Properties of a Family of Parallel Finite Element Simulations

David R. O'Hallaron and Jonathan Richard Shewchuk

December 23, 1996
CMU-CS-96-141

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This report characterizes a family of unstructured 3D finite element simulations that are partitioned for execution on a parallel system. The simulations, which estimate earthquake-induced ground motion in the San Fernando Valley of Southern California, range in size from 10,000–1,000,000 nodes and are partitioned for execution on 4–128 processors. The purpose of the report is to help researchers better understand the properties of unstructured tetrahedral finite element meshes and the sparse matrix vector product (SMVP) operations that are induced from them. The report is designed to serve as a comprehensive reference that researchers can consult for answers to the following kinds of questions: For a tetrahedral mesh with a particular number of nodes, how many elements and edges does it have? What is the distribution of node degrees in a tetrahedral mesh? What fraction of nodes in a partitioned mesh are interface nodes? What is the communication volume in a typical parallel SMVP? How many messages are there? How big are the messages? How many nonzeros are contained in the rows of a sparse matrices induced from tetrahedral meshes? The partitioned meshes described in the paper are available electronically.

Supported in part by the National Science Foundation under Grant CMS-9318163, by the Advanced Research Projects Agency/CSTO monitored by SPAWAR under contract N00039-93-C-0152, and by a grant from the Intel Corporation. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the U.S. government. Authors' email addresses: droh@cs.cmu.edu, jrs@cs.cmu.edu

Keywords: unstructured finite element meshes, parallel computing, sparse matrix

Contents

About this Report

An electronic copy of this report and the meshes described herein can be obtained through the Web page `http://www.cs.cmu.edu/˜quake/meshsuite.html`.

1 Introduction

The multiplication of a sparse matrix by a dense vector is central to many computer applications that simulate physical systems. However, the run-time properties of sparse matrix-vector product (SMVP) operations are poorly understood by researchers. For example, conventional wisdom holds that SMVP operations are communication intensive. However, when we measure the computation/communication ratios in realistic sparse codes, we find that these ratios can be quite large, as high as 50:1 even for simulations partitioned across 128 processing elements (PEs).

One reason for the generally poor understanding of SMVP operations is that performance depends heavily on the nonzero structure of the sparse matrix, and this structure depends on the physical system being simulated. Without access to real physical simulations, it is impossible to create credible SMVP test cases. The same is not true of dense matrix operations, where performance is independent of the data.

This report describes the properties of four unstructured tetrahedral finite element meshes partitioned for execution on 4, 8, 16, 32, 64, and 128 PEs. The meshes come from finite element simulations of earthquake-induced ground motion in the San Fernando Valley [1] and are partitioned using a recursive geometric bisection algorithm [7, 8]. Because the simulations model earthquakes, we refer to them as the *Quake simulations* and their corresponding meshes as the *Quake meshes*.

Our purpose is to help researchers better understand the properties of unstructured tetrahedral finite element meshes and the SMVP operations that are induced from them. The report is designed to serve as a comprehensive reference that researchers can consult for answers to the following kinds of questions: For a linear tetrahedral mesh with a particular number of nodes, how many elements and edges does it have? What is the distribution of node degrees in a tetrahedral mesh when the mesh is partitioned among multiple processors? What fraction of nodes are interface nodes? What is the communication volume in a typical parallel SMVP? How many messages are there? How big are the messages? How many nonzeros are in the rows of a sparse matrix induced from a tetrahedral mesh?

Section 2 describes the Quake meshes and their corresponding simulations. Section 3 details the basic structural properties of the Quake meshes. Sections 4 and 5 describe the communication and computation properties of SMVP operations that are induced from the Quake meshes.

2 The Quake simulations

There are four Quake simulations, denoted sf10, sf5, sf2, and sf1. The "sf" in the names is an abbreviation for San Fernando. The digit in the names indicates the highest frequency wave (in seconds) that the simulation is able to resolve. For example, sf10 resolves waves with 10 second periods, sf5 resolves waves with 5 second periods, and so on. Each program simulates 60 seconds of shaking as shock waves travel through a model of the San Fernando Valley. Each model employs a three-dimensional unstructured finite element mesh composed of thousands or millions of tetrahedra (i.e., pyramids with triangular bases). The mesh for sf10 is illustrated in Figure 1. The model corresponds to a volume of earth roughly 50 km x 50 km x 10 km. Beverly Hills is in the lower right-hand corner The town of San Fernando is in the midst of the darkly shaded region near the upper left corner.

Each tetrahedron in Figure 1 is called an *element*, and the vertices of the tetrahedra are called *nodes*. Some finite element simulations use structured meshes constructed from regular grids; however, the Quake simulations require *unstructured* meshes, which can accommodate the wildly varying densities of the soils

