sustained processor performance. This is a result of the relatively low communication/computation ratio that our model predicts. For example, using values of 1 \( \mu \text{s} \) and 400 MB/sec for the MPI latency and bandwidth, and a sustained processor speed of 0.5 GFLOPS, the communication time will only be 20% of the total runtime.
Figure 4. Left: Model-projected sensitivity of the billion-cell problem to MPI latency on a 100-Tflops system. Sustained CPU speed = 500 Mflops, B = 400 Mbytes/s. Right: Model-projected sensitivity of the billion-cell runtime to sustained processor speed on a hypothetical 100-Tflops. Latency=15us, B=400 MB/s.

| Table 1. Estimates of SWEEP3D Performance on a Future-Generation System as a Function of MPI Latency and Sustained Per-Processor Computing Rate |
|---------------------------------|---------------------|---------------------|
| **MPI Latency**                | **10% of Peak**     | **50% of Peak**     |
| 0.1 μs                         | 180                 | 16%                 |
| 1.0 μs                         | 198                 | 20%                 |
| 10 μs                          | 291                 | 20%                 |
| Amount of Communication        |                     |                     |
| 0.1 μs                         | 56                  | 52%                 |
| 1.0 μs                         | 74                  | 54%                 |
| 10 μs                          | 102                 | 58%                 |

6.2. The 20 million-cell problem

The model predicts that communication time ranges from one-half the total time to two-thirds of the total time depending on specific values for the latency and processor speed. The contribution of the bandwidth to the communication cost is, again, negligible. For this problem size latency and processor speed are equally important in decreasing the runtime. Actual plots are not presented here.

7. Summary

We introduced a performance model for parallel, multidimensional, wavefront calculations with machine performance characterized using the LogGP framework. The model accounts for overlap in the communication and computation components. The agreement with experimental data is very good under a variety of model sizes, data partitionings, blocking strategies, and on three different parallel architectures. Using our model, we analyzed performance of a deterministic transport code on a hypothetical 100 Tflops future parallel system of interest to ASCI.

8. Acknowledgements.

We acknowledge the use of resources at the Advanced Computing Laboratory, LANL, and support from the U.S. DOE under Contract No. W-7405-ENG-36. We thank SGI/CRAY for a grant of computer time on the CRAY T3E system. We acknowledge the use of the IBM SP2 at the LLNL.

9. References.

Please contact any of the authors for the complete reference list.