1988

Non-Algorithmic Load Imbalance Effects for Domain Decomposition Methods on a Hypercube

Dan C. Marinescu

John R. Rice
Purdue University, jrr@cs.purdue.edu

Report Number:
88-832
NON-ALGORITHMIC LOAD IMBALANCE EFFECTS FOR DOMAIN DECOMPOSITION METHODS ON A HYPERCUBE

Dan C. Marinescu
John R. Rice

CSD-TR-832
December 1988
Revised January 1989
NON-ALGORITHMIC LOAD IMBALANCE EFFECTS FOR DOMAIN DECOMPOSITION METHODS ON A HYPERCUBE

Dan C. Marinescu*
John R. Rice*

Computer Science Department
Purdue University
West Lafayette, IN 47907

Abstract

In this paper we discuss the effects of non-algorithmic load imbalance for three synchronization structures which appear in different algorithms using domain decomposition methods for solving PDE’s on a hypercube machine. The synchronization structures are determined from the conditions related to the partitioning of a hypercube. To characterize the load imbalance effects we introduce a factor \( \psi \) defined as the ratio of the load imbalance costs to the total parallel execution time in absence of any load imbalance. We develop non-deterministic models for SPMD (Same Program Multiple Data) execution model and compute the \( \psi \) factor for the three structures, for different types of distributions of the execution time.

1. INTRODUCTION

Recent experiments with highly efficient parallel solutions for three scientific problems on a hypercube machine, [4], have shown that the communication and control costs of a parallel computation can be significantly reduced. In the same time concerns that the load imbalance may be a more serious problem than it was previously considered have been raised, and several new potential sources of load imbalance have been identified. Also a new measure for the parallel speed up, the \( \text{scaled speed up} \), was defined [3], and it was conjectured that, as a first approximation, the parallel part of a program scales up with the problem size while the scalar part does not.

In a previous paper, [6], we have proposed a nondeterministic model of computation which takes into account the load imbalance among processors executing in parallel. We have pointed out that algorithmic as well as non-algorithmic effects may contribute to load imbalance in case of a synchronized parallel execution. The algorithmic load imbalance effects can be minimized by redistributing computations among processors, and by overlapping communication with computation. But a perfect load balance is unattainable since non-algorithmic effects like hardware and software errors and retries, or data dependent execution time, cause load imbalance. It is clear that such effects are more difficult to control and impossible to eliminate entirely. Though communication and control costs are likely to be the primary source of inefficiency for medium size parallel systems, the load imbalance will probably be a factor of increasing importance in massively parallel systems. We have concluded that synchronization

* Work supported in part by the Strategic Defense Initiative under Army Research Office contract DAAL03-86-K-0106.
may be the most important effect in many important applications and that further empirical and theoretical studies of the load imbalance and its effects are necessary.

In our model we discuss the issue of load imbalance in the context of synchronization. We view a parallel computation as a sequence of \( n \) synchronization epochs each using \( I_i \) processors, and we express the expected execution time of a parallel computation as

\[
T^* = E(T_c) = \alpha + n\beta + \sum_{i=1}^{n} \mu_i(1 + \Delta_i)
\]

(1.1)

In this expression \( \alpha = \alpha_1 + \alpha_2 \sum_{i=1}^{n} q(I_i) \) represents the cost due to control of the parallel computation. The communication costs are denoted by \( \beta \). Note that \( \beta \) reflects only the cost of communication which cannot be overlapped with computations. It is assumed that the execution time of the \( I_i \) processors active in any epoch \( i \), \( X_{i,j} \) are independent, identically distributed random variables with mean \( \mu_i \), and variance \( \sigma_i \). The average cost attributed to the load imbalance in epoch \( i \) is \( \Delta_i = \Delta(C_i, I_i) \), with \( C_i \) the coefficient of variation of the distribution of \( X_{i,j} \), \( C_i = \frac{\sigma_i}{\mu_i} \).

The expected serial execution time of a computation consisting of \( n \) synchronization epochs, with \( I_i \) processors active in each epoch, and with \( \mu_i \) the expected execution time per processor in each epoch is:

\[
T_S = \alpha_1 + \sum_{i=1}^{n} \mu_i I_i
\]

(1.2)

with \( \alpha_1 \) the algorithmic component of the control cost in case of serial execution. Note that \( \mu_i I_i \) is the expected total processor time per epoch, the time necessary for a single processor to carry out the parallel component of the computation.

In the framework of our model the speed up with \( P \) processors, \( S_P \), \( (I_i = P \text{ for all } i) \) is defined as:

\[
S_P = \frac{\alpha_1 + P \sum_{i=1}^{n} \mu_i}{\alpha + n\beta + \sum_{i=1}^{n} \mu_i(1 + \Delta_i)}
\]

(1.3)

It is interesting to note that \( S_P \) as defined by expression (1.3) corresponds to the same concept of scaled execution as discussed in [4]. When defining \( S_P \) we consider the serial execution time for a given parallel computation instead of determining the parallel execution time of a given computation on a single processor as the standard speed up is defined. In our model, the costs related to a parallel execution are dependent upon the number of processors executing in parallel. For example the load imbalance costs increase in case of a scaled execution [6].

To characterize the load imbalance effects we introduce in this paper a factor \( \psi \), the \textit{load imbalance factor} defined as the ratio of the total load imbalance costs to the
parallel execution time in absence of any load imbalance effects. To specialize the model to non-algorithmic load imbalance we assume that all processors active in any synchronization epoch execute the same program upon different data. Then it is reasonable to consider that the execution times of all processors active in any synchronization epoch $i$ are independent identically distributed random variables with mean $\mu_i$, and variance $\sigma_i$. Then the load imbalance factor $\psi$ is defined as

$$\psi = \frac{\sum_{i=1}^{n} \mu_i \times \Delta_i}{\sum_{i=1}^{n} \mu_i}$$

with $\Delta_i$ the load imbalance costs for synchronization epoch $i$ with $I_i$ processors active.

