NAL TR-1362T

NAL TR-1362T

NAL Technical Report TR-1362T

ISSN 0389-4010
UDC 681.32

TECHNICAL REPORT OF NATIONAL AEROSPACE LABORATORY

TR-1362T

The characteristic parameters of the NWT computer system in the global memory access

Shigeki HATAYAMA

OCTOBER 1998

NATIONAL AEROSPACE LABORATORY

CHŌFU, TOKYO, JAPAN

Characteristic parameters of the NWT computer system in global memory access *

Shigeki HATAYAMA *1

ABSTRACT

The NWT computer system available at the NAL since February 1993 comprises two system administrators, n processing elements (where n was 140 at the beginning, and is 166 at present) and a crossbar network, and operates as a distributed-memory message-passing MIMD computer. Each processing element itself is a vector computer. This paper reports measurements of two pairs of the characteristic parameters, $(r_\infty, n_{1/2})$ and $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$, of the NWT with the communication performance between the global and local memory spaces through the medium of the crossbar network and with MIMD computing in the global memory access, respectively. The significance of the results is interpreted, and several hardware parameters are estimated. The results in this paper apply only to the NWT system software during the period April to June 1993.

Keywords : communication performance, message bandwidth, latency, parallel processing, global memory access, MIMD computing, synchronization overhead, communication overhead, maximum performance, half-performance grain size, half-performance intensity, NWT

概　　要

1993年2月に運用を開始した航空宇宙技術研究所の数値風洞（NWT）計算機システムは2台の制御計算機n台の要素計算機（nは運用開始時において140台、現在は166台）、並びにクロスバーネットワークから構成されていて、分散メモリ型メッセージパッシング方式のMIMD計算機として動作する。各要素計算機自体はベクトル計算機である。ただし、利用者の並列プログラミングを容易にすることを目的として、プログラミングのための階層メモリ型並列計算機システムを仮定した論理モデルを備えているので、利用者は仮想的グローバルメモリを利用することが出来る。従って、“グローバル”と宣言された変数は、並列処理のために洗濯された要素計算機の間において、論理的共有メモリ上に存在するものとしてプログラミングすることが出来る。

本稿ではこのグローバル・メモリ・アクセスの場合における、NWTの運用開始初期の1993年4月から6月の間に測定したNWTの基本的処理性能を表わす2つの特性パラメーターの測定結果について報告する。その1つはクロスバーネットワークを媒体として、グローバルとローカル・メモリ空間の間での情報伝達処理性能を表わす特性パラメータ（$r_\infty, n_{1/2}$）であり、他はグローバル・メモリ・アクセスにおけるNWTの並列処理性能を表わす特性パラメータ（$\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$）である。ここで、$r_\infty$はグローバルとローカル・メモリ空間の間での情報伝達のバンビ幅を、$n_{1/2}$は$r_\infty/2$を達成するに必要なメッセージ長を、$\hat{r}_\infty$はピーク最大処理性能を、$\hat{s}_{1/2}$は$r_{\infty/2}$を達成するに必要なグレイン・サイズを、$f_{1/2}$は$\hat{r}_\infty/2$を達成するに必要なインテンシティ（1度参照されたグローバル変数が何度計算に利用されるかを表わす度合）を表わす。更に、得ら

＊　平成10年4月8日受付（Received 8 April 1998）

＊1　Computational Sciences Division

れた結果の意味について考察するとともに、いくつかのハードウェア・パラメータを推定し、別稿で示したローカル・メモリ・アクセスにおけるNWTの並列処理性能の結果として比較する。なお、本稿で示した値はその後のNWTシステム・ソフトウェアの改善努力によって、大巾に改善されたが、その成果は稿をあらためて報告する。

1. Introduction

A parametric description of the communication performance between different nodes of the communication network has been used to characterize the dependence of the speed of communication on a variety of message length, using the parameters $(r_\infty, n_{1/2})$ [3,5,10,11,12,13]. Similarly the degradation of performance due to synchronization overheads and bottlenecks in memory access that is incurred when programming with MIMD computing has been described by the prameters, $(r_\infty, s_{1/2})$ and $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$, and measured on several parallel computers [1,2,4,6,7,8,9,10].

In this paper we apply the same techniques to the performance study of the Numerical Wind Tunnel (NWT) computer system available at the National Aerospace Laboratory since February 1993. This computer system comprises two system administrators, n processing elements (where n was 140 at the beginning, and is 166 at present) and a crossbar network, and operates as a distributed-memory message-passing MIMD computer. Each processing element itself is a vector computer. The language to describe parallel processing is the NWT Fortran. Main memories of the NWT are physically distributed across the processing elements, but to ease programming, the logical model of the NWT assumed a hierarchical memory parallel computer system for programming offers the virtual global space (or global memory) shared by the selected processing elememts to users. On the other hand, each local space (or local memory) is the memory specific to each processing element. Details about the machine architecture and logical model of the NWT, the NWT Fortran, communication and synchronization are shown in Appendix.

One of the purposes of this paper is to measure the parameters $(r_\infty, n_{1/2})$ of the communication properties between the global and local memory spaces through the medium of the crossbar network of the NWT. The definiton of a variation of the COMMS1 pingpong benchmark and measurements of the parameters are given in Section 2. The other purpose is to measure the parameters $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ of the NWT with MIMD computing in the global memory access. The definition of the $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark and measurements of the parameters are given in Section 3. Results obtained in Section 3 are analyzed in Section 4, and the overall performance of the NWT in the global memory access is estimated in Section 5. The significance of the obtained results is discussed in Section 6 and 7, several hardware parameters are estimated, and compared with the results obtained in [4]. The obtained results in this paper apply only to the NWT system software during the period April to June 1993. The results of an improvement on the NWT system software will be reported in another paper. By the way, since the benchmark program was run with other users on the system, we may be not to ensure the most consistent measurements for the study in case of the global memory access. Bad results were thrown away, and reruns were continued until a reasonable result was obtained. The terminology in this paper follows one in [9].

2. Avariation of the pingpong benchmark

The test program of a variation of the COMMS1 pingpong benchmark [5,12,13] is the following reference and assignment operations which are read and write operations in substance, respectively:

reference (read) : $AL(I) = A(I)$,
assignment (write) : $A(I) = AL(I)$,

where A(I) is in the global memory space and AL(I) is in the local memory space.

A message A(I) in the global memory is divided as evenly as possible between the selected number of processing elements, pe, and is sent from the global memory space to each local memory space of selected processing elements. Each processing element returns each partitioned message AL(I) to the global memory space, immediately that the data has been received into its Fortran array and is available for use by each Fortran program. This pingpong exchange is repeated typically 10,000 times to give a sufficiently large time interval, and timed by a wall clock on the master processing element. Half the time for a single exchange is recorded as the time to read a message A(I) or write a partitioned message AL(I) between the global and local memory spaces. In these experiments on the NWT computer, the Fortran code is implemented with repeated loops containing two pairs of the compiler directives, PARALLEL REGION/END PARALLEL and SPREAD MOVE/END SPREAD. The SPREAD MOVE and END SPREAD are also barrier synchronization points in a parallel program.

The experiment is repeated for a variety of message lengths, and the time, t, to read or write a message of length, n, between the global and local memory spaces is measured and fitted by least-squares to the straight line described as follows [11,12]:

$$t=a_0+a_1 n. \tag{1}$$

From this equation we can obtain the characteristic parameters of the communication properties between the global and local memory spaces through the crossbar network of the NWT as follows:

$t_0=a_0$ = message start-up time or latency(μsec),

$\pi_0=t_0^{-1}$ = specific performance or bandwidth (MHz)
= short-message communication performance,

$r_\infty=(1.024^2 * a_1)^{-1}$ = maximum or asymptotic performance or bandwidth (Mbyte/sec)
= long-message communication performance,

$n_{1/2}=a_0/a_1$ = half-performance message length(byte)
= ratio of the best communication bandwidth to the worst bandwidth,

where the message start-up time includes the synchronization overhead and the time that is required to make packets for every read or write demands. The former, furthermore, includes the time that is required to specify the starting and ending values of the indices of each array in the local memory to be used by selected processing elements.

The actual or average data transfer rate or bandwidth r, for a message of length n, can be computed from

$$r=r_\infty pipe(n/n_{1/2})=\frac{\pi_0 n}{1.024^2} pipe(n_{1/2}/n), \tag{2}$$

where

$$pipe(x)=1/(1+x^{-1})=pipeline\ function. \tag{3}$$

Table 1 shows measurements of the characteristic parameters of the communication performance for the data transfer of the NWT when $pe=1\sim16$. Figure 1 shows (a) the message-transfer timing relations as a function of n on the NWT for $pe=1\sim16$, (b) the actual transfer rates as a function of $n/n_{1/2}$, (c) the maximum transfer rate against pe, (d) the half-performance message length against pe, and (e) the start-up time of data transfer against pe. The solid lines in Figure 1 (c) and (d) show the linear fits

$$r_\infty=7.554390*10^2 pe\ (Mbyte/sec), \tag{4}$$

$$n_{1/2}=-3.315542*10^4+1.834119*10^5 pe\ (byte), \tag{5}$$

respectively, and the solid line in Figure 1 (e) shows the following approximate fit function obtained from (1), (4) and (5)

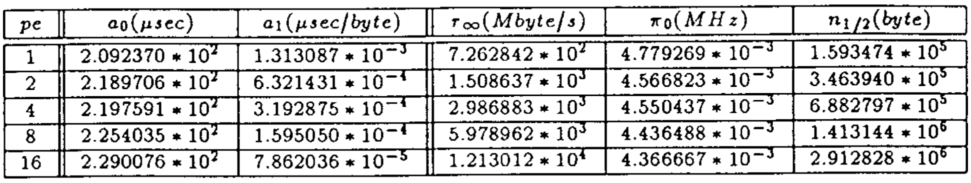
$$n_{1/2}=-3.315542*10^4+1.834119*10^5 pe\ (byte), \tag{6}$$

where this is the time included synchronization overheads, and the data transfer overhead only in Figure 1 (e) is the time subtracted the values of synchronization overheads for pe = 1 ~ 16 that were obtained in [4].

The above parameters give rise to the complete formula for the timing relation

$$t=2.315412*10^2-\frac{4.185576*10^1}{pe}+\frac{1.262411*10^{-3}}{pe}n\ (\mu sec). \tag{7}$$

Table 1 The characteristic parameters for the data transfer of the NWT.



pe	$a_0(\mu sec)$	$a_1(\mu sec/byte)$	$r_\infty(Mbyte/s)$	$\pi_0(MHz)$	$n_{1/2}(byte)$
1	$2.092370*10^2$	$1.313087*10^{-3}$	$7.262842*10^2$	$4.779269*10^{-3}$	$1.593474*10^5$
2	$2.189706*10^2$	$6.321431*10^{-4}$	$1.508637*10^3$	$4.566823*10^{-3}$	$3.463940*10^5$
4	$2.197591*10^2$	$3.192875*10^{-4}$	$2.986883*10^3$	$4.550437*10^{-3}$	$6.882797*10^5$
8	$2.254035*10^2$	$1.595050*10^{-4}$	$5.978962*10^3$	$4.436488*10^{-3}$	$1.413144*10^6$
16	$2.290076*10^2$	$7.862036*10^{-5}$	$1.213012*10^4$	$4.366667*10^{-3}$	$2.912828*10^6$

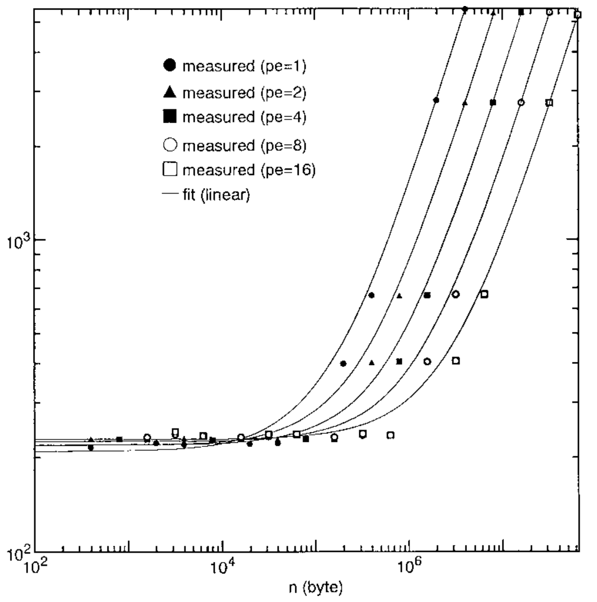


Fig. 1 The pingpong benchmark test ($pe = 1 \sim 16$).
(a) The timing relations as a function of message length.

