Atmospheric Modeling on a SIMD Computer

by

M. J. Suarez
Goddard Space Flight Center/611
Greenbelt, MD 20771

As technical limits are approached on the speed of processors used in super-computers, we may expect that greater parallelism will be sought to enhance their power. The parallel execution of varied tasks is to some degree used in all modern mainframes. In the new class of parallel machines, however, parallelism is a central design concept, and its presence is much more apparent to the user. Efficient use of a two processor XMP, for example, to run a single, large task such as an atmospheric GCM, is a very different problem than that of running the same task on its predecessor, the CRAY-1. As the number of processors increases from two to say eight or sixteen, we cannot expect the total computing power to increase proportionably, but the difficulty of programming probably will. Furthermore, the increase in computational speed we get from additional processors will depend both on our task (and how we organize it) and on the organization of the machine itself. It thus behooves us at these early stages in the development of parallel super-computers to consider how well our models could be made to perform on various computer organizations. At Goddard Space Flight Center we started a project to explore the possibilities of using machines like Goddard's Massively Parallel Processor (MPP) in atmospheric modeling.

In his landmark paper on computer organization, Flynn (1972) discussed two types of parallel processors that he termed: single instruction stream/multiple data stream (SIMD) and multiple instruction stream/multiple data stream (MIMD). In a MIMD each processor can work on a different set of instructions and access different data. Problems in its design and use

revolve around the assignment and coordination of tasks among the processors and the avoidance of conflicts in their access to data. Most parallel supercomputers produced or planned are of this type.

The MPP is an example of the SIMD organization. At any one time all its processors are executing the same instruction, or doing nothing. Each processor accesses different data, kept in local memory. But addressing, like any other instruction, is done with all processors in lock-step, so they all work on the same portion of the locally held data. Because of their relatively simple organization SIMD machines can be built of a very large number of processors (the MPP consists of 16K processing elements (PEs).) The rigidity of their lock-step execution of instructions, however, makes them suitable only for the most parallel of tasks.

It is clear that the wide range of computing tasks encountered in the atmospheric sciences requires that large, general purpose computers be available at major centers. It is unlikely SIMD machines will ever be able to meet this need. However, a large fraction of our computing resources is devoted to a single large task: the execution of high resolution general circulation and weather prediction models. Since these models are used repetitively for years in both research and operational forecasting, one can afford to devote considerable effort to optimizing their codes. The question underlying our discussion is: Should one also consider the optimization of the organization of the computers on which they are run? Judging by the attention given to the question, the answer has been - no, it is preferable to devote our attention to improving and adjusting our codes to the best available general purpose computers. It is our starting premise that increased availability and use of parallelism could alter this answer. In the rest of this note we give a brief discussion of this qustion. We will identify - probably naively- MIMD with general purpose and SIMD with specialization, since the latter is the simplest and most economical way

of achieving a large degree of parallelism and produces a consequent decrease in generality.

The main argument against considering a specialized organization is that it could be justified only for our largest tasks, general circulation models (GCMs). Because these attempt a comprehensive description of atmospheric processes, they involve most of the tasks of the field as a whole; consequently they could not be made sufficiently parallel to make efficient use of the SIMD specialization. In particular these models, in addition to solving discrete versions of the equations of motion, deal with a number of sub-grid scale procesess such as radiative transfer through cloudy atmospheres, condensation, and small scale turbulent and convective motions. Because existing parameterizations for these processes are highly conditional (do this if there is a cloud, otherwise do that) they will be very inefficient in a highly parallel SIMD machine. Two questions arise: How punitive are such "non-parallel" tasks? and how much may their formulation be modified to make them more amenable to parallel calculation? We will devote our attentive to the first and easier of the two, assuming (conservatively for the SIMD case) that the answer to the second is - not much.

