Cluster Comput (2008) 11: 133–149
DOI 10.1007/s10586-007-0038-3

A heterogeneity-aware approach to load balancing of computational tasks: a theoretical and simulation study

Jun Huang · Soo-Young Lee

Received: 20 July 2007 / Accepted: 20 July 2007 / Published online: 8 November 2007

Abstract One of the distinct characteristics of computing platforms shared by multiple users such as a cluster and a computational grid is heterogeneity on each computer and/or among computers. Temporal heterogeneity refers to variation, along the time dimension, of computing power available for a task on a computer, and spatial heterogeneity represents the variation among computers. In minimizing the average parallel execution time of a *target* task on a spatially heterogeneous computing system, it is not optimal to distribute the target task linearly proportional to the average computing powers available on computers. In this paper, effects of the temporal and spatial heterogeneity on performance of a target task have been analyzed in terms of the mean and standard deviation of parallel execution time. Based on the analysis results, an approach to load balancing for minimizing the average parallel execution time of a target task is described. The proposed approach whose validity has been verified through simulation considers temporal and spatial heterogeneities in addition to the average computing power on each computer.

Keywords Average execution time · Load balancing · Random variables · Spatial heterogeneity · Stochastic model · Temporal heterogeneity

1 Introduction

A cluster of computers is commonly used for high performance computing these days [1, 2]. Such a system is often shared by multiple users and computers in the system may not be all "identical." One of the most distinct characteristics of such an environment compared to a dedicated multiprocessor system is its heterogeneity [3], which may be classified into two types: *temporal* (variation with time on a computer) and *spatial* (variation with computer) heterogeneity.

Heterogeneity of such systems has been addressed from various viewpoints of high performance computing. Much of the efforts has gone into scheduling a stream of tasks (i.e., from the system's viewpoint) [4–6]. One of the important issues is how to utilize a heterogeneous computing system to optimize performance of a target task. This issue involves modelling the heterogeneity of the system [7, 8], performance estimation [9–12], reliability [13], scheduling and load balancing [14], etc.

The concepts of "machine heterogeneity" and "task heterogeneity" are employed in an effort to model task execution time on heterogeneous computing systems [7, 8]. The machine heterogeneity refers to the degree to which the machine execution times vary for a given task and the task heterogeneity refers to the degree to which the task execution times vary for a given machine. The main objective was to simulate different heterogeneous computing environments in evaluating the behavior of the task mapping heuristics.

A performance estimation model for heterogeneous parallel applications is proposed in [9]. They addressed the heterogeneity among the varying platforms and operating environments used and the possibility of having multiple tasks of the parallel application on each machine.

In [10], a model that calculates the slowdown imposed on applications in time-shared multi-user clusters is defined. Three kinds of "slowdown" are identified and the effects of slowdown on application performance are verified with emulated loads.

J. Huang · S.-Y. Lee (✉)
Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849, USA
e-mail: leesooy@eng.auburn.edu

J. Huang
e-mail: joey.huang@wdc.com

In [11], the issue of predicting parallel execution time of non-deterministic bulk synchronous computations, where computation time on each computer is modelled as a random variable, is considered. A tighter bound of the average parallel execution time has been derived. Though the randomness of computation time does not originate from heterogeneity of the computing system in their model, the results may be applicable to estimating parallel execution time on a heterogeneous computing system with minor modification.

In [12], the issue of modelling and characterizing parallel computing performance of heterogeneous Network Of Workstations (NOW) is considered, and the effects of heterogeneity on efficiency and scalability are investigated. The computing power (speed) "weight" is used to quantify the spatial heterogeneity among workstations, but the temporal heterogeneity of computing power on each workstation is not considered.

In [13], the probability of application failure is defined to quantify the reliability of a heterogeneous computing system. Minimizing execution time usually conflicts with improving reliability, and algorithms capable of trading off execution time for reliability are proposed.

In [14], a "stochastic scheduling" strategy, where the performance variability is considered in partitioning a data parallel application to be executed on a heterogeneous computing platform, is described. The mean and standard deviation of the predicted completion time of a task are used in scheduling, along with a "tuning factor" which specifies how "conservative" the scheduling is to be.

In most of the previous work on performance modelling, and scheduling or load balancing, (i) all types of heterogeneity were not taken into account, (ii) heterogeneity was often considered only implicitly, and (iii) effects of heterogeneity were not thoroughly analyzed.

In this paper, effects of spatial and temporal heterogeneity on the performance of a target task are first analyzed in detail. Also, it is shown that it is not optimal to partition a task over computers linearly proportional to their average computing powers in order to minimize the average parallel execution time. A heterogeneity-aware approach to load balancing, designed to minimize the average parallel execution time of a target task in a spatially heterogeneous computing environment, is then proposed in order to provide a theoretical foundation for developing a practical load balancing scheme. The proposed approach explicitly takes into account the standard deviation of available computing power in addition to the average computing power available on each computer. It has been shown that the average parallel execution time of a target task can be significantly reduced by the approach. The early results of this work were presented at two conferences [15, 16].

The organization of this paper is as follows: In Sect. 2, the system and task models are described, and quantitative definitions of temporal and spatial heterogeneities are given. In Sect. 3, performance measures are formulated with theoretical models. In Sect. 4, a two-step heuristic load balancing approach is described. In Sect. 5, simulation results are provided with discussion to show the effects of heterogeneity on performance, and improvement by using the proposed load balancing scheme. A summary is given in Sect. 6.

2 Models

2.1 System model

The high performance computing system to be employed in this study consists of N heterogeneous computers interconnected via a communication network. The speed of processor, speed and capacity of memory (including cache), and bandwidth of I/O system, etc. may not be the same on all computers. Different software, including the OS, may be available on different computers.

All these hardware and software components collectively affect the effective computing power available for applications on each computer. It is the available computing power for a target task, that eventually determines execution time of the task. Hence, in this paper, "availability" of computing power and communication bandwidth is employed in defining heterogeneity.

2.1.1 Availability

Let the maximum computing power of computer C_i be denoted by α_i for a target task for $i = 1, \ldots, N$, where N is the number of computers. The computing power available for the task at time t may be expressed by $\alpha_i A_i(t)$ where $A_i(t)$ is the "availability" of C_i for the task at time t and $0 \leq A_i(t) \leq 1$. The mean and standard deviation of $A_i(t)$ are denoted by a_i and σ_{A_i}, respectively.

In the steady state, a_i and σ_{A_i} are assumed to be fixed and do not vary with time while $A_i(t)$ varies with time. In the time-varying state, not only $A_i(t)$ but also a_i and/or σ_{A_i} vary with time, i.e., one needs to use the notations $a_i(t)$ and $\sigma_{A_i}(t)$. The time-varying state is not considered in this paper.

Availability of communication bandwidth can be defined similarly with notations β_i, $B_i(t)$, b_i, and σ_{B_i} for the maximum, instantaneous, mean and standard deviation of bandwidth, respectively. To avoid redundancy, they are not defined separately. One difference is that $B_i(t)$ tends to decrease as N increases due to the contention on the shared media, while $A_i(t)$ is independent of N, especially for data parallel applications.

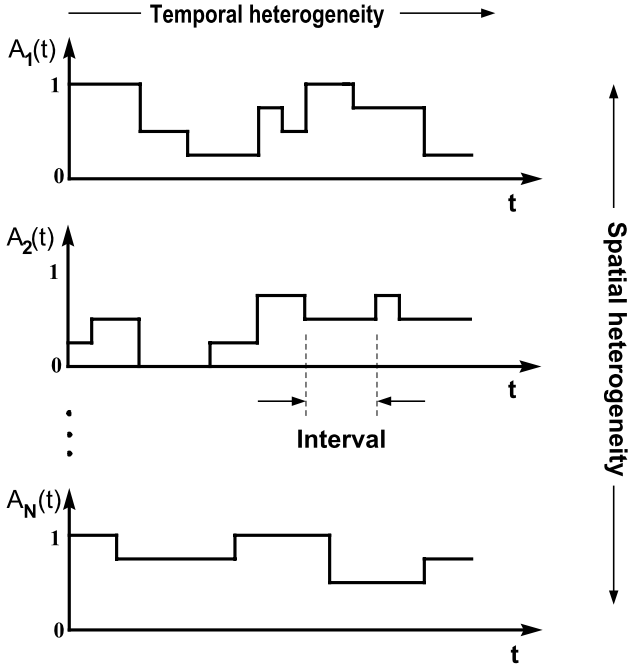


Fig. 1 Temporal and spatial heterogeneity. $A_i(t)$ is the availability of computing power on C_i for a target task, where $i = 1, \ldots, N$

2.1.2 Heterogeneity

When a set of heterogeneous computers is shared by multiple independent users, workload on each computer would vary with time and computer. Therefore, workload coupled with the heterogeneous hardware and software components makes availability (computing power) vary spatially and temporally. *Temporal heterogeneity* refers to variation of availability along the time dimension on a computer while *spatial heterogeneity* refers to that among computers as illustrated in Fig. 1.

With the notations given in the definition of availability above, a computer (C_i) exhibits temporal heterogeneity when $A_i(t)$ is a non-uniform function of time. A system consisting of multiple computers shows spatial heterogeneity when $a_i \neq a_j$ and/or $\sigma_{A_i} \neq \sigma_{A_j}$ for some i, j where $i \neq j$. These two types of heterogeneity will be quantified in Sect. 3.5.

The temporal heterogeneity is due to the fact that users submit their tasks at any time and the size of a task is variable, i.e., the task arrival and size are random. There are two sources of the spatial heterogeneity. One is system-related: hardware and software components are not the same for all computers. Also, the interconnection network may not be "symmetric." The other is task-related: the task (workload) characteristics (arrival time and size) may vary from one computer to another.

An interconnection network or networks are usually shared by computers. As a result, the spatial heterogeneity of communication bandwidth tends to be lower than that of computing power.

2.2 Task model

Two task models are employed in this paper: the *base* and *augmented* task models. In the *base* task model, a target task is assumed to be linearly partitionable over N computers, i.e., the sum of computational loads of subtasks is equal to that of the target task. In order to focus on effects of heterogeneous computing power on the performance of a target task, communication is not considered in the base model. Many data-parallel applications such as low-level image processing, (Monte Carlo) simulations, matrix computations, etc. would fit to this model well.

However, in many other applications, as a task is partitioned, communication among subtasks is unavoidable for sharing data and results, collecting local results, etc. The *augmented* task model is derived by incorporating "periodic" communication phases into the base task model. That is, execution of a target task consists of a sequence of alternating computation and communication phases. Hence, a communication phase can be considered as a synchronization point at which all N computers are required to be synchronized for data communication before proceeding to the next (computation) phase. There are many applications which can be represented by this model, e.g., iterative optimization, simulation of a physical or chemical phenomenon, medical image reconstruction, etc. The number of synchronization points is denoted by N_s.

3 Performance measures

Performance of a target task on a heterogeneous computing system may be measured in a few different ways. One popular metric is to use the execution time of the target task, whose minimization is the main objective of high performance computing. In a heterogeneous computing environment, execution time varies significantly due to the temporal and spatial heterogeneity and, therefore, the *average execution time* may be employed as a performance measure. From the stability point of view, one may want to minimize variation of the execution time, in which case the standard deviation of execution time can be used. Another way to quantify stability is to specify the probability that the execution time longer than a certain threshold is obtained. In this section, the performance measures to be employed in this paper are derived.

Let's denote the size of a target task by X and the portion of the target task assigned to the computer C_i by X_i. The execution time of X_i on C_i is referred to as T_i, which is a random variable, and its mean and standard deviation are

denoted by t_i and σ_{T_i}, respectively. In this paper, the same notation is used for a random variable and its observation in order to minimize the number of variables.