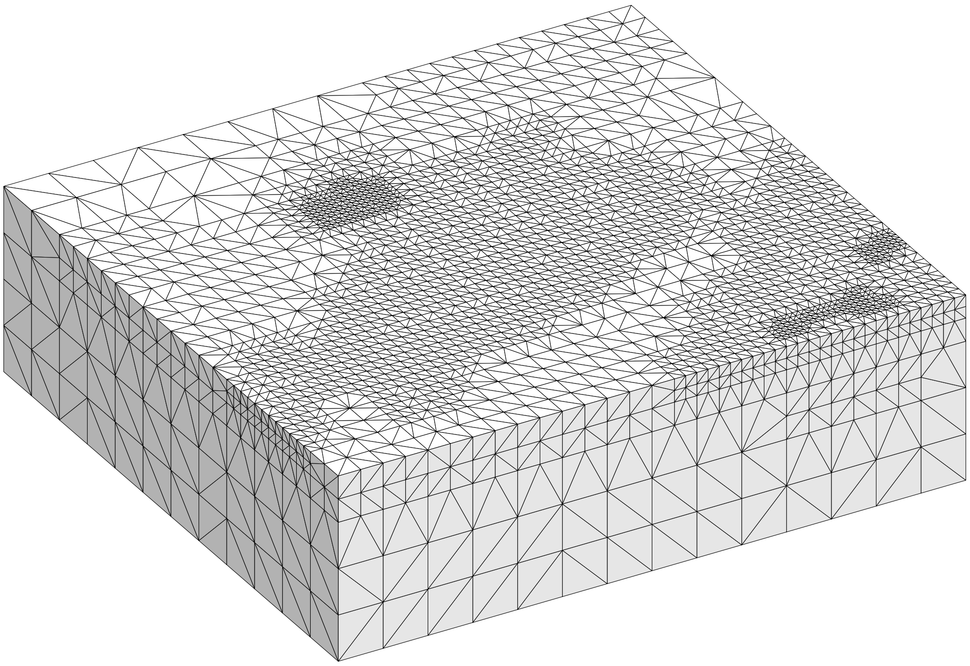

Figure 1: Finite element mesh for the sf10 model of the San Fernando Valley (side view).

in the valley. Wavelengths are shorter in softer soils, thus softer soils need a higher density of nodes and elements. The unstructured nature of the meshes can be seen clearly in Figure 1.

A Quake simulation estimates the ground motion during 60 seconds of shaking. Each simulated second consists of 100 time steps, for a total of 6000 time steps. During each time step, the simulation executes three sparse matrix-vector product (SMVP) operations of the form $\mathbf{y} = \mathbf{K}\mathbf{x}$, where $\mathbf{x}$ and $\mathbf{y}$ are vectors of length $3n$ (representing three degrees of freedom—x, y, and z displacements—for each node of the mesh), and $\mathbf{K}$ is a sparse $3n \times 3n$ *stiffness matrix*. $\mathbf{K}$ can be likened to an adjacency matrix of the nodes of the mesh; $\mathbf{K}$ contains a 3×3 submatrix for each pair of nodes connected by an edge of the mesh (including self-edges).

The simulations are parallelized using a domain-specific tool chain for finite element problems called Archimedes [1]. To generate a simulation that will run on p PEs, Archimedes partitions the mesh into p disjoint sets of elements. Each set is called a *subdomain* and is assigned to some PE (We will use the terms subdomain and PE interchangeably). The partitioner is based on a recursive geometric bisection algorithm [7, 8] that divides the elements equally among the subdomains while attempting to minimize the total number of nodes that are shared by subdomains, and hence the total communication volume. The geometric partitioning algorithm has provable upper bounds on the separator sizes and in practice usually generates partitions that are as good as those produced by other modern partitioning algorithms [2, 3, 4, 5, 6, 9, 10].

To compute $\mathbf{y} = \mathbf{K}\mathbf{x}$ on a set of PEs, we must consider the data distribution by which vectors and matrices are stored. The vectors $\mathbf{x}$ and $\mathbf{y}$ are stored in a distributed fashion according to the mapping of nodes to PEs induced by the partition of elements among PEs. If a node i resides in several PEs (because i is a vertex of several elements mapped to different PEs), the values x_i and y_i are replicated on those PEs. The matrix $\mathbf{K}$ is distributed so that K_{ij} resides on any PE on which nodes i and j both reside. Figure 2

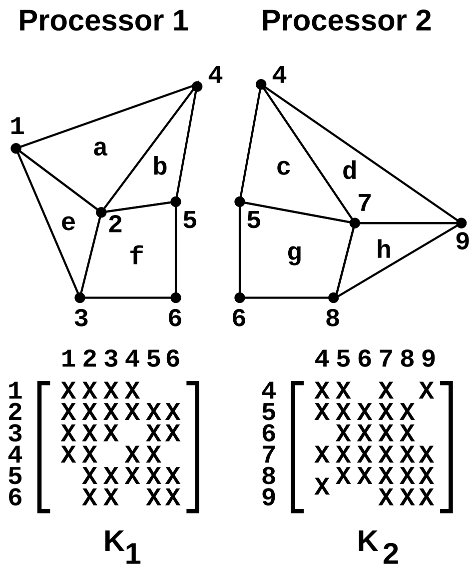

Figure 2: A finite element mesh and corresponding stiffness matrix $\mathbf{K}$, distributed among two PEs. Xs represent nonzero 3×3 submatrices. Note that some nodes are shared by both PEs, as are some stiffness matrix entries (corresponding to shared edges).

demonstrates this method of distributing data. Given this method of distributing data, the multiplication $\mathbf{y} = \mathbf{Kx}$ is performed in two steps: (1) *Computation phase:* each PE computes a local matrix-vector product over the subdomain that resides on that PE. (2) *Communication phase:* PEs that share nodes communicate and combine their nodal $\mathbf{y}$ values into correct global values for each node. In Figure 2, PEs 1 and 2 must communicate to resolve the values of the interface nodes 4, 5, and 6.

3 Mesh properties

This section describes the global and local properties of the Quake meshes. By global properties we mean properties of the entire mesh. By local properties we mean properties of the subgraphs on the individual PEs. For example, the number of nodes in a mesh is a global property, while the average number of nodes per PE is a local property.

