SLALOM: Surviving Adolescence

by

John Gustafson, Diane Rover, Stephen Elbert, and Michael Carter
Ames Laboratory, Ames, Iowa 50011

Introduction

Benchmarks have a way of taking on lives of their own, and SLALOM is in the throes of puberty. In last month's *Supercomputing Review*, Petter Bjørstad and Erik Boman showed the conjugate gradient method to be a better way of solving the system of equations in SLALOM, with spectacular improvements in the problem size one can solve in one minute. These are taken in stride by the flexible rules of SLALOM, which specify a problem to solve and not a particular algorithm. SLALOM accepts algorithm changes, and is not locked to methods that might become obsolete or unrepresentative of mainstream scientific computing. This reflects real-life use of computers: Algorithms change. Languages change. The challenge, for us, is to continue to make available every improvement that people discover, and to avoid unfair comparisons.

Shrinking SLALOM's Appetite for Storage

When Bjørstad and Boman first communicated their work to us, we began a furious effort to reduce the memory requirements of SLALOM so that computers would still be able to stay busy for one minute without using secondary storage. (Their 8,192-processor MasPar ran out of memory with a 34.5-second run.) The result is a radical change to the preferred method for SLALOM; the matrix is no longer explicitly constructed, but represented by tables of unique patch couplings. A problem of size n patches used to take order n^2 storage, but now takes only order $n^{1.5}$ storage. This solves the problem mentioned in last month's issue: that SLALOM might become a test of memory management instead of arithmetic performance. Also, it allows computers to make good use of data caches. Computers of all sizes, from laptops to supers, now benefit from the new problem setup and solver combination.

The storage reduction is made possible by slightly changing the way walls are divided into patches. The new decomposition method cuts the room's walls into patches by passing three sets of parallel planes through the room, one set along each coordiante axis, subject to the constraint that no patch can be more than twice as wide as it is high. The result is a set of three numbers which represent the number of patches along each coordinate axis. (see Figure 1).

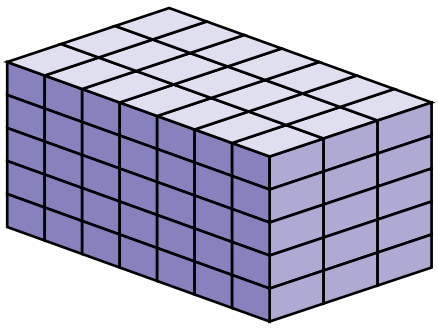

Figure 1. New SLALOM patch decomposition (regular grid, no staggering)

Some values of *npatch* become inaccessible due to this decomposition, but most even numbers are still valid. The uniform grid on each face of the room stemming from this decomposition gives rise to a great deal of regularity in the patch coupling matrix. This regularity, in turn, suggests a compact storage scheme that reduced memory requirements from order n^2 to order order $n^{1.5}$ for orthogonal patch couplings, and order n for parallel patch couplings (see Figure 2).

HISTORICAL PERSPECTIVE

The Incredible Shrinking Exponent

The change to a preconditioned conjugate gradient method is one of a long and continuing series of algorithmic improvements to SLALOM. SLALOM's evolution has proceeded through several stages in its 15 months. Here, we recount some significant events leading up to SLALOM's development, and notable changes which have been made since then.