3.1 Mean and standard deviation of execution time

Referring to the definition of availability, the relationship between X_i and T_i may be expressed as follows:

$$\int_0^{T_i} \alpha_i A_i(t)\,dt = X_i \tag{1}$$

By taking the expectation on both sides,

$$t_i = \frac{X_i}{\alpha_i a_i} \tag{2}$$

Assuming the uncorrelatedness between $A_i(t)$ and $A_i(t')$, the standard deviation, σ_{X_i}, of the work completed during t_i can be shown to be $\sigma_{X_i} = \sqrt{t_i}\alpha_i\sigma_{A_i}$. Then, the standard deviation of T_i may be given by

$$\sigma_{T_i} = \sqrt{t_i}\frac{\sigma_{A_i}}{a_i} \tag{3}$$

The parallel execution time of X in a run, denoted by T, which is also a random variable, is given by $T = \max_i\{T_i\}$ where the notation { } refers to a set. The mean (τ) and standard deviation (σ_T) of parallel execution time of X are computed as follows:

$$\tau = E[T]$$
$$\sigma_T = \sqrt{E[(T-\tau)^2]}$$

where $E[\ ]$ is the expectation operator.

3.2 Spatially homogeneous environment

When all N computers have the same distribution of execution time (T') and the corresponding *cdf* (cumulative distribution function) is a monotonically increasing function, τ may be expressed as follows (note that the subscript i is not used in this subsection since all computers are identical and that $T_i = T'$ and $X_i = X'$ for all i):

$$\tau = \int NT'F(T')^{N-1} f(T')\,dT' \tag{4}$$

where $F(T')$ and $f(T')$ are the *cdf* and *pdf* (probability density function) of T', respectively.

It is possible to derive τ in the following form [11, 17], referring to (2) and (3):

$$\begin{aligned}\tau &= t + K(N)\sigma_{T'} \\ &= t + K(N)\sqrt{t}\frac{\sigma_A}{a} \\ &= \frac{X'}{\alpha a} + K(N)\sqrt{\frac{X'}{\alpha a}}\frac{\sigma_A}{a}\end{aligned} \tag{5}$$

where $K(N)$ is an increasing function of N and $K(1) = 0$.

Notice that the average parallel execution time of a target task consists of two components, one depending on the mean of availability and the other on the standard deviation of availability, i.e., temporal heterogeneity. It is to be noted that temporal heterogeneity makes the average parallel execution time increase beyond the average (sequential) execution time on each computer. The increase is larger when the number of computers employed is greater. Also, as will be shown later, a higher spatial heterogeneity leads to a longer average parallel execution time.

3.3 Synchronization

When the communication phase in the augmented task model does not include data communication, it can be considered as a synchronization point. That is, at each synchronization point, all computers need to wait for the "slowest" computer before they proceed to next computation phase. Suppose that there are N_s synchronization points in a target task, such that the amount of work to be done between two successive synchronization points is $\frac{X'}{N_s}$. In order to extract effects of synchronization only, let's assume that no data communication is actually carried out at each synchronization point. Referring to (5), the mean parallel execution time in the jth phase, $\tau^{(j)}$, can be expressed as $\frac{X'}{\alpha a N_s} + K(N)\sqrt{\frac{X'}{\alpha a N_s}}\frac{\sigma_A}{a}$ for $j = 1, \ldots, N_s$. Then, the average parallel execution time τ is $N_s\tau^{(j)}$. Hence,

$$\tau = \frac{X'}{\alpha a} + K(N)\sqrt{\frac{X'N_s}{\alpha a}}\frac{\sigma_A}{a} \tag{6}$$

It can be seen that τ increases as N_s increases even though no synchronization overhead (e.g., communication overhead for synchronization) is taken into account.

3.4 Stability of performance

Variation of T may be quantified by its standard deviation, σ_T, which is to be minimized when one wants to achieve a stable performance. Also, one may be interested in knowing the probability, to be referred to as "risk factor," that T is longer than a certain desirable threshold, τ_d. The risk factor, denoted by RF, is given by $Prob[T > \tau_d]$.

3.5 Temporal and spatial heterogeneity

In this section, temporal heterogeneity and spatial heterogeneity are quantified for computing power only, since the definitions would be identical for communication bandwidth. A difference is that spatial heterogeneity would generally be lower for communication bandwidth than for computing power. This is mainly because the communication paths are usually shared by computers.

3.5.1 Temporal heterogeneity

Temporal heterogeneity is defined on an individual computer, which indicates variability of computing power available for a task along the time dimension. Therefore, the standard deviation of the availability may be used to quantify temporal heterogeneity on each computer. Noting that the average availability may vary with computer (and also time) and that τ depends on the ratio of σ_{A_i} to a_i (refer to (5)), temporal heterogeneity, to be denoted by TH_i, on C_i is defined to be the normalized standard deviation of availability.

$$TH_i \triangleq \bar{\sigma}_{A_i} = \frac{\sigma_{A_i}}{a_i} \tag{7}$$

The notation TH will be used when TH_i is the same for all i (computers), i.e., spatially homogeneous, or, when the mean of TH_i among all computers is to be referred to.

3.5.2 Spatial heterogeneity

Spatial heterogeneity is defined for a group of computers to be employed to execute a target task. It represents variability of computing power among the computers.

Let's denote the mean and maximum of $\bar{\sigma}_{A_i}$ among C_i by $\bar{\sigma}_A^{mean}$ and $\bar{\sigma}_A^{max}$, respectively, i.e., $\bar{\sigma}_A^{mean} = \frac{1}{N}\sum_{i=1}^{N}\bar{\sigma}_{A_i} = TH^{mean}$ and $\bar{\sigma}_A^{max} = \max_i\{\bar{\sigma}_{A_i}\}$. Spatial heterogeneity denoted by SH for a set of computers $\{C_i\}$ is defined as

$$SH \triangleq \bar{\sigma}_A^{max} - \bar{\sigma}_A^{mean} \tag{8}$$

That is, SH quantifies variation of temporal heterogeneity among computers.

4 Two-step heuristic approach

Let's denote the computing power (*speed*) of C_i at t by $S_i(t)$ which has the mean s_i and the standard deviation σ_{S_i}. Then, noting that $S_i(t) = \alpha_i A_i(t)$, $s_i = \alpha_i a_i$ and $\sigma_{S_i} = \alpha_i \sigma_{A_i}$. When $\sigma_{S_i} = 0$ for all i, a target task is to be partitioned proportional to s_i, in order to achieve the minimum τ. However, such partitioning does not lead to the minimum τ in general when $\sigma_{S_i} \neq 0$ for some i.

Let's consider cases where $\sigma_{S_i} \neq 0$ for some i. Suppose that X is partitioned such that X_i is linearly proportional to s_i. Then, t_i (the average execution time on C_i) would be the same for all i, but σ_{T_i} (the standard deviation of execution time on C_i) would not be. It is to be pointed out that σ_{T_i} is linearly proportional to $\bar{\sigma}_{A_i}$ (or TH_i). Noting that τ is given by $E[\max_i\{T_i\}]$ rather than $\max_i\{T_i\}$ or $\max_i\{t_i\}$, it is possible to further reduce τ by taking $\{\sigma_{T_i}\}$ into account. Therefore, load balancing may be carried out in two steps as follows:

(1) X is partitioned over $\{C_i\}$ such that X_i is proportional to s_i,
(2) $\{X_i\}$ is further adjusted considering $\{\sigma_{T_i}\}$ and $\{s_i\}$.

In the following, this two-step load balancing approach is elaborated for different cases.

4.1 When $s_i = s_j$ for all i, j:

Consider the case of $N = 2$ (two computers). In Step (1) of the two-step load balancing approach, X is divided equally between C_1 and C_2 since $s_1 = s_2$. That is, after Step (1), $X_1 = X_2 = \frac{X}{2}$ and $t_1 = t_2$. Suppose that $\sigma_{T_1} > \sigma_{T_2}$ after Step (1). Then, it is possible to reduce τ further by transferring a certain amount of work ($\Delta X \geq 0$) from C_1 to C_2 in Step (2), i.e., $X_1' = X_1 - \Delta X$ and $X_2' = X_2 + \Delta X$, where X_i' is X_i after Step (2). Then,

$$\begin{aligned} t_1' &= t_1 - \frac{\Delta X}{s_1} \triangleq t_1 - \Delta t_1 \quad \text{and} \\ t_2' &= t_2 + \frac{\Delta X}{s_2} \triangleq t_2 + \Delta t_2 \end{aligned} \tag{9}$$

where t_i' is t_i after Step (2).

Note that $\Delta t_1 = \Delta t_2$ since $s_1 = s_2$. Also, from (3), it can be shown that

$$\sigma_{T_1}' = \sigma_{T_1}\sqrt{1 - \frac{\Delta X}{s_1 t_1}} \quad \text{and} \quad \sigma_{T_2}' = \sigma_{T_2}\sqrt{1 + \frac{\Delta X}{s_2 t_2}} \tag{10}$$

where σ_{T_i}' is σ_{T_i} after Step (2).

A heuristic scheme, to be referred to as *equalization scheme*, that can be employed in Step (2) equalizes the sum of the mean and standard deviation of execution time between two computers. That is, it determines ΔX such that

$$t_1' + \sigma_{T_1}' = t_2' + \sigma_{T_2}' \tag{11}$$

where t_i and σ_{T_i}' are given by (9) and (10), respectively.

The equalization scheme is illustrated in Fig. 2. The scheme attempts to reduce the probability that the parallel execution time of a target task, $T = \max_i\{T_i\}$, is large, in order to minimize its average parallel execution time.

Note that the heuristic equalization scheme can be generalized for an arbitrary number of computers ($N \geq 2$). Let $X_i' = X_i + \Delta X_i$ where ΔX_i can be negative or non-negative for $i = 1, \ldots, N$. Then, the scheme finds $\{\Delta X_i\}$ such that $(t_i' + \sigma_{T_i}')$ is the same for all i subject to $\sum_{i=1}^{N}\Delta X_i = 0$, where $t_i' = t_i + \frac{\Delta X_i}{s_i}$ and $\sigma_{T_i}' = \sigma_{T_i}\sqrt{1 + \frac{\Delta X_i}{s_i t_i}}$ (see (9) and (10)).

In cases where $\Delta X_i \ll X_i$, $(t_i' + \sigma_{T_i}')$ can be approximated as a linear function of ΔX_i using the property of $\sqrt{1 + \frac{\Delta X_i}{s_i t_i}} \simeq (1 + \frac{\Delta X_i}{2 s_i t_i})$, i.e.,

$$t_i' + \sigma_{T_i}' \simeq (t_i + \sigma_{T_i}) + \Delta X_i\left(\frac{1}{s_i} + \frac{\sigma_{T_i}}{2 s_i t_i}\right) \tag{12}$$

Table 1 Possible reduction in τ

	$s_1 = s_2$	$s_1 < s_2$		
	$\sigma_{T_1} > \sigma_{T_2}$	$\sigma_{T_1} > \sigma_{T_2}$	$\sigma_{T_1} < \sigma_{T_2}$ $(s_1 \ll s_2)$	$\sigma_{T_1} < \sigma_{T_2}$ $(s_1 \simeq s_2)$
Direction of ΔX	$C_1 \rightarrow C_2$	$C_1 \rightarrow C_2$	$C_1 \rightarrow C_2$	$C_1 \leftarrow C_2$
Possible reduction in τ	Smallest	Largest	Small	Small

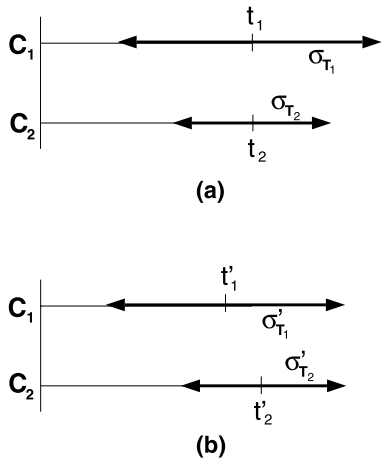


Fig. 2 Equalization scheme: (**a**) after Step (1) where a target task is partitioned such that the average execution time is the same on all computers and (**b**) after Step (2) where the partitioning is adjusted such that $t'_i + \sigma'_{T_i}$ is equalized for all computers

Then, it becomes the problem of solving linear equations of $\{\Delta X_i\}$.

