Analysis of Algorithmic Structures with Heterogeneous Tasks

Linda F. Wilson*
Institute for Computer Applications in Science and Engineering
Mail Stop 132C
NASA Langley Research Center
Hampton, VA 23681

Abstract

Developing efficient programs for distributed systems is difficult because computations must be efficiently distributed and managed on multiple processors. In particular, the programmer must partition functions and data in an attempt to find a reasonable balance between parallelism and overhead. Furthermore, it is very expensive to code an algorithm only to find out that the implementation is not efficient. As a result, it is often necessary to determine and examine those characteristics of an algorithm that can be used to predict its suitability for a distributed computing system.

In earlier work [7, 8], we presented a framework for the study of synchronization and communication effects on the theoretical performance of common homogeneous algorithmic structures. In particular, we examined the synchronous, asynchronous, nearest-neighbor, and asynchronous master-slave structures in terms of expected execution times. In this paper, we examine the effects of synchronization and communication on the expected execution times of heterogeneous algorithmic structures. Specifically, we consider structures containing two different types of tasks, where the execution times of the tasks follow one of two different uniform distributions or one of two different normal distributions. Furthermore, we compare the expected execution times of the heterogeneous algorithmic structures with times for corresponding homogeneous structures. Finally, we develop bounds for the expected execution times of the heterogeneous structures and compare those bounds to simulated execution times.

*This work was supported by the National Aeronautics and Space Administration under NASA Contract NAS1–19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681.

1 Introduction

It is well known that interprocessor communication can be costly, particularly in distributed systems where the distance between nodes is large. High interprocessor communication is the cause of the "saturation effect", which occurs when allocating additional processors to a problem causes a *decrease* in performance [6].

Communication overhead is often due to task *synchronization*, which is required to ensure that task precedence constraints are fulfilled. Synchronization is a property of an algorithm or problem. If one process finishes earlier than its synchronization partners, it must sit idle until the others finish, which is why synchronization has been called "a major cause of wasted computing cycles and of diminished performance in parallel computing" [3].

Many problems for parallel and distributed systems have two or more potential implementations, so it is particularly important to consider synchronization and communication when evaluating which implementation will give the best performance. This research examines the effects of synchronization and communication on execution time for common algorithmic structures. In particular, we will consider heterogeneous structures in which the execution times of the tasks follow known probability distributions and communication times are zero or constant.

2 Background

2.1 Algorithmic Structures

It is known that many algorithms possess an identifiable structure and that many algorithms share communication patterns [1]. Figure 1 gives sample precedence graphs for four parallel structures commonly found in algorithms: asynchronous, nearest-neighbor, synchronous, and asynchronous master-slave. In each of these structures, arcs from one node to another indicate that the first node (task) must complete execution before the second node (task) can begin execution. Notice that tasks in a row can execute concurrently if there are enough processors. Furthermore, notice that the asynchronous master-slave structure is related to the asynchronous structure in that there is no explicit synchronization between the slave processes. These and other algorithmic structures are discussed in more detail in [2, 5, 7].

2.2 Analysis of Algorithmic Structures with Homogeneous Tasks

In [7, 8], we examined the effects of synchronization and communication on execution times for different categories of homogeneous algorithmic structures. In particular, we analyzed the synchronous, asynchronous, nearest-neighbor, and asynchronous master-slave structures under the assumption that all of the tasks had execution times that were independent and identically distributed (i.i.d.) according to the uniform $u(0, 1)$ distribution or the normal $n(\frac{1}{2}, \frac{1}{12})$ distribution.

At first, we considered the effect of synchronization alone by assuming that communication times were zero. For very small problems, the results presented in [7, 8] demonstrated that the algorithmic structure used makes little difference in the expected execution time.

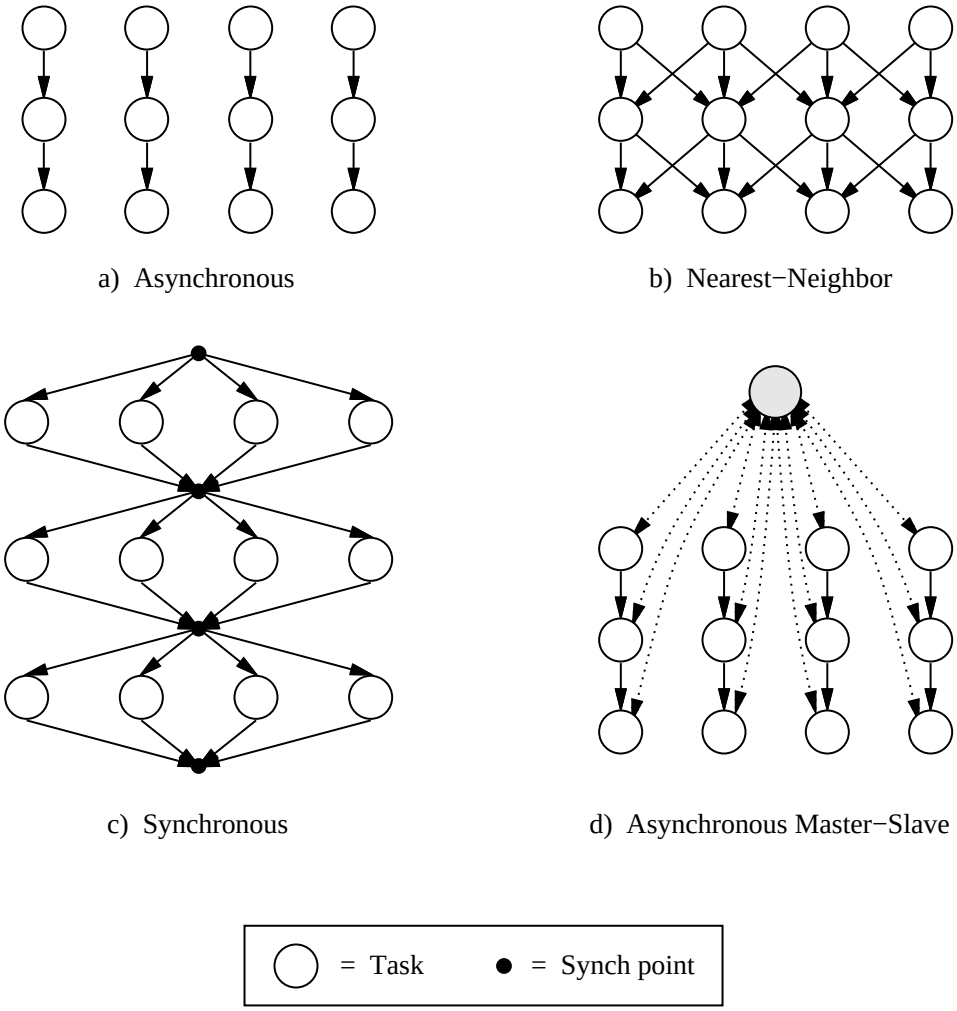

Figure 1: Examples of Algorithmic Structures

However, the results demonstrated that for large problems the cost of synchronization can be quite significant. In particular, the synchronous structure required the longest execution times while the nearest-neighbor structure had execution times that were only slightly better. As expected, the asynchronous structure gave the best execution times.[1]

Next, we analyzed the structures in Figure 1 under the assumption that communication times were constant and non-zero. By considering the possible combinations for communication and computation, we developed bounds for the expected execution time for each of these structures and compared the bounds with simulated executions. The simulation results showed that the nearest-neighbor structure suffered little performance degradation as the amount of work and number of processors were increased. The synchronous and asynchronous master-slave structures both showed performance degradation as the amount of work and number of processors were increased. The asynchronous master-slave structure

[1]When communication times are zero and the master's work is negligible, the asynchronous master-slave structure is the same as the asynchronous structure. Thus, the discussion of the cost of synchronization alone does not mention the asynchronous master-slave structure.

showed better performance with small numbers of processors and smaller communication times, while the synchronous structure showed better performance with large numbers of processors and larger communication times. However, the nearest-neighbor structure was better than all but the asynchronous structure.

