Analysis of Multigrid Methods on Massively Parallel Computers: Architectural Implications

Lesley R. Matheson
Robert E. Tarjan[1]

Department of Computer Science
Princeton University
Princeton, New Jersey 08544

NEC Research Institute
Princeton, New Jersey, 08540

Abstract

We study the potential performance of multigrid algorithms running on massively parallel computers with the intent of discovering whether presently envisioned machines will provide an efficient platform for such algorithms. We consider the domain parallel version of the standard V-cycle algorithm on model problems, discretized using finite difference techniques in two and three dimensions on block structured grids of size 10^6 and 10^9, respectively. Our models of parallel computation were developed to reflect the computing characteristics of the current generation of massively parallel multicomputers. These models are based on an interconnection network of 256 to 16,384 message passing, "workstation size" processors executing in an SPMD mode. The first model accomplishes interprocessor communications through a multistage permutation network. The communication cost is a logarithmic function which is similar to the costs in a variety of different topologies. The second model allows single stage communication costs only. Both models were designed with information provided by machine developers and utilize implementation derived parameters. With the medium grain parallelism of the current generation and the high fixed cost of an interprocessor communication, our analysis suggests an efficient implementation requires the machine to support the efficient transmission of long messages, (up to 1000 words) or the high initiation cost of a communication must be significantly reduced through an alternative optimization technique. Furthermore, with variable length message capability, our analysis suggests the low diameter multistage networks provide little or no advantage over a simple single stage communications network.

[1] Research at Princeton University partially supported by the National Science Foundation, Grant No. CCR-8920505, the Office of Naval Research, Contract No. N0014-91-J-1463, and by DIMACS (Center for Discrete Mathematics and Theoretical Computer Science), a National Science and Technology Center, Grant No. NSF-STC88-09648

0. Introduction

In the current generation of massively parallel (MP) computers there is a convergence towards a common set of architectural characteristics. From the standpoint of a computational scientist, this convergence presents the opportunity to study the class of machines as a whole, in order to determine whether or not they can be efficient platforms for the solution of various computationally intensive tasks.

We studied the potential use of these machines for the solution of multigrid algorithms. Our study included a wide range of multigrid algorithms and encompassed several different architectural characteristics.

In this paper we present the architectural ideas suggested by this study which would enable the current generation of MP machines to become efficient platforms for various multigrid applications.

Our approach was to develop a set of models of parallel computation based on the common characteristics of the current generation of MP machines. We implemented a representative set of structured multigrid algorithms on these models. We then looked at the performance predictions and tried to understand their implications.

The remainder of the paper is organized as follows. First, the models of computation are developed, followed by a brief description of the multigrid algorithms and their implementations. Next, the performance predictions are presented and finally their implications are summarized.

1. The Current Generation of Massively Parallel Computers

The power and availability of RISC microprocessor chips have increased dramatically over the past several years. The proliferation and decreased cost of these "workstation-size" processors have spawned the current generation of multicomputers. Some of the major architectural similarities of this generation are summarized below.

Multicomputers These multicomputers are interconnection networks of physically distributed processors and memory, linked in a variety of different topological configurations.

Powerful Microprocessors The processors are generally "off the shelf" single chip RISC microprocessors. They can perform integer and floating point computation significantly faster than the bit-serial processors which characterized many machines of the previous generation.

Medium Grain Size The increased size, cost and speed of the individual processing elements has delineated a medium grain size for the current generation. Most of the machines are targeted for the range of 1K processors, with larger machines possibly ranging up to 16K processors.

Slow Network Communication The current machines generally exhibit slow interprocessor communication speeds relative to on-chip events. This is frequently a result of handling the network communications processing in the software layer.

Single Program Multiple Data Mode of Execution Unlike the more rigid SIMD and asynchronous MIMD patterns of the previous generation most of the newer machines execute the same program on each processing element with different data, enforcing synchronization only as required by interprocessor communication.