In general, the communication, control and load imbalance costs, $\alpha$, $\beta$, and $\Delta$ depend upon the algorithm, the architecture of the parallel system, and the number of processors executing in parallel.

Explicit expressions for these costs for a given algorithm, or a class of algorithms, and for a particular architecture, or a class of machines are of interest. In this context we have investigated a class of problems related to the domain decomposition methods for solving of PDE's. We have considered only models for the nearest neighbor communication, when the communication costs are independent of the number of processors executing in parallel, and of the epoch, $\beta_i = \beta = const$. We have made a conservative assumption concerning the control costs related to the parallel execution, namely we have assumed that they are a linear function of the number of processors.

As far as the load imbalance is concerned we have shown that for any distribution of the execution time the load imbalance costs for a synchronization epoch can be expressed as $\Delta_i = f(I_i) \times g(C_X)$ where $C_X$ is the coefficient of variation of the distribution $F_X$ of the execution times. For several distributions, we have computed exact expressions for $\Delta_i$ [6].

For the uniform distribution we have $g(C_X) = C_X \sqrt{3}$ and

$$f(I_i) = \frac{I_i - 1}{I_i + 1}.$$  \hspace{1cm} (1.5)

For the exponential distribution we have $g(C_X) \equiv 1$ and

$$f(I_i) = \log I_i + C$$ \hspace{1cm} (1.6)

with $C$ is Euler's constant, $C = 0.577$. For the standard normal distribution we have $g(C_X) \equiv 1$ and

$$f(I_i) = (2 \log I_i)^\nu - \frac{1}{2} \left[ 2 \log I_i \right]^{-\nu} \left( \log \log I_i + \log 4\pi - 2C \right)$$ \hspace{1cm} (1.7)

We have also [6] derived upper bounds for general distributions and we have observed that for particular distributions, for example the uniform distribution, the cost of load imbalance does not depend upon number of processors running in parallel, when this number is large. That is $f(I_i) = const$, and hence effective use of massive parallelism is
possible.

We have recognized data dependent MFLOP $S$ rates as a potential source of non-algorithmic load imbalance. Several additional sources of non-algorithmic load imbalance in massively parallel computing are discussed for the first time in [4]. They are: (1) hardware induced effects like error correction related to the failures of memory modules, (2) high memory refresh rates and (3), hardware-software interactions like communication errors causing message retransmissions.

In view of these results it seems that the control, the communication, and the algorithmic load imbalance costs of a parallel computation can be minimized by a complex of programming techniques and algorithm design methods, but it seems rather difficult to control the effects of non-algorithmic load imbalance. Empirical studies based on real applications are necessary in this area.

Our goal is to facilitate algorithm design by investigating different synchronization structures and to estimate the cumulative effects of load imbalance. The paper is organized as follows. The synchronization structures are presented in the next section, followed by a discussion of applications involving these structures. The final section contains the analysis of the three structures.

2. THREE SYNCHRONIZATION STRUCTURES

We discuss three synchronization structures which can be used in conjuncture with domain decomposition techniques for solving PDE's. We restrict our analysis to the effects of load imbalance. The three synchronization structures use at most $N = a^K$ processors. These structures are presented in Figure 1 for the particular case $a = 2$ and $K = 3$.

The first structure (Figure 1a) is characterized by the following properties:

P1. The computation consists of $K + 1$ epochs and the number of active processors in the $i$-th epoch is $I_i = a^{K-i}$ with $a > 1$. In the first epoch there are $I_0 = N = a^K$ active processors. It follows that $\Delta_i > 0$ for $i > 0$ and $\Delta_K = 0$ since there is only one processor active during the last epoch.

P2. The execution time of all tasks in all synchronization epochs are independent, identically distributed random variables $X_{i,j}$ with mean $\mu_X$, variance $\sigma_X$ and coefficient of variation $C_X$.

P3. There is a global synchronization among all tasks of a given epoch.

The second structure (Figure 1b) is characterized by the following properties:

P1: The same number of active processors as the first structure.

P2: The same assumption on the execution times as for this structure.

P3: Within a given epoch, the tasks are synchronized in groups of $a$ processors. There is no global synchronization between epochs and, as soon as a related set of tasks completes in epoch $i$, the descendent task commences in epoch $i + 1$. 
Figure 1: The three synchronization structures for $a = 2$ and $k = 3$. The structures are (1a) an exponentially decreasing number of processors with rate $a$, (1b) a tree with $a$ processors in parallel at each branch, and (1c) parallel sequential streams with $a$ processors in parallel at each epoch in each stream.
The third structure (Figure 1c) is characterized by the following properties:

P1: The number of processors active in the same $N$ in all epochs, except the last, where only 1 is active.

P2: The same assumption on the execution times as for the first structure.

P3: Within a given epoch, the tasks are synchronized in groups of $a$ processors within $N/a$ sequential streams of task pairs. There is global synchronization just before the last task begins.

3. APPLICATIONS INVOLVING THESE SYNCHRONIZATION STRUCTURES

We believe that the synchronization structures shown in Figure 1 are common among those that arise in parallel computation. We note some applications here to illustrate the variety that exists. We do not attempt to explain them in detail, as that detracts from the object of this paper.