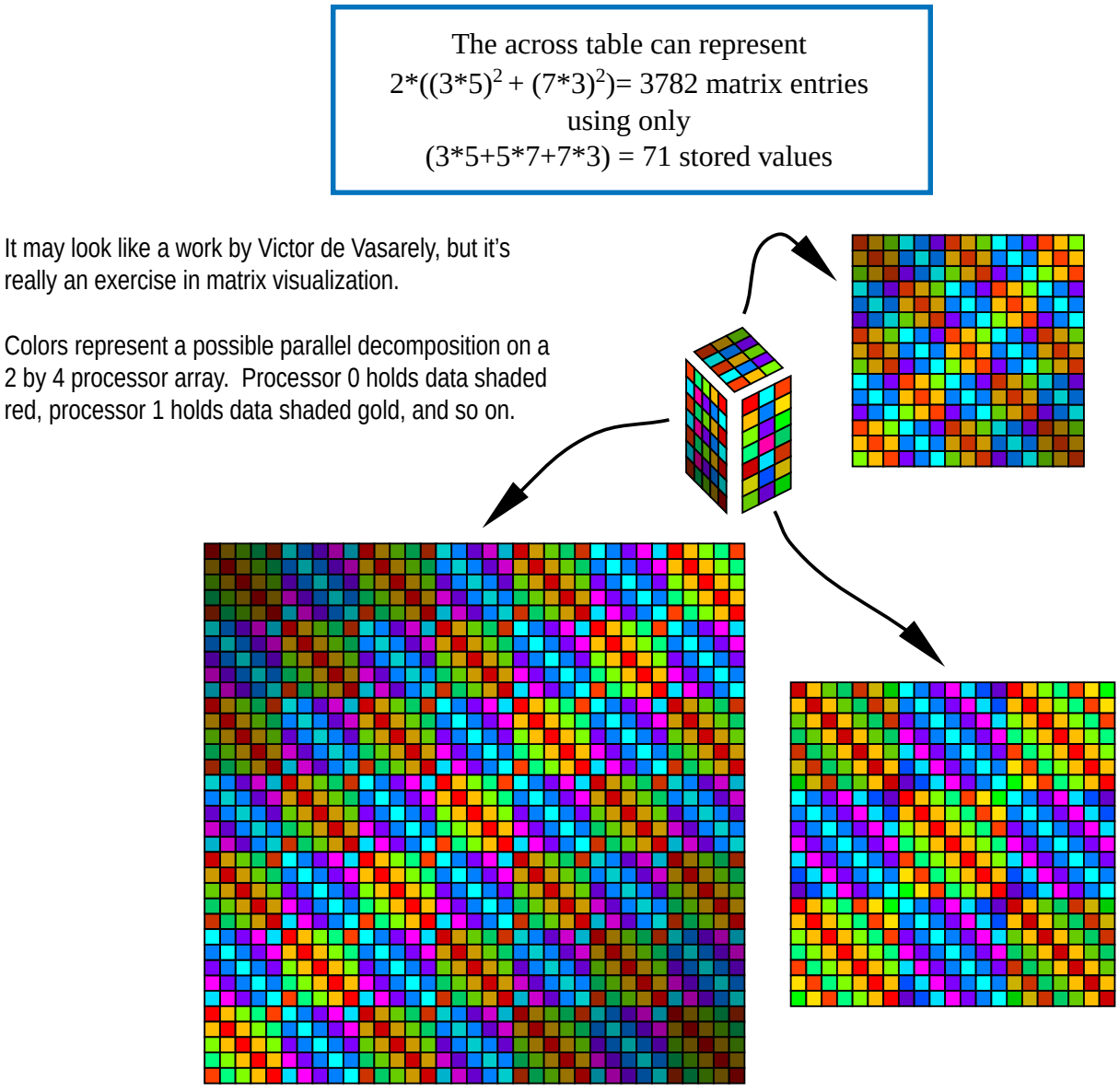

Figure 2. Matrix visualization

In his original 1984 paper, Goral [2] described how a complete solution to light transport in a closed room could be obtained by constructing a matrix of *form factors*, and solving the resulting linear system. No performance results were given. However, the matrix solver was assumed to be asymmetric and to require partial pivoting. The form factors were approximated by Gaussian quadrature instead of computed from exact expressions. The cost of solving the equations war order n^3.

Cohen [1] in 1985 published a paper that described the *hemi-cube* technique for approximating form factors between arbitrary patches, and used an iterative Gauss-Seidel solver. A room composed of 1,740 patches took 180 minutes to set up and 10 minutes to solve on a VAX 11/780 using this technique.

In designing SLALOM, we sought a real application for which the best known serial method would resemble LINPACK. Radiosity offered a nearby dense system, since every patch in a room interacts with every other patch, except those on the same wall. But with every improvement we've discovered, SLALOM looks less and less like LINPACK. When we originally conceived of the idea of using a radiosity solver as a computer benchmark, we noted that the form factor matrix could be made symmetric by dividing each row by area of its corresponding patch. This symmetry can be exploited effectively in a direct solver. As originally implemented, SLALOM used a direct solver to protect against worst-case behavior in highly reflective environments.

Jeff Brooks of Cray Research was the first to apply a blocked solver using Strassen matrix multiplication to reduce the solver complexity from order n^3 to order $n^{2.8}$. The improvement, which only affected the two or three fastest machines on the list, occurred about February 1991.

Bjørstad and Boman of the University of Bergen then hypothesized that the conjugate gradient method of linear system solution would still be faster than direct solvers for all cases allowed by SLALOM. So it is! It is so fast, in fact, that virtually all computers run out of main memory long before they reach a one-minute run if the patch coupling matrix is instantiated! It has been shown that the conjugate gradient method converges in order n^2 time, thus lowering the solver exponent again. (For fewer than about 50 patches, direct solvers still use fewer operations than conjugate gradient for the SLALOM problem.)