3.1 Global mesh properties

Figure 3 lists the basic global properties of the Quake meshes. Notice that when the wave period is halved, its frequency doubles, and the number of nodes increases by a factor of nearly eight—a factor of two in each of three dimensions. Another way to appreciate the size of these meshes is by the amount of memory their corresponding simulations consume. As a general rule, for each node in the mesh a simulation uses about 2 KBytes of memory at runtime to accommodate the storage of several double-precision vectors and sparse matrices. Thus sf10 requires about 15 MBytes and sf1 requires about 5 GBytes.

Figure 4 shows the node degrees for the Quake meshes, where the degree of a node is the number of neighboring nodes. It is somewhat surprising that the average and maximum node degrees grow with the

property	sf10	sf5	sf2	sf1
nodes	7,294	30,169	378,747	2,461,694
edges	44,922	190,377	2,509,064	16,684,112
faces	72,698	311,514	4,198,057	28,202,581
elements	35,025	151,239	2,067,739	13,980,162

Figure 3: Properties of the Quake meshes.

mesh size. Further, the average node degree of 12–13 is less than average node degree of 16 that one would expect.

	sf10	sf5	sf2	sf1
min	3.0	4.0	3.0	3.0
avg	12.3	12.6	13.2	13.6
max	25.0	29.0	31.0	34.0

Figure 4: Node degrees

Figure 5 shows the distributions of the node degrees in the form of a histogram. The numbers are somewhat counterintuitive after Figure 4, since the proportion of high-degree nodes decreases with problem size. For example, 5% of the sf2 nodes have 17–32 neighbors, compared to 3% of the sf1 nodes.

node degree	sf10	sf5	sf2	sf1
3 – 4	4	17	4	4
5 – 8	716	2,284	3,978	9,524
9 – 16	5,812	24,638	358,466	2,383,411
17 – 32	762	3,230	16,299	68,754
33 – 64	0	0	0	1

Figure 5: Histograms of node degrees

Another important property of meshes is the aspect ratio of the elements, which we define as the longest edge divided by the shortest altitude. In general, smaller aspect ratios are better than larger asect ratios. Histograms of the element aspect ratios for the Quake meshes are shown in Figure 6. The maximum aspect ratios of roughly 5 are small enough to guarantee well-conditioned stiffness matrices.

3.2 Global partitioned mesh properties

Figure 7 shows the distribution of interface and interior nodes for the partitioned Quake meshes. We say that a node is an interface node if it is shared by more than one PE. Otherwise, we say that it is an interior node. Notice by this definition that nodes on the external boundary of the domain can be either interior or interface nodes. Interface nodes are interesting because they represent communication at runtime.

aspect ratio	sf10	sf5	sf2	sf1
1.2 – 1.5	940	5,317	112,815	830,415
1.5 – 2	433	1,365	7,929	33,358
2 – 2.5	12,459	55,985	812,263	5,559,973
2.5 – 3	10,751	45,804	599,956	3,860,114
3 – 4	10,413	42,573	534,111	3,693,199
4 – 6	51	195	665	3,103

Figure 6: Histograms of element aspect ratios

subdomains		sf10 (7,294)	sf5 (30,169)	sf2 (378,747)	sf1 (2,461,694)
4	interface	561	1,686	11,501	36,553
	interior	6,733	28,483	367,246	2,425,141
8	interface	1,116	3,029	17,453	62,933
	interior	6,178	27,140	361,294	2,398,761
16	interface	1,690	4,385	26,173	102,317
	interior	5,604	25,784	352,574	2,359,377
32	interface	2,441	6,340	38,953	147,593
	interior	4,853	23,829	339,812	2,314,101
64	interface	3,367	8,809	56,280	216,157
	interior	3,927	21,360	322,467	2,245,537
128	interface	4,319	11,713	75,522	288,257
	interior	2,975	18,456	303,225	2,173,437

Figure 7: Interface and interior nodes. The total number of nodes is shown in parentheses.

Figure 8 shows the distribution of interface and interior edges for the partitioned Quake meshes. Interface edges are interesting because they represent redundant computation.

3.3 Local partitioned mesh properties

Figure 9 shows the minimum number k of elements that are assigned to each PE. The partitions are perfect in the sense that each PE is assigned either k or $k+1$ elements. This is not unexpected, since the element is the unit of partitioning.

Figure 10 shows the distribution of interface and interior nodes per PE. At runtime, each interface node corresponds to some data that must be transferred to other PEs and so we would like the interface nodes to be balanced evenly across the PEs. However we see that the number of nodes on different PEs can vary by a factor of three. Thus we can expect the communication phase of the SMVP operation to be similarly unbalanced. This imbalance is an artifact of the way modern mesh partitioners work, optimizing the total volume of communication across all PEs rather than minimizing the maximum communication on any PE.

An edge in the mesh corresponds to a nonzero entry in the coefficient matrix of the SMVP operation, and each interface edge corresponds to redundant nonzero entries. So in general, we want the number of interface edges to be small and we want each PE to have about the same number of edges. Indeed, Figure 11

subdomains		sf10 (44,922)	sf5 (190,377)	sf2 (2,509,064)	sf1 (16,684,112)
4	interface	1,549	4,714	33,544	47,105
	interior	43,373	185,663	2,475,520	16,637,007
8	interface	3,117	8,577	51,108	107,975
	interior	41,805	181,800	2,457,956	16,576,137
16	interface	4,822	12,650	77,331	186,499
	interior	40,100	177,727	2,431,733	16,497,613
32	interface	7,216	18,606	115,713	304,679
	interior	37,706	171,771	2,393,351	16,379,433
64	interface	10,294	26,511	169,097	441,036
	interior	34,628	163,866	2,339,967	16,243,076
128	interface	13,696	36,147	230,294	649,726
	interior	31,226	154,230	2,278,770	16,034,386

Figure 8: Interface and interior edges. The total number of edges is shown in parentheses.

subdomains	sf10	sf5	sf2	sf1
4	8,756	37,793	516,934	3,495,040
8	4,378	18,896	258,467	1,747,520
16	2,189	9,448	129,233	873,760
32	1,094	4,724	64,616	436,880
64	547	2,362	32,308	218,440
128	273	1,181	16,154	109,220

Figure 9: Elements per subdomain

shows that the edges are indeed well balanced across the PEs, and thus we can expect the computation phase of the SMVP operations to be well balanced.