A principal source is in models of physical phenomena (e.g., heat flows, electromagnetic forces, stresses, air flows) which are modeled by differential equations in 1 to 4 physical dimensions. These phenomena are inherently local in time and space. They are inherently synchronized in time, but loosely synchronized in space; space synchronization comes through the time for effects to propagate through space via local interactions.

Computations modeling these phenomena can exploit this loose coupling in space to achieve parallelism. The principal technique is called domain decomposition, where physical space is decomposed into a large number of domains. Since interactions between these domains is local, this allows one to use locally connected computer architectures effectively. See [6], [7], [8], and [10] for previous work of ours, which include descriptions of this approach at a fairly high level. There is an enormous literature on the mathematical analysis of specific instances of this general approach, this is currently one of the most intensively studies areas of numerical computation.

The basic technique is to compute the results in the interior of a particular domain and then communicate the new state to neighboring domains for their use. Important characteristics of such computation are as follows:

1. The interior computations and data are usually large compared to data to be communicated. One of the objectives of algorithm design is to be sure this is so, it follows naturally if one chooses domain shapes that have small "surface" compared to "volume" (e.g., nearly spheres or cubes).

2. The interior computations are usually similar (use the same program), but rarely identical (have different data) due to variations in shapes, materials, intensities of physical effects, etc.

3. Synchronization in time is essential. Some models of the physics may compensate for small time asynchronizations locally, and more as domains become separated in space. Many algorithms have an artificial time (e.g., iteration numbering) which has the same characteristics as real world time.
Thus the real world seems well suited to partitioned or hierarchical parallel machines such as the Butterfly, Cedar [1], hypercubes, Multi-FLEX [12], PASM [13], as well as the shared memory, "homogeneous" machines such as the Elexi, RP3 or Sequent.

There are other applications such as graphics, image processing, searching unstructured data and text processing where parallelism is inherent in the nature of the problem. The structures considered all occur in specific examples of these applications.

The key ingredients in the analysis of these applications for parallel computations are:

* the ratio of computing \((X_{i,j})\) costs to communication costs \((\beta)\),
* the ability to partition the problem so processors have equal expected computational loads (the \(X_{i,j}\) have the same mean \(\mu_i\)),
* the distribution of the \(X_{i,j}\) execution times.

In many of the applications, it is practical to allocate the expected computational loads fairly equally. Each processor is assigned an equal volume of space, an equal area of a display or an equal amount of text to typeset. Of course, one can easily construct examples where this allocation is not easy, but even then, one can expect that an effective allocation is made. The distribution of the \(X_{i,j}\) can also vary greatly, but again we expect that many, if not most, applications will have the \(X_{i,j}\) distributed "tightly" about the mean \(\mu_i\). Thus, a model assuming a uniform or normal distribution for the \(X_{i,j}\) will be appropriate for many applications.

In Table 1, we present a list of applications along with rough orders for the ratio of computation and communication costs. The entries are only orders of magnitude, numerical factors can vary greatly. The quantity \(S\) merely measures the size or "bulk" of the computation assigned to a processor, it is not the same from line to line, nor is it necessarily a variable that appears in an algorithm for these applications. We see that this ratio increases with problem sizes for all these applications, sometimes dramatically. In machines that can overlap communication with computing, the effective communication cost (the value of \(\beta\)) may be much less than the values indicated in Table 1.

4. ANALYSIS OF THREE SYNCHRONIZATION STRUCTURES

In this section we investigate the load imbalance costs for the three synchronization structures discussed previously. It is assumed that the execution times in all synchronization epochs have the same distribution \(X\). The strategy is to compute the total load imbalance costs and then the load imbalance factor, \(\psi\), for each structure and for several distributions of the execution times.

4.A The First Synchronization Structure

Let us denote by \(\Delta_{a,k}^{(1)}\), the total load imbalance for the first structure

\[
\Delta_{a,k}^{(1)} = \sum_{i=0}^{K-1} \Delta_i
\]
Table 1. Applications Using Hierarchical Synchronization Structures. The order of magnitude of the computing/communication ratio is given as a function of the size of the computation assigned to a processor.

<table>
<thead>
<tr>
<th>Application</th>
<th>Ratio of Computing to Communication</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Dimensions</td>
</tr>
<tr>
<td>Domain decomposition for PDEs</td>
<td></td>
</tr>
<tr>
<td>* using Gauss elimination (multifront methods)</td>
<td>$S^4/S = S^3$</td>
</tr>
<tr>
<td>* using SOR iteration</td>
<td>$S^3/S = S^2$</td>
</tr>
<tr>
<td>* using FFT method</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>* multigrid iteration</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>Time dependent PDE solutions</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>* explicit methods</td>
<td></td>
</tr>
<tr>
<td>Gauss elimination iteration</td>
<td>$S^4/S = S^3$</td>
</tr>
<tr>
<td>* implicit methods</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>* ADI methods</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>Newton iteration for nonlinear PDE</td>
<td></td>
</tr>
<tr>
<td>* Gauss elimination</td>
<td>$S^4/S = S^3$</td>
</tr>
<tr>
<td>* iteration</td>
<td>$S^3/S = S^2$</td>
</tr>
<tr>
<td>* FFT methods</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>Graphics</td>
<td></td>
</tr>
<tr>
<td>* simple display</td>
<td>$S^2/0 = S$</td>
</tr>
<tr>
<td>* connected displays (contours)</td>
<td>$S^2/S = S$</td>
</tr>
<tr>
<td>* feature extraction</td>
<td>$(S^2 \rightarrow S^3)/S = S \rightarrow S^2$</td>
</tr>
<tr>
<td>Search in amorphous data</td>
<td>$S/1 = S$</td>
</tr>
<tr>
<td>Text processing (Tex, troff)</td>
<td>$S/1 = S$</td>
</tr>
<tr>
<td>Numerical integration</td>
<td>$S^2/1 = S^2$</td>
</tr>
<tr>
<td>Discretization of PDEs</td>
<td>$S^2/S = S$</td>
</tr>
</tbody>
</table>
In this case the speed up is

\[ S_p^{(1)} = \frac{\alpha_1 + \mu_X \frac{a^{K+1} - 1}{a - 1}}{\alpha^{(1)} + (K\beta^{(1)}) + \mu_X(K + 1) + \mu_X\Delta^{(1)}_{a,k}} \]  