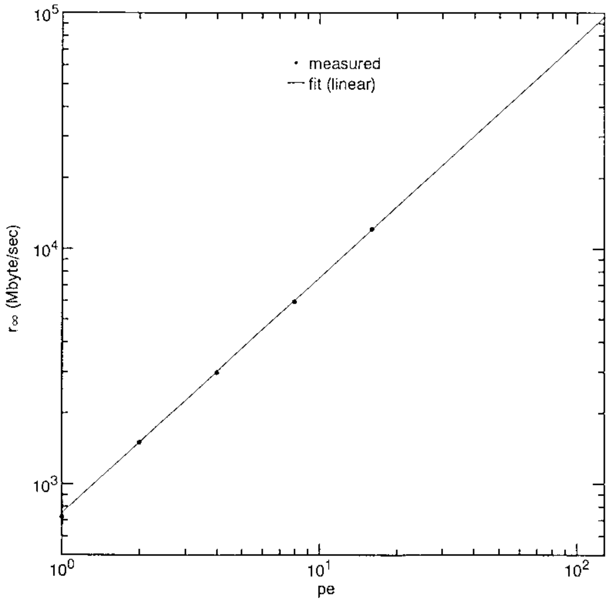


Fig. 1 The pingpong benchmark test ($pe = 1 \sim 16$).
(c) The maximum data transfer rate against the number of processing elements.

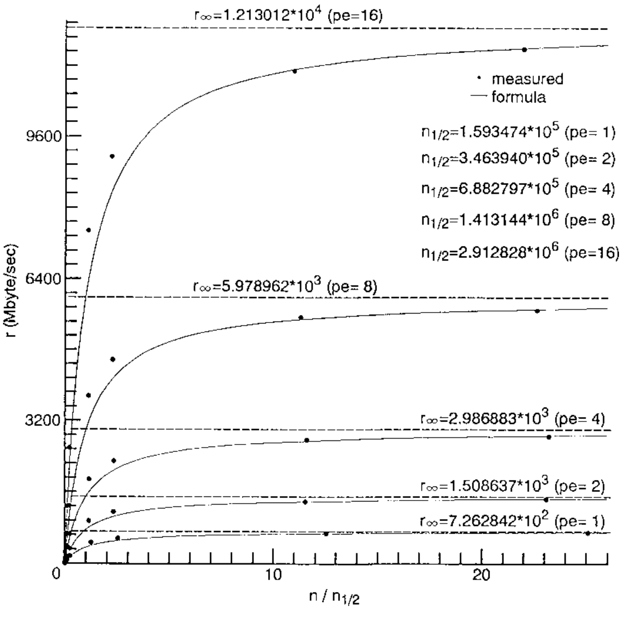


Fig. 1 The pingpong benchmark test ($pe = 1 \sim 16$).
(b) The actual data transfer rates as a function of message length.

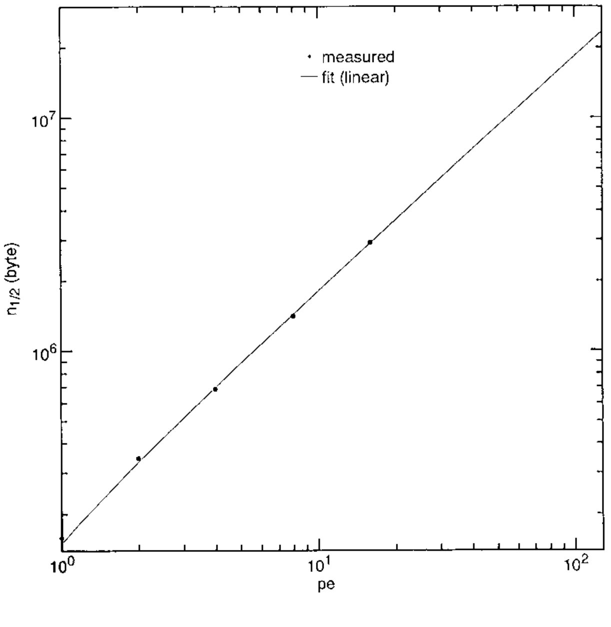


Fig. 1 The pingpong benchmark test ($pe = 1 \sim 16$).
(d) The half-performance message length against the number of processing elements.

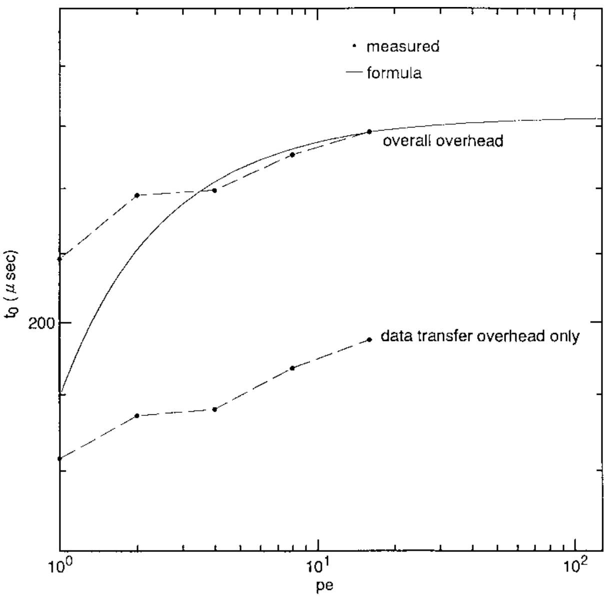


Fig. 1 The pingpong benchmark test ($pe = 1 \sim 16$).
(e) The start-up time of data transfer against the number of processing elements.

3. The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark

The test problem which we call the $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark is dyadic and triadic operations as follows:

dyads : $A(I) = B(I) * C(I)$,
triads : $A(I) = B(I) * C(I) + D(I)$.

Two data (B(I) and C(I)) or three data (B(I), C(I) and D (I)) in the global memory are divided as evenly as possible between the selected number of processing elements pe, and are sent from the global memory space to each local memory space of selected processing elements. Each processing element performs each work of a dyadic or triadic operation immediately that the data has been received into its Fortran array and is available for use by each Fortran program, and returns each result to a Fortran array A(I) in the global memory as soon as possible after completion of execution. In these experiments on the NWT computer system, the Fortran code is implemented with insertion of three pairs of the compiler directives, PARALLEL REGION/END PARALLEL, SPREAD MOVE/END SPREAD and SPREAD DO/END SPREAD. The SPREAD MOVE, SPREAD DO and END SPREAD are also barrier synchronization points in a parallel program.

First of all we introduce the following parameter f, which plays an important part in the $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark:

$$f = s/m, \tag{8}$$

where

f = computational intensity (flop/I/O word)
= number of floating-point operations in a processing element per word transferred to it,
s = amount of computational work in the parallel section (flop),
m = number of I/O data words in the parallel section (I/O word).

In the above-mentioned dyads there are three data transfers for every multiplying operation, hence $f = 1/3$. Similarly $f = 1/2$ in case of the triads. In the $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark, f is varied by repeating the arithmetic assigned to each processing element $3f$ times (dyads) or $2f$ times (triads).

We repeat the experiment for a variety of the computational intensity f, and measure the elapsed time t, for a single parallel section as a function of the amount of computational work s in the parallel section measured in floating-point operations and fit the results by least-squares to the expression [9,11]

$$t = a_0 + a_1 s. \tag{9}$$

From this equation we can obtain the characteristic parameters with MIMD computation in the global memory access for parameter f as follows:

$t_0 = a_0$ = set-up plus pipeline start-up plus synchronization plus communication start-up time (μsec)
= time for null job when $s = m = 0$, which is dependent on both the hardware and software (i.e. operating system and compiler),
$\pi_0 = t_0^{-1}$ = specific performance (Mflop/sec),
$r_\infty = a_1$ = maximum or asymptotic performance degraded by communication delays (Mflop/sec),
$s_{1/2} = a_0/a_1$ = half-performance grain size (flop)
= amount of maximum possible arithmetic operations lost during synchronization

and communication start-up, where the communication start-up time is the time that is required to make packets for the message-passing data transfer and other overheads.

The actual or average performance r, is given by the pipeline function as follows:

$$r = r_{\infty} pipe(s / s_{1/2}). \tag{10}$$

Table 2 shows measurements of the characteristic parameters with MIMD computing in the global memory access when $pe = 1 \sim 16$ and $f = 1/3$ or $1/2 \sim 11$. Figure 2 shows the timing relations as a function of the amount of arithmetic operations for parameter f when $pe = 16$.

Table 2 The characteristic parameters of the NWT with parameter f.

(a)$pe = 1$

f	*operation*	$a_0(\mu sec)$	$a_1(\mu sec/flop)$	$r_{\infty}(Mflop/s)$	$\pi_0(Mflop/s)$	$s_{1/2}(flop)$
1/3	*dyadic op.*	$5.358462 * 10^2$	$1.708670 * 10^{-2}$	$5.852505 * 10^1$	$1.866207 * 10^{-3}$	$3.136043 * 10^4$
1/2	*triadic op.*	$5.858259 * 10^2$	$1.205861 * 10^{-2}$	$8.292830 * 10^1$	$1.706992 * 10^{-3}$	$4.858154 * 10^4$
1	*dyadic op.*	$5.358462 * 10^2$	$7.505013 * 10^{-3}$	$1.332443 * 10^2$	$1.866207 * 10^{-3}$	$7.139844 * 10^4$
	triadic op.	$5.858259 * 10^2$	$6.891143 * 10^{-3}$	$1.451138 * 10^2$	$1.706992 * 10^{-3}$	$8.501143 * 10^4$
3	*dyadic op.*	$5.358462 * 10^2$	$4.196946 * 10^{-3}$	$2.382685 * 10^2$	$1.866207 * 10^{-3}$	$1.276753 * 10^5$
	triadic op.	$5.858259 * 10^2$	$3.430382 * 10^{-3}$	$2.915127 * 10^2$	$1.706992 * 10^{-3}$	$1.707757 * 10^5$
5	*dyadic op.*	$5.358462 * 10^2$	$3.558121 * 10^{-3}$	$2.810472 * 10^2$	$1.866207 * 10^{-3}$	$1.505981 * 10^5$
	triadic op.	$5.858259 * 10^2$	$2.791923 * 10^{-3}$	$3.581761 * 10^2$	$1.706992 * 10^{-3}$	$2.098288 * 10^5$
7	*dyadic op.*	$5.358462 * 10^2$	$3.280493 * 10^{-3}$	$3.048322 * 10^2$	$1.866207 * 10^{-3}$	$1.633432 * 10^5$
	triadic op.	$5.858259 * 10^2$	$2.509075 * 10^{-3}$	$3.985533 * 10^2$	$1.706992 * 10^{-3}$	$2.334828 * 10^5$
9	*dyadic op.*	$5.358462 * 10^2$	$3.133503 * 10^{-3}$	$3.191317 * 10^2$	$1.866207 * 10^{-3}$	$1.710055 * 10^5$
	triadic op.	$5.858259 * 10^2$	$2.355679 * 10^{-3}$	$4, 245061 * 10^2$	$1.706992 * 10^{-3}$	$2.486866 * 10^5$
11	*dyadic op.*	$5.358462 * 10^2$	$3.071519 * 10^{-3}$	$3.255718 * 10^2$	$1.866207 * 10^{-3}$	$1.744564 * 10^5$
	triadic op.	$5.858259 * 10^2$	$2.256116 * 10^{-3}$	$4.432396 * 10^2$	$1.706992 * 10^{-3}$	$2.596612 * 10^5$