The current generation includes the CM5 by Thinking Machines, a network of Sun SPARC processor nodes, potentially with vector accelerators, connected in a fat tree topology; the Touchstone Delta, developed by Intel and Caltech, a three dimensional mesh of two Intel i860s per node; the Paragon by Intel, a 3D mesh topology with one to four i860 processors per node; the Kendall Square Research machines, a hierarchy of concentric rings with shared virtual memory, with two custom designed chips per node. Cray Research is building a machine with DEC Alpha processors connected by a yet unrevealed topology.

2. Models of Parallel Computation

The models of parallel computation presented in this paper were designed to capture the salient characteristics of the current generation of massively parallel computers. The guiding philosophy behind the development of these models was to strike a reasonable balance between machine independence and practicality, simplicity and accuracy. The goal is to find a set of models which facilitates efficient algorithm design, and ideally, provides feedback into the machine design process itself.

The models of computation reflect the paradigm of the multicomputer: processors and memory are physically distributed throughout an interconnection network. Motivated by the large disparity between the speeds of on-chip and network events, the models reflect the costs of a two level memory hierarchy. The cost of a local memory access is included in the cost of an arithmetic operation while the cost of a remote memory access is treated separately. The models were parameterized to facilitate analysis under different ratios of problem to machine size. In addition, this parameterization allows the incorporation of changes in technology, such as increases in on-chip computation speed or a decrease in network communication latency. The models assume the processors operate in a Single Program Multiple Data mode of execution.

The analysis of this paper utilizes three of the models developed. The characteristics of these models are similar, differing only in their treatment of communication costs. The different treatment of communications costs ranges from assigning a simple topologically-blind cost for a network communication to a more complex function which potentially provides more accuracy. The cost of a floating point computation, treated similarly in each of the models, is separated from the cost of a remote memory access.

3. Communication Costs

Accurately and simply accounting for inter-processor communication is the toughest challenge in the development of a useful model of parallel computation. The three alternative treatments presented here are based on the common components of network communication costs exhibited by the current generation of multicomputers.

1. ***Fixed Start-Up Costs*** There is a large fixed start-up cost associated with any message passing, packet-based communication. To execute a network communication often requires a processor interupt, complete with a full context switch. The message must be packaged and tagged with destination information and injected into the network.

2. ***Variable Cost Per Node*** This component of communications cost is the time to route the message through the network to its destination. Cut-through, circuit switched routing, a common general technique, for example, imposes a per node path formation cost. These routing and path formation costs are actually a complex function of the routing algorithm, the communications pattern and network topology. Taken in sum, these comprise the different aspects of contention. In these models, this complex distance-related component is simplified. It is approximated as the product of a machine-dependent constant and the number of processor nodes along the required communications path. Sensitivity analysis is used to potentially understand the impact of different degrees of contention.

3. ***Spooling Costs Per Node*** A third component of network communications costs is the cost to physically spool the message through the network. Experimental results suggest the spooling cost can be approximated by a linear function of the message length, up to a message size of 1000 words. In these models the spooling cost of a

message is treated as the product of a per-word cost and the length of the message in four-byte words.

4. Fixed Costs of Receipt Finally, receiving a message generates a set of costs on the receiving processor, analogous to those required by the originating processor, namely, interrupts, context switches and message unpacking.

4. Three Models of Computation

The three models of computation used in this analysis were based on the common architectural characteristics discussed above and differ only in their approximation of the components of network communications costs. The following descriptions assume the models use only fixed constant size messages to accomplish all network communication.

The GAP Model The first model, the GAP model, is a simple, topologically blind model which grew out of a set of discussions with a group of researchers at Berkeley. So named because of the "gap" in processor utilization caused by the initiation of a network communication, the GAP model charges one fixed cost for every communication regardless of its source and destination.

The LOG Model The second model, the LOG model, introduces a variable topologically-based cost to the fixed cost component. The LOG model assumes the processors are physically connected in a 2D mesh with an overarching multi-stage permutation network. To approximate the distance a message must travel, the LOG model uses the logarithm of the Manhattan distance (or L_1 norm) between the sending and receiving processors on the 2D mesh. The motivation for the use of this function is twofold. First, it realizes the lower bound on path length between any two nodes in a network with a bounded branching factor. Second, it generally approximates the behavior of a variety of networks which realize logarithimic communication distances, such as butterfly and shuffle-exchange networks. Thus, communications cost in the LOG model, with fixed length messages, is approximated by the following function.