3 Algorithmic Structures with Heterogeneous Tasks

Our earlier analysis assumed that all of the tasks in the structure were identically distributed and that the tasks were executed on n identical processors, where n was the maximum width of the structure. This work examines the effect on performance when either one of these assumptions is relaxed. Notice that we still require that the task execution times be independent.

Suppose we have a program composed of i.i.d. tasks that currently runs on n identical workstations. Next, suppose that m of these workstations ($m < n$) are replaced with workstations that are identical in all respects except for clock speed. In particular, suppose that the new workstations have a faster clock speed. What is the expected execution time now that the system has changed? In this case, the capability of each machine is the same, but some machines will complete the tasks earlier than the others due to the difference in clock speed. The effect of this change is that the tasks on the faster machines now have task times that follow a different distribution than the tasks on the slower machines. Thus, we can model this new system as a system of n identical processors with task times that follow two different distributions.[2] The result is an algorithmic structure with heterogeneous task times.

Similarly, consider a program in which two different types of tasks need to be executed concurrently on n workstations. It is possible that we have two different types of workstations such that the execution times are identically distributed when the tasks are appropriately matched to the capabilities of the workstations. If so, the system can be modeled using an algorithmic structure with homogeneous task times. However, it is more likely that the resulting task times will not be identically distributed, so the system should be modeled using an algorithmic structure with heterogeneous task times.[3] Notice that the number of task distributions for the structure will depend on the homogeneity or heterogeneity of the n workstations. If the n workstations are identical, the heterogeneous structure will have tasks from two different distributions. If not, the algorithmic structure will have tasks from two or more distributions.

4 Analysis of Synchronization

In this section, we will consider the effects of synchronization on execution time for algorithmic structures with heterogeneous tasks. Our results are based on the following assumptions:

[2] In this example, the distributions will be of the same type (uniform, normal, etc.) but will have different means and variances.

[3] In this example, the distributions could be of different types and have different means and variances.

- Each task in the structure has an execution time that is independent and distributed according to one of two known distributions, where the two distributions are of the same type (e.g. both uniform or both normal) but have different means and variances.
- Communication between tasks requires zero units of time.[4]
- For a precedence graph of width n (i.e. the graph contains up to n tasks that may be executed concurrently), there are n identical processors which may be used to execute the algorithmic structure.

Following our work in [7, 8], we are using the uniform and normal distributions for task execution times because they are "appropriate for many applications" [4].

4.1 Uniform Distribution

Consider a random variable X from the uniform distribution $(0, a)$ (i.e. uniform on the interval $(0, a)$)[5]. The probability density function (pdf) $f(x)$ and cumulative distribution function (cdf) $F(x)$ are given by:

$$f(x) = \begin{cases} \frac{1}{a}, & 0 < x < a \\ 0, & \text{otherwise} \end{cases} \quad \text{and} \quad F(x) = \int_{-\infty}^{x} f(z)\,dz = \begin{cases} 0, & x \leq 0 \\ \frac{x}{a}, & 0 < x < a \\ 1, & x \geq a. \end{cases}$$

The expected value $E(X)$ of X is computed as

$$E(X) = \int_{-\infty}^{\infty} x \cdot f(x)\,dx = \int_{0}^{a} \frac{1}{a}\,x\,dx = \frac{a}{2}.$$

Synchronous Structure

Consider the synchronous structure in Figure 1c. Notice that tasks in one level must synchronize before any task proceeds to the next level. As discussed in [7, 8], the overall execution time for this structure is given by a sum of **max** terms, where each term is the maximum of the execution times for a particular iteration.

Suppose we have n random variables, where m of the variables (X_i, $1 \leq i \leq m$) are distributed uniformly on $(0, a)$ and $n-m$ of the variables (Y_j, $1 \leq j \leq n-m$) are distributed uniformly on $(0, 1)$.[6] Assuming that the variables are all independent, what is the expected value of $Z = \max(X_1, X_2, \ldots, X_m, Y_1, Y_2, \ldots, Y_{n-m})$? Notice that Z is the execution time for an iteration of the synchronous structure in which the X_i and Y_j are execution times of the concurrent tasks. Furthermore, a can be considered to be a scaling factor to account for the increased "speed" of the processors corresponding to the X_i tasks.

[4] This assumption will be removed in the analysis in Section 5.

[5] We could have easily picked a distribution uniform on the interval (c, d), but the equations are simpler if we normalize to the interval $(0, a)$.

[6] It is straightforward to generalize this to any m variables from $u(0, a)$ and $n-m$ variables from $u(0, b)$, where $0 < a < b$. For simplicity, we chose $u(0, a)$ and $u(0, 1)$, where $0 < a < 1$.

We can determine the expected value $E(Z)$ by finding the distribution function $G(z)$ and taking the derivative to obtain the density function $g(z)$. It can be shown that

$$G(z) = \Pr(Z < z) = \begin{cases} \left(\frac{1}{a}\right)^m z^n & 0 < z < a < 1 \\ z^{n-m} & 0 < a \le z < 1, \end{cases} \tag{1}$$

which leads to

$$g(z) = \begin{cases} n\left(\frac{1}{a}\right)^m z^{(n-1)} & 0 < z < a < 1 \\ (n-m)\, z^{(n-m-1)} & 0 < a \le z < 1. \end{cases} \tag{2}$$

The expected value $E(Z)$ is computed as follows:

$$\begin{aligned} E(Z) &= \int_0^a n\left(\frac{1}{a}\right)^m z^n\, dz + \int_a^1 (n-m)\, z^{(n-m)}\, dz \\ &= \left(\frac{1}{a}\right)^m \left(\frac{n}{n+1}\right) a^{(n+1)} + \left(\frac{n-m}{n-m+1}\right)\left[1 - a^{(n-m+1)}\right] \\ &= \left(\frac{n}{n+1}\right) a^{(n-m+1)} + \left(\frac{n-m}{n-m+1}\right) - \left(\frac{n-m}{n-m+1}\right) a^{(n-m+1)} \\ &= \left(\frac{n-m}{n-m+1}\right) + a^{(n-m+1)}\left[\frac{n}{n+1} - \frac{n-m}{n-m+1}\right] \\ &= \left(\frac{n-m}{n-m+1}\right)\left[1 + a^{(n-m+1)}\left[\frac{(n-m+1)(n)}{(n-m)(n+1)} - 1\right]\right]. \end{aligned} \tag{3}$$

Notice that when n is large or a is very small,

$$E(Z) \approx \frac{n-m}{n-m+1}. \tag{4}$$

In [7,8], we showed that the expected value for the maximum of n i.i.d. $u(0,1)$ variables is

$$E(Y_n) = \frac{n}{n+1}. \tag{5}$$

Comparing Equations 4 and 5, we see that Equation 4 is significant because it tells us that the expected execution time for n heterogeneous tasks (where $n-m$ tasks are $u(0,1)$ and m are $u(0,a)$, where $0 < a < 1$) is approximately the same as the expected execution time for $n-m$ homogeneous tasks (where all are $u(0,1)$). Intuitively, this result is understandable since the m tasks from $(0,a)$ will generally be shorter that the $n-m$ tasks from $(0,1)$ and hence will have minimal effect on the maximum value.

