Network computing performance evaluation in message passing environment

Guo Qingping* a, Yakup Paker b, Dennis Parkinson b & Xiao JinSheng a

a Department of Computer Science and Engineering, Wuhan University of Technology, 430063, Wuhan, People's Republic of China
b Department of Computer Science, Queen Mary, University of London, E1 4NS, London, UK

Version of record first published: 11 Oct 2011

To cite this article: Guo Qingping*, Yakup Paker, Dennis Parkinson & Xiao JinSheng (2003): Network computing performance evaluation in message passing environment, Parallel Algorithms and Applications, 18:4, 217-224

To link to this article: http://dx.doi.org/10.1080/10637190310001639767

Please scroll down for article

Full terms and conditions of use: http://www.tandfonline.com/page/terms-and-conditions

This article may be used for research, teaching, and private study purposes. Any substantial or systematic reproduction, redistribution, reselling, loan, sub-licensing, systematic supply, or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The accuracy of any instructions, formulae, and drug doses should be independently verified with primary sources. The publisher shall not be liable for any loss, actions, claims, proceedings, demand, or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.
NETWORK COMPUTING PERFORMANCE EVALUATION IN MESSAGE PASSING ENVIRONMENT

GUO QINGPING\textsuperscript{a,}\textsuperscript{*}, YAKUP PAKER\textsuperscript{b,}\textsuperscript{†}, DENNIS PARKINSON\textsuperscript{b} and XIAO JINSHENG\textsuperscript{a}

\textsuperscript{a}Department of Computer Science and Engineering, Wuhan University of Technology, Wuhan 430063, People’s Republic of China; \textsuperscript{b}Department of Computer Science, Queen Mary, University of London, London E1 4NS, UK

(Received 11 March 1999; In final form 11 October 2001)

This paper is based on joint research between Queen Mary, University of London, and Wuhan University of Technology, People’s Republic of China, on the use of network computing for solving non-linear heat flow problems. A general formulation for system performance evaluation has been derived from measured results, which covers the network computing as well as multiprocessor computing. The Amdahl’s law and Gustafson’s modification for speedup have been uniformly explained from this formula. Some criterions of speedup, efficiency and granularity for network computing also have been suggested.

Keywords: Network computing; Message passing environment; PVM; Performance evaluation

INTRODUCTION

The increased performance of high end workstations, not to mention PCs, over the last decade as well as the progress in local area network technologies have made it possible to use a collection of such machines as a “virtual” parallel computer to run numerically intensive applications. Based on this approach we have seen specialized machines built such as cluster computers [1]. People have also tried to capitalize on existing computing infrastructure used normally for other reasons to run parallel applications at idle periods such as during night time. From rather modest beginnings the so called network computing has taken an added interest recently with the computational grid where, over high bandwidth links, a large number of heterogeneous computers are to work as a transparent resource [2]. PVM was one of the earliest systems to introduce the metacomputing concept in terms of providing a flexible programming environment for a range of heterogeneous machines connected to...
a network [3]. MPI followed this to provide an international standard acceptable to the computer industry [4]. Thus, the PVM/MPI consisted of an integrated set of software tools and libraries for message passing, process/processor management and so on. This permitted a heterogeneous collection of computers networked together to be viewed by a user’s program as a single parallel computer. From the need to manage distributed applications in an open manner came middleware architectures such as CORBA [5]. Finally, we can mention HARNESS as a Java-centric metacomputing framework for dynamic management of networked virtual machines [6].

There are many factors that affect the performance of applications running on network computers [7]. In the message passing paradigm the two aspects, computation and communication, and their inter-relationships affect performance [8]. This paper analyses this for networked computing and illustrates the results based on some experiments performed using PVM on a networked PCs. Corresponding concepts such as speedup, efficiency, granularity and their behaviors are also addressed. Essential difference between network computing and multiprocessor computing has been explored, which explains different forms of the Amdahl’s law and Gustafson’s modification of speedup.

NETWORK COMPUTING PERFORMANCE MEASUREMENT

Network Structure and PVM Version

The network we used is a local area network with 30 PCs connected by an Ethernet, part of a large teaching laboratory at Computer Science Department of Queen Mary, University of London. At the time of experiments, each PC had a Pentium 166 processor and 64 Mbytes EDO RAM. The PVM version used was PVM3.3. In order to reduce interference from other users, we used remote login to one of the 30 PCs in the middle of the night, then taking that PC as a host (master) which spawned a number of slave processes running on corresponding slave PCs on the same LAN. Obviously, the best way of measurement would have been to isolate the LAN from the outside network, severing any random traffic impacts from other networks. However, this was not practical.