(4.2)

The effects of the load imbalance are characterized by \( \psi^{(1)}_{a,k} \) defined as the ratio of the expected increase of the execution time due to load imbalance, to the parallel execution time in absence of any load imbalance.

\[ \psi^{(1)}_{a,k} = \frac{\mu_X\Delta^{(1)}_{a,k}}{\mu_X (\log_a N + 1)} = \frac{\Delta^{(1)}_{a,k}}{K + 1} \]  

(4.3)

The case when \( a = 2 \) and \( K = \log_2 N \) is of special interest. Then we have

\[ S_p^{(1)} = \frac{\alpha_1 + \mu_X(2N - 1)}{\alpha^{(1)} + \beta^{(1)} \log_2 N + \mu_X \log_2 N + 1 + \mu_X\Delta^{(1)}_{a,N}} \]  

(4.4)

\[ \psi^{(1)}_{2,k} = \frac{\Delta^{(1)}_{2,k}}{K + 1} \]  

(4.5)

Exact expressions for \( \Delta^{(1)}_{a,k} \) can be obtained for the uniform, the exponential and the normal distributions of the execution time in each epoch. After presenting these results we give an upper bound for the case when the distribution \( F_X \) of \( X \) is not known.

**The Uniform Distribution**

In this case the ratio of the load imbalance costs to the parallel execution time is given by

\[ \psi^{(1)}_{a,k} = C_X \sqrt{3} \frac{[K - 2 \times Q_{a,k}]}{K + 1} \]  

(4.6)

with

\[ Q_{a,k} = \sum_{i=0}^{K-1} \frac{1}{1 + a^i} \]  

(4.7)

This expression is derived using the load imbalance cost for a synchronization epoch with a uniform distribution of the execution time

\[ \Delta_i = C_X \sqrt{3} \frac{I_i - 1}{I_i + 1} = C_X \sqrt{3} \frac{a^{K-i} - 1}{a^{K-i} + 1} \]

Then the total load imbalance costs are

\[ \Delta^{(1)}_{a,k} = \sum_{i=0}^{K-1} \Delta_i = C_X \sqrt{3} [K - 2Q_{a,k}] \]
Let us observe that

$$\sum_{i=0}^{\infty} \frac{1}{1 + a^i} = \frac{1}{2} \Phi_{1,1} (1, -1; -a; 1)$$

with $\Phi$ the basic generalized hypergeometric function. It seems nontrivial to derive exact expressions for $Q_{a,K}$ and we derive bounds for it. We see immediately that

$$\frac{A}{a} \leq Q_{a,K} \leq A,$$

with

$$A = \frac{a^K - 1}{a^{K-1}(a-1)} = \frac{a}{a-1} \left( 1 - \frac{1}{N} \right) = \frac{a(N-1)}{N(a-1)}.$$

It follows that

$$C_X \sqrt{3} (K - 2A) \leq \Delta_{a,K}^{(1)} \leq C_X \sqrt{3} (K - 2 \frac{A}{a}).$$

Figure 2. The ratio of load imbalance costs to the parallel execution time in absence of any load imbalance effects as function of the problem size for the first synchronization structure. The execution time has a uniform distribution with coefficient of variation $C_X$. 
When \( a = 2 \) and \( N \) is large \( A = 2 \). In this case

\[
C_X \sqrt{3} (K - 4) \leq \Delta_{2}^{(1)} \leq C_X \sqrt{3} (K - 2).
\]

To conclude the discussion of the uniform distribution case for the first synchronization structure we present \( \psi_{2,k}^{(1)} \) in Figure 2 for the binary case, \( a = 2 \), and for different values of \( C_X \). For small values, say \( C_X = 0.01 \), the effect of load imbalance is hardly noticeable. For larger \( C_X \), the load imbalance can add as much as 30\% to the parallel execution time when \( N \) is large \( (N > 2^{32}) \).

![Figure 3](image.png)

**Figure 3.** The ratio load imbalance costs to the parallel execution time in absence of any load imbalance effects as function of the problem size for the first synchronization structure. The execution time has an exponential distribution and \( a = 2 \).

**The Exponential Distribution**

In this case the ratio of the load imbalance costs to the parallel execution time is given by

\[
\psi_{a,k}^{(1)} = \frac{K}{K + 1} \left( C + \frac{K + 1}{2} \times \log a \right)
\]  

(4.8)
The total load imbalance costs are

\[ \Delta_{\text{a,k}}^{(1)} = K \left[ C + \frac{K + 1}{2} \times \log a \right] \]

This expression is derived using the fact that

\[ \Delta_i = \log I_i + C \]

with \( C = 0.577 \), Euler's constant, and \( I_i = a^i \). It follows immediately that when \( N = 2^K \) that

\[ \Delta_{\text{a,k}}^{(1)} = K \left[ C + \frac{K + 1}{2} \times \log 2 \right] \]

and

\[ \psi_{\text{a,k}}^{(1)} = 0.0752 \times K + 0.577 \times \frac{K}{K + 1} \quad (4.9) \]

Figure 3 presents the ratio between the load imbalance costs to the parallel execution time in absence of any load imbalance effects as function of the problem size for the exponential distribution. We observe the linear increase in the load imbalance factor shown by (4.9).