For a synchronous structure with p identical levels, the expected execution time $E(\text{synch}_{n,m})$ is given by

$$E(\text{synch}_{n,m}) = p \cdot E(Z) = p\left(\frac{n-m}{n-m+1}\right)\left[1 + a^{(n-m+1)}\left[\frac{(n-m+1)(n)}{(n-m)(n+1)} - 1\right]\right]. \tag{6}$$

When n is large or a is very small, we can use a simpler approximation for $E(\text{synch}_{n,m})$.

$$E(\text{synch}_{n,m}) \approx p\left(\frac{n-m}{n-m+1}\right) \tag{7}$$

Asynchronous Structure

Analysis of the synchronous structure was fairly straightforward because tasks in one level must synchronize before any task proceeds to the next level. For the asynchronous structure in Figure 1a, the analysis is more difficult because of the lack of synchronization.

Figure 1a shows an asynchronous structure that has four streams of execution, each of which contains three tasks in series. The execution time of a single stream depends on the sum of three tasks, while the execution time of the entire structure depends on the maximum of four sums. In [7,8], we discussed how the calculation of the expected value of a maximum of n i.i.d. variables requires knowledge of the pdf for the variables. The problem with analyzing the asynchronous structure is that it is difficult to determine the pdf for a sum of random variables. For example, the pdf for the sum of just three $u(0,1)$ variables is given by:

$$g(x) = \begin{cases} 4\left(\frac{1}{6}x^3\right)^3\left(\frac{1}{2}x^2\right) = \frac{1}{108}x^{11}, & 0 < x < 1 \\ 4\left(\frac{-x^3}{3} + \frac{3x^2}{2} - \frac{3x}{2} + \frac{1}{2}\right)^3\left(-x^2 + 3x - \frac{3}{2}\right), & 1 < x < 2 \\ 4\left(\frac{x^3}{6} - \frac{3x^2}{2} + \frac{9x}{2} - \frac{7}{2}\right)^3\left(\frac{x^2}{2} - 3x + \frac{9}{2}\right), & 2 < x < 3. \end{cases} \tag{8}$$

Notice that Equation 8 is not a simple function of the underlying $u(0,1)$ distribution. In practice, algorithms consist of many levels of tasks, and the exact computation of the underlying distribution of the sum can be tedious.

In [7], we mentioned that the normal distribution can be used to approximate the sum of k tasks, where k is large. We also demonstrated that standard tables can be used to determine the expected value for the maximum of n i.i.d. variables that are normally distributed. Thus, we were able to obtain close approximations for the expected execution time of the asynchronous structure. For the current work, we are not aware of any standard tables that can be used to determine the expected value of the maximum of variables from two or more normal distributions. Thus, we will use simulation to determine the behavior of the asynchronous structure with heterogeneous tasks.

Before performing the simulation, it is useful to anticipate what the results will be. In the analysis of the synchronous structure, we observed that the expected value for a combination of variables from two different uniform distributions with the same lower bound depends primarily on the variables from the distribution with the larger mean. This behavior was due to the fact that the variables from the "smaller" distribution will have little effect on the maximum value. With the asynchronous structure, we would expect the same behavior, particularly since the "faster" processors will likely race ahead of the "slower" ones.

Nearest-Neighbor Structure

Figure 1b shows a nearest-neighbor structure in which each task must synchronize with one or two neighbor tasks. In [7, 8], we discussed how analytical methods cannot be used to determine the expected value of the nearest-neighbor structure because tasks in each level of the nearest-neighbor structure cannot be isolated such that the final result is a sum of **max** terms or the **max** of identical and independent random variables. Thus, simulation was used to determine the behavior of the homogeneous nearest-neighbor structure. In Section 4.3, we will present simulation results that demonstrate how the expected execution time of the nearest-neighbor structure is affected by the presence of heterogeneous tasks.

4.2 Normal Distribution

As noted in [4], it is often reasonable to assume that tasks in a program have execution times that follow a normal distribution $n(\mu, \sigma^2)$, where μ is the mean and σ^2 is the variance. The normal distribution $n(\mu, \sigma^2)$ has a pdf given by:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \qquad -\infty < x < \infty. \tag{9}$$

In Section 4.1, we used the simple pdf of the uniform distribution to derive an equation for the expected execution time of the heterogeneous synchronous structure. With the normal distribution, a similar approach will not work because the pdf does not have an antiderivative. Thus, simulation must be used to examine all of the heterogeneous structures based on the normal distribution, just as it is needed for the asynchronous and nearest-neighbor structures based on the uniform distribution.

4.3 Discussion of Results

In this section, we present simulation results for the heterogeneous asynchronous, synchronous, and nearest-neighbor structures in which communication times are assumed to be zero. Since the uniform distribution $u(0, 1)$ has mean $\frac{1}{2}$ and variance $\frac{1}{12}$, we examine the normal distribution $n(\frac{1}{2}, \frac{1}{12})$ so that comparisons can be made between the two distributions. Using the scaling factor a, $0 < a < 1$, the uniform $u(0, 1)$ distribution will be paired with the scaled $u(0, a)$ distribution. Similarly, the normal $n(\frac{1}{2}, \frac{1}{12})$ distribution will be paired with the scaled $n(\frac{a}{2}, \frac{a^2}{12})$ distribution.[7] Notice that $u(0, a)$ and $n(\frac{a}{2}, \frac{a^2}{12})$ have the same mean but different variances, even though the same scaling constant a is used.

Table 1 provide simulation results[8] for the heterogeneous asynchronous, synchronous, and nearest-neighbor structures in which tasks are drawn from the uniform $u(0, 1)$ and $u(0, a)$ distributions. Table 2 provides similar results for the normal $n(\frac{1}{2}, \frac{1}{12})$ and $n(\frac{a}{2}, \frac{a^2}{12})$ distributions. For comparison, simulation results for the relevant homogeneous structures (where $m = 0$) are also included.

[7] If Y is $n(\mu, \sigma^2)$ and $X = kY$, X is $n(k\mu, k^2\sigma^2)$.

[8] Results were taken to be the average of 10,000 simulated executions, where each processor executed 1000 tasks and all of the tasks on a particular processor came from one distribution.