4.2 When $s_i \neq s_j$ for some i, j:

Again, consider the case of $N = 2$, and suppose that $s_1 < s_2$. After Step (1), $X_1 = \frac{s_1 X}{s_1+s_2}$ and $X_2 = \frac{s_2 X}{s_1+s_2}$. What is to be done in Step (2) to further reduce τ depends on $\{\sigma_{T_i}\}$ and $\{s_i\}$, as discussed below, and can be generalized for cases where $N > 2$.

(a) $\sigma_{T_1} > \sigma_{T_2}$: In this case, ΔX is to be moved from C_1 to C_2 and the equalization scheme may be employed in determining ΔX. One difference is that $\Delta t_1 > \Delta t_2$. In other words, the same increase in t_2 (by moving ΔX from C_1 to C_2) results in a larger decrease in t_1, leading to a larger reduction in τ, compared to the case where $s_1 = s_2$.

(b) $\sigma_{T_1} < \sigma_{T_2}$ and $s_1 \ll s_2$: It is still possible to reduce τ by transferring ΔX from C_1 to C_2 though the equalization scheme may not be used since the condition, $t'_1 + \sigma'_{T_1} = t'_2 + \sigma'_{T_2}$, cannot be satisfied. Reduction in τ would be smaller than that in (a).

(c) $\sigma_{T_1} < \sigma_{T_2}$ and $s_1 \simeq s_2$: In this case, ΔX is to be moved from C_2 to C_1 and the equalization scheme can be used. Reduction in τ would be smaller than that in (a) or (b).

It should be clear that transferring ΔX in the other direction than specified above would result in a longer τ. The above discussion on reduction in τ for different cases is summarized in Table 1.

5 Simulation results and discussion

5.1 Simulation

Availability is assumed to have a uniform distribution (it was observed that other distributions such as a "truncated" Gaussian distribution resulted in similar trends). Then, the distribution of execution time on each computer looks similar to a Gaussian or Gamma distribution which was adopted also in another study [7]. The average availability a_i (or average computing power $\alpha_i a_i$) may vary with computer (C_i). However, a_i (or $\alpha_i a_i$) can be effectively normalized in task partitioning, i.e., load balancing for the average availability (or average computing power) varying with computer can be done easily by assigning X_i, proportional to a_i (or $\alpha_i a_i$), to C_i. Hence, a_i is set to 0.5 with $\alpha_i = 1.0$ for all i in the simulation in order to focus on the effects of heterogeneity in the availability (instead of the average availability or computing power) and to maximize the range of variation of σ_{A_i} (note that $0 \leq a_i \leq 1.0$). Then, the maximum σ_{A_i} ($\bar{\sigma}_{A_i}$) is $\frac{1}{2\sqrt{3}}$ ($\frac{1}{\sqrt{3}}$). The computing power α_i of C_i is set to 1.0 also for normalization purpose.

The notion of "interval" is adopted to quantify the time duration in which availability ($A_i(t)$) remains constant. The interval is mostly affected by other tasks (distributions of their arrival time and size) and the local scheduler on C_i and is to be modelled by a random variable. Note that decreasing the interval given a fixed task size is equivalent to increasing the task size with the interval fixed, and vice versa. A larger interval (for a given task size) leads to a higher chance for load imbalance among computers. The interval is generated from a uniform distribution of which range is $[0, 20]$ except when it is varied.

Simulation was repeated 1000 times for each case and results were averaged for the graphs in the following sections.

5.2 Effects of heterogeneity on a target task

In this section, some of the simulation results are presented to discuss effects of temporal and spatial heterogeneity on

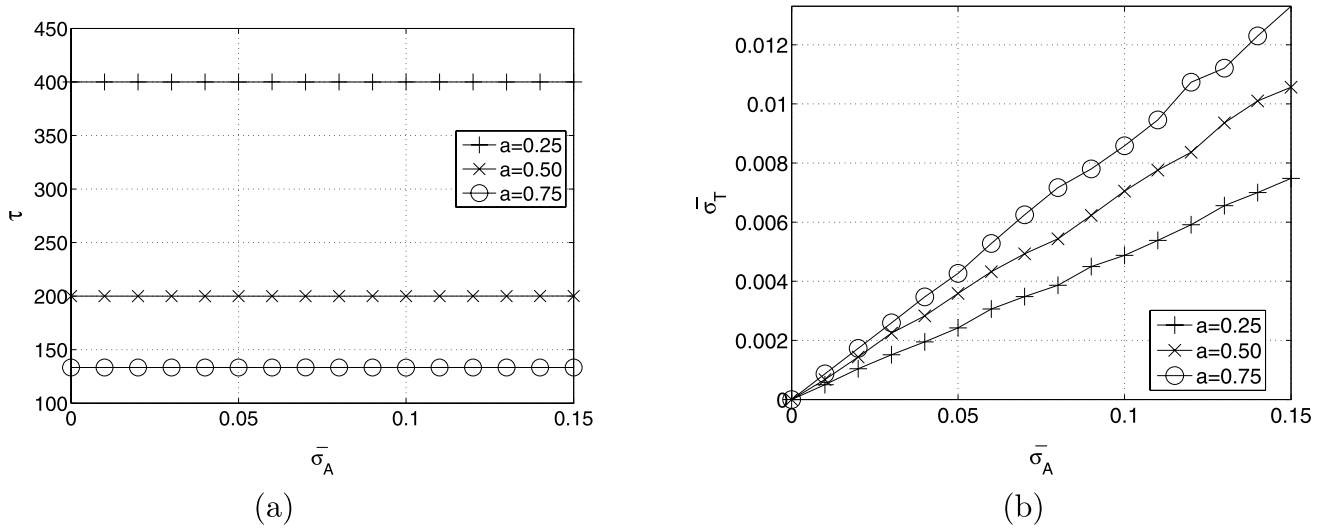


Fig. 3 (**a**) Average execution time and (**b**) normalized standard deviation of execution time on a single computer where $X = 100$

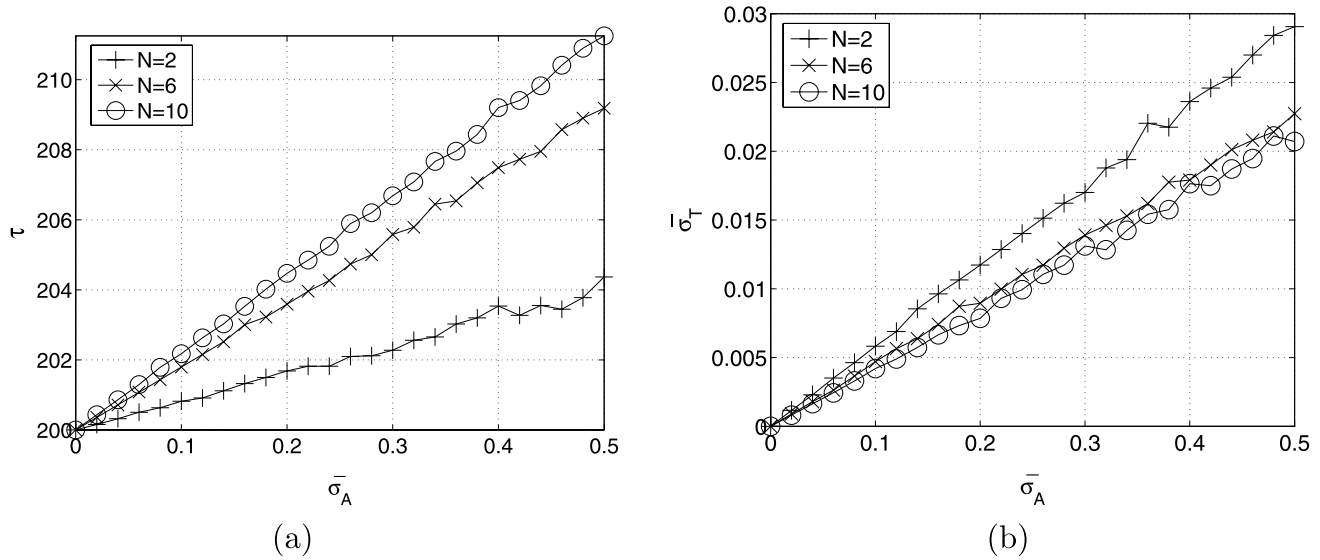


Fig. 4 (**a**) Average parallel execution time and (**b**) normalized standard deviation of parallel execution time on N computers where $X_i = 100$ for $i = 1, \ldots, N$

the performance of a target computing task. In the graphs where variation of the parallel execution time (T) is analyzed, the standard deviation of T normalized by τ is used, i.e., $\frac{\sigma_T}{\tau} \triangleq \bar{\sigma}_T$. The results without communication overhead (but with synchronization in some cases) are provided in Figs. 3–12, and those with communication overhead are in Figs. 13 and 14.

5.2.1 *Spatially homogeneous and temporally heterogeneous*

In Fig. 3, τ and $\bar{\sigma}_T$ are plotted when $N = 1$. As can be seen in the figure, it is clear that the average sequential execution time is not affected by $\bar{\sigma}_A$, but its variation increases proportional to $\bar{\sigma}_A$, as shown in (2) and (3).

In Fig. 4, effects of $\bar{\sigma}_A$ on τ and $\bar{\sigma}_T$ are shown for different values of N. As predicted in (5), τ and $\bar{\sigma}_T$ increase almost linearly proportional to $\bar{\sigma}_A$ (which is TH) when multiple computers are employed. The increase due to $\bar{\sigma}_A$ is larger for a larger N.

5.2.2 *Spatially and temporally heterogeneous*

In Fig. 5, dependency of τ and $\bar{\sigma}_T$ on SH and TH is shown when N is fixed to 8. In this graph, when SH is zero (i.e., on the TH axis), $\bar{\sigma}_A$, which is TH_i, is the same

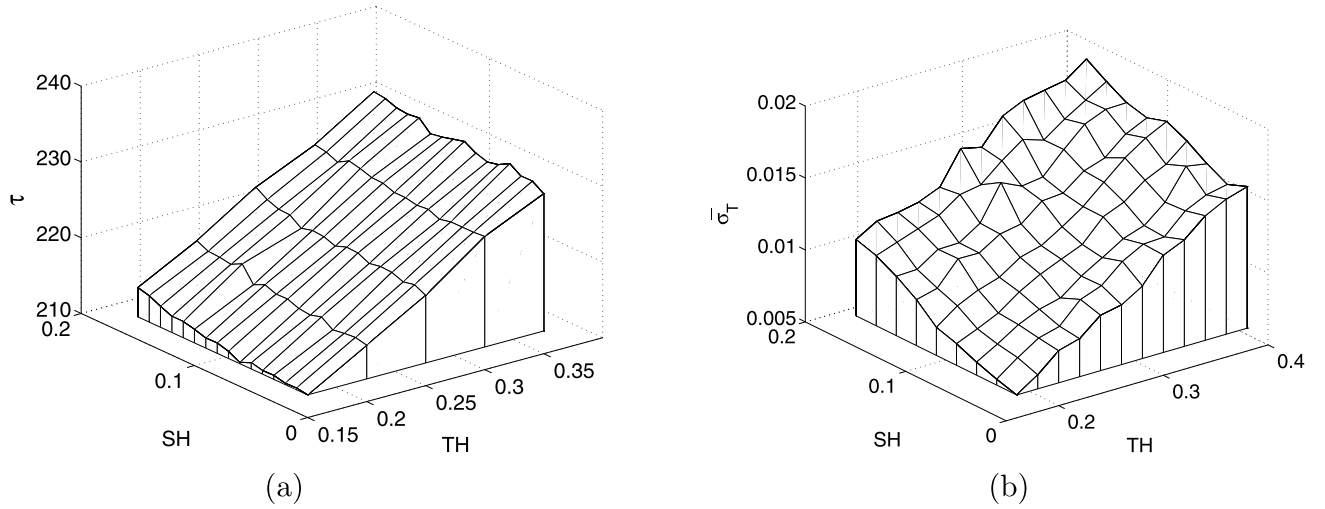


Fig. 5 (**a**) Average execution time and (**b**) normalized standard deviation of execution time when *SH* and *TH* are varied with N fixed at 8 where $X_i = 100$ for $i = 1, \ldots, 8$