4 Communication properties

This section describes the communication properties of SMVP operations that are induced from the Quake meshes. All sizes and volumes are presented in units of words per degree of freedom (dof) in the simulation. In general, if a simulation models k dof, then there are k quantities associated with each node in the corresponding mesh and k words of data are exchanged for each interface node shared by a pair of PEs.

4.1 Global communication properties

Figure 12 shows the total volume of data transferred by all PEs during the communication phase of an SMVP operation. The total communication volume is related to, but not identical to, the global number of interface nodes. The reason they are not identical is that a node might be shared by multiple subdomains.

Figure 13 shows the bisection volume V for the Quake SMVP operations, where bisection volume is defined as follows. We are given a symmetric $p \times p$ matrix m such that m_{ij} is the number of words

		sf10		sf5		sf2		sf1	
subdomains		total	interface	total	interface	total	interface	total	interface
4	min	1,947	211	7,828	432	96,123	3,765	614,564	13,621
	avg	1,970	287	7,968	847	97,572	5,760	624,583	18,297
	max	1,997	366	8,207	1,274	99,613	9,185	630,394	30,944
8	min	1,025	181	3,971	521	48,046	2,368	306,611	9,974
	avg	1,061	288	4,164	772	49,559	4,397	315,659	15,814
	max	1,081	373	4,280	1,102	50,990	5,676	323,728	24,650
16	min	542	148	2,081	324	24,213	1,911	155,093	7,656
	avg	575	225	2,182	571	25,367	3,331	160,383	12,922
	max	620	285	2,273	767	26,278	4,471	165,556	19,254
32	min	295	103	1,115	276	12,366	1,471	78,918	4,202
	avg	321	169	1,167	422	13,120	2,501	81,675	9,360
	max	349	246	1,245	617	13,788	3,551	85,963	13,779
64	min	162	66	583	145	6,441	988	40,405	2,751
	avg	184	123	637	303	6,870	1,832	41,994	6,908
	max	222	184	760	479	7,199	2,869	44,132	11,146
128	min	91	51	314	95	3,408	523	20,803	2,110
	avg	107	84	353	209	3,622	1,253	21,632	4,652
	max	133	125	406	324	3,997	2,232	22,683	7,579

Figure 10: Total nodes and interface nodes per subdomain

		sf10		sf5		sf2		sf1	
subdomains		total	interface	total	interface	total	interface	total	interface
4	min	11,548	565	48,378	1,193	631,492	10,974	4,169,622	40,262
	avg	11,624	781	48,777	2,361	635,662	16,782	4,198,043	54,008
	max	11,703	1,000	49,464	3,537	641,529	26,799	4,213,857	91,207
8	min	5,912	479	24,319	1,442	315,794	6,829	2,083,537	29,454
	avg	6,013	788	24,882	2,158	320,054	12,809	2,108,905	46,704
	max	6,072	1,018	25,230	3,095	324,028	16,557	2,131,610	72,806
16	min	3,033	393	12,423	889	158,507	5,561	1,047,039	22,444
	avg	3,122	616	12,710	1,603	161,708	9,724	1,061,930	38,216
	max	3,244	811	12,977	2,161	164,204	13,120	1,076,253	57,141
32	min	1,576	274	6,419	752	79,986	4,243	527,572	12,258
	avg	1,645	466	6,556	1,188	82,091	7,299	535,294	27,698
	max	1,719	706	6,783	1,768	83,863	10,462	547,042	41,060
64	min	822	171	3,270	392	40,731	2,836	266,474	7,987
	avg	879	338	3,415	855	41,918	5,356	270,992	20,454
	max	968	518	3,736	1,381	42,807	8,601	276,949	33,471
128	min	438	130	1,691	249	20,904	1,479	134,982	6,122
	avg	473	229	1,794	589	21,473	3,670	137,318	13,800
	max	538	367	1,936	943	22,525	6,601	140,132	22,681

Figure 11: Total edges and interface edges per subdomain

transferred from PE i to PE j. If we assume that PEs $0, \ldots, p/2-1$ are on one side of the bisection and

subdomains	sf10	sf5	sf2	sf1
4	1,226	3,440	23,154	73,438
8	2,540	6,522	35,986	128,442
16	4,314	10,264	56,306	213,130
32	7,264	16,234	86,768	312,662
64	11,826	25,406	131,750	471,952
128	18,854	38,324	190,042	654,294

Figure 12: Global communication volume (words per dof)

PEs $p/2, \ldots, p-1$ are on the other side, then

$$V = 2 \sum_{i=0}^{p/2-1} \sum_{j=p/2}^{p-1} m_{ij}$$

words cross the bisection during the communication phase. Notice that the bisection volume is quite small relative to the total communication volume and in absolute terms as well, especially on more than a few PEs. This is not surprising, given the locality of physical simulations.

subdomains	sf10	sf5	sf2	sf1
4	624 (51%)	1,718 (50%)	10,916 (47%)	32,188 (44%)
8	676 (27%)	1,786 (27%)	11,196 (31%)	32,954 (26%)
16	758 (18%)	1,960 (19%)	11,690 (21%)	33,624 (16%)
32	882 (12%)	2,188 (13%)	12,306 (14%)	34,294 (11%)
64	1,014 (9%)	2,346 (9%)	12,888 (10%)	35,393 (7%)
128	1,308 (7%)	2,744 (7%)	13,802 (7%)	36,682 (6%)

Figure 13: Bisection communication volume (% of global communication volume) (words per dof)

Figure 14 shows the total number of messages transferred between PEs during the communication phase of the Quake SMVPs. Figure 15 summarizes the sizes of those messages. Notice that for large numbers of PEs the average message size is only several hundred words per dof. Also, there is a large variance in the sizes of messages. For example, the message sizes for the sf1 simulation can vary by three orders of magnitude. So again we see imbalance in the communication phase.