(b)$pe = 2$

f	*operation*	$a_0(\mu sec)$	$a_1(\mu sec/flop)$	$r_{\infty}(Mflop/s)$	$\pi_0(Mflop/s)$	$s_{1/2}(flop)$
1/3	*dyadic op.*	$5.463351 * 10^2$	$8.945350 * 10^{-3}$	$1.117899 * 10^2$	$1.830378 * 10^{-3}$	$6.107476 * 10^4$
1/2	*triadic op.*	$6.027210 * 10^2$	$5.940494 * 10^{-3}$	$1.683361 * 10^2$	$1.659142 * 10^{-3}$	$1.014597 * 10^5$
1	*dyadic op.*	$5.463351 * 10^2$	$3.859042 * 10^{-3}$	$2.591317 * 10^2$	$1.830378 * 10^{-3}$	$1.415727 * 10^5$
	triadic op.	$6.027210 * 10^2$	$3.638937 * 10^{-3}$	$2.748055 * 10^2$	$1.659142 * 10^{-3}$	$1.656311 * 10^5$
3	*dyadic op.*	$5.463351 * 10^2$	$2.163291 * 10^{-3}$	$4.622587 * 10^2$	$1.830378 * 10^{-3}$	$2.525481 * 10^5$
	triadic op.	$6.027210 * 10^2$	$1.750714 * 10^{-3}$	$5.711955 * 10^2$	$1.659142 * 10^{-3}$	$3.442715 * 10^5$
5	*dyadic op.*	$5.463351 * 10^2$	$1.811562 * 10^{-3}$	$5.520098 * 10^2$	$1.830378 * 10^{-3}$	$3.015823 * 10^5$
	triadic op.	$6.027210 * 10^2$	$1.421533 * 10^{-3}$	$7.034659 * 10^2$	$1.659142 * 10^{-3}$	$4.239937 * 10^5$
7	*dyadic op.*	$5.463351 * 10^2$	$1.679116 * 10^{-3}$	$5.955515 * 10^2$	$1.830378 * 10^{-3}$	$3.253707 * 10^5$
	triadic op.	$6.027210 * 10^2$	$1.277734 * 10^{-3}$	$7.826355 * 10^2$	$1.659142 * 10^{-3}$	$4.717109 * 10^5$
9	*dyadic op.*	$5.463351 * 10^2$	$1.603903 * 10^{-3}$	$6.234791 * 10^2$	$1.830378 * 10^{-3}$	$3.406285 * 10^5$
	triadic op.	$6.027210 * 10^2$	$1.199727 * 10^{-3}$	$8.335230 * 10^2$	$1.659142 * 10^{-3}$	$5.023818 * 10^5$
11	*dyadic op.*	$5.463351 * 10^2$	$1.540639 * 10^{-3}$	$6.490813 * 10^2$	$1.830378 * 10^{-3}$	$3.546159 * 10^5$
	triadic op.	$6.027210 * 10^2$	$1.153695 * 10^{-3}$	$8.667802 * 10^2$	$1.659142 * 10^{-3}$	$5.224266 * 10^5$

(c)$pe = 4$

f	*operation*	$a_0(\mu sec)$	$a_1(\mu sec/flop)$	$r_{\infty}(Mflop/s)$	$\pi_0(Mflop/s)$	$s_{1/2}(flop)$
1/3	*dyadic op.*	$5.498325 * 10^2$	$4.398520 * 10^{-3}$	$2.273492 * 10^2$	$1.818736 * 10^{-3}$	$1.250040 * 10^5$
1/2	*triadic op.*	$6.024799 * 10^2$	$2.953622 * 10^{-3}$	$3.385674 * 10^2$	$1.659806 * 10^{-3}$	$2.039800 * 10^5$
1	*dyadic op.*	$5.498325 * 10^2$	$1.864600 * 10^{-3}$	$5.363081 * 10^2$	$1.818736 * 10^{-3}$	$2.948796 * 10^5$
	triadic op.	$6.024799 * 10^2$	$1.701371 * 10^{-3}$	$5.877613 * 10^2$	$1.659806 * 10^{-3}$	$3.541144 * 10^5$
3	*dyadic op.*	$5.498325 * 10^2$	$1.081254 * 10^{-3}$	$9.248521 * 10^2$	$1.818736 * 10^{-3}$	$5.085137 * 10^5$
	triadic op.	$6.024799 * 10^2$	$8.904061 * 10^{-4}$	$1.123083 * 10^3$	$1.659806 * 10^{-3}$	$6.766350 * 10^5$
5	*dyadic op.*	$5.498325 * 10^2$	$9.111465 * 10^{-4}$	$1.097518 * 10^3$	$1.818736 * 10^{-3}$	$6.034513 * 10^5$
	triadic op.	$6.024799 * 10^2$	$7.206707 * 10^{-4}$	$1.387596 * 10^3$	$1.659806 * 10^{-3}$	$8.359989 * 10^5$
7	*dyadic op.*	$5.498325 * 10^2$	$8.384131 * 10^{-4}$	$1.192729 * 10^3$	$1.818736 * 10^{-3}$	$6.558014 * 10^5$
	triadic op.	$6.024799 * 10^2$	$6.453565 * 10^{-4}$	$1.549531 * 10^3$	$1.659806 * 10^{-3}$	$9.335614 * 10^5$
9	*dyadic op.*	$5.498325 * 10^2$	$7.954631 * 10^{-4}$	$1.257129 * 10^3$	$1.818736 * 10^{-3}$	$6.912106 * 10^5$
	triadic op.	$6.024799 * 10^2$	$6.049308 * 10^{-4}$	$1.653082 * 10^3$	$1.659806 * 10^{-3}$	$9.959485 * 10^5$
11	*dyadic op.*	$5.498325 * 10^2$	$7.715897 * 10^{-4}$	$1.296026 * 10^3$	$1.818736 * 10^{-3}$	$7.125970 * 10^5$
	triadic op.	$6.024799 * 10^2$	$5.788175 * 10^{-4}$	$1.727660 * 10^3$	$1.659806 * 10^{-3}$	$1.040881 * 10^6$

(d)$pe = 8$

f	*operation*	$a_0(\mu sec)$	$a_1(\mu sec/flop)$	$r_\infty(Mflop/s)$	$\pi_0(Mflop/s)$	$s_{1/2}(flop)$
1/3	*dyadic op.*	$5.573111 * 10^2$	$2.304755 * 10^{-3}$	$4.338856 * 10^2$	$1.794330 * 10^{-3}$	$2.418093 * 10^5$
1/2	*triadic op.*	$6.155499 * 10^2$	$1.498470 * 10^{-3}$	$6.673474 * 10^2$	$1.624564 * 10^{-3}$	$4.107856 * 10^5$
1	*dyadic op.*	$5.573111 * 10^2$	$1.015463 * 10^{-3}$	$9.847725 * 10^2$	$1.794330 * 10^{-3}$	$5.488246 * 10^5$
	triadic op.	$6.155499 * 10^2$	$8.445490 * 10^{-4}$	$1.184064 * 10^3$	$1.624564 * 10^{-3}$	$7.288504 * 10^5$
3	*dyadic op.*	$5.573111 * 10^2$	$5.681015 * 10^{-4}$	$1.760249 * 10^3$	$1.794330 * 10^{-3}$	$9.810062 * 10^5$
	triadic op.	$6.155499 * 10^2$	$4.392422 * 10^{-4}$	$2.276648 * 10^3$	$1.624564 * 10^{-3}$	$1.401391 * 10^6$
5	*dyadic op.*	$5.573111 * 10^2$	$4.672316 * 10^{-4}$	$2.140266 * 10^3$	$1.794330 * 10^{-3}$	$1.192794 * 10^6$
	triadic op.	$6.155499 * 10^2$	$3.560626 * 10^{-4}$	$2.808495 * 10^3$	$1.624564 * 10^{-3}$	$1.728769 * 10^6$
7	*dyadic op.*	$5.573111 * 10^2$	$4.289453 * 10^{-4}$	$2.331300 * 10^3$	$1.794330 * 10^{-3}$	$1.299259 * 10^6$
	triadic op.	$6.155499 * 10^2$	$3.150848 * 10^{-4}$	$3.173749 * 10^3$	$1.624564 * 10^{-3}$	$1.953601 * 10^6$
9	*dyadic op.*	$5.573111 * 10^2$	$4.015839 * 10^{-4}$	$2.490140 * 10^3$	$1.794330 * 10^{-3}$	$1.387782 * 10^6$
	triadic op.	$6.155499 * 10^2$	$2.956608 * 10^{-4}$	$3.382254 * 10^3$	$1.624564 * 10^{-3}$	$2.081946 * 10^6$
11	*dyadic op.*	$5.573111 * 10^2$	$3.906035 * 10^{-4}$	$2.560141 * 10^3$	$1.794330 * 10^{-3}$	$1.426795 * 10^6$
	triadic op.	$6.155499 * 10^2$	$2.834716 * 10^{-4}$	$3.527690 * 10^3$	$1.624564 * 10^{-3}$	$2.171469 * 10^6$

(e)$pe = 16$

f	*operation*	$a_0(\mu sec)$	$a_1(\mu sec/flop)$	$r_\infty(Mflop/s)$	$\pi_0(Mflop/s)$	$s_{1/2}(flop)$
1/3	*dyadic op.*	$5.636702 * 10^2$	$1.152806 * 10^{-3}$	$8.674486 * 10^2$	$1.774087 * 10^{-3}$	$4.889549 * 10^5$
1/2	*triadic op.*	$6.114180 * 10^2$	$7.706485 * 10^{-4}$	$1.297608 * 10^3$	$1.635542 * 10^{-3}$	$7.933812 * 10^5$
1	*dyadic op.*	$5.636702 * 10^2$	$4.750824 * 10^{-4}$	$2.104898 * 10^3$	$1.774087 * 10^{-3}$	$1.186468 * 10^6$
	triadic op.	$6.114180 * 10^2$	$4.236324 * 10^{-4}$	$2.360537 * 10^3$	$1.635542 * 10^{-3}$	$1.443275 * 10^6$
3	*dyadic op.*	$5.636702 * 10^2$	$2.669454 * 10^{-4}$	$3.746084 * 10^3$	$1.774087 * 10^{-3}$	$2.111556 * 10^6$
	triadic op.	$6.114180 * 10^2$	$2.229791 * 10^{-4}$	$4.484725 * 10^3$	$1.635542 * 10^{-3}$	$2.742042 * 10^6$
5	*dyadic op.*	$5.636702 * 10^2$	$2.258583 * 10^{-4}$	$4.427555 * 10^3$	$1.774087 * 10^{-3}$	$2.495681 * 10^6$
	triadic op.	$6.114180 * 10^2$	$1.788920 * 10^{-4}$	$5.589965 * 10^3$	$1.635542 * 10^{-3}$	$3.417805 * 10^6$
7	*dyadic op.*	$5.636702 * 10^2$	$2.081038 * 10^{-4}$	$4.805294 * 10^3$	$1.774087 * 10^{-3}$	$2.708601 * 10^6$
	triadic op.	$6.114180 * 10^2$	$1.610078 * 10^{-4}$	$6.210879 * 10^3$	$1.635542 * 10^{-3}$	$3.797443 * 10^6$
9	*dyadic op.*	$5.636702 * 10^2$	$2.016318 * 10^{-4}$	$4.959535 * 10^3$	$1.774087 * 10^{-3}$	$2.795542 * 10^6$
	triadic op.	$6.114180 * 10^2$	$1.502826 * 10^{-4}$	$6.654130 * 10^3$	$1.635542 * 10^{-3}$	$4.068455 * 10^6$
11	*dyadic op.*	$5.636702 * 10^2$	$1.937937 * 10^{-4}$	$5.160126 * 10^3$	$1.774087 * 10^{-3}$	$2.908610 * 10^6$
	triadic op.	$6.114180 * 10^2$	$1.436749 * 10^{-4}$	$6.960158 * 10^3$	$1.635542 * 10^{-3}$	$4.255566 * 10^6$