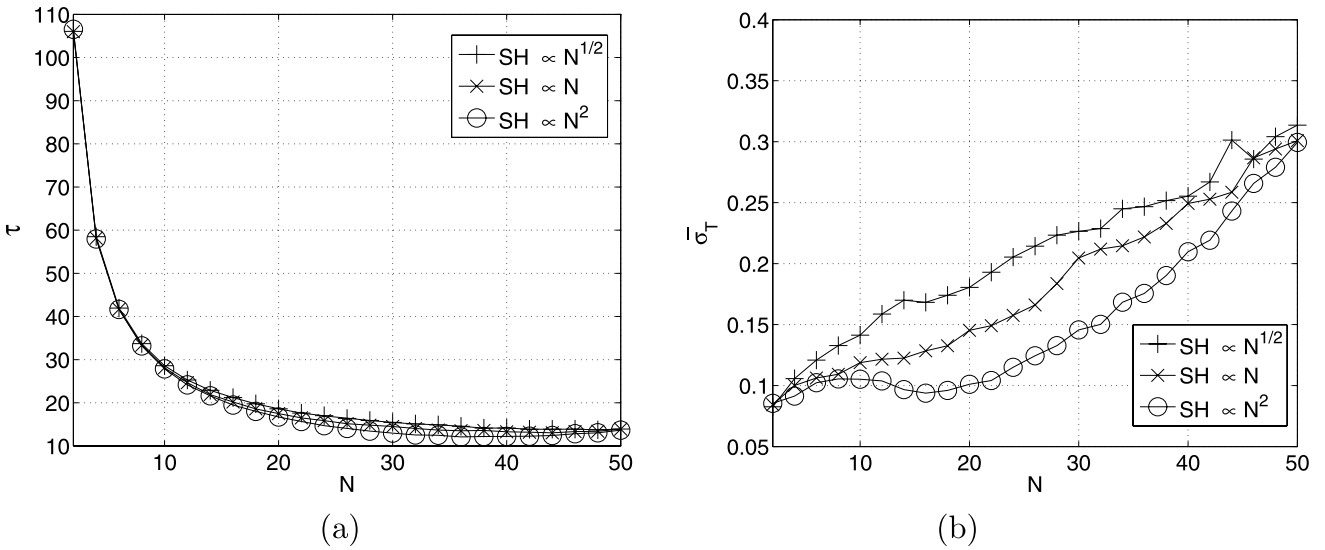


Fig. 6 (**a**) Average execution time and (**b**) normalized standard deviation of execution time. $X_i = \frac{X}{N}$ for $i = 1, \ldots, N$ and $X = 100$. As N increases, $\bar{\sigma}_A^{mean}$ remains fixed

for all computers. When *SH* is greater than zero, distribution of $\bar{\sigma}_{A_i}$ among computers is *linear* such that $\bar{\sigma}_{A_i} = \bar{\sigma}_A^{mean} + 2(\frac{i-1}{N-1} - 0.5)(\bar{\sigma}_A^{max} - \bar{\sigma}_A^{mean})$ for $i = 1, \ldots, N$. That is, $TH = \bar{\sigma}_A^{mean}$ in these graphs. As *SH* increases, τ increases significantly, especially when $\bar{\sigma}_A^{mean}$ also increases (going diagonally from the origin on the $SH - TH$ plane).

Now, suppose that the size of a target task, X, is fixed independent of N and it is uniformly distributed over N computers. The larger the set of computers employed for the target task is, the larger heterogeneity (*SH*) among computers becomes in general. Two cases are considered: (i) when $\bar{\sigma}_A^{mean}$ is fixed while *SH* increases and (ii) when both of *SH* and $\bar{\sigma}_A^{mean}$ increase. In both cases, three different situations, in terms of how $\bar{\sigma}_A^{max}$ is increased, are considered: proportional to $\sqrt{N}$, N, and N^2. Simulation results for cases (i) and (ii) are provided in Figs. 6 and 7, respectively. As N increases, τ decreases almost inversely proportional to N and then starts to increase beyond a certain N especially when *SH* increases rapidly. In contrast, $\bar{\sigma}_T$ monotonically increases as N increases. It increases sharply after a certain value of N in the case that *SH* increases proportional to N^2.

Speed-up is shown for the cases of (i) and (ii) in Figs. 8a and 8b, respectively. The reason why the three curves meet when $N = 50$ is that the simulation was set up such that the distribution of $\bar{\sigma}_A$ among computers becomes identical in all three situations when N is increased to 50. Hence, what

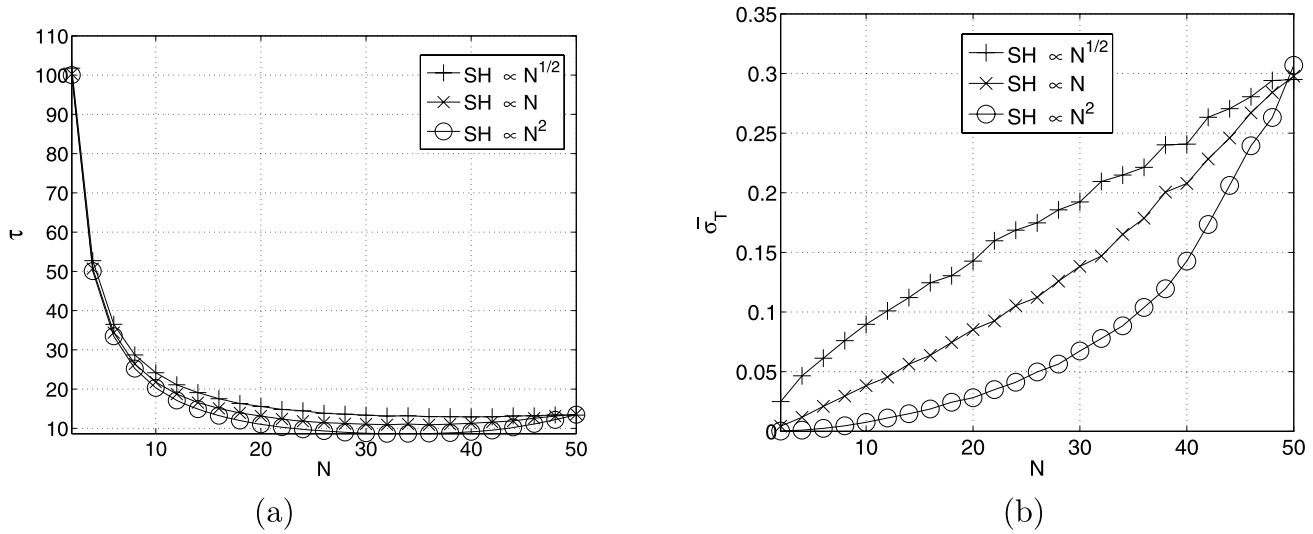


Fig. 7 (**a**) Average execution time and (**b**) normalized standard deviation of execution time. $X_i = \frac{X}{N}$ for $i = 1, \ldots, N$ and $X = 100$. As N increases, $\bar{\sigma}_A^{mean}$ increases

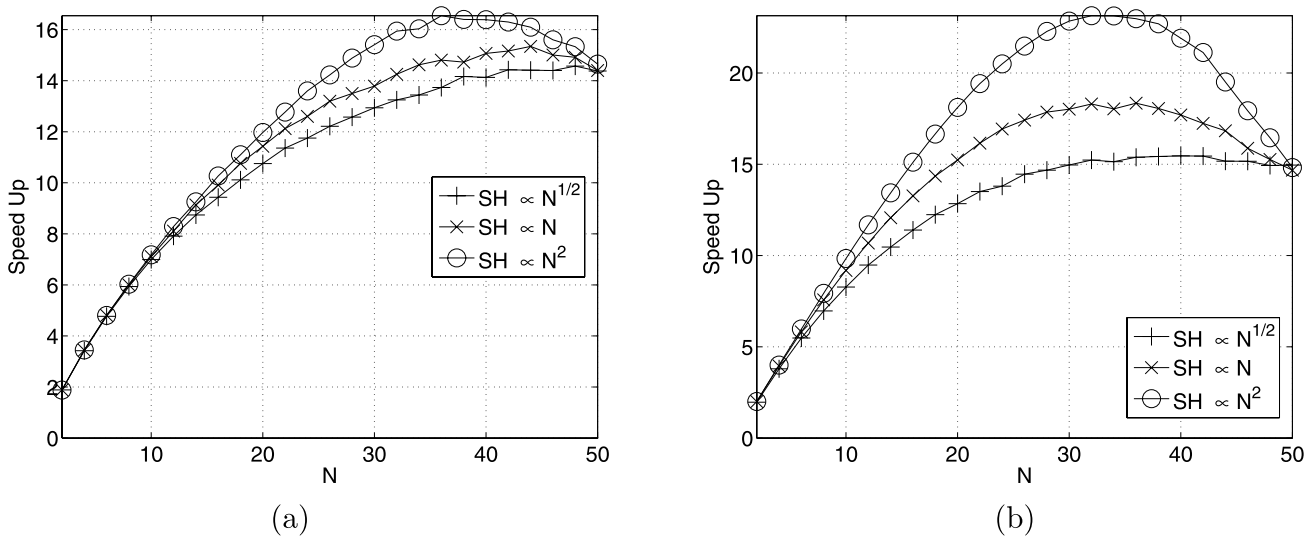


Fig. 8 Speed-up when (**a**) $\bar{\sigma}_A^{mean}$ remains fixed and (**b**) $\bar{\sigma}_A^{mean}$ increases as N increases. $X_i = \frac{X}{N}$ for $i = 1, \ldots, N$. $X = 100$

is to be observed in these graphs is the "shape" (trend) of each curve. It is clear that spatial heterogeneity *SH* alone keeps one from employing more than a certain number of computers even for an embarrassingly parallel task. Also, it can be induced that the complexity of the function $K(N)$ in (5) is at least $O(\sqrt{N})$.

5.2.3 Synchronization

The number of synchronization points, N_s, is varied for different $\bar{\sigma}_{A_i}$, to observe its effects on τ and $\bar{\sigma}_T$ when $SH = 0$ in Fig. 9 and when $SH > 0$ in Fig. 10. In Fig. 9, it is seen that as N_s increases τ increases since the idle time due to synchronization increases. However, $\bar{\sigma}_T$ decreases. This can be explained as follows. The increased T_i (or t_i) due to a larger N_s leads to the increased σ_T (refer to (3)). However, T_i increases faster than σ_T, causing $\bar{\sigma}_T$ to decrease. The increase rate in τ is larger for a larger $\bar{\sigma}_A$ (*TH*) and a higher *SH*, as shown in Fig. 10.