This situation motivated us to find a way of reducing memory requirements. As necessity is the mother of invention, the conjugate gradient method is the progenitor of the new patch decomposition method and compact form factor storage tables (see Figure 2).

Of late, we have been informed by James Shearer of IBM that is it possible to implement the matrix-vector multiply kernel of conjugate gradient as a convolution operation using FFTs. This further reduces the solver complexity to order $n^{1.5} \ln n$. Dave Schneider, of CSRD, has suggested that the multipole technique might apply to SLALOM, someday bringing the complexity down to $n^{1.5}$. These further improvements will put a new emphasis on the task of writing out the answer file, previously a near-negligible part of the one-minute run. The algorithmic improvements to SLALOM in only one year of its existence are equivalent to perhaps a decade of improvements one might achieve from hardware alone.

Are Direct Solvers Dead?

Our prejudice was much like that of other computer users we've spoken with: "Iterative solvers have their place, but for dense systems of equations, you still want a direct solver." Now we're wondering if there is any place for LINPACK-type methods. For nearly all real applications, LU factorization seems to us as dead as Cramer's Rule and Bubble Sort. Yet, it remains the focus of supercomputer designers and purchasers! We also thought that making the system only weakly diagonally dominant (using highly reflective faces) in SLALOM would

bring iterative methods to their knees. Not so. We suspect direct solvers are still the method of choice when there are many right-hand sides to solve against, more than perhaps 10 percent of the size of the matrix. If anyone can show us a real problem for which a LINPACK-type method is the best method, we's like to hear about it.

THE REPORT

Changes to the July SR Performance List

Instead of the usual list of computers we've published in the past, here are the top 10 computers for the older SLALOM method. The NEC and 484-processor Intel entries are new; the 256-processor and 64-processor Intel entries have improved slightly.

Table 1 shows the top 10 computers with the old version, including some changes since the July publication of SLALOM. The NEC SX-3/14 now has the highest performance using the direct solver version of SLALOM. We look forward to comparing the high-end computers using the new version soon.

TABLE 1

TOP 10 SLALOM ENTRIES (DIRECT SOLVER ONLY)

Machine, environment	Processors	Patches	MFLOPS	Measurer	Date
NEC SX-3/14, 350 Mhzf77sx (pi 'solver:mxv'...)	1	6011	3344	Y. Kobayashi (v)NEC	11/1/91
Intel Delta (i860) 40 MHzFortran+coded Ddot	484	5750	2957	E. Kushner (v)Intel	7/3/91
Siemens S600/20, 312 MHzFortran 77+LAPACK	1	5610	2727	A. Rohnfelder (v)Siemens	4/22/91
Cray Y-MP8D, 167 MHzFortran+LAPACK (Strassen)	8	5120	2130	J. Brooks (v)Cray Research	9/21/90
Intel Delta (I860) 40 MHzFortran+coded Daxpy	256	4320	1260	E. Kushner (v)Intel	6/12/91
Cray-2S/4, 244 MHzFortran+LAPACK (Strassen)	4	4204	1160	M. Ess (v)Cray Computer	5/27/92
Cray Y-MP8D, 167 MHzFortran+LAPACK (Strassen)	4	4096	1190	J. Brooks (v)Cray Research	9/21/90
nCUBE 2, 20 MHzFortran+assembler	1024	3736	821	J. Gustafson,Ames Lab	2/8/91
Intel Delta (I860) 40 MHzFortran+coded Ddot	64	3420	639	E. Kushner (v) Intel	6/12/91
Cray-2S/4, 244 MHzFortran+LAPACK (Strassen)	2	3280	560	M. Ess(v)Cray Computer	5/27/92

Notes

A "(v)" after the name of the person who made the measurement indicates a vendor.Vendors frequently have access to compilers, libraries and other tools that make the performance higher than that achievable by a customer.

The measurements above use the old method, and hence the old operation count to estimate MFLOPS. The MFLOPS should not be compared with thoseachieved by the new iterative method!

The FLOPS Are Down, but the Performance Is Up: the New Algorithm

The new method brings up the point we've been making in all these articles: MFLOPS ratings are a misleading goal. With the new method, many computers have lowered their "MFLOPS" ratings, where we count the operations of the best known method. But who cares? The computers are running bigger problems than ever! Progress is supercomputing can happen with no benefit from FLOPS improvement.