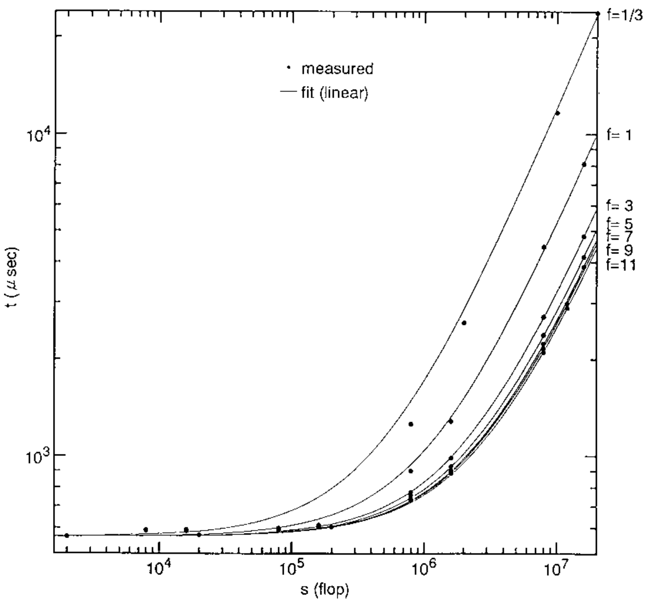


Fig. 2 The ($\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$) benchmark test with parameter f **(**$pe = 16$**)**

(a) *dyads*

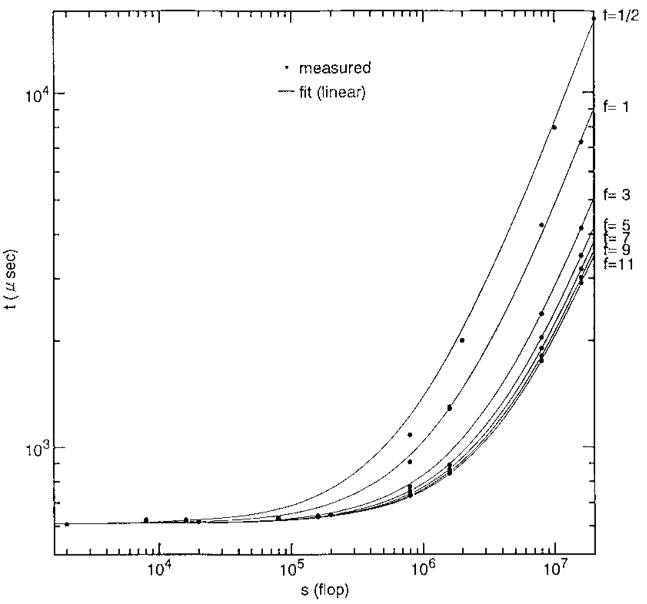


Fig. 2 The ($\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$) benchmark test with parameter f **(**$pe = 16$**)**

(b) *triads*

4. Analyses of results of the $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark

Since arithmetic and communication cannot be overlapped in time, the time t of (9) can be separated the set-up plus pipeline start-up plus synchronization plus communication start-up, communication, and calculation parts of the time as follows:

$$t = t_0 + t_c m + t_a s, \qquad (11)$$

where

$t_0 =$ time for null job when $s = m = 0$ (μsec),
$tc =$ time per I/O word on average (μsec),
$t_a =$ time per floating-point arithmetic operation on average (μsec).

From (9) and (11), we obtain the following equation:

$$a_1 s = r_\infty^{-1} s = t_c m + t_a s, \qquad (12)$$

from which

$$r_\infty = \hat{r}_\infty pipe(f/f_{1/2}), \qquad (13)$$

where

$\hat{r}_\infty = 1/t_a$ = peak maximum performance (Mflop/sec)
= inverse of the arithmetic time,
$f_{1/2} = t_c/t_a$ = half-performance intensity (flop/I/O word)
= ratio of the communication time to the arithmetic time.

On the other hand we can obtain the following equation from (9) and (11):

$$a_0 = t_0 = r_\infty^{-1} s_{1/2}, \qquad (14)$$

hence from (13)

$$s_{1/2} = \hat{s}_{1/2} pipe(f/f_{1/2}), \qquad (15)$$

where

$\hat{s}_{1/2} = t_0/t_a$ = peak half-performance grain size (flop)
= ratio of the synchronization time to the arithmetic time
= peak possible arithmetic lost during synchronization and communication start-up.

Table 3 The characteristic parameters of the NWT with MIMD computing in the global memory access ($pe = 1 \sim 16$).

(a) f/r_∞ versus f.

pe	operation	$a_0(\mu sec/w)$	$a_1(\mu sec/flop)$	$\hat{r}_\infty(Mflop/s)$	$f_{1/2}(flop/w)$
1	dyadic op.	$4.813236 * 10^{-3}$	$2.602747 * 10^{-3}$	$3.842095 * 10^{2}$	$1.849291 * 10^{0}$
	triadic op.	$5.045908 * 10^{-3}$	$1.792651 * 10^{-3}$	$5.578331 * 10^{2}$	$2.814774 * 10^{0}$
2	dyadic op.	$2.542006 * 10^{-3}$	$1.313694 * 10^{-3}$	$7.612123 * 10^{2}$	$1.935006 * 10^{0}$
	triadic op.	$2.571595 * 10^{-3}$	$9.150726 * 10^{-4}$	$1.092809 * 10^{3}$	$2.810263 * 10^{0}$
4	dyadic op.	$1.241929 * 10^{-3}$	$6.592441 * 10^{-4}$	$1.516889 * 10^{3}$	$1.883868 * 10^{0}$
	triadic op.	$1.253582 * 10^{-3}$	$4.659327 * 10^{-4}$	$2.146233 * 10^{3}$	$2.690479 * 10^{0}$
8	dyadic op.	$6.881063 * 10^{-4}$	$3.281024 * 10^{-4}$	$3.047829 * 10^{3}$	$2.097230 * 10^{0}$
	triadic op.	$6.334515 * 10^{-4}$	$2.258190 * 10^{-4}$	$4.428325 * 10^{3}$	$2.805129 * 10^{0}$
16	dyadic op.	$3.131489 * 10^{-4}$	$1.651551 * 10^{-4}$	$6.054914 * 10^{3}$	$1.896090 * 10^{0}$
	triadic op.	$3.208815 * 10^{-4}$	$1.146905 * 10^{-4}$	$8.719118 * 10^{3}$	$2.797804 * 10^{0}$

(b) $f/s_{1/2}$ versus f.

pe	operation	$a_0(flop/w)$	$a_1(1/flop)$	$\hat{s}_{1/2}(flop)$	$f_{1/2}(flop/w)$
1	dyadic op.	$8.982354 * 10^{-6}$	$4.857277 * 10^{-6}$	$2.058767 * 10^{5}$	$1.849257 * 10^{0}$
	triadic op.	$8.613319 * 10^{-6}$	$3.060048 * 10^{-6}$	$3.267923 * 10^{5}$	$2.814766 * 10^{0}$
2	dyadic op.	$4.653150 * 10^{-6}$	$2.404507 * 10^{-6}$	$4.158857 * 10^{5}$	$1.935178 * 10^{0}$
	triadic op.	$4.266768 * 10^{-6}$	$1.518264 * 10^{-6}$	$6.586470 * 10^{5}$	$2.810294 * 10^{0}$
4	dyadic op.	$2.259059 * 10^{-6}$	$1.198954 * 10^{-6}$	$8.340604 * 10^{5}$	$1.884192 * 10^{0}$
	triadic op.	$2.080698 * 10^{-6}$	$7.733730 * 10^{-7}$	$1.293037 * 10^{6}$	$2.690420 * 10^{0}$
8	dyadic op.	$1.234421 * 10^{-6}$	$5.887775 * 10^{-7}$	$1.698434 * 10^{6}$	$2.096583 * 10^{0}$
	triadic op.	$1.028971 * 10^{-6}$	$3.668824 * 10^{-7}$	$2.725669 * 10^{6}$	$2.804634 * 10^{0}$
16	dyadic op.	$5.557034 * 10^{-7}$	$2.929589 * 10^{-7}$	$3.413448 * 10^{6}$	$1.896865 * 10^{0}$
	triadic op.	$5.247379 * 10^{-7}$	$1.875845 * 10^{-7}$	$5.330931 * 10^{6}$	$2.797341 * 10^{0}$

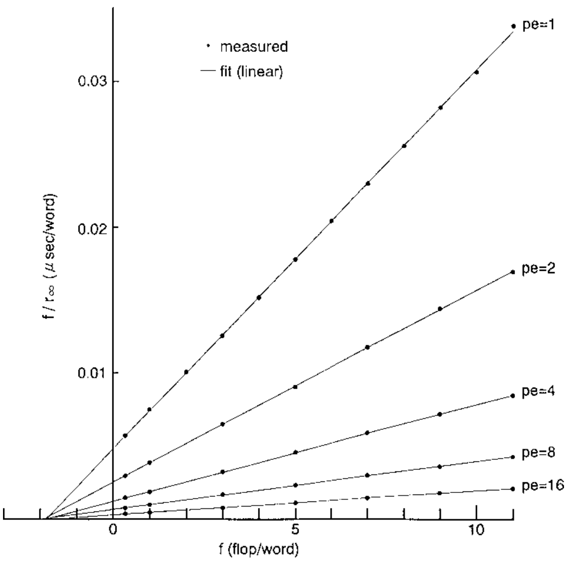


Fig. 3 f/r_∞ **versus** f (pe = **1 ~ 16).**
(a) *dyads*

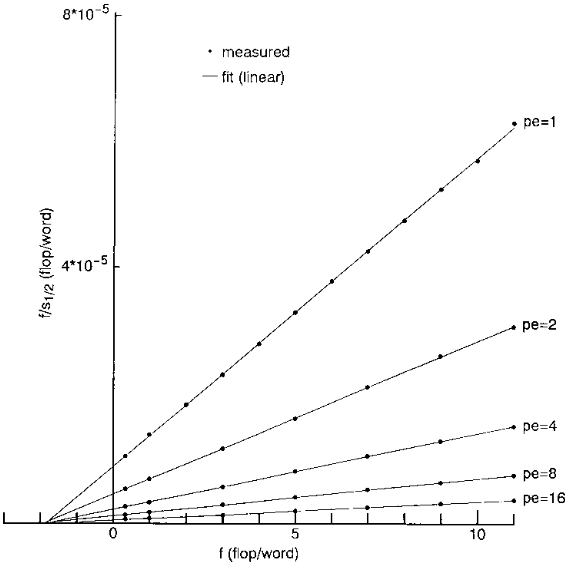


Fig. 4 $f/s_{1/2}$ **versus** f (pe = **1 ~ 16).**
(a) *dyads*

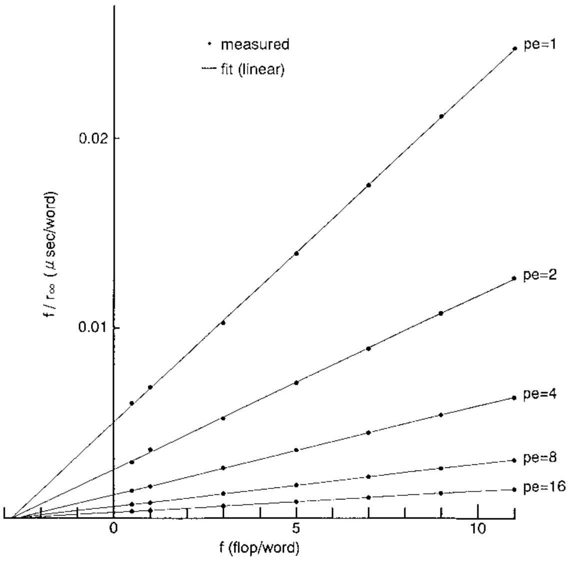


Fig. 3 f/r_∞ **versus** f (pe = **1 ~ 16).**
(b) *triads*

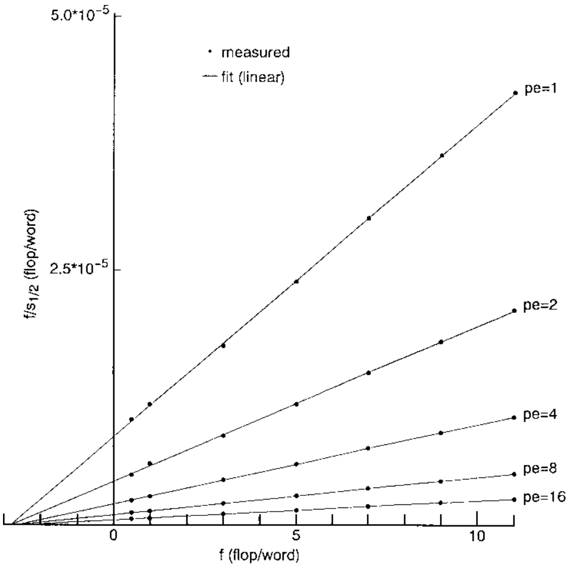


Fig. 4 $f/s_{1/2}$ **versus** f (pe = **1 ~ 16).**
(b) *triads*

Equations (13) and (15) can be rewritten in the following forms:

$$f/r_\infty = \hat{r}_\infty^{-1}(f+f_{1/2}), \tag{16}$$

$$f/s_{1/2} = \hat{s}_{1/2}^{-1}(f+f_{1/2}), \tag{17}$$

From these equations we can finally compute the values of the parameters ($\hat{r}_\infty$, $\hat{s}_{1/2}$, $f_{1/2}$), which is easy because both equations are linear for the computational intensity f.