5.2.4 Granularity of availability

When the size of a target task (in terms of its execution time) is much larger than an *interval* (the duration when availability remains constant, refer to Sect. 5.1) (equivalently, the *interval* is much smaller than the target task size), effects

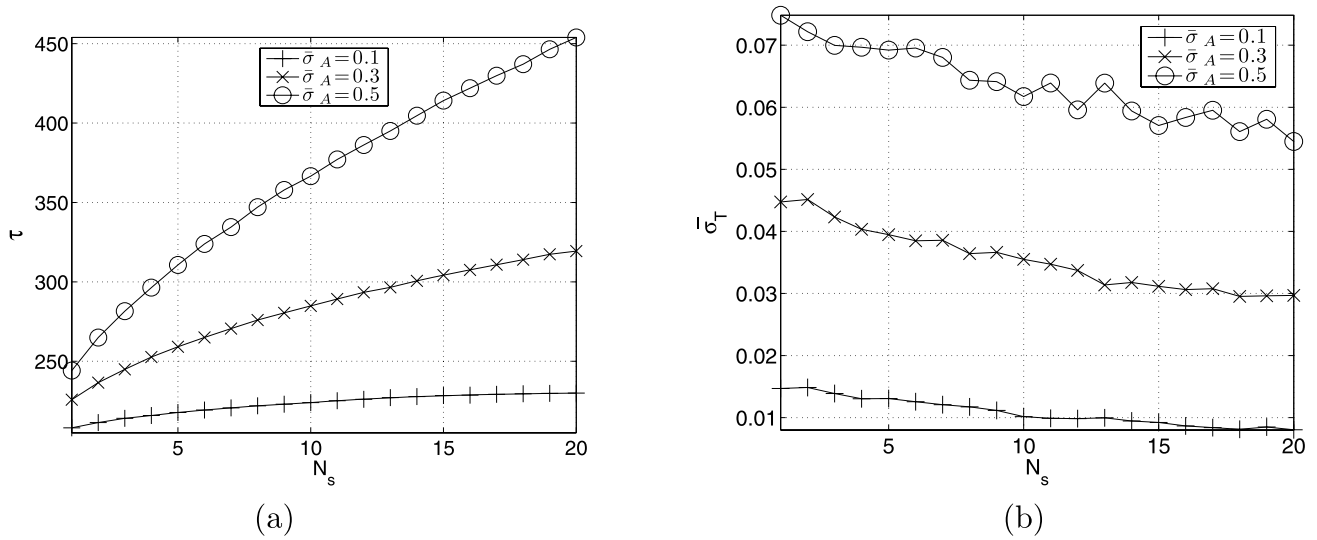


Fig. 9 (**a**) Average execution time and (**b**) normalized standard deviation of execution time as functions of N_s. $X = 1000$, $N = 10$, and $X_i = \frac{X}{N} = 100$

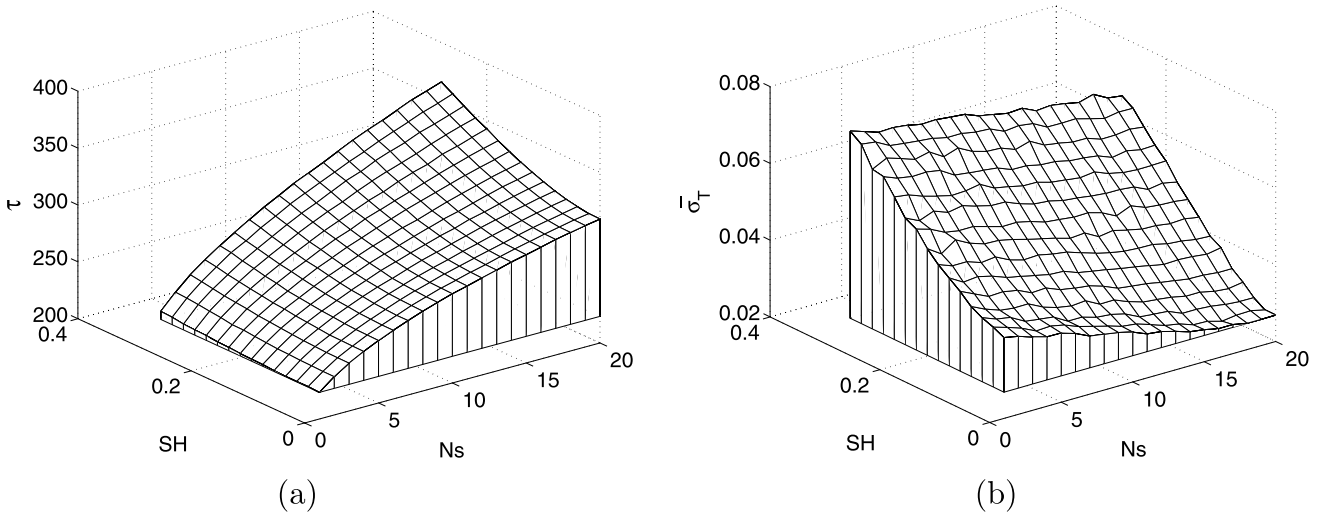


Fig. 10 (**a**) Average parallel execution time and (**b**) normalized standard deviation of parallel execution time as functions of SH and N_s. $X = 1000$, $N = 10$, $X_i = \frac{X}{N} = 100$, and $\bar{\sigma}_A^{mean} = 0.28$

of heterogeneity (temporal or spatial) are reduced since the probability of load imbalance decreases. Dependency on interval is shown when $SH = 0$ in Fig. 11 and when $SH > 0$ in Fig. 12. As the *interval* increases (i.e., availability varies more slowly with time) for a given target task, effects of TH and SH become larger making τ and $\bar{\sigma}_T$ larger. Equivalently, a smaller task would be more sensitive to heterogeneity.

5.2.5 *Communication overhead*

Simulation results when the augmented task model is adopted are provided in Figs. 13 and 14. In these graphs, τ includes both computation and communication times. In Fig. 13, SH_{comp} and SH_{comm} represent the spatial heterogeneity of availability in computing power and communication bandwidth, respectively. As expected for the simulation model, SH_{comp} and SH_{comm} have the similar effect on τ and $\bar{\sigma}_T$ (refer to Sect. 5.2.2). In this simulation, the ratio of computation time to communication overhead is set to 6. That is why τ increases more slowly along SH_{comm} than SH_{comp}.

In Fig. 14, SH is the same for both the computing power availability and bandwidth availability. As N increases, τ decreases initially and then turns around to increase after a certain value of N. This is mainly because communication overhead becomes dominant as N increases. It

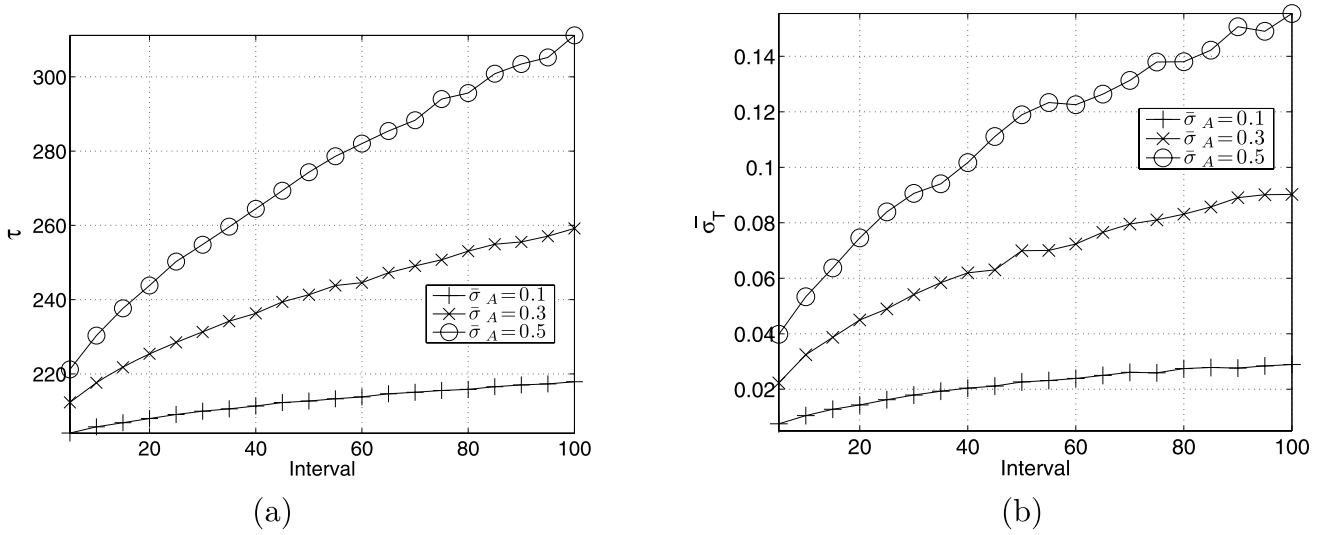


Fig. 11 (**a**) Average execution time and (**b**) normalized standard deviation of execution time as functions of interval. $X = 1000$, $N = 10$, and $X_i = \frac{X}{N} = 100$

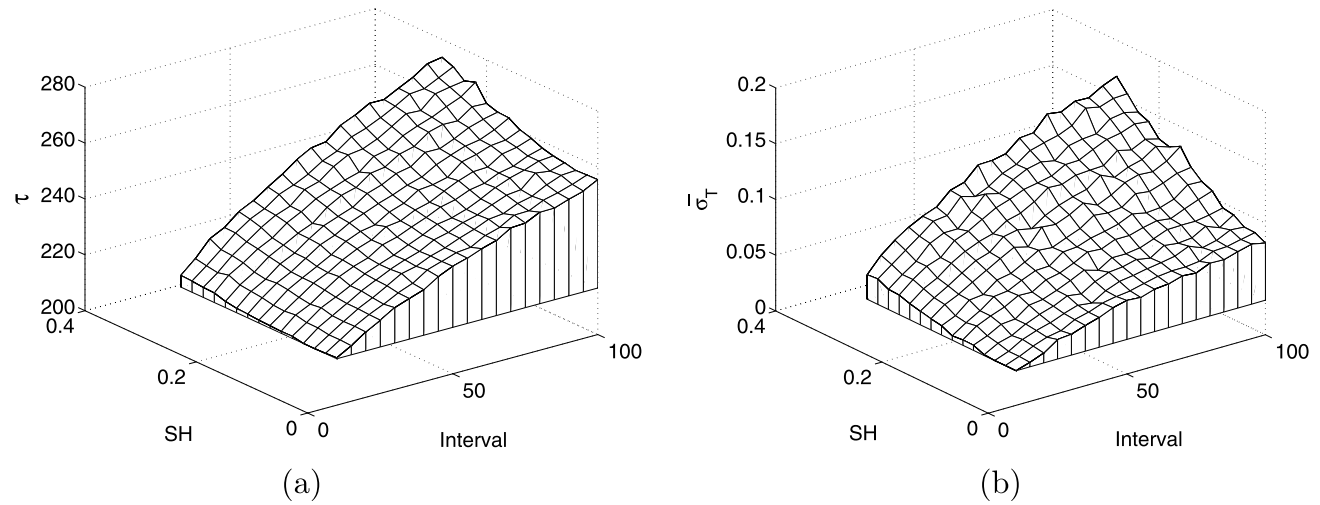


Fig. 12 (**a**) Average parallel execution time and (**b**) normalized standard deviation of parallel execution time as functions of *SH* and interval. $X = 1000$, $N = 10$, $X_i = \frac{X}{N} = 100$, and $\bar{\sigma}_A^{mean} = 0.28$

is also observed that the trend is more visible when the bandwidth availability (and computing power availability) shows a higher spatial heterogeneity (*SH*). For example, in Fig. 14a, it is seen that τ turns around to increase when *SH* is relatively high while it monotonically decreases when $SH = 0$.