Figure 16 drives this point home about the imbalance in the communication phase even more clearly. For example, consider sf2 running on 128 PEs. This is a large finite element problem, and yet fully one third of the messages are smaller than 64 words per degree of freedom. For the tetrahedral earthquake models, with three dof, this means that one third of the messages are smaller than 192 words. An important implication is that we cannot expect to amortize message latencies with large messages.

4.2 Local communication properties

This section describes the communication properties on the individual PEs. Although the literature often cites global communication properties such as total communication volume when comparing the quality of

subdomains	sf10	sf5	sf2	sf1
4	10	8	8	8
8	32	28	26	28
16	82	90	88	86
32	250	230	210	232
64	618	564	516	522
128	1,626	1,340	1,246	1,296

Figure 14: Global number of messages

subdomains		sf10	sf5	sf2	sf1
4	min	43	401	2,206	5,692
	avg	123	430	2,894	9,180
	max	197	458	3,765	14,933
8	min	1	4	269	173
	avg	79	233	1,384	4,587
	max	176	423	2,569	8,746
16	min	2	3	5	13
	avg	53	114	640	2,478
	max	102	249	1,549	7,161
32	min	1	1	1	1
	avg	29	71	413	1,348
	max	85	211	1,160	4,617
64	min	1	1	1	2
	avg	19	45	255	904
	max	62	146	825	3,108
128	min	1	1	1	1
	avg	12	45	153	505
	max	40	146	607	2,037

Figure 15: Global message sizes (words per dof)

mesh partitions, these properties are probably less important to running time than the local communication properties on each PE. The reason is that the PE with the highest communication time is the bottleneck PE during the SMVP. Thus, we would like to balance the communication times by minimizing the maximum communication time on any PE. Unfortunately, what we see in this section is that the communication properties on each PE can be highly unbalanced.

Figure 17 shows the communication volume per dof on each PE. There are several interesting aspects to these statistics. First, the reduction in communication volume per subdomain is smaller than expected. Since a cube with a volume of n has a surface area of about $6n^{2/3}$, we would expect a reduction in the number of nodes per subdomain by a factor of k to reduce the communication volume per subdomain by a factor $k^{2/3}$. Thus, for the sf1 simulation, we would expect the factor of 32 reduction in nodes per subdomain to result in a factor of 10 reduction in the communication volume per subdomain. However, what we actually see in Figure 17 is a factor of 3.6 reduction, which is significantly less than expected.

Another important aspect of Figure 17 is the large difference between the PE with the smallest communication volume and the PE with the largest communication volume. Again, we see this potential problem

sf10	subdomains					
msg size	4	8	16	32	64	128
1		2		18	72	266
2		0	2	10	48	148
3–4		0	8	32	66	228
5–8		2	2	20	74	228
9–16		2	6	18	64	272
17–32		2	4	46	144	394
33–64	2	8	30	78	150	90
65–128	2	6	30	28		
129–256	6	10				
257–512						

(a) sf10

sf5	subdomains					
msg size	4	8	16	32	64	128
1				6	36	96
2				6	18	60
3–4		2	6	16	24	94
5–8		0	6	12	38	128
9–16		0	2	14	80	184
17–32		0	8	24	74	258
33–64		2	8	42	116	366
65–128		0	22	72	168	154
129–256		10	38	38	10	
257–512	8	14				

(b) sf5

sf2	subdomains					
msg size	4	8	16	32	64	128
1				4	14	32
2				0	4	12
3–4				4	8	28
5–8			4	2	16	42
9–16			4	8	22	50
17–32			4	8	22	108
33–64			6	12	48	142
65–128			6	28	62	230
129–256			2	22	94	338
257–512		2	10	40	142	256
513–1024		6	30	78	84	8
1025–2048		14	22	4		
2049–4096	8	4				
4097–8192						
8193–16384						

(c) sf2

sf1	subdomains					
msg size	4	8	16	32	64	128
1				2		12
2				6	2	12
3–4				2	8	24
5–8				4	12	30
9–16			2	10	6	44
17–32				2	14	52
33–64			2	12	24	96
65–128			4	14	30	110
129–256		2	2	18	38	150
257–512			6	12	94	250
513–1024			8	46	102	290
1025–2048		4	12	34	128	226
2049–4096		4	30	66	64	
4097–8192	6	16	20	4		
8193–16384	2	2				

(d) sf1

Figure 16: Histograms of global message sizes (words per dof)

with modern partitioners, which work hard to balance computation and to minimize global communication volume, but make no effort to balance the communication on each PE. The imbalance in the communication volume on each PE is shown even more dramatically in Figure 18.

Figure 19 shows the total number of messages sent and received by the individual PEs. The number of messages is an even number because pairs of PEs always exchange pairs of messages. If there are k messages for a given PE, then that PE has $k/2$ neighbors with whom it exchanges a pair of equal sized messages. For example, we see that there is some PE in the sf1 simulation running on 128 PEs that has 23 neighbors, which is about 20% of the total number of PEs. Thus, the Quake simulations are an interesting middle ground between regular grid computations with a constant 4 neighbors and complete exchange algorithms where each PE communicates with every other PE.