Table 3, and Figure 3 and 4 show measurements of the characteristic parameters ($\hat{r}_\infty$, $\hat{s}_{1/2}$, $f_{1/2}$) of the NWT with MIMD computing in the global memory access when pe **= 1 ~ 16, and fits to the following expressions:**

$$f/r_\infty = a_0 + a_1 f, \tag{18}$$

$$f/s_{1/2} = a_0 + a_1 f, \tag{19}$$

respectively.

5. Estimations of the overall performance of the NWT

In order to estimate the characteristic parameters ($\hat{r}_\infty$, $\hat{s}_{1/2}, f_{1/2}$) for $pe = 1 \sim 128$, we performed experiments corresponding to $f = 1/3$ (dyads) and $f = 1/2$ (triads) for the benchmark program mentioned in Section 3. Table 4 shows measurements of the parameters ($\hat{r}_\infty$, $\hat{s}_{1/2}$) and estimations of the parameters ($\hat{r}_\infty$, $\hat{s}_{1/2}$) of the NWT with MIMD computation in the global memory access, where the values of $f_{1/2}$ are the average values of $f_{1/2}$ in Table 3, and the values of the parameters ($\hat{r}_\infty$, $\hat{s}_{1/2}$) are computed from (13) and (15), respectively.

Figure 5 shows the variation of maximum performance r_∞ in Table 4 as a function of the number of processing elements pe. Solid lines show the linear fits

$$dyads : r_\infty = 5.857613 * 10^1 pe \ (Mflop/sec), \tag{20}$$

$$triads : r_\infty = 8.619480 * 10^1 pe \ (Mflop/sec), \tag{21}$$

Similarly Figure 6 shows the variation of half-performance grain size $s_{1/2}$ in Table 4 as a function of pe. Solid lines show the quadratic fits

$$dyads : s_{1/2} = 4.698217 * 10^2 + 3.081760 * 10^4 pe + 7.300503 * 10^1 pe^2 \ (flop), \tag{22}$$

$$triads : s_{1/2} = -1.021786 * 10^3 + 4.948197 * 10^4 pe + 1.213604 * 10^2 pe^2 \ (flop), \tag{23}$$

When $pe = 1$, the value of $s_{1/2}$ is about 110 times as large as one of $n_{1/2}$ in Table 1 (b) of [4], and about 3 times of the value of $s_{1/2}$ in Table 2 of [4].

Figure 7 shows the variation of set-up plus pipeline start-up plus synchronization plus communication start-up time t_0 ($= a_0$ in Table 4) as a function of pe. These approximate fit functions are obtained from (14), (20), (21), (22) and (23) as follows:

$$dyads : t_0 = 5.261119 * 10^2 + 1.246327 * 10^0 pe + 8.020702 * 10^0 / pe \ (\mu sec), \tag{24}$$

$$triads : t_0 = 5.740714 * 10^2 + 1.407978 * 10^0 pe - 1.185438 * 10^1 / pe \ (\mu sec). \tag{25}$$

The above parameters give rise to the complete formulas for the timing relations

$$dyads : t = 5.261119 * 10^2 + 1.246327 pe + \frac{8.020702}{pe} + \frac{1.707180 * 10^{-2}}{pe} s \ (\mu sec), \tag{26}$$

$$triads : t = 5.740714 * 10^2 + 1.407978 pe - \frac{1.185438 * 10^1}{pe} + \frac{1.160163 * 10^{-2}}{pe} s \ (\mu sec), \tag{27}$$

From (10), (13) and (15) we can obtain the following equa-

Table 4 The characteristic paramenters of the NWT with MIMD computing in the global memory access ($pe = 1 \sim 128$).

pe	operation	$a_0(\mu sec)$	$a_1(\mu sec/flop)$	$r_\infty(Mflop/s)$	$s_{1/2}(flop)$	$\hat{r}_\infty(Mflop/s)$	$\hat{s}_{1/2}(flop)$	$f_{1/2}(flop/w)$
1	dyadic op.	$5.358462 * 10^2$	$1.708670 * 10^{-2}$	$5.852505 * 10^1$	$3.136043 * 10^4$	$3.977987 * 10^2$	$2.131590 * 10^5$	$1.932356 * 10^0$
	triadic op.	$5.858259 * 10^2$	$1.205861 * 10^{-2}$	$8.292830 * 10^1$	$4.858154 * 10^4$	$5.446052 * 10^2$	$3.190438 * 10^5$	$2.783590 * 10^0$
2	dyadic op.	$5.463351 * 10^2$	$8.945350 * 10^{-3}$	$1.117899 * 10^2$	$6.107476 * 10^4$	$7.598436 * 10^2$	$4.151293 * 10^5$	$1.932356 * 10^0$
	triadic op.	$6.027210 * 10^2$	$5.940494 * 10^{-3}$	$1.683362 * 10^2$	$1.014597 * 10^5$	$1.105494 * 10^3$	$6.663042 * 10^5$	$2.783590 * 10^0$
4	dyadic op.	$5.498325 * 10^2$	$4.398520 * 10^{-3}$	$2.273492 * 10^2$	$1.250040 * 10^5$	$1.545308 * 10^3$	$8.496607 * 10^5$	$1.932356 * 10^0$
	triadic op.	$6.024799 * 10^2$	$2.953622 * 10^{-3}$	$3.385674 * 10^2$	$2.039800 * 10^5$	$2.223433 * 10^3$	$1.339574 * 10^6$	$2.783590 * 10^0$
8	dyadic op.	$5.573111 * 10^2$	$2.304755 * 10^{-3}$	$4.338856 * 10^2$	$2.418093 * 10^5$	$2.949150 * 10^3$	$1.643594 * 10^6$	$1.932356 * 10^0$
	triadic op.	$6.155499 * 10^2$	$1.498470 * 10^{-3}$	$6.673474 * 10^2$	$4.107856 * 10^5$	$4.382591 * 10^3$	$2.697703 * 10^6$	$2.783590 * 10^0$
16	dyadic op.	$5.636702 * 10^2$	$1.152806 * 10^{-3}$	$8.674486 * 10^2$	$4.889549 * 10^5$	$5.896107 * 10^3$	$3.323460 * 10^6$	$1.932356 * 10^0$
	triadic op.	$6.114180 * 10^2$	$7.706485 * 10^{-4}$	$1.297608 * 10^3$	$7.933812 * 10^5$	$8.521627 * 10^3$	$5.210278 * 10^6$	$2.783590 * 10^0$
32	dyadic op.	$6.121628 * 10^2$	$5.614663 * 10^{-4}$	$1.781051 * 10^3$	$1.090293 * 10^6$	$1.210592 * 10^4$	$7.410796 * 10^6$	$1.932356 * 10^0$
	triadic op.	$6.503874 * 10^2$	$3.890572 * 10^{-4}$	$2.570316 * 10^3$	$1.671701 * 10^6$	$1.687973 * 10^4$	$1.097836 * 10^7$	$2.783590 * 10^0$
64	dyadic op.	$6.223780 * 10^2$	$2.751105 * 10^{-4}$	$3.634903 * 10^3$	$2.262284 * 10^6$	$2.470668 * 10^4$	$1.537690 * 10^7$	$1.932356 * 10^0$
	triadic op.	$6.773932 * 10^2$	$1.831644 * 10^{-4}$	$5.459576 * 10^3$	$3.698280 * 10^6$	$3.585402 * 10^4$	$2.428727 * 10^7$	$2.783590 * 10^0$
128	dyadic op.	$6.775643 * 10^2$	$1.317650 * 10^{-4}$	$7.589269 * 10^3$	$5.142218 * 10^6$	$5.158478 * 10^4$	$3.495201 * 10^7$	$1.932356 * 10^0$
	triadic op.	$7.477181 * 10^2$	$8.992630 * 10^{-5}$	$1.112022 * 10^4$	$8.314788 * 10^6$	$7.302850 * 10^4$	$5.460472 * 10^7$	$2.783590 * 10^0$

tion for the actual or average performance, r, with MIMD computing in the global memory access:

$$r=\hat{r}_{\infty}pipe(f/f_{1/2})pipe(s/s_{1/2})=\frac{\hat{r}_{\infty}}{1+(\hat{s}_{1/2}/s)+(f_{1/2}/f)}, \quad (28)$$

where the first term in the denominator comes from the arithmetic time, the second term comes from the synchronization time and the third term comes from the communication time. Equation (28) gives the degree of degradation of the peak maximum rate due to synchronization and communication delays, and inadequate grain size.

Figure 8 shows (a) the timing relations as a function of the amount of arithmetic operations and (b) the actual rates as a function of the arithmetic operations, when $pe = 1 \sim 128$.

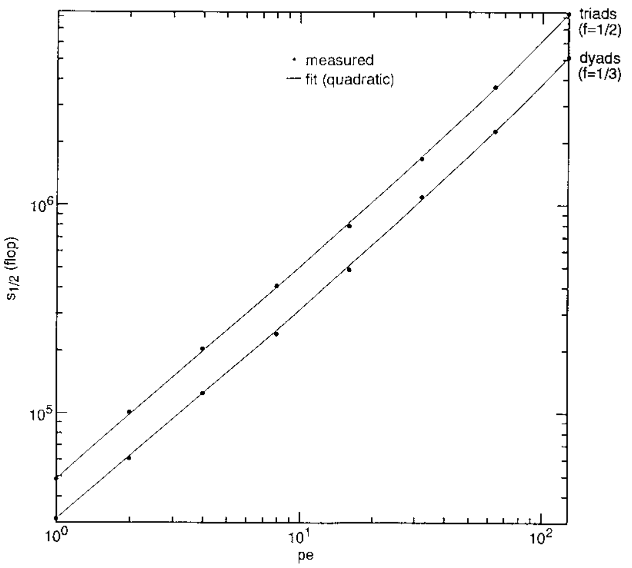


Fig. 6 The half-performance grain size against the number of processing elements with MIMD comuting in the global memory access.