**The Normal Distribution**

Let us consider first the case of a standard normal distribution. In Appendix 1 we show that

\[ \psi_{\text{a,k}}^{(1)} = \frac{1}{K + 1} \left[ A(a) \cdot S_1(K) - B(a) \cdot S_2(K) - \frac{1}{2A(a)} \cdot S_3(K) \right] \quad (4.10) \]

with

\[ A(a) = (2 \log a)^{1/2} \]

\[ B(a) = \frac{1}{2A(a)} \left[ \log 4\pi - 2C + \log \log a \right] \]

\[ S_1(K) = \sum_{i=1}^{K-1} (K - i)^{1/2} = \sum_{i=1}^{K-1} (i)^{1/2} \]

\[ S_2(K) = \sum_{i=1}^{K-1} (K - i)^{-1/2} = \sum_{i=1}^{K-1} (i)^{-1/2} \]

\[ S_3(K) = \sum_{i=1}^{K-1} (K - i)^{-1/2} \log (K - i) = \sum_{i=1}^{K-1} i^{-1/2} \log i \]
When \( a = 2 \), the coefficients \( A \) and \( B \) have the following values: \( A = 0.779 \), and \( B = 0.3713 \).

![Figure 4. The ratio load imbalance costs to the parallel execution time in absence of load imbalance effects as function of the problem size for the first synchronization structure. The execution time has a \((\mu, \sigma)\) normal distribution with \( \mu = 1 \).](image)

Let us now consider the case of a \((\mu, \sigma)\) normal distribution. The derivation of the results is presented in Appendix 1. The ratio of load imbalance costs to the parallel execution time for the first structure is

\[
\psi^{(1)}_{2, K} = \frac{C_X \cdot \mu_X}{K + 1} \left( A(a) \cdot S_1(K) - B(a) \cdot S_2(K) - \frac{1}{2A(a)} \cdot S_3(K) \right) \quad (4.11)
\]

with \( A(a) \), \( B(a) \), \( S_1(K) \), \( S_2(K) \), \( S_3(K) \) defined previously.

The results are presented in Figure 4. We see that for relative small values of the coefficient of variation, e.g., for \( C_X < 0.05 \), the load imbalance increases the execution time only slightly by 10 to 20\% even for large computations. For larger coefficients of variation the increase in the execution time grows more rapidly with the number \( N \) of
processors and is about 100% for $C_X = 0.19$, and $N = 2^{48}$.

**An Upper Bound for A General Distribution**

We conclude the discussion of the first structure by deriving an upper bound for $\psi_{a,k}^{(1)}$ for the case when the distribution function of $X$ is continuous, strictly increasing. We show that in this case

$$\psi_{a,k}^{(1)} = \sigma_X \times \frac{D(a,K)}{\sqrt{2} \times (K + 1)} \quad (4.12)$$

with $D(a,N)$ given by the following expressions

$$D(a,K) = \begin{cases} 
D' & \text{if } K = 2K' \quad (K \text{ even}) \\
D' + \left[ a^{K' + 1} - \frac{1}{a^{K'}} \right] & \text{if } K = 2K' + 1 \quad (K \text{ odd})
\end{cases} \quad (4.13)$$

where

$$D' = \frac{1 + \sqrt{a}}{a - 1} \left[ a^{K'} - 1 \right] \left[ a - \frac{1}{\sqrt{a} \, a^{K' - 1}} \right]$$

According to [6] we have

$$\Delta_i \leq C_X \frac{I_i - 1}{\sqrt{2} I_i - 1}$$

In our case $I_i = a^i$ and it can be seen easily that

$$\frac{a^i - 1}{\sqrt{2} a^i - 1} \leq \frac{a^{i-1} - 1}{\sqrt{2} a^{i-1}} \quad \text{for } a > 1$$

But

$$\frac{a^i - 1}{\sqrt{2} a^{i-1}} = \frac{1}{\sqrt{2}} \left[ a^{\sqrt{a}^{i-1}} - \frac{1}{\sqrt{a}^{i-1}} \right]$$

and

$$\Delta_{a,k}^{(1)} \leq \frac{C_X}{\sqrt{2}} D(a,K)$$

Then (4.12) follows immediately.

**4.B The Second Synchronization Structure**

Let us now consider the second structure, presented in Figure 1b. Its speed up is given by an expression similar to (4.2) but with a coefficients $\beta^{(2)}$ to reflect the communication costs for this structure and $\alpha^{(2)}$ for control costs. Let us call $\Delta_{a,N}^{(2)}$ the total load imbalance cost for this structure.
Proposition

For any distribution of the execution time the load imbalance costs for the first synchronization structure are an upper bound for the load imbalance costs of the second structure with the same number of elements.

$$\Delta_{d,k}^{(2)} \leq \Delta_{d,k}^{(1)}$$

The proof is based upon the following observation. For any distribution of the execution time the execution time including any load imbalance effects for the second structure is smaller than the execution time for the first structure. Since the expected execution time of a given processor is the same in both cases it follows that the load imbalance costs for the second case are smaller.

To prove that the execution time for the second structure is always smaller than the one for the first structure let us consider the simple structure presented in Figure 5. Let $X_i, X_j, X_k, X_l, X_m$ and $X_n$ be independent, identically distributed, random variables representing the execution times on processors executing in parallel, subject to synchronization condition as shown in Figure 5.