Table 1: Simulation Results for Expected Execution Times of Heterogeneous Structures with Tasks from $u(0,1)$ and $u(0,a)$ Distributions and Zero Communication Times

a	Alg. Struct.	$(n,m)^a$								
		(3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	Asynch	508	508	510	510	511	511	513	513	516
	Synch	750	753	800	801	833	834	889	890	941
	NN	732	733	765	766	785	785	813	813	836
$\frac{2}{3}$	Asynch	508	508	510	510	511	511	513	513	516
	Synch	750	760	800	804	833	838	889	891	941
	NN	732	737	765	767	785	788	813	814	836
$\frac{4}{5}$	Asynch	508	508	510	510	511	511	513	513	516
	Synch	750	770	800	811	833	848	889	896	941
	NN	732	744	765	771	785	789	813	816	836

[a] The notation (n,m) refers to a structure of width n in which $n-m$ tasks are $u(0,1)$ and m tasks are $u(0,a)$, $0 < a < 1$.

Table 2: Simulation Results for Expected Execution Times of Heterogeneous Structures with Tasks from $n(\frac{1}{2}, \frac{1}{12})$ and $n(\frac{a}{2}, \frac{a^2}{12})$ Distributions and Zero Communication Times

a	Alg. Struct.	$(n,m)^a$								
		(3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	Asynch	508	508	510	510	511	511	514	514	517
	Synch	744	747	797	798	836	837	911	912	1010
	NN	727	728	762	762	783	784	817	818	848
$\frac{2}{3}$	Asynch	508	508	510	510	511	511	514	514	517
	Synch	744	754	797	802	836	844	911	917	1010
	NN	727	732	762	764	783	785	817	818	848
$\frac{4}{5}$	Asynch	508	508	510	510	511	511	514	514	517
	Synch	745	766	797	811	836	860	911	937	1010
	NN	727	739	762	768	783	789	817	820	848

[a] The notation (n,m) refers to a structure of width n in which $n-m$ tasks are $n(\frac{1}{2}, \frac{1}{12})$ and m tasks are $n(\frac{a}{2}, \frac{a^2}{12})$, $0 < a < 1$.

For the asynchronous structure, the expected execution time for the heterogeneous (n, m) structure is the same as that for the homogeneous $(n-m, 0)$ structure. This is not surprising since the structure's lack of synchronization allows some processors to race ahead of others. Notice, however, that there is very little difference between the execution times for the (n, m) and $(n, 0)$ structures. Because the asynchronous structure is relatively insensitive to increases in width (when the number of tasks per processor is constant), there is little to be gained by moving some of the work from slower processors to faster processors as long as the amount of work per processor is constant. Instead, performance gains may be made by allocating more work to the faster processors and less work to the slower processors. For example, in the case where $a = \frac{1}{2}$, each processor executing $u(0, \frac{1}{2})$ tasks can process on average twice as many tasks as a processor executing $u(0, 1)$ tasks.[9]

The simulation results for the synchronous structure agree with the analytical results obtained from Equation 6 for the uniform distribution. When a is small or n is fairly large, the expected execution time for the heterogeneous (n, m) structure is reasonably approximated by the homogeneous $(n–m, 0)$ structure. Notice that we predicted this behavior in Equation 7. When a is large and n is small, the approximation is not very close, but the execution time for (n, m) is still closer to the time for $(n-m, 0)$ than it is to the time for $(n, 0)$. Even though the approximation was based on the uniform distribution, the results for the normal distribution indicate that it still applies. Thus, the results indicate that the performance of a homogeneous synchronous structure can be improved by using m faster processors to make it heterogeneous.

Based on the results for the asynchronous and synchronous structures, we would expect that the nearest-neighbor (n, m) structure can be approximated by the corresponding $(n-m, 0)$ structure. The results in Table 1 and Table 2 indicate that this is true when a is small or n is large. Like the synchronous structure, the results for the (n, m) nearest-neighbor structure are closer to those for the $(n-m, 0)$ structure even when a is large and n is small. Thus, the nearest-neighbor structure also has the property that performance gains can be made by moving some of the tasks from slower processors to faster ones. Notice, however, that the gains for the nearest-neighbor structure are not as great as those for the synchronous structure.

5 Analysis of Synchronization and Communication

The previous section examined the cost of synchronization for the asynchronous, synchronous, and nearest-neighbor structures. The cost of synchronization alone was determined by assuming communication between tasks required zero units of time. This section will investigate the execution times of these structures and the asynchronous master-slave structure when communication times are constant and non-zero.

[9] Because of differences in variances, a stream of $2n$ tasks from $u(0, \frac{1}{2})$ will not have the same distribution as a stream of n tasks from $u(0, 1)$. However, the two streams will have the same expected value if n is large since the distribution of each stream can then be approximated by a normal distribution.

5.1 Bounds on Execution Time

Asynchronous Structure

For the asynchronous structure in Figure 1a, communication occurs only between tasks on the same processor. Since communication time between tasks on the same processor is zero or minimal compared to communication between different processors, it is reasonable to assume that communication times for the purely asynchronous structure are zero. Thus, the overall execution time for the asynchronous structure is not affected by communication.

Synchronous Structure

In the synchronous structure in Figure 1c, communication occurs between tasks on different processors and hence communication times are non-zero. Consider Figure 2, which shows a Gantt chart for the execution of heterogeneous tasks in the synchronous structure on four processors. In each iteration, communication is required to initiate execution of the tasks and synchronize the tasks before proceeding to the next iteration. Notice that the tasks have variable execution times. Assume that a processor may communicate with exactly one processor at a time and a processor may communicate only when it is not busy with computation.

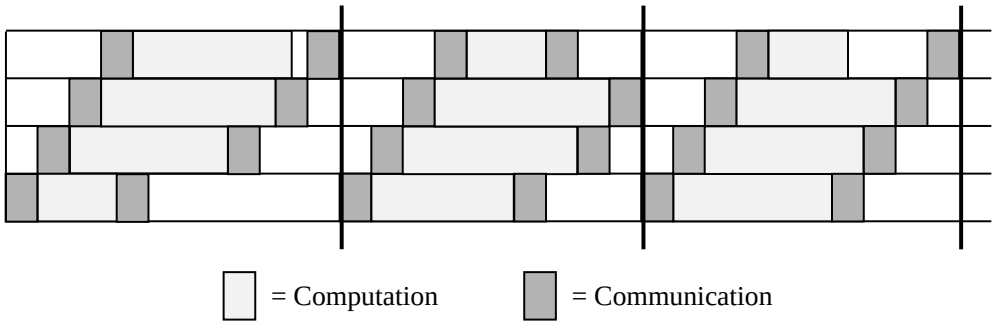

Figure 2: Gantt Chart for Synchronous Structure with Heterogeneous Tasks

Suppose that the shortest task in an iteration comes from the scaled $u(0, a)$ distribution[10] and the other tasks come from the $u(0, 1)$ distribution. In the first iteration, the shortest task is the first one to begin execution. Notice that the initial communication for the shortest task affects the execution time for the iteration since the longer tasks must wait to begin execution. In the second iteration, the same tasks are rearranged such that the shortest task is the last one to begin execution. Notice that the shortest task's communication and computation have no effect on the iteration's execution time. Specifically, the execution time for the second iteration depends only on the time required for the three longest tasks. Thus, the second iteration has an execution time that is shorter than the first iteration, even though the same tasks are executed in both iterations.