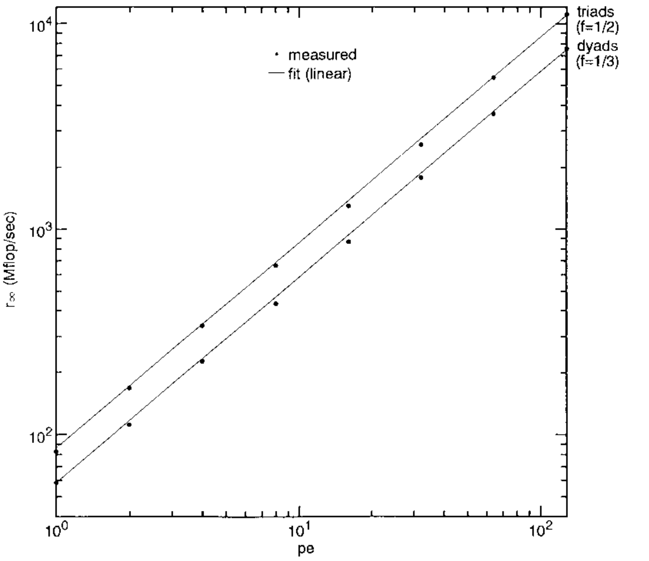


Fig. 5 The maximum rates against the number of processing elements with MIND comuting in the global memory access.

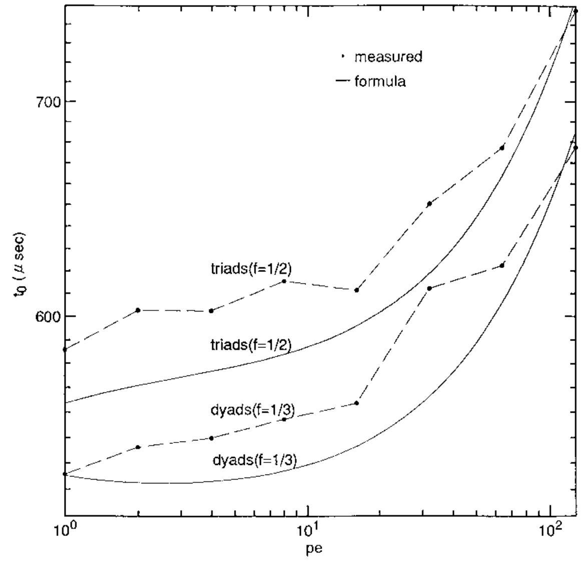


Fig. 7 The start-up time against the number of processing elements with MIMD comuting in the global memory access.

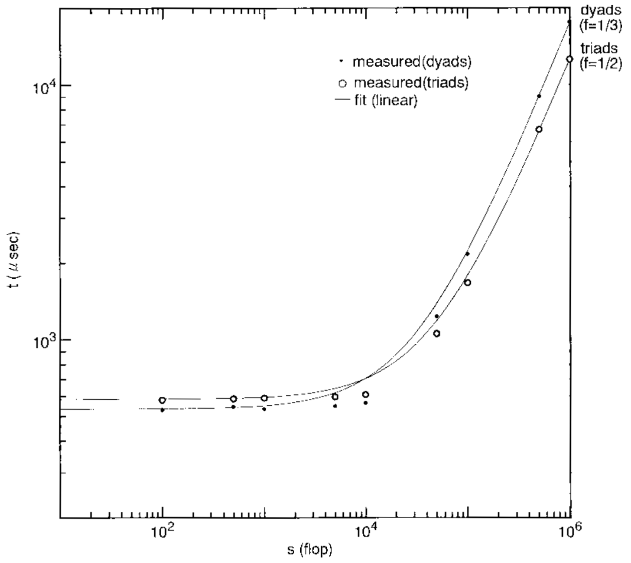


Fig. 8.1 The ($\hat{r}_\infty$, $\hat{s}_{1/2}$, $f_{1/2}$) benchmark test ($pe = 1$).

(a) The timing relations as a function of the amount of arithmetic.

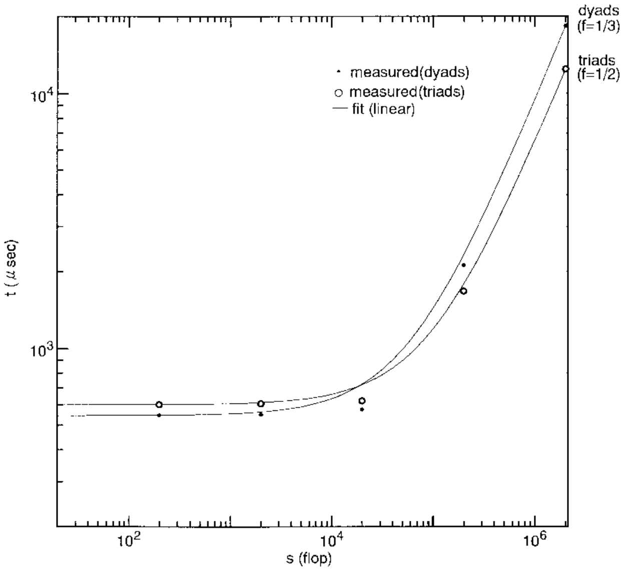


Fig. 8.2 The ($\hat{r}_\infty$, $\hat{s}_{1/2}$, $f_{1/2}$) benchmark test ($pe = 2$).

(a) The timing relations as a function of the amount of arithmetic.

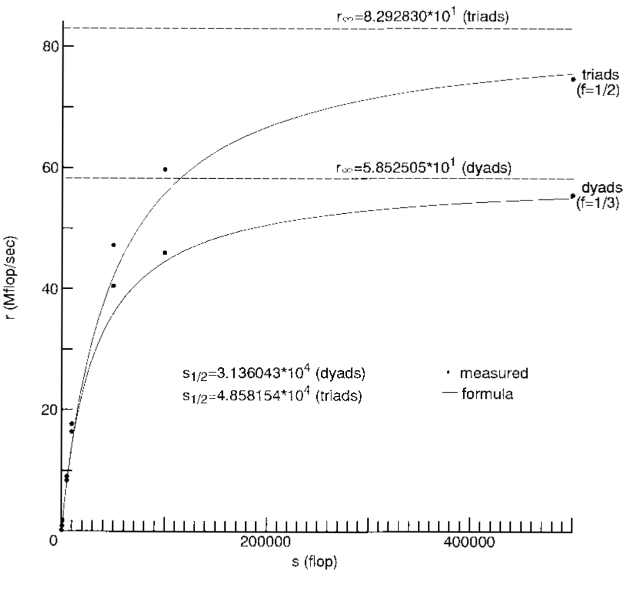


Fig. 8.1 The ($\hat{r}_\infty$, $\hat{s}_{1/2}$, $f_{1/2}$) benchmark test ($pe = 1$).

(b) The actual processing rate as a function of the amount of arithmetic.

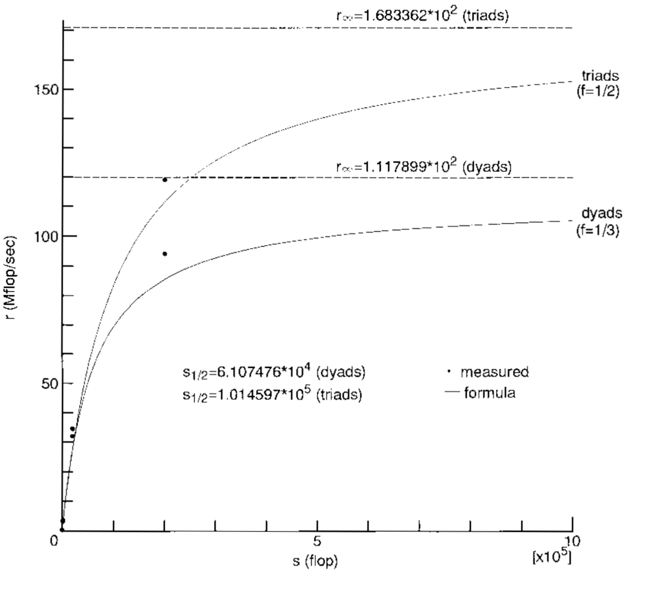


Fig. 8.2 The ($\hat{r}_\infty$, $\hat{s}_{1/2}$, $f_{1/2}$) benchmark test ($pe = 2$).

(b) The actual processing rate as a function of the amount of arithmetic.

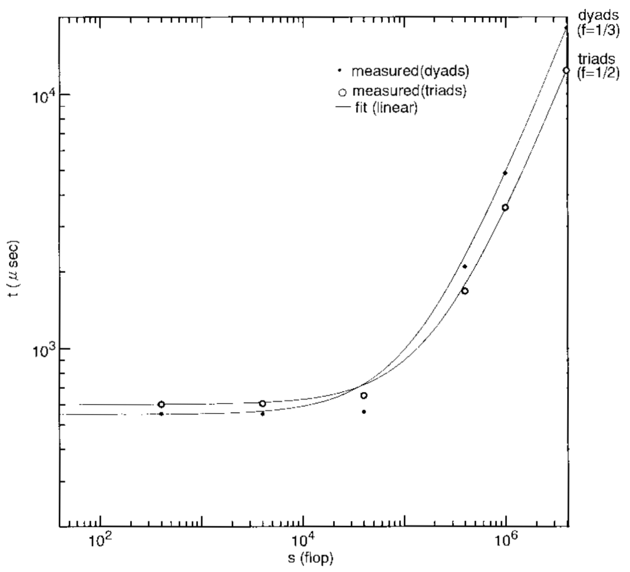


Fig. 8.3 The ($\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$) benchmark test ($pe = 4$).
(a) The timing relations as a function of the amount of arithmetic.

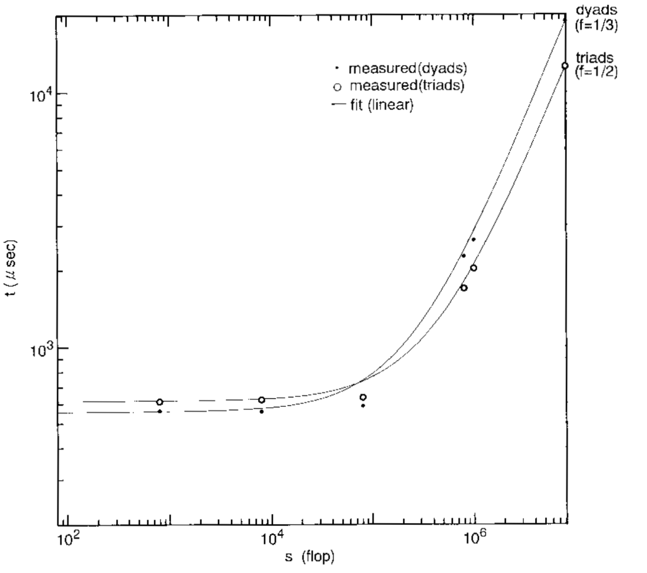


Fig. 8.4 The ($\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$) benchmark test ($pe = 8$).
(a) The timing relations as a function of the amount of arithmetic.

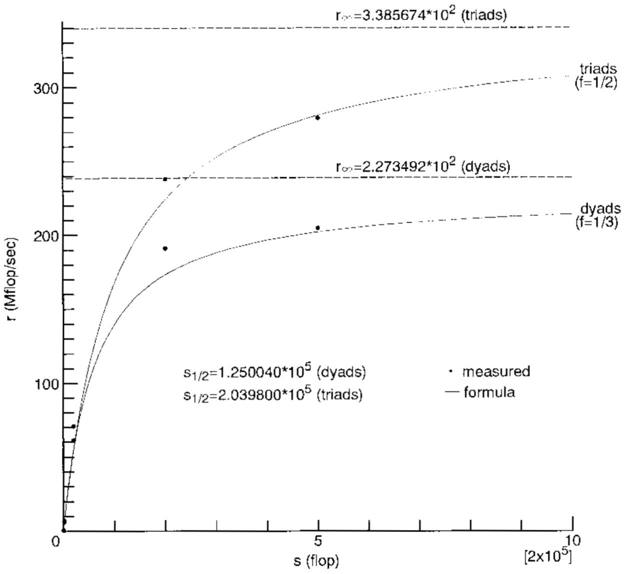


Fig. 8.3 The ($\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$) benchmark test ($pe = 4$).
(b) The actual processing rate as a function of the amount of arithmetic.