Here are some measurements for single processors. The parallel versions of the new algorithm are still under development, and we await figures for very high-end computers. This list is admittedly bottom-heavy for a magazine devoted to supercomputing! However, the slower machines show the benefit of the new algorithm. All runs are close to 60 seconds.

TABLE 2

Slalom Performance Using The New Algorithm Patches

Machine	New Method	Old Method	MFLOPS	Measurer	Date
IBM RS/6000 550, 41.6 Mhz xlf-0	4252	1610	22.5	S. Elbert, Ames Lab	11/5/91
Intel iPSC/860, 40 Mhz, if77 2.0 -02	2128	647	5.42	E. Kushner (v), Intel	10/28/92
Silicon Graphics 4D/380S, 33 Mhz, R3000, C	2048	700	5.02	S. Elbert, Ames Lab	10/22/91
DECStation 5000/200, 25 MHz R3000, C -0	1742	534	3.66	M. Carter, Ames Lab	10/22/91
DECStation 5000/120, 20 Mhz R3000, C -0	1534		2.81	M. Carter, Ames Lab	10/22/91
Silicon Graphics 4D/25, 20 Mhz R3000, C -0	1390	507	2.33	S. Elbert, Ames Lab	10/13/91
SUN 4/370, 25 Mhz C (ucc -dalign -fast ...)	1248	451	1.86	M. Carter, Ames Lab	10/14/91
DECStation 2100, 12.5 Mhz, C (cc -0)	1192	377	1.71	M. Carter, Ames Lab	10/14/91
NeXT Cube '040, 68040, 25 Mhz, gnu C ...	1080		1.39	M. Lades, Inst. Füer Neur.	10/21/91
nCUBE 2, 20 Mhz, C (ncc -0)	792	354	0.761	J. Gustafson, Ames Lab	10/17/91
Mac IIfx, (40 Mhz 68030+68882) C (-opt full ...)	518	235	0.325	R. Zurcher, ISU	11/04/91
VAXStation 3520, C (-0)	434	181	0.234	J. Gustafson, Ames Lab	10/17/91
Amiga 2500/30 (25 Mhz 68030+68882), C ...	396		0.196	R. Bless, U. Karlsruhe	10/23/91
Mac IIci (25 Mhz 68030+68882) Think C	334	190	0.143	J. Gustafson, Ames Lab	10/17/91
Mac SE/30, (16.7 68030+68882) C ...	328	163	0.124	R. Zurcher, ISU	11/04/91
Mac IIsi (20 Mhz 68030+68882) Think C	304	175	0.116	J. Gustafson, Ames Lab	10/13/91
Mac IIsi (20 Mhz 68030 only) Think C	102	73	0.0154	J. Gustafson, Ames Lab	10/17/91

We Reserve the Right ...

We use the same input data (the "geom" file) for all computers to get a fair comparison, but the benchmark must still be treated as thought *any input data within the prescribed ranges might be used*. So far, everyone has adhered to the spirit of the rules and not exploited potential loopholes in wording. We simply wish to avoid tuning performance to a particular input file.

SLALOM's New Address

The SLALOM distribution Internet address has changed to **tantalus.scl.ameslab.gov** (IP address 147.155.32.1).

The directories

/pub/Slalom/Source/Serial/C
/pub/Slalom/Source/Serial/Fortran77

contain versions of SLALOM that use the new setup and solver. We are in the process of updating the data parallel, message-passing and shared-memory parallel versions tomatch the new serial method. Please send your results, suggestions, and comments to **slalom@tantalus.scl.ameslab.gov**

For those who do not have access to "anonymous ftp," send e-mail to **netlib@tantalus.scl.ameslab.gov** for instructions on how to get SLALOM files sent to you automatically.

Acknowledgements

We have Petter Bjørstad, Erik Boman, and James Shearer to thank for many of the ideas that now form the current version of the SLALOM benchmark. We are, as always, grateful for the efforts of many individuals who have contributed the performance data in our reports.

References

[1] M. Cohen and D. Greenberg, "The Hemi-Cube: A Radiosity Solution for Complex Environments," *Computer Graphics*, Vol. 19, No. 3, 1985, pp. 31-40.

[2] C. M. Goral, K.E. Torrance, D. P. Greenberg and B. Battaile, "Modeling the Interaction of Light Between Diffuse Surfaces," *Computer Graphics*, Vol. 18, No. 3, July 1984.