Figure 2 demonstrates that the execution time for an iteration may be reduced by starting the longest tasks before the shortest tasks. When task execution times come from known

[10] In the discussion that follows, it is assumed that $0 < a < 1$.

distributions, the tasks with the largest means should be the first ones initiated in order to increase the probability of minimizing the overall execution time. However, notice from the third iteration that there is no guarantee that starting the longest tasks first will automatically reduce the overall execution time. In this example (which does not use the same task times as the first two iterations), the shortest task's final communication must be added to the overall execution time for the other tasks. Starting the tasks that are likely to be the longest at the beginning of the iteration merely presents the opportunity for reducing the overall execution time. Results presented later in this paper will be based on the policy that tasks from the distribution with the smaller mean (such as $u(0, a)$) will be initiated before tasks from the distribution with the larger mean (such as $u(0, 1)$).

In [7, 8], we developed bounds on the expected execution time for the homogeneous synchronous structure in which each iteration contains n tasks (executed on n identical processors) and the structure contains p iterations. The expected time for the entire structure $E(X)$ follows the bounds

$$E(\text{synch}) + 2\,p\,C \leq E(X) \leq E(\text{synch}) + (n+1)\,p\,C, \tag{10}$$

where $E(\text{synch})$ is the expected execution time without communication and C is the time required for one communication.

While Equation 10 was developed for the homogeneous synchronous structure, it is also applicable to the heterogeneous structure. In particular, the communication requirements are the same for both structures, so the terms containing C do not change.[11] The only difference between the two structures is that the heterogeneous (n, m) structure requires less execution time than the corresponding homogeneous $(n, 0)$ structure. Thus, Equation 10 can be modified slightly to obtain bounds for the heterogeneous synchronous structure.

$$E(\text{synch}_{n,m}) + 2\,p\,C \leq E(X) \leq E(\text{synch}_{n,m}) + (n+1)\,p\,C, \tag{11}$$

where $E(\text{synch}_{n,m})$ represents the expected execution time for the heterogeneous structure with communication that was determined in Section 4. In the next section, simulation results will be used to examine the usefulness of these bounds.

Nearest-Neighbor Structure

Consider Figure 3, which shows a Gantt chart for the execution of tasks from the heterogeneous nearest-neighbor structure (Figure 1b). In this case, neighbor processors must communicate with each other to assure that both have completed the current iteration before proceeding to the next iteration. Notice that this communication could be synchronous, in which both must be done with computation before any exchange can occur, or asynchronous, in which the send and receive operations each take one communication unit of time. The overall effect is that synchronization between neighbors requires two communication units. Thus, a processor synchronizing with two other processors spends four communication units before proceeding to the next iteration. It should be noted that it is assumed that communication between different pairs of processors may occur simultaneously.

[11] Notice that we assume that communication times for the "faster" processors are the same as those for the "slower" processors. Faster computation does not always result in proportionally-faster communication. Similarly, differences in task times do not necessarily cause differences in communication times.

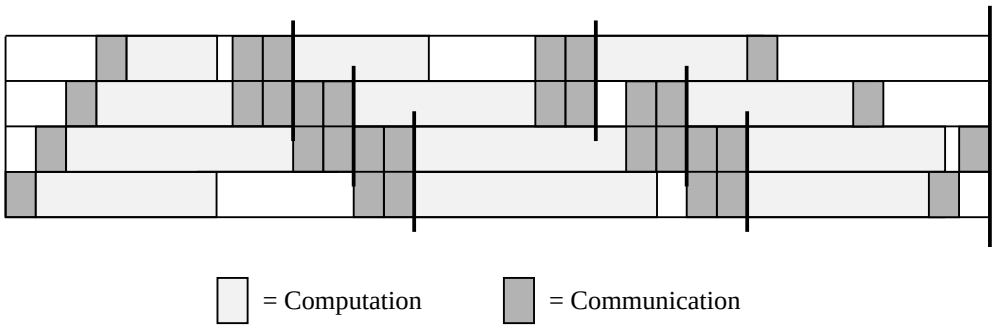

Figure 3: Gantt Chart for Nearest-Neighbor Structure with Heterogeneous Tasks

For the homogeneous nearest-neighbor structure in which each processor has one or two neighbors, the expected execution time $E(X)$ was shown in [7] to be bounded by

$$E(\mathrm{nn}) + 2\,pC \leq E(X) \leq E(\mathrm{nn}) + C(2n + 6(p-1)), \tag{12}$$

where $E(\mathrm{nn})$ is the expected execution time without communication that was determined by simulation. For the heterogeneous structure, the communication requirements are the same as those for the homogeneous structure. Thus, it is reasonable to assume that the bounds in Equation 12 still hold when $E(X)$ represents the heterogeneous execution time and $E(\mathrm{nn})$ is replaced by $E(\mathrm{nn}_{n,m})$. Simulated execution times will be compared to these bounds in the next section.

Asynchronous Master-Slave Structure

It was fairly straightforward to determine bounds for the synchronous and nearest-neighbor structures because each processor completed the same number of tasks. With the asynchronous master-slave structure, processors can receive different amounts of work, so the analysis is more complicated.

Consider Figure 4, which shows a Gantt chart for the execution of slave tasks in the heterogeneous asynchronous master-slave structure on four processors.[12] Like the synchronous and nearest-neighbor structures, the initial tasks in the asynchronous master-slave structure are staggered since communication from the master is required. Communication from the slave to the master and from the master to the slave is required between each iteration. Since the master can communicate with one slave at a time, there may be delays as slaves wait to be serviced by the master object. Notice in Figure 4 that some processors complete more tasks than other processors. Because there is no synchronization between slave tasks, the master processor can "deal out" work to any processor that is idle.

In [7], we developed the following bounds for the expected execution time $E(X)$ of the homogeneous asynchronous master-slave structure:

$$E(\mathrm{asynch}) + 2pC \leq E(X) \leq E(\mathrm{asynch}') + 2n'p'C, \tag{13}$$

[12]Notice that Figure 4 shows only slave tasks and communication and does not indicate work by the master. Although it is not realistic to ignore the effects of the master's work and communication on performance, this approach can provide a lower bound on the structure's execution time.

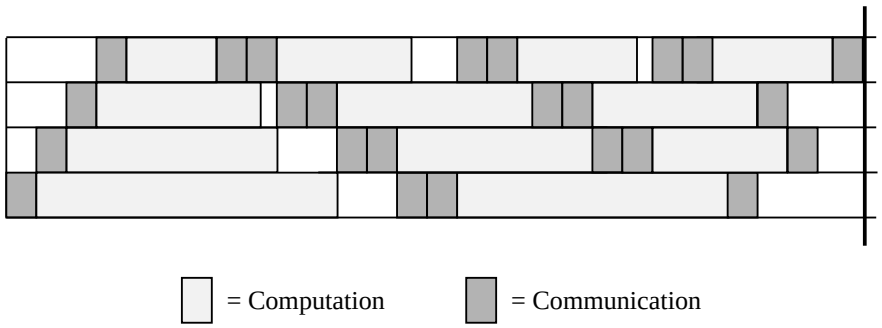

Figure 4: Gantt Chart for Asynchronous Master-Slave Structure with Heterogeneous Tasks

where $E(\text{asynch})$ is the expected execution time of the asynchronous structure (based on synchronization alone) with n processors and p iterations and $E(\text{asynch}')$ is the corresponding time with $n' = n - 1$ processors and $p' = \left\lceil \frac{np}{n-1} \right\rceil$ iterations per processor.