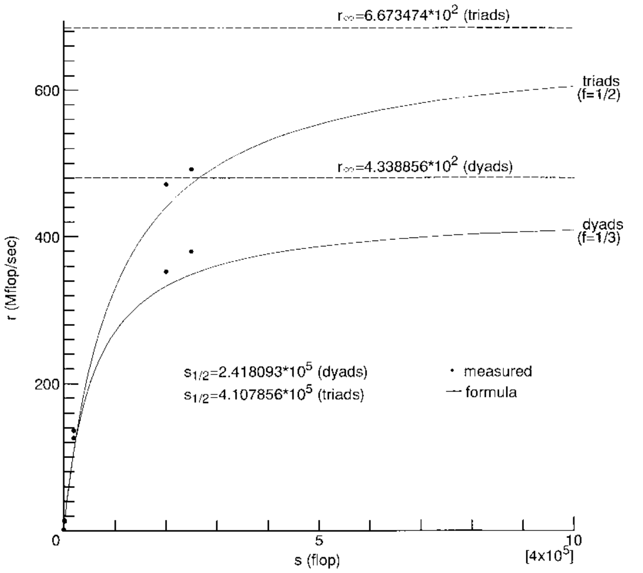


Fig. 8.4 The ($\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2}$) benchmark test ($pe = 8$).
(b) The actual processing rate as a function of the amount of arithmetic.

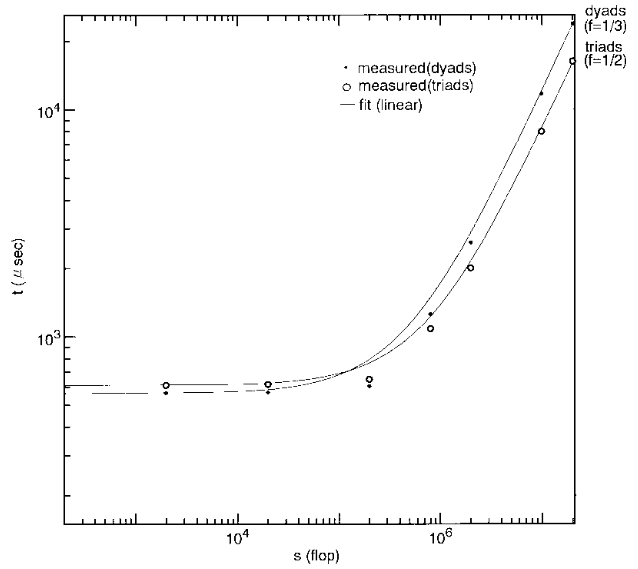


Fig. 8.5 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test (pe = 16).
(a) The timing relations as a function of the amount of arithmetic.

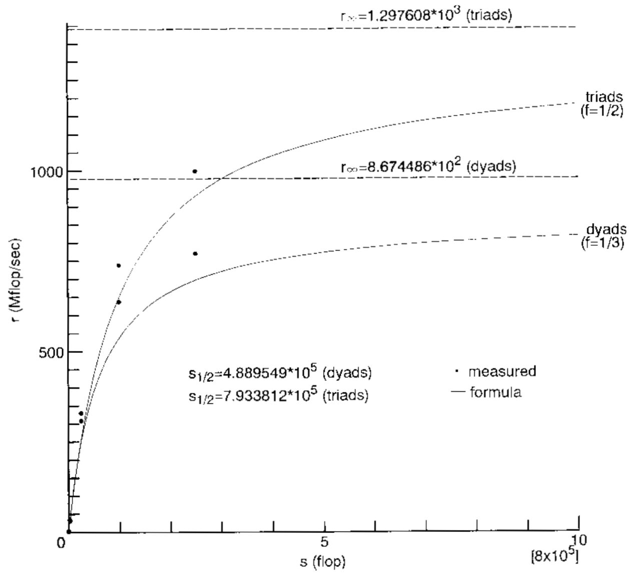


Fig. 8.5 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test (pe = 16).
(b) The actual processing rate as a function of the amount of arithmetic.

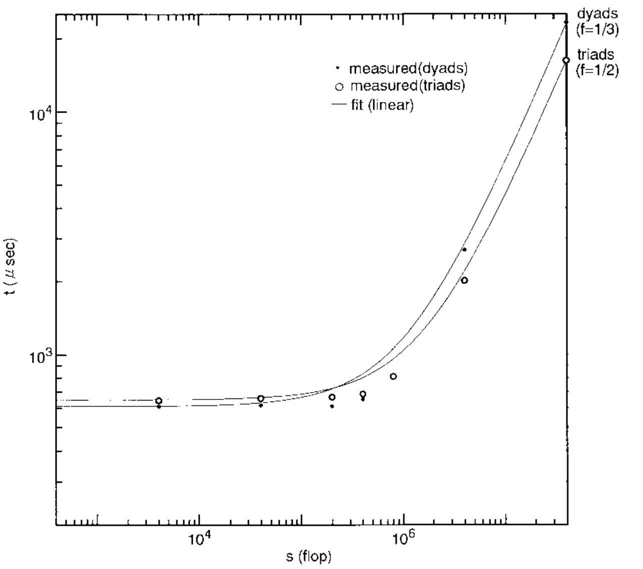


Fig. 8.6 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test (pe = 32).
(a) The timing relations as a function of the amount of arithmetic.

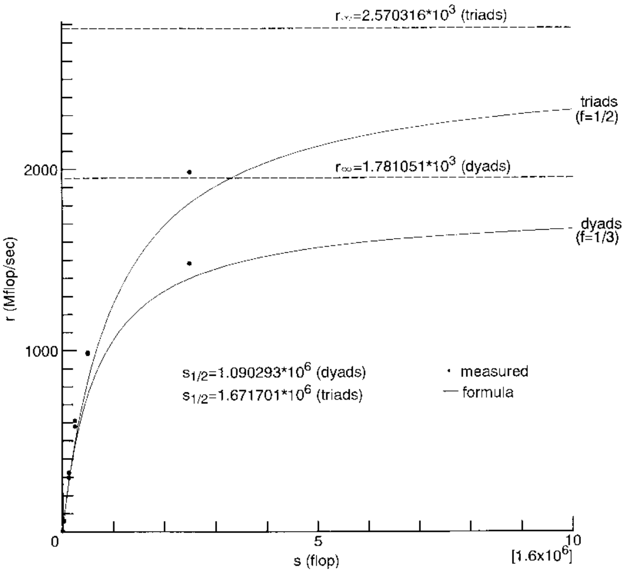


Fig. 8.6 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test (pe = 32).
(b) The actual processing rate as a function of the amount of arithmetic.

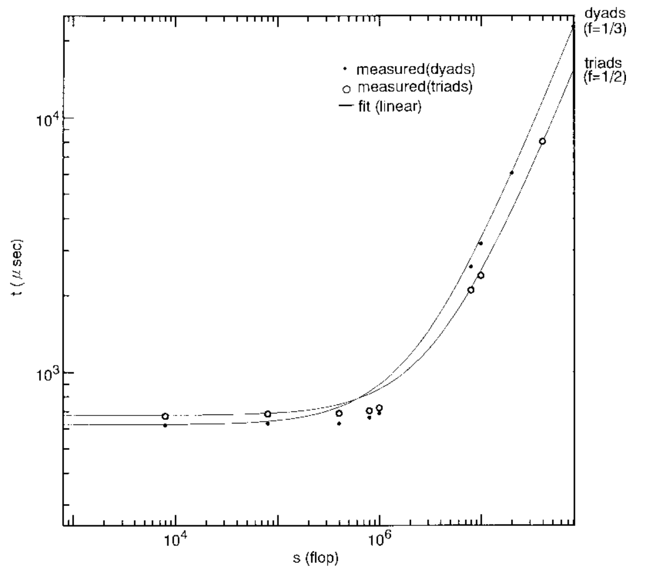


Fig. 8.7 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test $(pe = 64)$.
(a) The timing relations as a function of the amount of arithmetic.

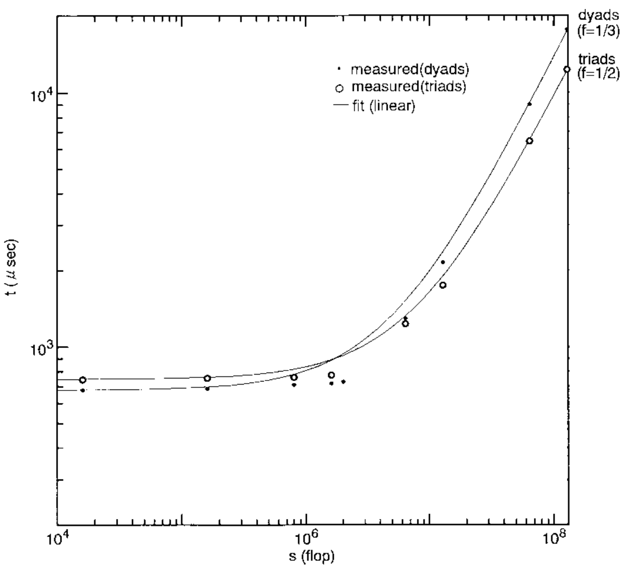


Fig. 8.8 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test $(pe = 128)$.
(a) The timing relations as a function of the amount of arithmetic.

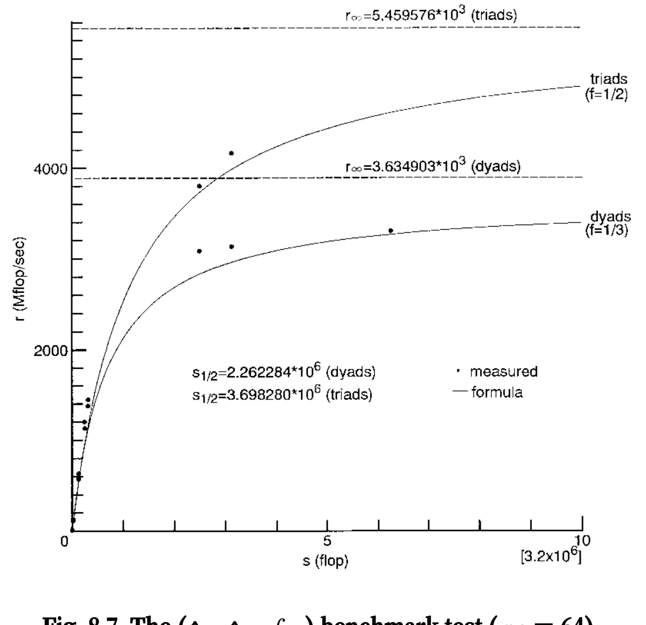


Fig. 8.7 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test $(pe = 64)$.
(b) The actual processing rate as a function of the amount of arithmetic.

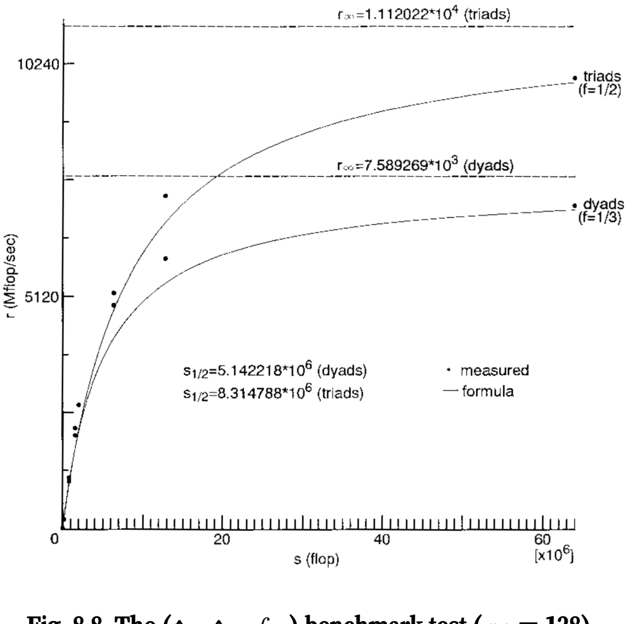


Fig. 8.8 The $(\hat{r}_\infty, \hat{s}_{1/2}, f_{1/2})$ benchmark test $(pe = 128)$.
(b) The actual processing rate as a function of the amount of arithmetic.

6. Discussions

In this section we interpret the significance of the results obtained in the previous sections, and estimate several hardware parameters.