The following expressions give the total execution time, $Z$, as well as the partial execution time, $Y_q$, and $Y_p$:

$$Y_q = \max(X_i, X_j)$$

$$Y_p = \max(X_k, X_l)$$

$$Z = \max[(X_m + Y_q), (X_n + Y_p)]$$

The following inequality follows immediately:

$$Z = \max[(X_m + \max(X_i, X_j)), (X_n + \max(X_k, X_l))] \leq \max(X_m, X_n) + \max(X_i, X_j, X_k, X_l).$$

But $\max(X_m, X_n) + \max(X_i, X_j, X_k, X_l)$ is precisely the execution time for the first structure with four processors active in the first epoch (the corresponding execution time are $X_i, X_j, X_k, X_l$), and two active in the second epoch (the corresponding execution time are $X_m, X_n$).
The results shows in Figures 2 to 4 are to estimate the load imbalance costs for the second structure. Note that closed form expressions for the distribution function of the parallel execution time can be derived but it is impractical to construct them.

4.C The Third Synchronization Structure

Let us now discuss the third structure presented in Figure 1c. The scaled speed up is given in this case by the following expression

\[ S^{(3)} = \frac{\alpha_1 + \mu_X (KA^K + 1)}{\alpha^{(3)} + (K\beta^{(3)}) + \mu_X(K + 1) + \mu_X\Delta_{\alpha,K}^{(3)}} \]

with \(\alpha^{(3)}\) the control costs, \(\beta^{(3)}\) the communication costs and \(\Delta_{\alpha,K}^{(3)}\) the load imbalance cost for this structure.

The methodology to compute the value of \(\Delta_{\alpha,K}^{(3)}\) is slightly different in this case. We first compute the first moment of the random variable \(Z\) representing the total parallel computation time, and, knowing the expected duration of the computation in absence of any load imbalance effects, \(\mu_X[1 + K]\), we then are be able to compute the load imbalance costs.

The structure in Figure 1c is a composite structure consisting of three substructures as shown in the Figures 6a, 6b and 6c.

P1. The first substructure consists of a synchronization epoch with \(K\) elements followed by a single execution as shown in Figure 6a. Its total duration is denoted by \(Z\).

P2. Each element of the synchronization structure at the previous level consists of a sequence of \(K\) elements (see Figure 6b). The execution time of one of these sequence is denoted by \(Y^{(i)}\).

P3. Each element at the previous level is a synchronization structure with \(a\) parallel processors, see Figure 6c. The execution time of the corresponding synchronization epoch is denoted by \(Y_{i,j}\).

For this structure we examine only the case when \(X_1, \ldots, X_a\) are independent random variables with a uniform distribution with mean \(\mu_X\) variance \(\sigma_X\) and coefficient of variation \(C_X\).

Appendix 2 presents the derivation of the load imbalance factor and shows that it has the following expression

\[ \psi_{\alpha,K}^{(3)} = \frac{K}{K + 1} \times C_X \times [q_1(a) + q_2(a,K,C_X) + q_1(a)q_2(a,K,C_X)] \quad (4.14) \]

with

\[ q_1(a) = \sqrt{\frac{a - 1}{a + 1}} \quad (4.15) \]
\[ q_2(a, K, C_x) = \frac{2\sqrt{\frac{3aK}{a+2}}}{(a+1) + C_x\sqrt{3(a-1)}} \times \left[ (2 \log K)^{\frac{1}{2}} - \frac{1}{2} (2 \log K)^{-\frac{1}{2}} \times (\log \log K + \log 4\pi - 2C) \right]. \] (4.16)

Figure 6. The details of the third synchronization structure.

Figure 6a.

Figure 6b.

Figure 6c.
Figure 7. The ratio of load imbalance costs to the parallel execution time in absence of any load imbalance effects as function of the problem size for the third synchronization structure. The execution time has a uniform distribution.

When $a = 2$ we have

\[ q_1(2) = \frac{1}{\sqrt{3}} \]

\[ q_2(2, K, C_X) = \frac{\sqrt{2K}}{\left(\sqrt{3} + C_X\right)} \times \left[ (2 \log K)^{1/3} - \frac{1}{2} (2 \log K)^{-1/3} \times (\log \log K + \log 4\pi - 2C) \right] \]

with $C = 0.577$, the Euler’s constant.

Figure 7 presents the variation of the load imbalance factor for several values of $C_X$. We observe that for large values of $K$ the load imbalance costs are significant and the execution time doubles for $C_X = 0.1$.

The results are shown in Figure 7. The effects of load imbalance are much larger than for the previous two structures (compare with Figure 2). For example, with $C_X = 0.10$ and $N = 2^{48}$ we have a 110% increase here compared to 15% with the first synchronization structure. For $C_X = 0.20$ and $N = 2^{48}$ these numbers become 300% and 30%, respectively. The fact is not really surprising as the largely uncoupled nature
of this structure's synchronization allows large differences to build up prior to the final synchronization point.

CONCLUSIONS

We examine three synchronization structures which are useful for solving PDE's using domain decomposition methods on machines with a hypercube or similar architecture. Since existing instrumentation for parallel systems does not allow direct measurements of synchronization costs, we have restricted our study to analytical modeling. We construct non-deterministic models of SPMD (Same Program Multiple Data) parallel execution. We assume independent, identically distributed execution times on every processor. To investigate the relationship among different performance measures of the parallel execution and the problem size we have adopted the scaled execution view point as presented in [4] and [6].