Communications Cost = Fixed + Variable * Distance

where Distance = Log(Manhattan Distance)

The Single Stage Model The Single Stage model also treats the cost of a communication as the sum of fixed costs and a per-node distance dependent cost. This model, like the LOG model, also assumes the processors are physically connected in a two dimensional mesh. In the single stage model, however, there is no overarching multi-stage network. All communication is accomplished by single or multiple hops along the physical connections of the 2D mesh. The motivation for this model was the possibility of quantifying the impact of a multi-stage network on performance for various applications. The cost of a communication with fixed length messages, therefore, is approximated by the following function.

Communications Cost = Fixed + Variable * Distance

where Distance = Manhattan Distance

Model Parameters

The three models, with fixed length messages, are parameterized by a fixed component and a per-node variable component of communication, the cost of a floating point operation, and the machine size (number of processors). In this analysis eight different pairs of values for fixed and variable cost per node are used. Five pairs are used to represent different possible conditions in the LOG and Single Stage models, while the last three, where the variable costs are zero, represent the similar conditions in the GAP model. The values in the table below were based on timings of random end to end communication patterns on an early release of the CM5 performed by both an internal Thinking Machines applications group and more independent sources.

Table 1

Model Parameters

Fixed	Variable	Machine State
2500	200	Current
1000	200	Current-Low
500	200	Potential
500	100	Potential
100	50	Ideal
5000	0	Current
3600	0	Current Low
1000	0	Potential

The first two pairs of values approximate the current fixed and variable cost on working machines running "off the shelf" software. The first pair (2500, 200) is an averaged approximation while the second pair is more idealized. With a 33Mhz clock, such as the current clock speed of the SPARC chip used in the CM5, for example, a 2500 cycle fixed cost and a 200 cycle per-node variable cost translates to approximately 75 and 6-7 microsecond costs, respectively. The next two pairs represent reductions in cost which may be possible within this generation. The fifth pair represents an ideal. The last three pairs attempt to replicate the three different states within the GAP model. The cost of a 32-bit floating point operation, in machine cycles, is estimated at 6 cycles. While on-chip computing speeds are rapidly increasing, this value attempts to approximate the current state without accelerators which have a "peak" rate of two operations per cycle.

When the message size is allowed to vary up to approximately 1000 words, a fourth parameter, the per-word spooling rate, is introduced. Experimental data suggested a 4 cycle per-word cost would be a reasonable value, with a sensitivity analysis up to approximately 12 cycles per-word.

Parallel Machine Size

The current generation of massively parallel machines is characterized by "medium-grain" machines typically consisting of approximately 256 to 16K processors. This analysis considers machines with 2^8, 2^{10}, 2^{12}, and 2^{14} processors.

5. Multigrid Algorithms and Implementations

The analysis presented here considers the standard V and F-cycle in two and three dimensions. This analysis considers only the simplest problems and solution schemes: model problems are considered on structured meshes spanning square and cubic domains. Explicit weighted Jacobi schemes are used to solve problems discretized using second order finite difference techiniques. The hierarchy of structured meshes is constructed using a coarsening ratio of two in each dimension. The cycling schemes execute two relaxation sweeps onthe downstroke and one on the upstroke.

The problems were implemented on the parallel models using simple, practical domain partitioning strategies. In two dimensions the finest mesh was simply partitioned into load-balanced square subdomains and mapped to the analogous processor in the 2D mesh of processors. In three dimensions, the domain was analogously partitioned and the processor mapping was only slightly more complicated and was within a factor of two of optimal.