Equation 13 was developed by considering the minimum and maximum execution times of the slave processors and the master under the assumption that any task could be executed on any processor. If the number of processors n or the communication unit C is large, the execution time will be dominated by the master's communication with the slave processors. In this case, the master may have little time for executing ordinary tasks since most of its time will be spent servicing the slaves. At the other extreme, the master processor will spend less time on communication and can process tasks along with the slave processors.

To determine bounds for the heterogeneous asynchronous master-slave structure, we need to consider how the shorter task times will affect the overall execution time. In the worst case, communication will dominate the execution time, and the shorter task execution times will have little or no effect in reducing the overall time. Thus, the upper bound in Equation 13 for the homogeneous case will still serve as an upper bound for the heterogeneous structure.[13] In the best case, computation will dominate the overall execution time, so the shorter task times will have an effect on the lower bound.

To compute the lower bound for the heterogeneous case, we need to determine the optimal distribution of work. Recall that the heterogeneous (n, m) structure will have task execution times following one distribution (such as $u(0, 1)$) on n–m processors and a scaled distribution (such as $u(0, a)$) on the other m processors. If the np tasks are "equally" distributed among the processors according to processor speed,

$$(n - m)x + my = np, \tag{14}$$

where x is the number of tasks assigned to each slower processor and y is the number of tasks assigned to each faster processor.

Next, we must account for the minimal time spent on communication. Ideally, the processors will all start and stop computation at the same time. Suppose that the structure contains tasks from the $u(0, 1)$ and $u(0, a)$ distributions. Given that each task requires two

[13]The results from Section 4.3 indicate that there is little or no difference between $E(\text{asynch}')$ and $E(\text{asynch}'_{n,m})$. We will use $E(\text{asynch}')$ since it can be computed using the normal approximation [7].

units of computation (one to receive the task and one to return the result), the execution times for the processors will be identical if

$$x\left(\frac{1}{2}+2C\right)=y\left(\frac{a}{2}+2C\right), \tag{15}$$

where $\frac{1}{2}$ and $\frac{a}{2}$ are the expected values for the $u(0,1)$ and $u(0,a)$ distributions, respectively. Observe that Equation 15 assumes that processors are never idle.

Equations 14 and 15 form a set of simultaneous equations that can be solved to find the optimal values for x and y.

$$x=\left\lceil\frac{np-my}{n-m}\right\rceil \qquad y=\left\lfloor\frac{\left(\frac{np}{n-m}\right)\left(\frac{1}{2}+2C\right)}{\left(\frac{m}{n-m}\right)\left(\frac{1}{2}+2C\right)+\left(\frac{a}{2}+2C\right)}\right\rfloor \tag{16}$$

Notice that the ceiling function is used to obtain integer values for x and y. As a result, Equation 15 may not be an exact equality, so the lower bound is given by

$$E(X)\geq\max\left[x\left(\frac{1}{2}+2C\right),y\left(\frac{a}{2}+2C\right)\right]. \tag{17}$$

5.2 Discussion of Results

To examine the behavior of (n,m) heterogeneous structures with non-zero communication times, simulation was used to obtain expected execution times for the synchronous, nearest-neighbor, and asynchronous master-slave structures. In particular, constant communication times ranging from 0.025 to 0.25 were used with execution times distributed according to the normal $n(\frac{1}{2},\frac{1}{12})$ and $n(\frac{a}{2},\frac{a^2}{12})$ distributions.[14] Each structure contained $1000n$ tasks, and expected values were obtained by averaging the execution times of 10,000 simulated executions.

Table 3 presents simulation results for the synchronous structure. As expected, the execution times for the (n,m) structure fall between those for the $(n-m,0)$ and $(n,0)$ structures. In particular, observe that there is a noticeable difference between the execution times for (n,m) and $(n,0)$, even when the communication times are large. It is significant that the benefit from reducing computation times on m processors is not completely outweighed by the communication, even though the (n,m) structure requires just as much communication as the $(n,0)$ structure.

Table 4 presents simulation results for the nearest-neighbor structure. When communication times are small or the number of processors is large, there is a noticeable difference between the times for (n,m) and $(n,0)$. Thus, there is some benefit in allocating work to m faster processors. When communication times are large and the number of processors is small, the benefit disappears because the large amount of communication negates the reduced computation times.

[14] In this section, only results for the normal distributions will be presented. Results for the uniform distributions, which were similar to those of the normal distributions, are given in Appendix A.

Table 3: Simulation Results for Expected Execution Times of Heterogeneous Synchronous Structure with Tasks from $n(\frac{1}{2}, \frac{1}{12})$ and $n(\frac{a}{2}, \frac{a^2}{12})$ Distributions and Constant Communication Times

a	Comm. Time C	$(n,m)^a$ (3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	.025	837	864	916	936	977	1016	1106	1191	1343
	.05	929	982	1036	1075	1119	1203	1310	1549	1726
	.10	1118	1220	1277	1358	1408	1610	1741	2418	2567
	.15	1311	1463	1521	1650	1705	2066	2199	3321	3449
	.20	1508	1710	1769	1953	2009	2550	2671	4224	4344
	.25	1709	1961	2020	2265	2316	3040	3147	5125	5242
$\frac{4}{5}$	.025	837	884	916	950	977	1047	1106	1244	1343
	.05	929	1003	1036	1091	1119	1243	1310	1620	1726
	.10	1118	1242	1277	1378	1408	1667	1741	2479	2567
	.15	1311	1486	1521	1673	1705	2127	2199	3372	3449
	.20	1508	1733	1769	1977	2009	2605	2671	4273	4344
	.25	1709	1984	2020	2287	2316	3088	3147	5172	5242

[a] The notation (n,m) refers to a structure of width n in which $n-m$ tasks are $n(\frac{1}{2}, \frac{1}{12})$ and m tasks are $n(\frac{a}{2}, \frac{a^2}{12})$, $0 < a < 1$.

Table 4: Simulation Results for Expected Execution Times of Heterogeneous Nearest-Neighbor Structure with Tasks from $n(\frac{1}{2}, \frac{1}{12})$ and $n(\frac{a}{2}, \frac{a^2}{12})$ Distributions and Constant Communication Times

a	Comm. Time C	(n,m) (3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	.025	781	795	816	824	836	841	865	868	890
	.05	852	879	894	911	918	929	951	956	976
	.10	1005	1063	1069	1100	1103	1122	1143	1150	1169
	.15	1173	1258	1260	1302	1303	1330	1355	1364	1386
	.20	1349	1456	1458	1509	1510	1543	1576	1587	1616
	.25	1532	1657	1658	1716	1717	1754	1792	1805	1843
$\frac{4}{5}$	.025	781	800	816	826	836	844	865	869	890
	.05	852	882	894	912	918	932	951	957	976
	.10	1005	1063	1069	1100	1103	1125	1143	1151	1169
	.15	1173	1256	1260	1302	1303	1337	1355	1367	1386
	.20	1349	1455	1458	1509	1510	1553	1576	1591	1616
	.25	1532	1656	1658	1716	1717	1765	1792	1811	1843