Our analysis is focused on load imbalance effects which appear in such systems. A factor \( \psi \) representing the ratio of the total load imbalance costs to the parallel execution time in absence of any load imbalance is introduced and we give closed form expressions for this factor for the first and third synchronization structures. We also show that the second structure load imbalance costs are bounded by those of the first structure.

We analyze three distributions of the execution time for the first structure, namely the uniform distribution, the exponential and the normal distribution. The corresponding expressions for \( \psi \) are given by (4.6), (4.8), (4.10) and (4.11). An upper bound for any continuous, strictly increasing distribution of the execution time is given by (4.12). The values of \( \psi \) for the first structure in the binary case, \( a = 2 \), for a range of values of \( C_X \), the coefficient of variation of the execution time are represented in Figures 2, 3, and 4 for the three distributions analyzed.

For the third structure we analyze only the case of the uniform distribution of the execution time, and (4.14) gives the corresponding expression for the load imbalance factor. The load imbalance costs are substantially higher for the third structure.

In general \( \psi \) depends upon the coefficient of variation of the distribution of the execution time on one processor and upon the computation size. We show that the load imbalance effects lead to an increase of the execution time ranging from a few tens of percent to a multifold increase. For example, with the first structure, problem size \( N = 2^{48} \) and a uniform distribution with coefficient of variation \( C_X = 0.13 \), then the increase in the execution time due to load imbalance effects is about 20%. In the same case, except for an exponential distribution we see a four fold increase in execution time.

APPENDIX 1 - The load imbalance factor for the first structure in case of a normal distribution.

Let us first consider the case of a standard normal distribution. In this case we have \( \mu = 0 \) and \( \sigma = 1 \) which leads to

\[
\Delta_t = (2 \log L_t)^{1/2} - \frac{1}{2} \left[ 2 \log L_t \right]^{-1/2} (\log 4\pi - 2C) \\
- \frac{1}{2} (2 \log L_t)^{-1/2} (\log \log L_t)
\]
Since $f_i = a^{K-i}$, $A_i$ becomes:

$$
\Delta_i(a,K) \approx A(a)(K - i)^{-i/a} - B(a)(K - i)^{-i/b} - \frac{1}{2A(a)} (K - i)^{-i/c} \log (K - i)
$$

with

$$
A(a) = (2 \log a)^{1/2}
$$

$$
B(a) = \frac{1}{2A(a)} \left[ \log 4\pi - 2C + \log \log a \right]
$$

Then

$$
\Delta_{a,b}^{(1)}(a,K) = \sum_{i=1}^{K-1} \Delta_i = A(a) \cdot S_1(K) - B(a) \cdot S_2(K) - \frac{1}{2A(a)} S_3(K)
$$

with

$$
S_1(K) = \sum_{i=1}^{K-1} (K - i)^{-i/a} = \sum_{i=1}^{K-1} (i)^{-i/a}
$$

$$
S_2(K) = \sum_{i=1}^{K-1} (K - i)^{-i/b} = \sum_{i=1}^{K-1} (i)^{-i/b}
$$

$$
S_3(K) = \sum_{i=1}^{K-1} (K - i)^{-i/c} \log (K - i) = \sum_{i=1}^{K-1} i^{-i/c} \log i
$$

According to Ramanujan [9]:

$$
S_1(K) = C_1 + \frac{2}{3} (K - 1)^{\sqrt{K} - 1} + \frac{1}{2} \sqrt{K} - 1 + \frac{1}{6} \left[ \left\{ \sqrt{K} - 1 + \sqrt{K} \right\}^{-3} + \left\{ \sqrt{K} + \sqrt{K} + 1 \right\}^{-3} + \cdots \right]
$$

with

$$
C_1 = -\frac{1}{4\pi} \left\{ \frac{1}{\sqrt{1}} + \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{3}} + \cdots \right\}
$$

The asymptotic expansion for large value of $K$ can be shown to be

$$
S_1(K) = C_1 + \frac{2}{3} (K - 1)^{\sqrt{K} - 1} + \frac{1}{2} \sqrt{K} - 1 + \frac{1}{\sqrt{K} - 1} \left\{ \frac{1}{24} - \frac{1}{1920(K - 1)^2} + \frac{1}{9216(K - 1)^4} + \cdots \right\}
$$
Then $S_2(K)$ is given by

$$S_2(K) = C_0 + \sqrt{K} - 1 + \frac{1}{2\sqrt{K} - 1} - \frac{1}{2} \left\{ \frac{\left(\sqrt{K} - 1 + \sqrt{K}\right)^{-3}}{\sqrt{K}(K - 1)} + \frac{\left(\sqrt{K} + \sqrt{K} + 1\right)^{-3}}{\sqrt{K}(K + 1)} + \cdots \right\}$$

with

$$C_0 = -(1 + \sqrt{2}) \left[ \frac{1}{\sqrt{1}} - \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{3}} - \frac{1}{\sqrt{4}} + \cdots \right]$$

To evaluate $S_3(K)$ we note that

$$\sum_{i=1}^{\infty} i^{-x} \log i = -\zeta'(x)$$

for $1 \leq x \leq \infty$. $\zeta(x)$ is Riemann's zeta function. It follows that

$$S_3(K) \leq -\zeta'(\sqrt{2}) = 307.8223572$$

When $N = 2^K$, the coefficients $A$ and $B$ have the following values: $A = 0.779$, and $B = 0.3713$.