Algorithm Characteristics and Measurement Method

In the performance measurement, we chose out PVM version of Modified Tridiagonal Matrix algorithm, which used an implicit method to solve cyclical temperature in ceramic/metal composites [9]. This algorithm adopts a master–slave paradigm, using data decomposition methodology. The algorithm needs to divide transient time as several time steps and space distance in a cylinder as several space segments. Because of the implicit method’s nature we can choose big time interval and fix the number of time steps. The experiment is carried out for a number of cases by changing the number of space segments from the largest one to the smallest one. In fact, the number of chosen space segments for the measurements range from 90,720, the largest one, to 5670, the smallest one. For each experiment the number of space segments is halved from the previous one. For each fixed number of space segments, a number program runs have been carried out on the LAN from one slave PC to at most 24 PCs. The execution times (s) are measured and plotted in Fig. 1.
NETWORK COMPUTING BEHAVIOR ANALYSIS

General Formula of Execution Time

Let us consider a typical Ethernet LAN with homogeneous PCs for simplicity. The execution time $T$ of an application can be represented as [10]:

$$T = \frac{A}{N} + C_2 N + C_3 + \frac{\varepsilon A}{N}$$  \hspace{1cm} (1)

where $A$ is the total amount of computation, $N$ is the number of PCs involved, $C_2 N$ is the communication time (non-overlapping), $C_3$ is a constant initial latency and $\frac{\varepsilon A}{N}$ represents a variable latency related to the concurrent amount. The formula (1) can be rewritten as

$$T = C_1/N + C_2 N + C_3$$  \hspace{1cm} (2)

Difference between Network Computing and Multiprocessor Computing

Essential difference between network computing and multiprocessor computing is the communication mechanism. Most advanced multiprocessor systems have employed a communication engine for message passing, therefore, concurrent communication. In this situation, for computing dominant applications, the formula (2) becomes

$$T = C_1/N + C_3$$  \hspace{1cm} (3)

Theoretical Results

In order to get the theoretical results of network computing we need to determine the coefficients $C_1$, $C_2$ and $C_3$ of Eq. (2) from measured results. There are two considerations: firstly, because of random background traffic, measured results vary. In order to reduce this effect after several measurements, we choose the one with the smallest execution time. Secondly, there are several methods, for example the least square method, the coefficient-determination method, etc. to calculate the coefficients. We chose the latter for this purpose.
Formulas (1) and (2) show that $C_1$ describes the influence of total computation amount and some communication latency related to the total computation amount, so it varies from case to case. However, the $C_2$ and $C_3$ represent communication and system overheads due to bus transfer and initial latency, which are application independent. Therefore, if we determine $C_2$ and $C_3$ from one case we can use them for other cases. In fact, we have chosen the number of space segments for our measurement each time dividing by half from 90,720, the largest one, to 5670, the smallest one. Since the computational load is proportional to the number of space segments, from formula (1) we can write for $i$th measurement

$$C_{1i} = 2(1 + \varepsilon)C_{1i+1}$$

$$C_{1i} = (1/2(1 + \varepsilon))C_{1i-1}$$

In practice, we first determine $C_1$, $C_2$ and $C_3$ of formula (2) from the measured results of case 4, that is the forth curve in Fig. 1. We choose $N$ (number of PCs) as 1, 12 and 24, respectively, and measure the corresponding execution times $T$. Replacing $T$ and $N$ by those values in formula (2) we get three equations for the three unknowns. Therefore, the $C_1$, $C_2$ and $C_3$ can be determined as follows:

$$C_1 = 122, \quad C_2 = 4.5, \quad C_3 = 4$$

Then, using neighbor cases the $C$ can be determined, and $\varepsilon = 0.1203512$ for the LAN used. Using relations (4) and (5) we can deduce the coefficient $C_1$ for all cases:

$$C_1 = 54.45 \text{ case 5}; \quad C_1 = 273.37 \text{ case 3}; \quad C_1 = 612.53 \text{ case 2}; \quad C_1 = 1372.5 \text{ case 1}.$$

A set of theoretical execution time curves can be plotted as shown in Fig. 2. Figure 3 plots both the measured results and the theoretical results in one figure. Those results explicitly show that the formula (2) approaches PVM environment behavior in network computing.
SPEEDUP, EFFICIENCY AND GRANULARITY

Turning Point of Execution Time or Speedup Curve of Network Computing

As can be seen from Figs. 1 and 2, the execution time of each case has a turning point. Assuming that $N$ is continuous in Eq. (2) we can easily determine the turning point of execution time, as well as the speedup. In fact, they are the same as given in Eq. (6) where $C_1$ mainly represents the computation amount and $C_2$ represents the sequential communication.

$$N_{\text{max}} = (C_1/C_2)^{1/2}$$  \hspace{1cm} (6)