A good starting model for the behavior of a task as the number of processors working on it is increased is "Amdahl's assertion". It states, in effect, that every task contains some fixed fraction of critical work that while being performed requires all other subtasks idle, awaiting either access or results. If this fraction is denoted by f, then the time required to perform the task with n processors, given in units of the time required to perform it with a single, identical processor is

$$t = \frac{(1 - f)}{n} + f \quad ,$$

and its inverse, the speed-up, is

$$t^{-1} = \frac{n}{(1 - f) + fn} \qquad (1)$$

For small n speed increases linearly, while at large n it saturates at a value of f^{-1} times the speed of the individual processors. The choices this model's assumptions offer is to either effectively eliminate critical tasks or use the complexity allowed the machine for a few fast processors. Less pessimistic results require relaxing the assumption of a fixed, single-tasking fraction.

To be definite we consider a simple heuristic model of a GCM. We assume that its tasks fall in three categories: local or nearly local unbranched calculations, such as those used in updating the equations of motion at grid points; nonlocal unbranched calculations, such as computing zonal FFT's; and local branched calculations, such as those used in sub-grid scale processes.

Machines like the MPP -- a SIMD with a nearest neighbor network -- are ideal for the local unbranched tasks of grid-point models, provided the model's horizontal grid can be mapped onto the processors array and its vertical grid fits into each processing element. In this case the only overhead is in accessing a near neighbor's data. The rest of the time is spent doing arithmetic and accessing local memory. If as n increases the grid is made finer or the mapping more sparse (fewer grid points per processor); the calculations per unit time increases like n and the overhead remains fixed, so

$$t^{-1} \sim n.$$

This is the ideal case.

As an example of non-local unbranched calculations we take an FFT along one direction of the array of processors. This would be the first step in solving a Poisson equation on a sphere with a latitude longitude mapping onto the array and the transform taken in the zonal direction. If for simplicity we take a square array, and again assume the grid is made finer with larger n, the result for large n is

$$t^{-1} \sim \sqrt{n} \log n \quad . \tag{2}$$

This is less than ideal but still an attractive speed-up, and suggests that non-local operations such as the solution of elliptic equations need not be a serious impediment to SIMD calculations. We note, however, that the proportionality constant in (2) would depend on the speed with which near neighbor transfers can be made compared to the arithmetic speed of the processors.

The third case, branched local calculations, are the most punitive to the SIMD machine. Flynn (1972) analyzed this case by considering a branched structure with equal probability of having to execute either side of the branch. With this assumption the number of processors active at branching level p is

$$n_p = \frac{n}{2^p} ,$$

and beyond level $p = \log_2(n) - 1$, processors are executing sequentialy. If f_p is the fraction of the calculations done at each level, the time required in a SIMD organization is

$$t = 1 + \sum_{p=o}^{\log_2(n)-1} f_p \left[\frac{1}{n_p} - 1 \right] , \; n > 1. \tag{3}$$

The speed-up is then crucially dependent on the choice of fp, which depends on the nature of the task. If the branching is only a few levels deep the summation would be terminated after these terms and we would have

$$t = \sum_{p=0}^{p1} \frac{f_p}{n_p} \quad .$$

Two idealized cases considered by Flynn serve to illustrate possible behaviors of the speed-up. In the first, the fraction of the work at successively deeper levels of branching is assumed to decrease exponentially, just matching the decrease in the number of processors working.

$$f_p = \frac{1}{2p+1} ,$$

that is, half the operations are fully parallel, one quarter are at the first level, and so on. This circumvents Amdahl's assertion by making the single-tasking fraction decrease as n increases, the speed-up for large n is then

$$t^{-1} \sim \frac{n}{\log (n)} \qquad (4)$$

this is probably the weakest dependence for which a large degree of parallism should be considered. The second case is to assume that an equal fraction of the work must be done at each level. The asymptotic speed up is then

$$t^{-1} \sim \log (n) \qquad (5)$$

this behavior is called Minsky's conjecture, and a task that exhibits it is clearly unsuitable for highly parallel calculations.

Where does this leave us? First the dynamics calculations should be amenable to large parallelism, for both local and non-local processes. If the branched fraction in the physics behaves like (5) and cannot be modified, we will be restricted to using a few fast processors. If on average it behaves like (4), however, then, depending on how economically processors can be replicated, a fairly large n may be optimal.

Reference:

Flynn, M. J., 1972. Some computer organizations and their effectiveness. IEEE Transactions on Computers, C-21, 948-960.