6. Analysis Overview

The remainder of this paper presents the results and implications of the implementation of the standard multigrid algorithms on the three models of parallel computation. The following two sections present the performance predictions for the two and three dimensional V-cycle when fixed length messages are used to execute all of the required network communication. Next, the results of the same analysis are repeated with variable length messages where the message size is allowed to vary up to 1000 words. The results of an implementation of the 3D

V-cycle on the Single Stage model are then presented. Finally, the implications of the set of predictions are summarized.

7. The Standard V-cycle in Two Dimensions

The performance predictions for the two dimensional V-cycle on both the LOG and GAP models were not encouraging. On moderate sized machines, those with 1K to 4K processors, with a 2500 fixed communications cost (approx. 75 microseconds) , the models predicted speed-ups of only 55 times over the serial implementation. For larger machines, the speed-ups do not even reach 200 times. The table below shows the speed-ups of the V-cycle for different machine sizes under different assumptions of fixed and variable communications costs. The problem size is 1,000,000 points or 1000 points per dimension.

Table 2

Speed-Up
Two Dimensional V-cycle
with Fixed Length Messages

N^2 = 1,000,000

GAP and LOG Model Predictions

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	27.1	55.1	103.2	172.6
1000, 200	58.3	125.5	238.6	387.1
500, 200	94.4	218.8	424.0	660.9
500, 100	94.9	223.5	450.5	755.0
100, 50	190.4	585.7	1462.1	2881.7
5000, 0	19.5	39.3	74.4	128.6
3600, 0	19.8	40.4	79.3	147.0
1000, 0	58.7	128.6	255.4	453.2

Because the information provided by these models attempts to bridge the gap between abstract models of computation and machine-dependent benchmarks, interpreting the data is not straightforward. From a theoretical perspective these speed-ups are far from linear. On the other hand computing the wall clock time associated with these predictions, then scaling these model problem times to reflect the increased complexity of actual applications, produces running times which are unacceptably slow.

If the fixed cost of a communication can be reduced to 500 cycles or 15 microseconds with a 33MHz clock, the models predict speed-ups in the range of 200 times. Only in the ideal case where the fixed cost of a communication is 100 cycles or approximately 3.3 microseconds, do the speed-ups become somewhat attractive.

These discouraging predictions are a result of very high communications latencies. With a fixed problem size of 1,000,000 points, in this range of processors, the fine grid communications costs dominate both the cost of the computation and the cost of coarse grid communications.

Very fast processors and relatively slow network communication create very poor processor utilization in this range of processors and problem sizes. The increased cost of the microprocessors in these machines makes efficiency an important performance criterion. We define efficiency here as the ratio of the time spent on computation to the total time on both computation and network communication. The table below shows the efficiency predicted by the models for the two dimensional V-cycle using the same eight pairs of values for the fixed and variable cost of a network communication.

Table 3

Efficiency

Two Dimensional V-cycle
with Fixed Length Messages

N^2= 1,000,000

GAP and LOG Model Predictions

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	10.6%	5.39%	2.55%	1.13%
1000, 200	22.77%	12.29%	5.91%	2.53%
500, 200	36.88%	21.44%	10.51%	4.32%
500, 100	37.10%	21.90%	11.17%	4.95%
100, 50	74.41%	57.38%	36.36%	17.54%
5000, 0	5.62%	2.80%	1.33%	.60%
3600, 0	7.63%	3.85%	1.84%	.85%
1000, 0	22.94%	12.60%	6.33%	2.97%

Both the LOG and the GAP models predict very low efficiency levels when the fixed cost of a communication is high. With a fixed cost of 2500 cycles, small to modest sized machines, consisting of 256-1024 processors, reach only 5%-10% efficiency. With a fixed cost of 1000 cycles (approximately 30.3 microseconds using a 33Mh clock), efficiency is still only 10%-20%. Driving the fixed cost down to 500 cycles (15 microseconds) produces more reasonable levels of 20%-30% for modestly sized machines. To reach 40%-60% efficiency where the machine begins to leverage the power of these new microprocessors, the fixed cost needs to be reduced all the way down to the 100 cycle range (3.3 microseconds).