Amdahl’s Law and Gustafson’s Modification of Speedup

From Eq. (2), the speedup of network computing $S_{\text{net}}$ can be written as

$$S_{\text{net}} = \frac{(C_1 + C_2 + C_3)}{(C_1/N + C_2N + C_3)}$$

It obeys the Amdahl’ law, i.e. as the number of PCs is increased $S_{\text{net}}$ has a turning point described by the formula (6). However, as pointed out in the “Difference between network computing and multiprocessor computing” section, for advanced multiprocessor systems $S_{\text{multi}}$ becomes

$$S_{\text{multi}} = \frac{T_1}{T_N} = \frac{(C_1 + C_3)}{(C_1/N + C_3)}$$

where $C_1/N$ represents the parallel portion of parallel algorithm, and $C_3$ represents the sequential portion of the algorithm. We assume the total execution time to be 1 for algebraic simplicity, that is

$$C_1/N + C_3 = 1$$  \hspace{1cm} (7)
Then the speedup $S_{\text{multi}}$ can be represented as

$$S_{\text{multi}} = C_1 + C_3 = N(1 - C_3) + C_3 \quad (8)$$

This speedup is proportional to the number of processors, which has no extreme point. It is exactly the same result of the Gustafson’s modification of Amdahl’s law for MPP machine’s speedup [11]. From this point of view we can say the Gustafson’s modification of Amdahl’s law for speedup is only suitable for non-sequential communication situation. In that case, as problem scale is increased the speedup is linear to the number of processors. In network computing environment, however, precondition of the Gustafson’s modification does not exist, and the speedup has a turning point, as shown in Fig. 4, which can be determined by Eq. (6).

**Optional Efficiency—a Magic Number of 0.50**

A definition of processor efficiency is

$$E = S/N \quad (9)$$

where $S$ represents speedup. According to formula (2) the $E$ can be written as

$$E = \frac{(C_1 + C_2 + C_3)}{(C_1 + C_2N^2 + C_3N)} \quad (10)$$

Replacing the $N$ as a value of turning point, that is the Eq. (6) then Eq. (10) becomes

$$E_{\text{turning}} = \frac{(C_1 + C_2 + C_3)}{(2C_1 + (C_1/C_2)^{1/2})} \quad (11)$$

As a common knowledge we have $C_1 \gg C_2, C_3$ and because $C_1^{1/2} \gg C_3/C_2^{1/2}$ from Eq. (11) we obtain

$$E_{\text{turning}} \sim 0.50$$
There comes an important argument: if efficiency is reduced to 0.50, involving more computers in network computing makes no sense, because more computers imply the performance will be reduced more, hence no further gain.

**Maximum Speedup**

Maximum speedup is achieved at the turning point. Therefore,

\[
S_{\text{max}} = \frac{(C_1 + C_2 + C_3)}{(2(C_1C_2)^{1/2} + C_3)}
\]

(12)

Consider \(C_1 \gg C_2\) and \(C_3\), \(S_{\text{max}}\) then becomes \(S_{\text{max}} \sim N_{\text{max}}/2\).

**Optional Granularity**

It is clear that for PVM environments based on networks, large granularity generally leads to better performance. The lowest limit of granularity is

\[
G_{\text{min}} \sim C_1/N_{\text{max}} = (C_1C_2)^{1/2}
\]

(13)

Therefore, poor quality network (large \(C_2\)) needs larger granularity.

From formulas of \(N_{\text{max}}\) and \(G_{\text{min}}\), that is Eqs. (1 and 3), the following relations can be easily derived:

\[
C_1 = G_{\text{min}}N_{\text{max}}
\]

(14)

\[
C_2 = G_{\text{min}}/N_{\text{max}}
\]

(15)

Therefore, the network communication characteristics, i.e. the \(C_2\), can be determined from application experience. If computation amount of other applications, i.e. \(C_1\), can be predicted, then \(G_{\text{min}}\) and \(N_{\text{max}}\) as well as behaviours of those applications can be predetermined.

**CONCLUSIONS**

From the above analysis we have shown the following arguments

1. The nature of network based PVM environment provides parallel computation and sequential communication, which is different than modern multiprocessor systems.
2. Speedup of network computing obeys Amadahl’s law, but the MPP’s can be described with Gustafson’s modification due to different communication features.
3. In network computing PVM environment, the maximum speedup \(S_{\text{max}}\) is equal to the half of the optimum number of computers, \(N_{\text{max}}\). The maximum number of computers involved \((N_{\text{max}})\), and minimum amount of granularity \((G_{\text{min}})\) for an application is simply determined by formulas (6) and (13).
4. The application parameter \(C_1\) and network communication parameter \(C_2\) have very simple relations to \(N_{\text{max}}\) and \((G_{\text{min}})\) given by Eqs (14) and (15).
Acknowledgements

Research supported by the UK Royal Society joint project (the Royal Society Q724) and the Natural Science Foundation of China (NSFC Grant No. 69773021).

References