In Figure 19, notice the large variance in the number of messages transferred by different PEs. The

subdomains		sf10	sf5	sf2	sf1
4	min	448	864	7,530	27,408
	avg	613	1,720	11,577	36,719
	max	784	2,582	18,446	62,054
8	min	384	1,086	4,828	5,214
	avg	635	1,631	8,997	25,989
	max	850	2,360	11,716	50,588
16	min	314	676	4,016	15,512
	avg	539	1,283	7,038	26,641
	max	736	1,764	9,494	39,760
32	min	250	642	3,024	8,504
	avg	454	1,015	5,423	19,541
	max	724	1,492	8,006	29,076
64	min	164	342	2,082	5,668
	avg	370	794	4,117	14,749
	max	588	1,432	6,840	24,354
128	min	132	254	1,118	4,468
	avg	295	599	2,969	10,223
	max	580	1,120	5,420	17,016

Figure 17: Communication volume per subdomain (words per dof)

sf10	subdomains					
msg size	4	8	16	32	64	128
129–256				1	9	41
257–512	1	1	7	23	52	84
513–1,024	3	8	9	8	3	3
1,025–2,048						
2,049–4,096						

(a) sf10

sf5	subdomains					
msg size	4	8	16	32	64	128
129–256						1
257–512					7	42
513–1,024	1		4	18	49	83
1,025–2,048	2	5	12	14	8	2
2,049–4,096	1	3				

(b) sf5

sf2	subdomains					
msg size	4	8	16	32	64	128
1,025–2,048						15
2,049–4,096			1	7	35	103
4,097–8,192	1	2	10	25	29	10
8,193–16,384	2	6	5			
16,385–32,768	1					
32,769–65,536						

(c) sf2

sf1	subdomains					
msg size	4	8	16	32	64	128
1,025–2,048						
2,049–4,096						
4,097–8,192		2			2	35
8,193–16,384			1	8	37	91
16,385–32,768	3	4	12	24	25	2
32,769–65,536	1	2	3			

(d) sf1

Figure 18: Histograms of communication volume per subdomain (words per dof)

partitioner is not doing a good job of balancing the number of messages sent by each PE. This could have a significant impact on performance when message latencies are high. Figure 20 expands on this point with the histograms for the number of messages per subdomain. The interesting aspect of these statistics is that

subdomains		sf10	sf5	sf2	sf1
4	min	4	2	2	2
	avg	5	4	4	4
	max	6	6	6	6
8	min	4	4	4	6
	avg	8	7	7	7
	max	12	12	10	14
16	min	4	4	4	4
	avg	10	11	11	11
	max	18	20	16	18
32	min	6	6	4	4
	avg	16	14	13	15
	max	30	30	26	26
64	min	6	8	4	4
	avg	19	18	16	16
	max	38	40	36	38
128	min	6	8	4	6
	avg	25	21	20	20
	max	62	52	50	46

Figure 19: Messages per subdomain

a significant number of PEs communicate with a large number of PEs.

The variance in communication can be seen even more dramatically in Figure 21, which shows the message sizes for the subdomain with the smallest average message size. Here we see that the message sizes on a single PE can vary by three orders of magnitude. Further, the distribution of message sizes is fairly uniform, with roughly as many small messages as large messages. This is shown in Figure 22, which details the histogram of the messages sizes in Figure 21.

In summary, modern mesh partitioners typically allocate a class of mesh entity such as nodes, elements, or edges evenly across the subdomains, and then attempt to minimize some communication metric, usually the total number of interface nodes. While partitioners generally do a good job of meeting these goals, it is not clear that they are using the appropriate optimization criteria. The communication properties of the Quake SVMPs show us that across PEs there is a wide variability in the volume of communication data, the number of messages, and the sizes of the individual meshes. Since the SMVP operations are synchronous, the PE with the longest communication phase will be the bottleneck PE. Thus, in addition to minimizing the total communication volume, partitioners should attempt to minimize the maximum communication time on each PE.

5 Computation properties

This section describes the distribution of nonzero entries in the sparse matrices that are induced from the Quake meshes. As we saw in Section 2, each nonzero entry in a sparse matrix corresponds to a mesh edge, and since the meshes are undirected graphs, each mesh corresponds to two nonzero matrix entries. If a simulation has k dof, then each nonzero matrix entry consists of a block of k^2 words, and each vector entry consists of a subvector of k words. In this section, the number of nonzero matrix entries is expressed in units of words per dof $\times$ dof.

sf10	subdomains					
msg size	4	8	16	32	64	128
2						
3–4	2	2	1			
5–8	2	3	6	5	4	2
9–16		3	8	13	18	26
17–32			1	14	39	74
33–64					3	26

(a) sf10

sf5	subdomains					
msg size	4	8	16	32	64	128
2	1					
3–4	2	2	1			
5–8	1	4	5	8	4	3
9–16		2	7	13	28	45
17–32			3	11	30	68
33–64					2	12

(b) sf5

sf2	subdomains					
msg size	4	8	16	32	64	128
2	1					
3–4	2	3	1	1	1	1
5–8	1	3	4	6	9	4
9–16		2	11	18	28	48
17–32				7	24	65
33–64					2	10

(c) sf2

sf1	subdomains					
msg size	4	8	16	32	64	128
2	1					
3–4	2		2	1	1	
5–8	1	7	3	2	7	4
9–16		1	10	22	31	49
17–32			1	7	24	66
33–64					1	9