8. The Standard V-cycle in Three Dimensions

The implementation and analysis of three dimensional problems differs from the two dimensional analysis in several ways. First, the additional dimension increases the computational burden by a factor of O(N), to O(N^3), while increasing the required communication by a factor of $N/P^{1/6}$. Second, mapping the three dimensional problem domain to a two dimensional machine model tends to increase not only the complexity, but the distance of interprocessor communications. Third, the problem size in the analysis is increased by a factor of 1000, while still considering the same range of machine sizes.

The LOG and GAP models predict only slightly improved levels of performance for the three dimensional V-cycle. Table 4 below lists the speed-ups predicted by the models for three dimensional problems with one billion points.

Table 4

Speed-Up

Three Dimensional V-cycle with Fixed Length Messages

N^3 = 1,000,000,000

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	49.1	130.6	338.1	859.3
1000, 200	86.7	240.3	636.8	1632.5
500, 200	116.2	333.7	902.7	2332.2
500, 100	129.5	389.2	1102.2	2969.2
100, 50	202.7	702.6	2288.7	6954.7
5000, 0	30.1	79.2	205.3	526.7
3600, 0	39.9	106.8	279.7	722.5
1000, 0	102.3	302.4	855.2	2333.5

Generally, the predictions are not encouraging. The slight increase in performance is due to the increased amount of computation relative to both the amount of communication and the number of processors. For a 1024 processor machine with a 2500 cycle fixed communication cost, the LOG model predicts a speed-up of only 130 times. If the fixed cost of a communication drops to 500 cycles, this improves by a factor of 2-3. Only in the ideal case of a 100 cycle fixed communication cost do the results approach acceptable levels for moderately-sized machines and design tool levels, with thousand-fold speed-ups, for very large machines.

As with the two dimensional predictions, the sluggish predictions are due mainly to the overwhelming costs of the network communication. Table 5 below shows the efficiency levels which coincide with these speed-up predictions.

Table 5

Efficiency

Three Dimensional V-cycle with Fixed Length Messages

N^3 = 1,000,000,000

Processors	256	1024	4096	16,384
Fixed, Variable				
2500, 200	19.19%	12.75%	8.25%	5.26%
1000, 200	33.85%%	23.46%	15.54%	9.96%
500, 200	45.40%	32.59%	22.03%	14.23%
500, 100	50.58%	38.01%	26.92%	18.12%
100, 50	79.17%	68.61%	55.87%	42.45%
5000, 0	11.7%	7.7%	5.01%	3.22%
3600, 0	15.59%	10.42%	6.83%	4.40%
1000, 0	39.95%	29.53%	20.87%	12.42%

The predictions of these models are in contrast to the asymptotic predictions of more abstract models of computation. Asymptotic analysis suggests the fine grid communications costs become negligible as the problem size gets large for a fixed range of machine sizes. These results suggest the huge imbalance between the cost of communication per word and the cost of a floating point computation causes communication time to dominate the time spent on computation, even with one billion points.

The standard V-cycle algorithm alternates between computation and communication systolically, placing a heavy communications burden on a multi-stage interconnection network. On medium-grain multiprocessors, those with 256 to 16K processors, for realistic problem sizes, local, fine-grid communication is predominant. By the time the grids have coarsened beyond one point per processor, only a small fraction of the computation remains. This magnifies the importance of a small fixed cost per word and de-emphasizes the importance of low variable per-node communications costs. Unfortunately, the models in the previous section show the demand for inexpensive local communication is answered in the current generation of massively parallel machines by a high fixed communications cost producing discouraging levels of performance for both two and three dimensional problems.

9. The Standard V-cycle with Variable Length Messages

The previous analysis assumed all communication was accomplished through fixed length messages consisting of only a small constant number of words. Sensitivity analysis suggested that acceptable levels of performance required lower fixed costs per word. The low spooling rate per word exhibited by these machines motivates potentially lowering the average communication cost per word by transmitting large blocks of words per message. With large messages, the fixed cost of initiating a network communication can be amortized over a larger number of words, lowering the effective fixed cost per word.