5.2.6 *Load distribution and stability (risk factor)*

So far, only the cases where a target task is uniformly distributed among computers have been discussed. Let's examine the effect of distribution of the target task with *SH* varied. In Fig. 15, τ and $\bar{\sigma}_T$ are plotted as functions of dX with $N = 2$ where $dX = \frac{X_1 - X_2}{2}$ and $\Delta\bar{\sigma}_A = \bar{\sigma}_{A_1} - \bar{\sigma}_{A_2}$. In Fig. 16, the risk factor defined in Sect. 3.4 is shown as a function of dX. The same set of results are provided when $N = 10$ in Figs. 17 and 18. In these simulations, $\bar{\sigma}_{A_i}$ is increased *linearly* with i among 10 computers as before (refer to Fig. 5). X_i is also varied linearly with respect to i, i.e., $X_i = X_0 + Slope_{X_i}(i - 1)$ for $i = 1, \ldots, N$. Therefore, a negative slope means that a computer with a smaller $\bar{\sigma}_{A_i}$ is assigned a larger fraction of X (a larger X_i).

It may be observed from these results that, by assigning more work (a larger fraction of target task) to a computer with a smaller variation of availability, (a) a shorter (minimum) τ is obtained, (b) $\bar{\sigma}_T$ is sharply decreased, and (c) the

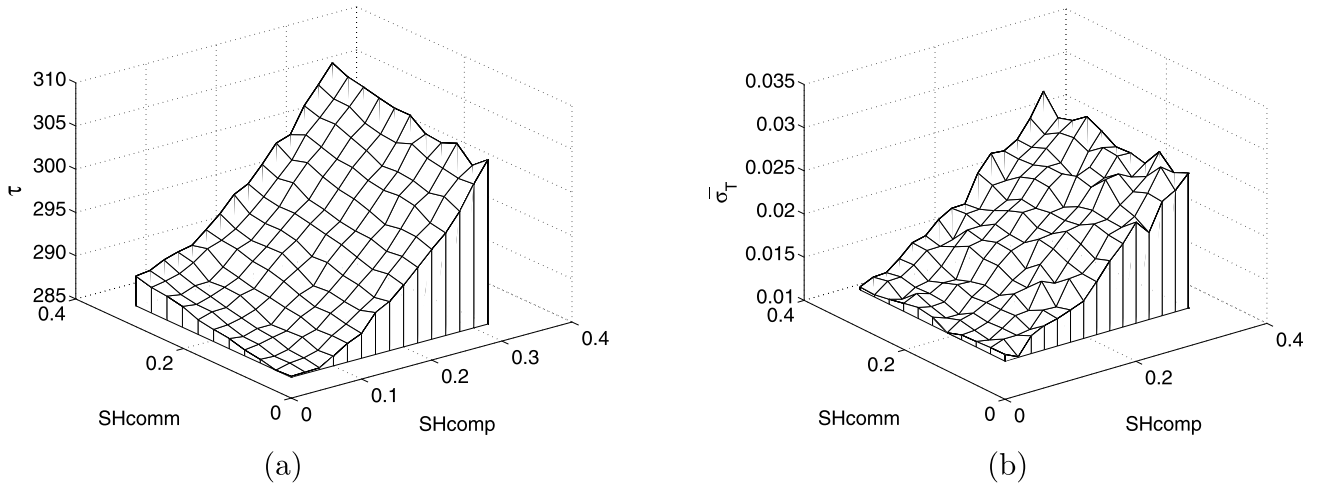


Fig. 13 (**a**) Average execution time (computation and communication times) and (**b**) normalized standard deviation of execution time as functions of SH_{comm} and SH_{comp}. $N = 20$, $X = 2000$, $Ns = 10$, $X_i = \frac{X}{N} = 100$, and $\bar{\sigma}_A^{mean} = \bar{\sigma}_B^{mean} = 0.28$

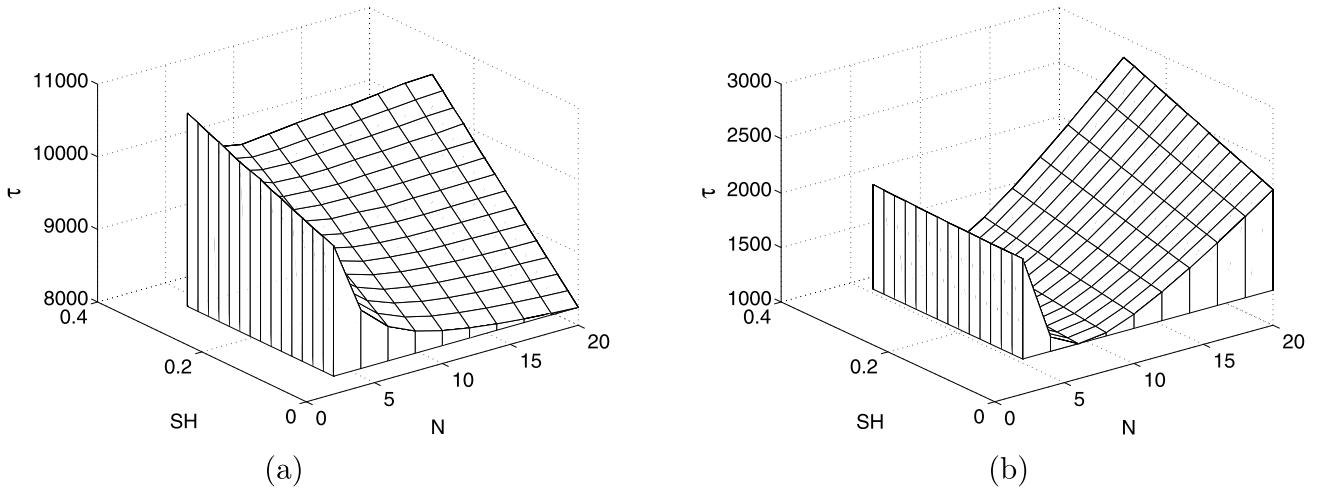


Fig. 14 Average execution time: (**a**) when communication overhead is independent of N and $X = 400$, and (**b**) when communication overhead increases linearly proportional to N and $X = 2000$. $Ns = 10$ and $\bar{\sigma}_A^{mean} = \bar{\sigma}_B^{mean} = 0.28$

risk factor is quickly reduced. These trends are more explicit for a larger N and a higher SH. It is also seen that the amount of work to be assigned to each computer to minimize τ, $\bar{\sigma}_T$, and RF depends on SH (and the distribution of $\bar{\sigma}_{A_i}$). That is, for achieving the minimum execution time on a spatially heterogeneous computing system, it is not optimal to distribute a target task linearly proportional to the *mean* availabilities of computers.

5.3 Load balancing

Let's first consider a case where the average computing speed is the same for all computers, i.e., $s_i = s_j$ for all i, j. First, $X_i = X_j$ for all i, j in Step (1) of the two-step load balancing approach described in Sect. 4. It is assumed that $t_i = 100$ and $\sigma_{T_i} = \frac{i-1}{N-1}\sigma_{max}$, for $i = 1, \ldots, N$, after Step (1). In Step (2), X_i is adjusted such that $t_i' = t_i + Slope_T(i - \frac{N-1}{2})$ for $i = 1, \ldots, N$. $Slope_T = 0$ corresponds to the case where X_i' is proportional to s_i (the *average* computing speed or power), i.e., Step (1) only. A negative $Slope_T$ indicates that a computer with a larger σ_{T_i} is assigned a larger ΔX_i (refer to Sect. 4) in Step (2) in addition to X_i allocated in Step (1).

In Fig. 19, τ and *percentage standard deviation* of $(t_i' + \sigma_{T_i}')$ are shown as functions of $Slope_T$ which specifies how a target task is distributed over N computers. The percentage standard deviation of $(t_i' + \sigma_{T_i}')$ is a measure which reflects the degree of equalization, and is defined as:

$$\%\ \text{standard deviation of } (t_i' + \sigma_{T_i}') = \frac{\sigma_{(t'+\sigma_T')}}{(t' + \sigma_T')} \times 100\% \tag{13}$$

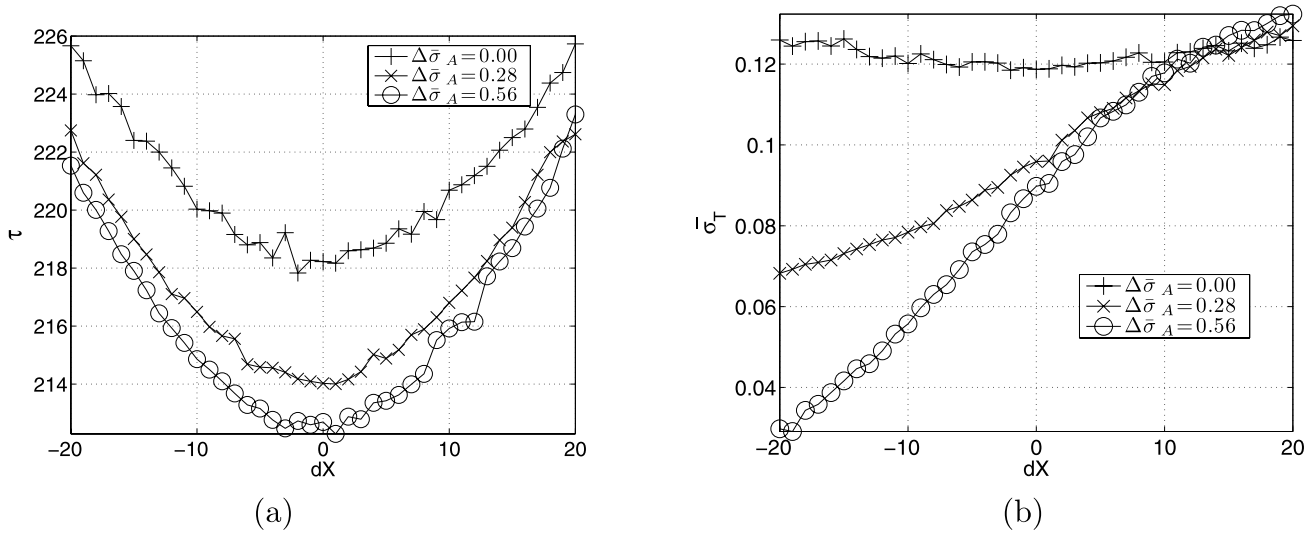


Fig. 15 (**a**) Average execution time and (**b**) normalized standard deviation of execution time on two computers where $X_1 = \frac{X}{2} - dX$, $X_2 = \frac{X}{2} + dX$, and $X = 100$

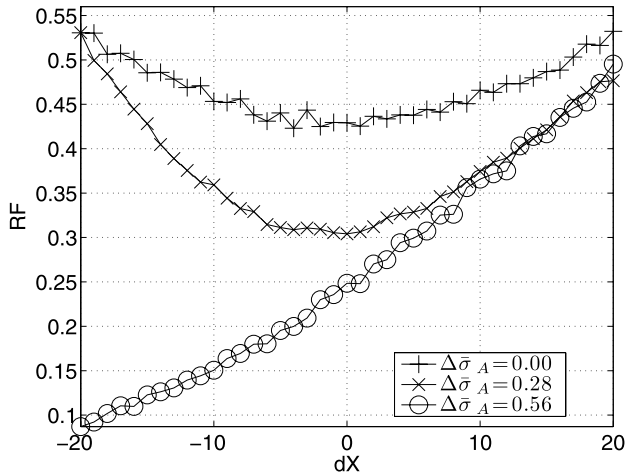


Fig. 16 Risk factor on two computers where $X_1 = \frac{X}{2} - dX$, $X_2 = \frac{X}{2} + dX$, and $X = 200$. $\tau_d = 221$. $\Delta\bar{\sigma}_A = \bar{\sigma}_{A_2} - \bar{\sigma}_{A_1}$

where $\overline{(t' + \sigma'_T)}$ and $\sigma_{(t'+\sigma'_T)}$ are average and standard deviation of $(t'_i + \sigma'_{T_i})$ over N computers:

$$\overline{(t' + \sigma'_T)} = \frac{\sum_{i=1}^{N} (t'_i + \sigma'_{T_i})}{N} \tag{14}$$

$$\sigma_{(t'+\sigma'_T)} = \sqrt{\frac{\sum_{i=1}^{N} (t'_i + \sigma'_{T_i} - \overline{t' + \sigma'_T})^2}{N}} \tag{15}$$

First of all, it is to be noted from Fig. 19 that it is not optimal to distribute a target task linearly proportional to the *average* computing power of computers. In Fig. 19a, it is clear that the performance improvement (reduction in τ) achieved by Step (2) (more precisely, Step (1) + Step (2)) over Step (1) is significant, and is larger for a larger $\Delta\sigma_T$ (equivalently, a higher spatial heterogeneity). Comparing Figs. 19a and 19b, it can be seen that the equalization scheme employed in Step (2) works well, i.e., the distribution of X minimizing τ closely matches with that minimizing variation in $t_i + \sigma_{T_i}$.