Finally we obtain

$$\psi_{2,K}^{(1)} = 0.779 \frac{S_1(K)}{K + 1} - 0.3713 \frac{S_2(K)}{K + 1} - 0.6444 \frac{S_2(K)}{K + 1}$$

Let us now consider the general case of a $(\mu, \sigma)$ normal distribution. The load imbalance costs for a synchronization epoch with $I_i$ processors active is in this case

$$\Delta_i \equiv \sigma_X [(2 \log I_i)^{1/2} - \frac{1}{2} (2 \log I_i)^{-1/2} (\log \log I_i + \log 4\pi - 2C)]$$

Consequently the ratio of load imbalance costs to the parallel execution time for the first structure is

$$\psi_{2,K}^{(1)} = \frac{\mu_X \sum_{i=1}^{K-1} \Delta_i}{\mu_X (K + 1)} = \frac{C_X \cdot \mu_X}{K + 1} \left[ A \cdot S_1(K) - B \cdot S_2(K) - \frac{1}{2A} \cdot S_3(K) \right]$$

with $S_1, S_2, S_3$ defined previously.

**APPENDIX 2 - The load imbalance factor for the third structure.**

Let us express the random variable $Y_{i,n}$ (see Figure 6c) as

$$Y_{i,n} = \max(X_{i,1}, X_{i,2}, \ldots, X_{i,a})$$
It is known that in this case
\[ E[Y_{i,j}] = \mu_X (1 + \Delta_X) \]
with
\[ \Delta_X = C_X \sqrt{3} \frac{a - 1}{a + 1} \]

Then
\[ \sigma_{Y_{i,j}}^2 = \text{var} [Y_{i,j}] = E[Y_{i,j}]^2 - E[Y_{i,j}]^2 \]

It can be shown that
\[ \sigma_{Y_{i,j}}^2 = \sigma^2 \frac{12a}{(a + 1)^2(a + 2)} \]

Hence the coefficient of variation of \( Y_{i,j} \) is
\[ C_{Y_{i,j}} = \frac{\sigma_{Y_{i,j}}}{\mu_{Y_{i,j}}} = C_X \frac{1}{(a + 1)} \frac{\sqrt{12a}}{a + 2} \]
\[ C_{Y_{i,j}} = C_X \frac{1}{(a + 1) + C_X \sqrt{3} (a - 1)} \]

Let us now examine \( Y^{(i)} \) (see Figure 6b). In this case \( Y^{(i)} \) is the sum of independent, identically distributed random variables, \( Y_{i,j} \).
\[ Y_{i,j} = Y_{i,1} + \cdots + Y_{i,K} \]

It is well known that the mean value and the variance of a sum of independent, identically distributed random variables can be expressed as
\[ \mu_{Y^{(i)}} = E[Y^{(i)}] = KE[Y_{i,j}] \]
\[ \sigma_{Y^{(i)}}^2 = K \sigma_{Y_{i,j}}^2 \]

with \( E[Y_{i,j}] \) and \( \sigma_{Y_{i,j}} \) the mean value and the variance of the \( Y_{i,j} \). Hence
\[ C_{Y^{(i)}} = \frac{C_{Y_{i,j}}}{\sqrt{K}} \]

Now if \( K \) is large, we can assume that \( Y^{(i)} \) has a normal distribution with parameters \( \mu_{Y^{(i)}} \) and \( \sigma_{Y^{(i)}} \).
Finally, the total duration of the computation presented in Figure 1c is the random variable \( Z \) (see Figure 5a)

\[
Z = X_p + \max(Y(1), Y(2), \ldots, Y(K)).
\]

It follows immediately that

\[
E[Z] = E[X_p] + \sum_{i=1}^K E[Y(i)] [1 + \Delta y^{(i)}]
\]

But

\[
E[Y(i)] = K \mu_X \left( 1 + C_X \sqrt{\frac{a - 1}{a + 1}} \right)
\]

But \( Y^{(i)} \) has a normal distribution hence

\[
\Delta y^{(i)} \approx C_{y^{(i)}} \left( (2 \log K)^{1/2} - \frac{1}{2} (2 \log K)^{-1/2} \times \right.
\]

\[
(\log \log K + \log 4\pi - 2C)\]

with

\[
C_{y^{(i)}} = C_X \frac{2 \sqrt{\frac{3aK}{a+2}}}{(a + 1) + C_X \sqrt{3(a - 1)}}
\]

Then let us denote

\[
q_1(a) = \sqrt{3} \frac{a - 1}{a + 1}
\]

\[
q_2(a,K,C_X) = \frac{2 \sqrt{\frac{3aK}{a+2}}}{(a + 1) + C_X \sqrt{3(a - 1)}} \times \left[ (2 \log K)^{1/2} - \frac{1}{2} (2 \log K)^{-1/2} \times (\log \log K + \log 4\pi - 2C) \right].
\]

\[
E[Z] = \mu_X + K \mu_X (1 + C_X q_1(a)) (1 + C_X q_2(a,K,C_X))
\]

or

\[
E[Z] = \mu_X [1 + K + KC_X (q_1(a) + q_2(a,K,C_X) + q_1(a)q_2(a,K,C_X))]\]

Without any load imbalance, the execution time is \( \mu_X [1 + K] \). It follows that the load imbalance costs are

\[
\Delta_{a,K}^{(3)} = \mu_X \times K \times C_X \times [q_1(a) + q_2(a,K,C_X) + q_1(a)q_2(a,K,C_X)]
\]

The load imbalance to parallel execution time factor is given by
\[ \psi_{a,k}^{(3)} = \frac{K}{K+1} \times C_x \times \left[ q_1(a) + q_2(a,K,C_x) + q_1(a)q_2(a,K,C_x) \right] \]

LITERATURE