In the analysis of this section, spooling costs are added to the communication cost functions of the previous section. The cost of a message is a function of the distance and the length in words, and is the sum of fixed start-up and receipt costs, variable per-node costs and spooling costs.

Experimental data suggest that approximating the total spooling costs as a linear function of message size is reasonable up to approximately 5000 words. The analysis here assumes a maximum message size of 1000 words and uses a per word spooling cost of 4 clock cycles. Approximating the spooling rate was accomplished with the help of timings provided by Pablo Tomayo of Thinking Machines, Inc. The rate was determined by a regression analysis on three node ping pong rates of message sizes ranging from 1 to 5000 words. Sensitivity analysis with rates up to 12 cycles per word showed the results of this section are relatively insensitive to small changes in the per-word spooling rate.

The predictions for the standard V-cycle algorithm in two dimensions with large message transmission were generally far more encouraging than the fixed message length predictions. The table below lists the speed-up and efficiency predictions for the same eight pairs of fixed and variable communications costs.

Table 6

Speed-Up

Efficiency

Two Dimensional V-cycle with Variable Length Messages

$N^2 = 1{,}000{,}000$

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	170.1 66.49%	314.0 30.76%	368.0 9.12%	354.5 2.32%
1000, 200	208.3 81.41%	501.8 49.17%	709.4 17.59%	715.5 4.69%
500, 200	225.1 87.99%	626.9 61.42%	1027.0 25.47%	1083.1 7.10%
500, 100	338.1 89.23%	667.1 65.36%	1197.3 29.69%	1360.8 8.92%
100, 50	446.1 96.21%	882.6 86.47%	2396.4 59.44%	3839.6 25.1%
5000, 0	132.5 51.77%	200.8 19.67%	216.5 5.36%	207.7 1.36%
3600, 0	152.8 59.72%	258.6 25.33%	294.2 7.29%	286.7 1.88%
1000, 0	213.8 83.55%	555.4 54.42%	882.9 21.9%	979.7 6.42%

These predictions show at least a factor of 6 speed-up on moderate-size machines and a factor of two speed-up on large machines over the fixed length predictions. For example, on a 1024 processor machine, with a fixed communications cost of 2500 cycles, with variable length messages, the speed-up predicted is 314 as compared to 55 on the models with constant message size. There is a corresponding improvement in the efficiency of 30% versus 5%. If fixed costs can be driven down to 500 cycles, the variable message length still provides approximately a factor of two improvement over the fixed length predictions.

With large messages reducing the fine grid communication costs, the coarse grid communications costs, which are proportional to $\log^2 P$, grow to counterbalance the computational speed-ups provided by additional processors. The increase in speed-up as the number of processors gets large is less pronounced. In addition, the optimal number of processors implied by this trade-off occurs in a more reasonable range. For example, with fixed and variable costs of 2500 and 200 cycles respectively, the models predict that the optimal number of processors for this computation is approximately 4900.

With the three dimensional V-cycle, the models suggest that the ability to send variable length messages, up to 1000 words, produces a marked increase in solution speed in this range of processors, on problems up to one billion points. The table below shows the speed-ups predicted for the three dimensional algorithm by the LOG and GAP models.

Table 7

Speed-Up

Three Dimensional V-cycle with Variable Length Messages

$N^3 = 1{,}000{,}000{,}000$

Processors (t_F, t_V)	256	1024	4096	16,384
2500, 200	253.3	1006.1	3965.4	15,192.1
1000, 200	253.9	1010.3	3999.2	15,555.7
500, 200	254.1	1011.7	4010.6	15,680.8
500, 100	254.2	1012.3	4016.9	15,773.9
100, 50	254.4	1013.7	4029.3	15,924.2
5000, 0	252.5	1000.3	3922.4	14,785.1
3600, 0	253.0	1004.2	3953.3	15,105.8
1000, 0	254.1	1011.5	4011.8	15,739.9

With variable length messages, the high fixed communications cost can be effectively amortized over a large number of words, driving down the average cost per word to a more ideal range. Computation costs dominate the total execution time, producing almost linear speed-ups in this range of problem to processor size. Almost all of the complementary efficiency levels are above 90% for each of the eight fixed, variable communications cost pairs throughout the entire range of machine sizes.