In Fig. 20, a system of two computers where $s_1 < s_2$ is considered. In these graphs, τ is plotted as a function of ΔX. Again, $\Delta X = 0$ corresponds to the cases where X is divided between two computers proportional to their average computing powers (s_1 and s_2). It can be observed that reduction in τ, which can be achieved by Step (2), is larger for a larger s or a larger $\Delta\sigma_T$. Let's define the percentage error (ε) of the equalization scheme as $\frac{\Delta\tau_{max} - \Delta\tau}{\Delta\tau_{max}} \times 100$ where $\Delta\tau_{max}$ and $\Delta\tau$ are the maximum possible and achieved (by the heuristic) reductions in τ, respectively. In this set of results, the average ε was 5.1%.

In Fig. 21, percentage reduction in τ, which is achieved by Step (2), is analyzed with the number of computers (N) varied. In this simulation, s_i increases and σ_{T_i} decreases linearly proportional to i, i.e., a faster computer has a smaller variation of execution time. The *percentage reduction in* τ is defined to be $\frac{\tau_{(1)} - \tau_{(2)}}{\tau_{(1)}} \times 100$ where $\tau_{(1)}$ and $\tau_{(2)}$ are τ achieved by Steps (1) and (2), respectively. It can be seen that the percentage reduction increases as N increases. That is, the performance improvement by Step (2) becomes greater when a larger number of computers are employed. In Fig. 21a, $\Delta\sigma_T$ is fixed independent of N. However, in reality, $\Delta\sigma_T$ would usually increase as N increases. Therefore, in such a case, one may expect a larger reduction in τ. Note that a larger $\Delta\sigma_T$ leads to a greater reduction in τ as shown in Fig. 21b.

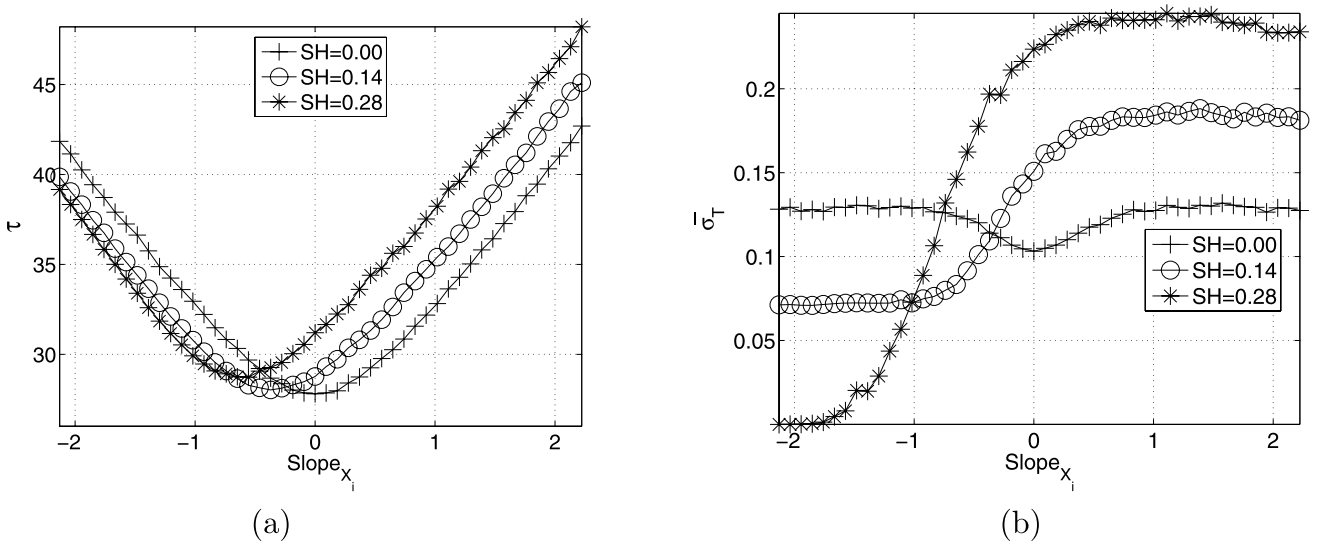


Fig. 17 (**a**) Average execution time and (**b**) normalized standard deviation of execution time on 10 computers. $X = 100$ and $X_i = X_0 + Slope_{X_i}(i-1)$ for $i = 1, \ldots, 10$

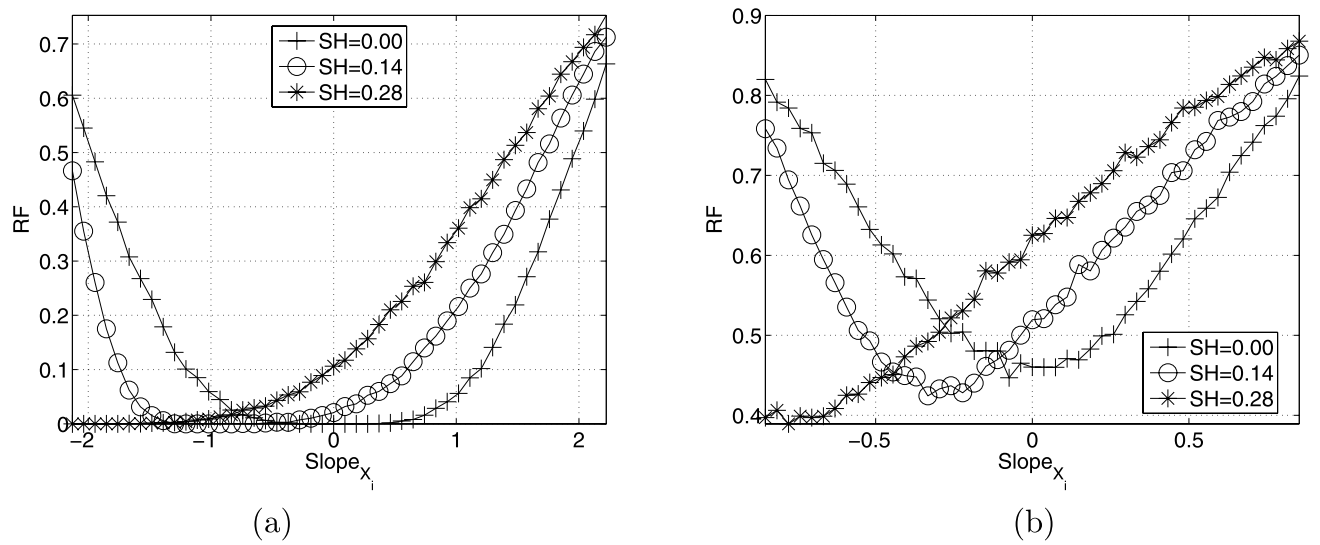


Fig. 18 Risk factor on 10 computers (**a**) $\tau_d = 40$ and (**b**) $\tau_d = 28$. $X = 100$ and $X_i = X_0 + Slope_{X_i}(i-1)$ for $i = 1, \ldots, 10$

5.4 Summary

If the computational load is perfectly balanced among computers in terms of the average computing power available, τ significantly increases as one or both types of heterogeneity (*TH* and *SH*) increase, while the average sequential execution time on a single computer is not affected by *TH*. As *SH* increases, τ increases more when the average *TH* (among computers) increases than when it remains fixed. As the number of computers employed for a target task increases, τ decreases initially and then may increase especially when *SH* increases fast. More frequent synchronization amplifies effects of heterogeneity and degrades performance of a target task more for a higher *TH* and/or *SH*. Performance degradation due to heterogeneity becomes even larger when the interval is longer (coarser granularity). The risk factor and variation of parallel execution time is quickly reduced as the fraction of a target task assigned to computers with smaller *TH* increases.

The reduction in τ by the proposed approach is greater when (i) variation of the average computing power among computers is larger; (ii) variation of the standard deviation of computing power among computers is larger; or (iii) the number of computers employed is greater.

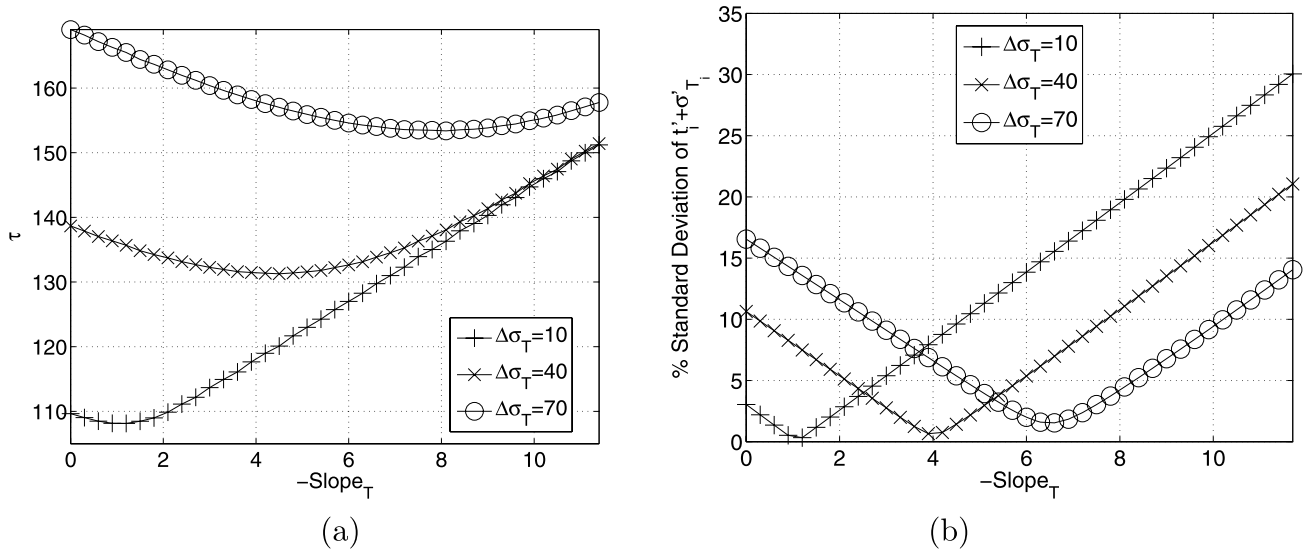


Fig. 19 (**a**) Average parallel execution time and (**b**) Percentage standard deviation of $t'_i + \sigma'_{T_i}$ after Step (2) in the load balancing, when the number of computers, N, is 10. After Step (1), $t_i = 100$ and $\Delta\sigma_T = \sigma_{T_N} - \sigma_{T_1}$ where $\sigma_{T_i} = \frac{i-1}{N-1}\sigma_{Tmax}$, where $i = 1, \ldots, 10$