(d) sf1

Figure 20: Histograms of messages per subdomain

subdomains		sf10	sf5	sf2	sf1
4	min	43	401	2,206	5,692
	avg	88	415	2,280	6,582
	max	189	429	2,354	8,012
8	min	1	4	733	173
	avg	61	181	1,166	3,365
	max	134	385	1,953	5,214
16	min	4	3	56	113
	avg	37	78	490	1,798
	max	64	189	1,115	5,645
32	min	1	2	4	2
	avg	20	49	261	961
	max	74	174	964	3,491
64	min	2	1	18	49
	avg	13	29	157	566
	max	40	146	433	1,874
128	min	1	1	1	4
	avg	7	19	99	295
	max	36	46	421	1,494

Figure 21: Message sizes for the subdomain with the smallest average message sizes (words per dof)

The matrices induced from the Quake meshes are symmetric. The data structures for these matrices can exploit the symmetry by storing only the upper (or lower) triangle. The penalty for such a space-efficient

sf10	subdomains					
msg size	4	8	16	32	64	128
1		2		4		6
2				2	6	10
3–4			2	4	4	16
5–8			2	2	2	12
9–16			2	2	2	2
17–32		2		4	2	6
33–64	2	2	12	4	4	2
65–128	2	4		2		
129–256	2	2				
257–512						

(a) sf10

sf5	subdomains					
msg size	4	8	16	32	64	128
1					36	2
2				2	18	2
3–4		2	6	6	24	4
5–8			2	4	38	8
9–16				2	80	12
17–32			2		74	14
33–64				8	116	10
65–128			4	6	168	
129–256		2	6	2	10	
257–512	4	2				

(b) sf5

sf2	subdomains					
msg size	4	8	16	32	64	128
1						2
2						
3–4				2		2
5–8						2
9–16				4		2
17–32					4	10
33–64			4	4		6
65–128			2	4	4	4
129–256				4	6	8
257–512			2	2	4	4
513–1024		4	6	6		
1025–2048		6	2			
2049–4096	4					
4097–8192						

(c) sf2

sf1	subdomains					
msg size	4	8	16	32	64	128
1						
2				2		
3–4						2
5–8				2		4
9–16						6
17–32						2
33–64				2	2	6
65–128			2	2	2	4
129–256		2		2		6
257–512			2		6	4
513–1024			2		4	2
1025–2048			4	2	2	6
2049–4096			2	4		
4097–8192	4	4	2			

(d) sf1

Figure 22: Histograms of message sizes for subdomain with smallest average message size (words per dof)

storage schemes is less locality during the SMVP. In this section, we assume a simpler but less efficient scheme based on nonsymmetric storage where the entire matrix is stored. Given m nonzero entries in the nonsymmetric scheme, there are $(m(m+1))/2$ nonzero entries in the symmetric scheme.

5.1 Global computation properties

Figure 23 shows the number of nonzero entries in the global sparse matrices for the Quake SMVP operations. If a mesh has n nodes and e edges, then there are $2e+n$ nonzero entries in the induced sparse matrix (assuming nonsymmetric storage).

	sf10	sf5	sf2	sf1
nonzero entries	97,138	410,923	5,396,875	35,829,918

Figure 23: Global nonzero matrix entries per dof $\times$ dof (assuming nonsymmetric storage)

5.2 Local computation properties

Of course the global sparse matrix is never actually constructed or stored. Rather a sparse local matrix is constructed on each PE. Figure 24 shows the number of nonzero matrix entries per subdomain. If the simulation has k dof, then each nonzero entry requires $2k^2$ floating point operations during the local computation phase of the SMVP. Notice that the computation is reasonably well balanced across the PEs, with only a few percent difference between the maximum and minimum number of nonzeros.

subdomains		sf10	sf5	sf2	sf1
4	min	25,043	104,584	1,359,107	8,953,808
	avg	25,218	105,522	1,368,895	9,020,668
	max	25,403	107,135	1,382,671	9,058,108
8	min	12,849	52,609	679,634	4,473,685
	avg	13,087	53,930	689,667	4,533,469
	max	13,225	54,740	699,046	4,586,948
16	min	6,608	26,927	341,227	2,249,171
	avg	6,819	27,604	348,782	2,284,243
	max	7,108	28,227	354,686	2,318,062
32	min	3,447	13,953	172,338	1,134,062
	avg	3,610	14,278	177,302	1,152,263
	max	3,784	14,811	181,514	1,180,047
64	min	1,806	7,123	87,903	573,353
	avg	1,942	7,468	90,706	583,977
	max	2,158	8,232	92,813	598,030
128	min	967	3,696	45,126	290,767
	avg	1,053	3,942	46,568	296,267
	max	1,208	4,278	49,047	302,947

Figure 24: Nonzero matrix entries per subdomain per dof $\times$ dof (assuming nonsymmetric storage)

Figure 25 shows the number of nonzero entries per row for the PE with the fewest average entries per row. These are interesting numbers because they show how extremely sparse the matrices are. The implication is that the inner loops of the SMVP will tend to be short and we can expect difficulty in amortizing the startup costs of these loops.