Table 5: Simulation Results for Expected Execution Times of Heterogeneous Asynchronous Master-Slave Structure with Tasks from $n(\frac{1}{2}, \frac{1}{12})$ and $n(\frac{a}{2}, \frac{a^2}{12})$ Distributions and Constant Communication Times

a	Comm. Time C	(n,m)								
		(3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	.025	589	**512**	596	**540**	602	**516**	619	800	800
	.05	686	**640**	707	**687**	728	803	834	1599	1600
	.10	904	935	972	1052	1061	1600	1600	3199	3199
	.15	1142	1264	1279	1503	1503	2399	2400	4798	4799
	.20	1391	1620	1623	2000	2000	3199	3199	6396	6397
	.25	1646	2002	2003	2499	2500	3999	3999	7997	7997
$\frac{4}{5}$	.025	589	**571**	596	**584**	602	**585**	619	800	800
	.05	686	687	707	716	728	820	834	1599	1600
	.10	905	965	977	1067	1070	1600	1600	3198	3199
	.15	1142	1275	1279	1503	1503	2400	2400	4798	4799
	.20	1391	1622	1623	2000	2000	3199	3199	6396	6397
	.25	1646	2003	2003	2499	2500	3999	3999	7997	7997

Table 5 presents simulation results for the asynchronous master-slave structure. Notice the bold-faced entries in Table 5, which identify (n,m) execution times that were less than the times for the corresponding $(n-m,0)$ and $(n,0)$ structures. These entries indicate that the heterogeneous asynchronous master-slave structure can significantly improve performance over both the $(n-m,0)$ and $(n,0)$ structures when communication times are very small. As communication times get large, communication with the master dominates the execution time and the (n,m) times quickly approach the $(n,0)$ times. Thus, the communication outweighs any benefit from faster computation.

Figures 5 through 10 demonstrate the bounds obtained in Section 5.1 by comparing them with simulated execution times for the synchronous, nearest-neighbor, and asynchronous master-slave structures. Figures 5 through 7 present results for the $(8,3)$ structure while Figures 8 through 10 present similar results for the $(16,8)$ structure. Notice that the synchronous and asynchronous master-slave structures closely follow the upper bound, while the execution time for the nearest-neighbor structure falls right between the upper and lower bounds. Results for other values of n and m followed the same patterns.

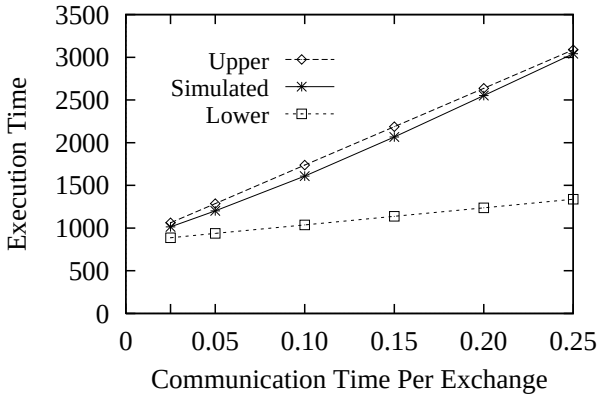

Figure 5: Comparison of Simulated $(8,3)$ Synchronous Structure with Upper and Lower Bounds

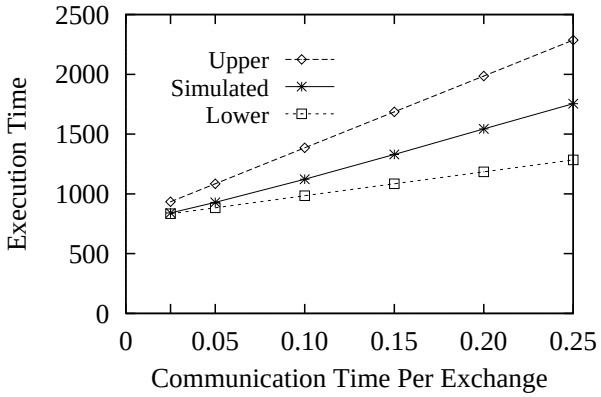

Figure 6: Comparison of Simulated $(8,3)$ Nearest-Neighbor Structure with Upper and Lower Bounds

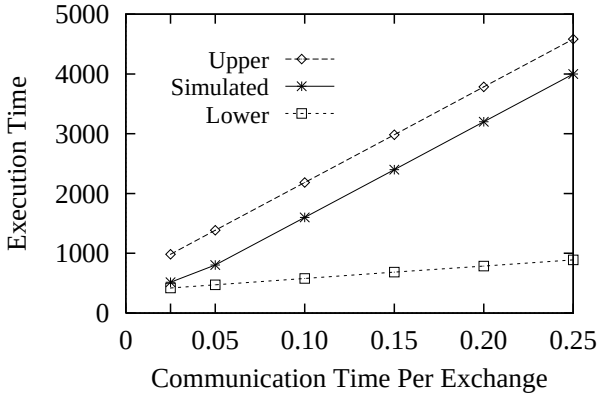

Figure 7: Comparison of Simulated $(8,3)$ Asynchronous Master-Slave Structure with Upper and Lower Bounds

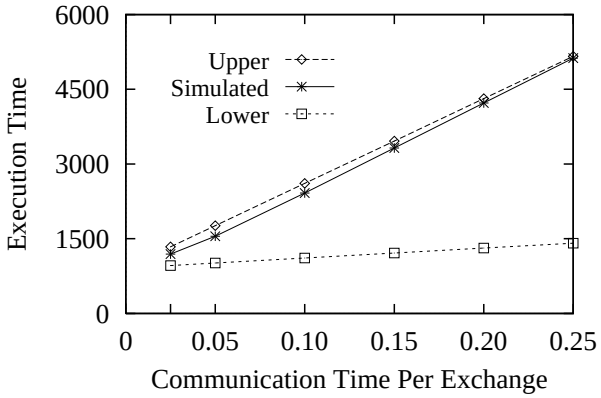

Figure 8: Comparison of Simulated $(16, 8)$ Synchronous Structure with Upper and Lower Bounds

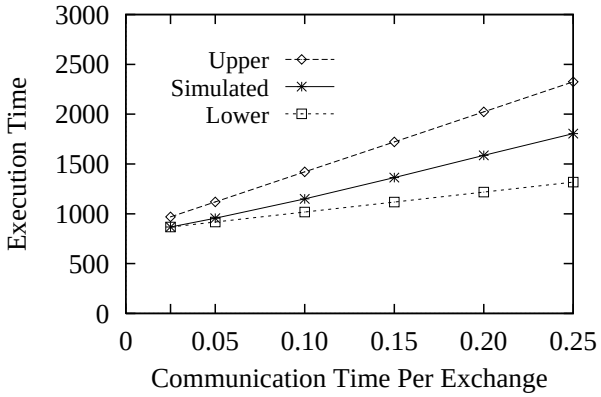

Figure 9: Comparison of Simulated $(16, 8)$ Nearest-Neighbor Structure with Upper and Lower Bounds

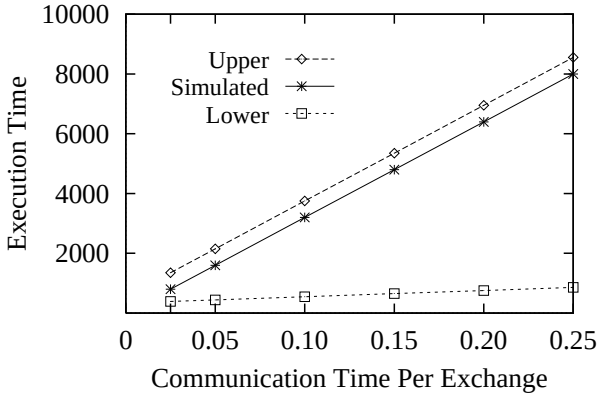

Figure 10: Comparison of Simulated $(16, 8)$ Asynchronous Master-Slave Structure with Upper and Lower Bounds