These results suggest the average communications cost per word can be driven down far enough through the efficient transmission of large messages to effectively leverage the increased computational speeds of the current generation of microprocessors. Thus, the ability to package messages into large blocks, up to a 1000 word maximum, can potentially bring these machines closer to the goal of design tool performance on these problems.

10. The F-Cycle

Performance predictions for the standard F-cycle were very similar to the V-cycle results. With both fixed, constant length and variable length message transmission, the F-cycle slightly outperformed the V-cycle. With fixed message lengths, this was due mainly to the reduced amount of fine grid communication of the F-cycle. With the ability to send large messages, the F-cycle out performed the V-cycle in three dimensions because of the reduction in the amount of required computation.

11. Standard V-cycle on a Single Stage Machine

The two and three dimensional V-cycle algorithms were implemented on the Single Stage model in order to try to determine the impact of a multi-stage network on the performance of multigrid algorithms. The single stage model assumes the processors are connected by a 2D mesh and all communication takes place along these physical connections. There is no overarching multi-stage communication network. The model is parameterized by the same machine dependent costs, namely, fixed and variable communications costs, spooling rates and floating point computation rates. The only difference in communications costs is in the variable, distance related cost component. In this model the distance a message must travel is simply the Manhattan distance (the L_1 norm) of the location of the sending and receiving processors on the mesh.

The results in both two and three dimensions suggest the impact of a multi-stage network on performance is very small, regardless of the maximum message length. The table below shows the increase in total time caused by sending messages through the mesh connections rather than through the logarithmic multi-stage network defined by the LOG model.

Table 8

The Percentage Increase in Total Time for the 3D V-cycle Implemented on the Single Stage Model from the Time Required On the Multi Stage LOG Model

Number of Processors	% Difference M=1	% Difference M=1000
256	5.88%	.05%
1024	8.17%	.19%
4096	11.54%	1.19%
16,384	16.47%	8.79%

The table shows a less than 10% increase on moderate sized machines with fixed message length communication, where the fixed and variable costs of a communication are 2500 and 200 cycles respectively. For machines with variable length message capability, the increase in total time is less than 1% for moderate machines.

In three dimensions, the increase in communications costs alone with variable length messages is small, except on very large machines. The table below isolates the communications costs and shows the percentage increases.

Table 9

The Percentage Increase in Communications Time for the 3D V-cycle Implemented on the Single Stage LPSS Model from the Time Required On the Multi Stage LOG Model

Number of Processors	% Difference M=1000
256	4.38%
1024	10.87%
4096	37.30%
16,384	120.91%

The table shows the small increase in communications costs with only a single stage permutation network. For very large machines the increase is only slightly over a factor of two. These results suggest that even for very large machines with fixed length messages, the addition of multi-stage networks does not seem to enhance performance enough to justify the additional machine complexity.

12. Conclusions

The performance predictions presented here suggest the fixed cost of a communication on the current generation of massively parallel machines needs to be driven down into the range of 15 microseconds to produce acceptable levels of performance. Ideally, the cost should be in the range of 3 microseconds. The computational speeds of the next generation of microprocessors appear to be increasing rapidly. Though these and other hardware advances may produce enhanced performance, they will certainly exacerbate the huge disparity between the speeds of on-chip and network events. Driving the average cost of a local communication appears to be imperative if these machines are to become efficient platforms for the the solution of multigrid applications.

One way to accomplish this reduction in the average cost per word of a network communication may be through the efficient transmission of large messages. This capability would allow the fixed cost of a communication to be amortized over a large number of words.

Finally, expensive multi-stage networks appear to have little impact on the performance of standard multigrid algorithms. In this range of problem to machine sizes, with both fixed and variable length message transmission, performance degrades only slightly when communication is forced to traverse the physical connections of a 2D mesh of processors.