The ratio of computation to communication on a PE can provide some insight into the relative cost of communication at runtime. Generally, high computation/communication ratios are desirable. Figure 26 shows the computation/communication ratios for the SPMV operations from the Quake simulations. For the SMVP, the floating point operation is a useful measure of work. Thus, each ratio (denoted fp/comm ratio in the figure) is computed by the dividing the average number of floating point operations per PE during the computation phase by the average number of words transferred per PE during the communication phase.

subdomains		sf10	sf5	sf2	sf1
4	min	4	4	4	4
	avg	13	13	14	14
	max	26	28	32	35
8	min	4	4	4	4
	avg	13	13	14	14
	max	25	27	27	33
16	min	4	4	4	4
	avg	12	12	14	14
	max	23	26	27	27
32	min	4	4	4	4
	avg	12	12	13	14
	max	23	23	27	27
64	min	4	4	4	4
	avg	10	11	12	14
	max	19	22	27	27
128	min	4	4	4	4
	avg	9	11	12	14
	max	22	24	30	27

Figure 25: Nonzero matrix entries/row per dof $\times$ dof for the subdomain with the fewest avg. entries/row (assuming nonsymmetric storage)

subdomains		sf10	sf5	sf2	sf1
4	fp ops	453,924	1,899,396	24,640,110	162,372,024
	comm wds	1,839	5,160	34,731	110,157
	fp/comm ratio	247	368	709	1,474
8	fp ops	235,566	970,740	12,414,006	81,602,442
	comm words	1,905	4,893	26,991	77,967
	fp/comm ratio	124	198	460	1,047
16	fp ops	122,742	496,872	6,278,076	41,116,374
	comm words	1,617	3,849	21,114	79,923
	fp/comm ratio	76	129	297	514
32	fp ops	64,980	257,004	3,191,436	20,740,734
	comm words	1,362	3,045	16,269	58,623
	fp/comm ratio	48	84	196	354
64	fp ops	34,956	134,424	1,632,708	10,511,586
	comm words	1,110	2,382	12,351	44,247
	fp/comm ratio	31	56	132	238
128	fp ops	18,954	70,956	838,224	5,332,806
	comm words	885	1,797	8,907	30,669
	fp/comm ratio	21	39	94	174

Figure 26: Computation/communication ratio per matrix-vector product per subdomain (assuming 3 dof)

There are some interesting points to make about the numbers in Figure 26. First, conventional wisdom holds that sparse codes like the SMVP are communication intensive. However, this is not always the case. As we see for sf2, which is a reasonably large problem, the computation/communication ratios vary from large (500:1) to moderate (50:1). This common misconception about sparse codes is probably due to the fact that researchers have not had the opportunity to run large enough problems.

Second, while the computation/communication ratios are reasonably high for large problems, as the problem sizes grow by a factor of ten, we see that the computation/communication ratios grow only by a factor of two. This is not surprising; consider that a good partition of an n-node tetrahedral mesh will produce $\mathcal{O}(n^{2/3})$ shared nodes (for the same reason that an $\mathcal{O}(n)$-node cube has a surface area of $\mathcal{O}(n^{2/3})$ nodes). Hence, the computation/communication ratio is $\mathcal{O}(n^{1/3})$, and a factor-of-ten increase in n will yield roughly a factor-of-two increase in that ratio. The point is that while large SMVPs do have reasonable computation/communication ratios, these ratios do not increase quickly with increasing problem size, as they do for cubic problems like dense matrix multiply.

6 Concluding remarks

This report has characterized a family of unstructured tetrahedral finite element simulations partitioned for execution on a parallel system. Our aim is to provide a comprehensive reference source for researchers who are interested in sparse and irregular computations. Along the way we have made a few observations about the properties of the meshes and their induced SMVP operations.

Computation is well balanced across PEs, but communication is not. The number of messages per PE, the communication volume on each PE, and the message sizes vary dramatically across PEs. Improving this balance suggests a potentially important area of improvement for designers of partitioning algorithms.

Further, the sparse matrix-vector product operations induced from the Quake meshes are not as communication intensive as conventional wisdom suggests. For a reasonable sized problem, the ratio of floating point operations to communication words can vary from 500:1 on 4 PEs to 50:1 on 128 PEs. This offers some hope for the efficient implementation of sparse matrix codes.

References

[1] BAO, H., BIELAK, J., GHATTAS, O., O'HALLARON, D. R., KALLIVOKAS, L. F., SHEWCHUK, J. R., AND XU, J. Earthquake ground motion modeling on parallel computers. In *Proceedings of Supercomputing '96* (Pittsburgh, PA, Nov. 1996).

[2] BARNARD, S., AND SIMON, H. A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. Tech. Rep. RNR-92-033, NASA Ames Research Center, Nov. 1992.

[3] FARHAT, C. A simple and efficient automatic FEM domain decomposer. *Comp. & Struct. 28*, 5 (1988), 579–602.

[4] GILBERT, J., MILLER, G., AND TENG, S.-H. Geometric mesh partitioning: Implementation and experiments. In *9th International Parallel Processing Symposium* (Santa Barbara, April 1995), IEEE, pp. 418–427.

[5] HENDRICKSON, B., AND LELAND, R. The Chaco user's guide Version 2.0. Tech. Rep. SAND95-2344, Sandia National Laboratories, July 1995.

[6] HENDRICKSON, B., AND LELAND, R. An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Sci. Comput. 16*, 2 (1995), 452–469.

[7] MILLER, G., AND THURSTON, W. Separators in two and three dimensions. In *Proceedings of the 22th Annual ACM Symposium on Theory of Computing* (Maryland, May 1990), ACM, pp. 300–309.

[8] MILLER, G. L., TENG, S.-H., AND VAVASIS, S. A. A unified geometric approach to graph separators. In *Proceedings of the 32nd Anuual Symposium on Foundations of Computer Science* (Puerto Rico, Oct 1991), IEEE, pp. 538–547.

[9] POTHEN, A., SIMON, H., AND LIOU, K. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications 11* (1990), 430–452.

[10] SIMON, H. Partitioning of unstructured problems for parallel processing. *Comput. Sys. Engr. 2*, 3 (1991), 135–148.