6 Conclusions

- Heterogeneous algorithmic structures can be used to model problems containing tasks with execution times from different probability distributions.
- Ignoring communication times, the expected execution time for a heterogeneous (n, m) structure is roughly the same as that for the corresponding homogeneous $(n - m, 0)$ structure.
- When non-zero communication times are considered, the heterogeneous (n, m) structure will require as much communication as the homogeneous $(n, 0)$ structure.
- The heterogeneous (n, m) synchronous structure will generally be faster than the corresponding $(n, 0)$ structure, even when communication times are large.
- When communication times are small, the heterogeneous nearest-neighbor structure is slightly faster than the homogeneous structure. When communication times are large, the execution times are the same, so there is no benefit in using a mixture of slower and faster processors.
- The (n, m) asynchronous master-slave structure can gain significant performance over the $(n-m, 0)$ and $(n, 0)$ structures when communication times are very small. The benefit of the (n, m) structure over the $(n, 0)$ structure decreases as communication times increase, until there is no benefit at all.
- The theoretical bounds developed in Section 5.1 can be used to predict the expected execution times for heterogeneous algorithmic structures with known task distributions.

References

[1] L. H. Jamieson, "Characterizing parallel algorithms," in *The Characteristics of Parallel Algorithms* (L. H. Jamieson *et al.*, eds.), pp. 65–100, Cambridge, MA: MIT Press, 1987.

[2] V. W. Mak and S. F. Lundstrom, "Predicting performance of parallel computations," *IEEE Transactions on Parallel and Distributed Systems*, vol. 1, pp. 257–270, July 1990.

[3] D. C. Marinescu and J. R. Rice, "The effects of communication latency upon synchronization and dynamic load balance on a hypercube," *Proc. 5th International Parallel Processing Symposium*, pp. 18–25, 1991.

[4] D. C. Marinescu and J. R. Rice, "Synchronization and load imbalance effects in distributed memory multi-processor systems," *Concurrency: Practice and Experience*, vol. 3, pp. 593–625, December 1991.

[5] J. Mohan, "Performance of parallel programs: Models and analyses," PhD dissertation, Tech. Rep. CMU-CS-84-141, Department of Computer Science, Carnegie Mellon University, July 1984.

[6] G. C. Sih and E. A. Lee, "Scheduling to account for interprocessor communication within interconnection-constrained processor networks," *Proceedings of the International Conference on Parallel Processing*, vol. I, pp. 9–16, 1990.

[7] L. F. Wilson, "Performance analysis of distributed algorithms based on communication and synchronization patterns." PhD dissertation, The University of Texas at Austin, May 1994.

[8] L. F. Wilson and M. J. Gonzalez, "Synchronization and communication in algorithmic structures," *Sixth IEEE Symposium on Parallel and Distributed Processing*, pp. 196–203, October 1994.

A Results for the Uniform Distribution

Tables 6 through 8 present results for the uniform distribution that are comparable to the results for the normal distribution given in Tables 3 through 5. Notice that the results for the two distributions are very similar. Correspondingly, the bounds for execution times are practically identical for the two distributions, and the graphs in Figures 5 through 10 represent both the uniform and normal distributions.

Table 6: Simulation Results for Expected Execution Times of Heterogeneous Synchronous Structure with Tasks from $u(0,1)$ and $u(0,a)$ Distributions and Constant Communication Times

a	Comm. Time C	$(n,m)^a$								
		(3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	.025	842	871	919	939	974	1014	1086	1178	1298
	.05	935	988	1039	1078	1117	1202	1293	1546	1699
	.10	1123	1226	1280	1361	1407	1608	1729	2422	2548
	.15	1315	1468	1524	1651	1703	2066	2186	3319	3427
	.20	1512	1714	1772	1953	2006	2551	2658	4219	4317
	.25	1712	1965	2022	2266	2314	3042	3134	5118	5209
$\frac{4}{5}$	.025	842	889	919	950	1014	1034	1086	1208	1298
	.05	935	1007	1039	1091	1117	1231	1293	1605	1699
	.10	1123	1247	1280	1378	1407	1658	1729	2469	2548
	.15	1315	1489	1524	1674	1703	2118	2186	3358	3427
	.20	1512	1736	1772	1978	2006	2597	2658	4254	4317
	.25	1712	1987	2022	2288	2314	3080	3134	5149	5209

[a] The notation (n,m) refers to a structure of width n in which $n-m$ tasks are $u(0,1)$ and m tasks are $u(0,a)$, $0 < a < 1$.

Table 7: Simulation Results for Expected Execution Times of Heterogeneous Nearest-Neighbor Structure with Tasks from $u(0,1)$ and $u(0,a)$ Distributions and Constant Communication Times

a	Comm. Time C	(n,m)								
		(3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	.025	786	800	820	828	839	844	865	868	886
	.05	856	884	898	914	921	932	952	956	974
	.10	1008	1067	1073	1103	1105	1124	1144	1151	1168
	.15	1174	1262	1264	1306	1306	1332	1356	1365	1385
	.20	1349	1461	1461	1513	1513	1546	1578	1588	1617
	.25	1531	1662	1661	1721	1722	1758	1796	1808	1845
$\frac{4}{5}$	.025	786	805	820	830	839	846	865	868	886
	.05	856	886	898	915	921	934	952	957	974
	.10	1008	1066	1073	1102	1105	1127	1144	1151	1168
	.15	1174	1260	1264	1304	1306	1338	1356	1367	1385
	.20	1349	1459	1461	1512	1513	1556	1578	1593	1617
	.25	1531	1660	1661	1721	1722	1770	1796	1813	1845

Table 8: Simulation Results for Expected Execution Times of Heterogeneous Asynchronous Master-Slave Structure with Tasks from $u(0,1)$ and $u(0,a)$ Distributions and Constant Communication Times

a	Comm. Time C	(n,m)								
		(3,0)	(4,1)	(4,0)	(5,1)	(5,0)	(8,3)	(8,0)	(16,8)	(16,0)
$\frac{1}{2}$	.025	589	**513**	596	**540**	602	**519**	622	800	803
	.05	687	**641**	709	**689**	731	809	843	1599	1600
	.10	905	938	977	1059	1070	1600	1600	3199	3199
	.15	1144	1279	1288	1505	1505	2400	2400	4798	4798
	.20	1396	1626	1629	2000	2000	3199	3199	6396	6397
	.25	1658	2000	2000	2500	2500	3999	3999	7997	7997
$\frac{4}{5}$	.025	589	**571**	596	**584**	602	**588**	622	800	803
	.05	687	689	709	719	731	828	843	1599	1600
	.10	905	965	977	1067	1070	1600	1600	3198	3199
	.15	1144	1283	1288	1505	1505	2400	2400	4798	4798
	.20	1396	1628	1629	2000	2000	3199	3199	6396	6397
	.25	1658	2000	2000	2499	2500	3999	3999	7997	7997