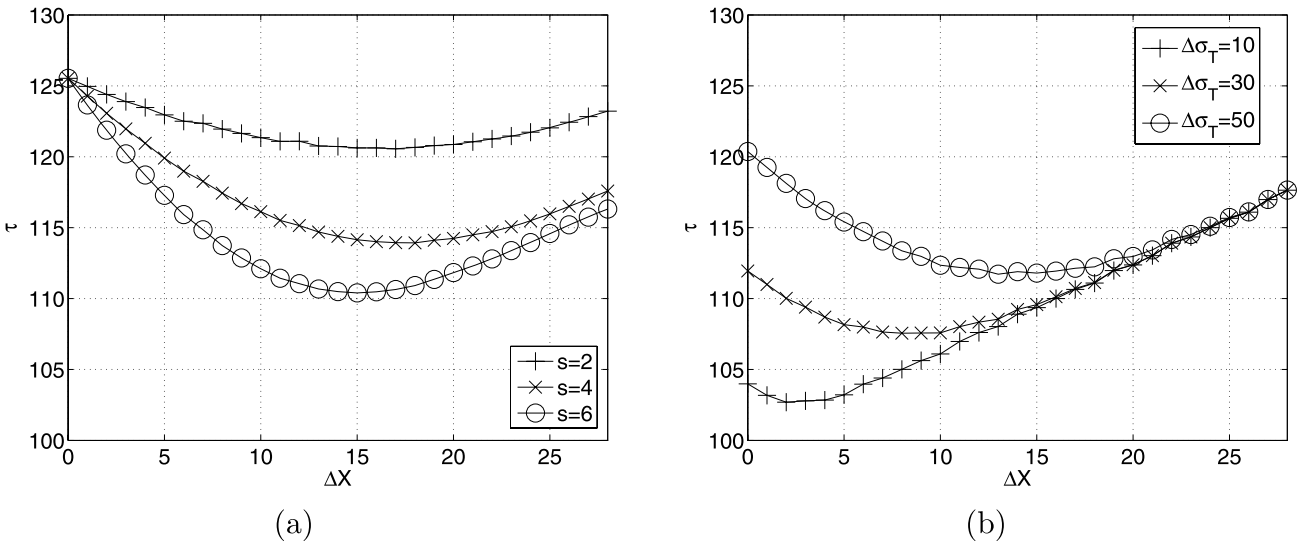


Fig. 20 Average parallel execution time on two computers (**a**) with a fixed $\Delta\sigma_T = \sigma_{T_1} - \sigma_{T_2} = 50$ ($\sigma_{T_2} = 10$) and (**b**) with a fixed $s = 4$, where $s = \frac{s_2}{s_1}$ ($s_1 = 1$) and $\Delta\sigma_T = \sigma_{T_1} - \sigma_{T_2}$ ($\sigma_{T_2} = 0$)

6 Conclusion

In this paper, two types of heterogeneity are defined for a heterogeneous computing environment. Temporal heterogeneity (*TH*) refers to variation in computing power or communication bandwidth available for a task along the time dimension on an individual computer or channel, while spatial heterogeneity (*SH*) refers to variation of the available computing power or communication bandwidth among computers. How they affect performance of a target task has been analyzed in detail through simulation as summarized in Sect. 5.4.

Also, it is shown that, for achieving the minimum average parallel execution time on a spatially heterogeneous computing system, it is not optimal to distribute a target task linearly proportional to the *average* availabilities (or *average* computing powers) of computers. A two-step heuristic approach to partitioning a target task for minimizing its average parallel execution time (τ) is described with a theoretical model. After a task is partitioned proportional to the

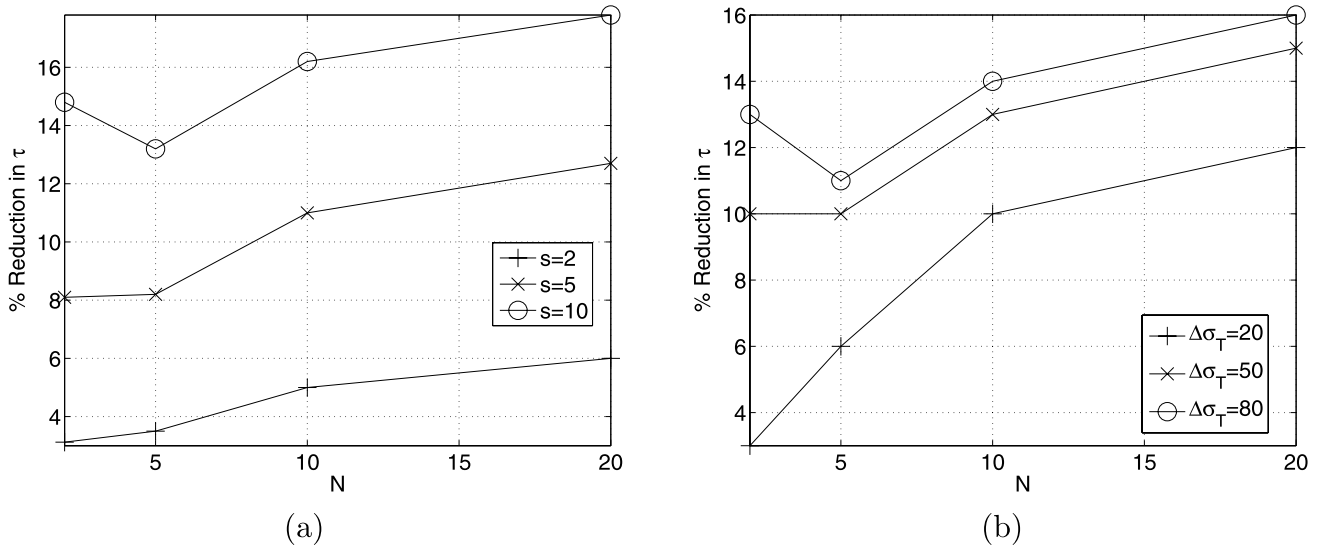


Fig. 21 Percentage reduction in average parallel execution time as a function of the number of computers (**a**) with a fixed $\Delta\sigma_T = \sigma_{T_1} - \sigma_{T_N} = 60$ ($\sigma_{T_N} = 0$) and (**b**) with a fixed $s = 5$, where $s = \frac{s_N}{s_1}$ ($s_1 = 1$) and $\Delta\sigma_T = \sigma_{T_1} - \sigma_{T_N}$ ($\sigma_{T_N} = 0$). After Step (1), $t_i = 100$ for all i

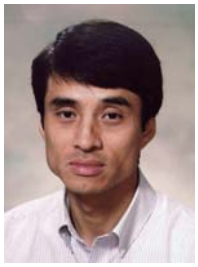
average computing power of each computer in the first step, the proposed approach achieves a further reduction in τ by re-allocating more load to computers with smaller *TH* or higher average computing power in the second step. This approach has been shown to achieve a significant additional performance improvement in terms of τ.

The future work includes developing efficient load balancing schemes for heterogeneous cluster or grid computing environments, based on the theoretical results reported in this paper.

References

1. Buyya, R.: High Peformance Cluster Computing (1999)
2. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Kaufmann (1988)
3. Freund, R.F., Siegel, H.J.: Heterogeneous processing. Computer, 13–17 (1993)
4. Maheswaran, M., Siegel, H.: A dynamic matching and scheduling algorithm for heterogeneous computing systems. In: Proceedings of Heterogeneous Computing, pp. 57–69 (1998)
5. Thanalapati, T., Dandamudi, S.: An efficient adaptive scheduling scheme for distributed memory multicomputers. IEEE Trans. Parallel Distributed Syst., 758–768 (2001)
6. Zhang, Y., Sivasubrmaniam, A., Moreira, J., Franke, H.: Impact of workload and system parameters on next generation cluster scheduling mechanisms. IEEE Trans. Parallel Distributed Syst., 967–985 (2001)
7. Ali, S., Siegel, H., Maheswaran, M., Hensgen, D., Ali, S.: Task execution time modeling for heterogeneous computing systems. In: Proceedings of the 9th Heterogeneous Computing Workshop, pp. 185–199, May 2000
8. Armstrong, R.: Investigation of effect of different run-time distributions on smartnet performance. Master's thesis, Department of Computer Science, Naval Postgraduate School (1997)
9. Al-Jaroodi, J., Mohamed, N., Jiang, H., Swanson, D.: Modeling parallel applications performance on heterogeneous systems. In: Proceedings of IPDPS 2003, Workshop on Advances in Parallel and Distributed Computational Models, Nice, France, April 2003
10. Figueira, S.M., Berman, F.: A slowdown model for application executing on time-shared clusters of workstations. IEEE Trans. Parallel Distributed Syst., 653–670 (2001)
11. Xu, C.-Z., Wang, L.Y., Fong, N.-T.: Stochastic prediction of execution time for dynamic bulk synchronous computations. In: Proceedings of International Parallel and Distributed Processing Symposium, San Francisco, April 2001
12. Zhang, X., Yan, Y.: Modeling and characterizing parallel computing performance on heterogeneous networks of workstations. In: Proceedings of Seventh IEEE Symposium. Parallel and Distributed Processing, pp. 25–34, October 1995
13. Dogon, A., Ozguner, F.: Trading off execution time for reliability in scheduling precedence-constrained tasks in heterogeneous computing. In: Proceedings of International Parallel and Distributed Processing Symposium, San Francisco, April 2001
14. Schopf, J., Berman, F.: Stochastic scheduling. In: CS Dept. Technical Report (CS-99-03), University of California, San Diego (1999)
15. Huang, J., Lee, S.-Y.: Effects of spatial and temporal heterogeneity on performance of a target task in heterogeneous computing environmentss. In: Proceedings of ISCA 15th International Conference on Parallel and Distributed Computing Systems, pp. 301–306, September 2002
16. Lee, S.-Y., Huang, J.: A theoretical approach to load balancing of a target task in temporally and spatially heterogeneous grid computing environment. In: The 3rd International Workshop on Grid Computing, pp. 70–81, November 2002
17. David, H.: Order Statistics. Wiley (1970)

Jun Huang received the B.S. degree in Automatic Control, and the M.S. degree in Control Theory and Control Engineering from University of Science and Technology of China, Hefei, in 1997 and 2000, respectively, and the Ph.D. degree in Electrical and Computer Engineering from Auburn University in 2005. His primary research interests are high performance parallel processing, heterogeneous computing and communication, and real-time embedded systems.



Soo-Young Lee received his B.S. degree in Electronics Engineering from Seoul National University in 1978, and M.S. degree in Electrical and Electronics Engineering from Korea Advanced Institute of Science in 1980. He received his Ph.D. degree in Electrical and Computer Engineering from the University of Texas at Austin in 1987. From 1980 to 1983, he was an instructor in the Department of Electronics Engineering, Kyung-Pook National University, Taegu, Korea. From 1987 to 1994, he was on the faculty of School of Electrical Engineering, Cornell University. In 1995, he joined the Department of Electrical Engineering, Auburn University, where he is a professor. He has worked on several topics in the area of parallel and distributed computing and communication, such as multiprocessor systems, parallel algorithms, task partitioning and mapping, scheduling, load balancing, multipath data transfer, source rate control, etc. More recent activities include heterogeneous computing and communication, sensor network applications, and large-scale scientific and engineering applications. Another active area of his research has been electron beam (e-beam) lithography. He has worked on e-beam proximity effect correction, developing the PYRAMID, a hierarchical and flexible approach to proximity correction, and nanofabrication using the e-beam lithographic process. He has published a significant number of journal and conference papers in the above areas. His researches have been supported by NSF, DARPA, SRC, NIH, Seagate, and NNFC. He is an associate editor of the ISCA International Journal of Computers and Their Applications, and a senior member of IEEE. He was the program co-chair for the 2007 ISCA International Conference on Parallel and Distributed Computing Systems.