6.1 Consideration with performance of the data transfer

Programs with a lot of data communication between processing elements may find that their overall performance is limited more by the performance of the communication network than by the arithmetic performance of the processing elements themselves. Hence the speed of communication of message between the global and local memory spaces, and latency of the crossbar network are of critical importance.

(1) Assessment of the long-message performance and estimation of the half-performance

The best bandwidth for long messages of the crossbar network of the NWT is given by (4), and seems to be a barely satisfactory value. From (4), and (29) and (30) of [4] , the values of the half-performance intensity $f_{1/2}$ (hardware parameter) defined by (13) are computed as follows:

$$dyads: f_{1/2}=1.948638 \ (flop/I/O\ word), \tag{29}$$

$$triads: f_{1/2}=2.761333 \ (flop/I/O\ word), \tag{30}$$

These values coincide very well with ones in Table 4.

(2) Assessment of the short-message performance

On the other hand the worst bandwidth for short messages is five to seven orders of magnitude smaller than the best bandwidth, as shown in Table 1. This occurs because not only the message start-up latency is rather large and increased proportionally to pe, but also because the maximum bandwidth is increased proportionally to pe, as shown in Figure 1 (c) and (e). To improve the π_0 and $n_{1/2}$ 10-fold would require a 10-fold decrease from the measured values of t_0. Since almost all the message start-up overhead is in the software, we must point the target for the next improvement of the NWT system software at this reduction.

(3) Estimation of the message start-up overhead

The average value of the data transfer overhead only in case of transferring one message is shown in Figure 1 (e). According to experiments for $pe=1 \sim 16$, this value in case of transferring two messages is increased by 41.8 μsec on an average, and similarly in case of transferring three messages is increased by 78.0 μsec on an average. Hence we can consider the average time that is required to transfer one message to be

$$40.4 \ \mu sec. \tag{31}$$

This value can be interpreted to be the sum of the time required to make a packet and the pure transfer time of one data. Hence the values subtracted this value from the values of Figure 1 (e) become the time that is required for the operating system to complete the processing on a read or write demand, and e.g., when $pe=1$, it is a rather large value of 144.3 μsec.

6.2 Consideration with MIMD computation in the global memory access

(1) Assessment of a_0 in Table 4

The value of a_0 in (9) can be assessed by the following equations:

$$dyads: a_0=a_{01}+2a_{02}+a_{03} \ (\mu sec), \tag{32}$$

$$triads: a_0=a_{01}+2a_{02}+2a_{03} \ (\mu sec), \tag{33}$$

where

a_{01} = value of a_0 of Table 2 in [4],
a_{02} = value to be computed from (6),
a_{03} = value of (31).

The values of a_0 computed from the above equations roughly coincide with those in Table 4.

(2) Assessment of $\hat{r}_\infty$ in Table 4

The values of $\hat{r}_\infty$ in Table 4 well coincide with those of r_∞ of Table 2 in [4]. This shows correctness of methodology used in this paper.

(3) Degree of degradation due to synchronization and communication overheads

The degree of degradation of the peak maximum rate, d_{pr}, due to synchronization and communication overheads is dependent on f, and given by

$$d_{pr}=pipe(s/s_{1/2})pipe(f/f_{1/2}). \tag{34}$$

Furthermore we define the degree of degradation of the actual rate of the parallel processing in the global memory access to the actual rate of the vector processing, d_{pgv}, as follows:

$$d_{pgv}=\frac{pipe(s/s_{1/2})pipe(f/f_{1/2})}{pipe(n/n_{1/2})}, \tag{35}$$

where the numerator comes from (28), the denominator comes from (2) in [4], and $s=n$. Then we can approximately express the actual rate of the parallel processing in the global memory access r_{pg}, by

$$r_{pg}=d_{pgv}r_v pe, \tag{36}$$

where r_v is the actual rate of the vector processing. When $s, n \to \infty$, equation (35) reduces to the following:

$$d_{pgv}=pipe(f/f_{1/2}), \tag{37}$$

from which, using the values of $f_{1/2}$ in Table 4, the values of d_{pgv} for dyadic ($f=1/3$) and triadic ($f=1/2$) operations are computed as follows:

$$dyads : d_{pgv}=1/6.797068\ (flop/I/O\ word), \tag{38}$$

$$triads : d_{pgv}=1/6.567181\ (flop/I/O\ word), \tag{39}$$

6.3 Comparison

We note the followings:

(i) The value of $a_0(=t_0)$ for $pe=1$ in Table 1 is equal to 205.7 μsec (Intel iPSC/860, p.392 of [13]), but larger than 132.0 μsec (Intel Touchstone Delta, p.392 of [13]), 172.1 μsec (Intel Paragon, p.392 of [13]) and 87.2 μsec (Meiko CS-2, p.392 of [13]). The ratio of r_∞ for $pe=1$ in Table 1 to r_∞ in p.392 of [13] is 260 (iPSC/860), 108 (Delta) , 31 (Paragon) and 17 (CS-2), and hence the value of $n_{1/2}$ for $pe=1$ in Table 1 becomes about a 277-fold value of iPSC/860, a 179-fold value of Delta, a 40-fold value of Paragon and a 43-fold value of CS-2 of $n_{1/2}$ in p.392 of [13], because $n_{1/2}$ is proportional to r_∞.

(ii) There is no data pertinent to comparison with the values of the characteristic parameters in Table 4 except data on the IBM LCAP parallel computer system, of which the characteristic parameters are the followings [10]:

$$dyads : \hat{r}_\infty=1.08pe\ (Mflop/sec), \tag{40}$$

$$dyads : \hat{s}_{1/2}=1.08pe\ (2500/4777pe)\ (flop), \tag{41}$$

$$dyads : f_{1/2}=2.02\ (flop/I/O\ word), \tag{42}$$

where $pe=1\sim 10$.

It is obvious to prefer the NWT to the LCAP.

(iii) However, all sorts of the start-up time of the NWT in the global memory access seem to be rather large, and hence we must continue to make efforts with the decrease of every kind of start-up time commensurate with an increased maximum performance.

7. Summary of the characteristic parameters of performance of the NWT

We can summarize the significance of the characteristic parameters of performance of the NWT measured in practice through Fortran in this paper and [4] as follows:

(1) With SIMD computation, the maximum performance is 390Mflop/s (dyads) and 550Mflop/s (triads), the vector breakeven length is 25 (dyads) and 42 (triads),and a vector length to be required to reach 90% of the maximum performance is 2600 (dyads) and 4000 (trids).

(2) With MIMD computation in the local memory access and for $pe=2^i$ (i = 0 ∼ 7, the maximum performance is 386peMflop/s (dyads) and 547peMflop/s (triads), the breakeven grain size is 25,000∼13,000 (dyads) and 35,000 ∼ 18,000 (triads), a grain size to be required to reach 90% of the maximum performance is 90,000 ∼ 15,300,000 (dyads) and 135,000∼21,600,000 (triads), and the ratio of synchronization overhead to pipeline start-up time is 47∼ 43.

(3) With data transfer between the global and local memory spaces and for $pe=2^i$ (i = 0 ∼ 4), the maximum performance is 755peMbyte/s, and a message length to be required to reach 90% of the maximum performance is 1,440,000 ∼ 26,000,000byte.

(4) With MIMD computation in the global memory ac-

cess and for $pe=2^i$ (i=0∼7), the maximum performance is 58*pe*Mflop/s (dyads) and 86*pe*Mflop/s (triads), a grain size to be required to reach 90% of the maximum performance is 270,000 ∼ 45,000,000 (dyads) and 450,000 ∼ 74,700,000 (triads), the ratio of synchronization plus communication overhead to pipeline start-up time is 700∼900, and the ratio of synchronization plus communication overhead to synchronization overhead is 20.

As mentioned above, the degree of degradation of peak maximum rate when $s \to \infty$ is 1/6.8 (dyads) and 1/6.6 (triads) in case that data transferred from the global memory to the local memory uses only once. A number of reference of data transferred to the local memory to be required to reach 90% of the peak maximum rate when $s \to \infty$ is 52 (dyads) and 50 (triads).

(5) We finally mention that we had been vigorously continuing subsequent improvements to the NWT system software, and accomplished an 1.2 to 1.3-fold decrease of synchronization overheads and a 2.5 to 3.5-fold decrease of communication overheads at April 1994, which we will report in another paper.

8. Conclusions

The principal conclusion of this work is that the communication overheads and maximum performance of the NWT in the global memory access could be characterized by relatively simple timing equations for a wide range of problem conditions, and also several hardware parameters of the NWT could be estimated in the process of analyzing results of experiments.

The NWT with its system software at the moment of this experiment is definitely a large amount of work and long-message computer system with MIMD computation in the global memory access, but in the present state of art this is the decree of fate with every parallel computer that has the very large arithmetic capabilities of the nodes. In order to eliminate this restriction on the field of application, it is imperative to tackle the problem of drastically reducing the synchronization and communication overheads due to almost all being in the system software function.

That is to say, in order to decrease the excution time of programs with a lot of data communication between different nodes, not only we must increase communication speed, but also in particular decrease communication delays in proportion to the increase in the arithmetic capabilities of the processing elements. If it were not so, such programs may find that their overall performance is limited more by the performance of communication network than by the arithmetic performance of the processing elements themselves. Hence the decrease of latency of the crossbar network is of critical importance.

Generally speaking, at this moment there is a huge gap beween the communication performance of distributed-memory parallel computers and that of traditional shared-memory parallel vector computers. The reduction of this communication gap is the main challenge now facing us. When this wish of us would be realized, the distributed-memory parallel computer may be competitive in performance and generality with the traditional parallel vector computer. By the way, at this moment the huge-scale application program can not be executed by the latter, while on the other hand the NWT can execute the program of the maximum capacity of 35*Gbyte*.

As a concluding remark, we mention that we accomplisded a 2.5 to 3.5-fold decrease of communication overheads at April 1994, as a result of hading been still vigorously continuing subsequent improvements on the NWT system software, which we will report in another paper.

Finally, most of the overheads incurred arise from the time spent in system software routines supporting user programs. Hence the obtained results in this paper apply only to the system software available at the NAL during the period April to June 1993.

9. References

[1] E.A. Carmona and M.D. Rice, Modeling the serial and parallel fractions of a parallel algorithm, *J. Parallel Distributed Computing 13* (1991) 286-298.

[2] I.J. Curington and R.W. Hockney, Synchronization and pipeline overheadmeasurements on the FPS-5000 MIMD computer, in: M. Feilmeier, et al., eds., *Parallel Computing 85* (North-Holland, Amsterdam, 1986) 469-476.

[3] D. Dent, C-90 performance measurements, *Supercomputer 45VIII-(5)* (1991) 8-14.

[4] S. Hatayama, The characteristic parameters of the NWT computer system in the local memory access, *NAL TR* (submitted for publication).

[5] A.J.G. Hey, The Genesis distributed-memory benchmarks, *Parallel Computing 17* (1991) 1275-1283.

[6] R.W. Hockney and D.F. Snelling, Characterizing MIMD computers : e.g. Denelcor HEP, in: M.

Feilmeier et al., eds., *Parallel Computing 83* (North-Holland, Amsterdam, 1984) 521-526.

[7] R.W. Hockney, (r_∞, $n_{1/2}$, $s_{1/2}$) measurements on the 2-CPU CRAY X-MP, *Parallel Computing 2* (1985) 1-14.

[8] R.W. Hockney, Performance characterization of the HEP, in: J.S. Kowalik, ed., *MIMD Computation : HEP Supercomputer and its Applications* (MIT Press, Cambridge, MA, 1985) 59-90.

[9] R.W. Hockney and C.R. Jesshope, *Parallel Computers 2* (Adam Hilger, Bristol, 1988).

[10] R.W. Hockney, Synchronization and communication overheads on the LCAP multiple FPS-164 computer system, *Parallel Computing 9* (1988/89) 279-290.

[11] R.W. Hockney, Performance parameters and benchmarking of supercomputers, *Parallel Computing 17* (1991) 1111-1130.

[12] R.W. Hockney and E.A. Carmona, Comparison of communications on the Intel iPSC/860 and Touchstone Delta, *Parallel Computing 18* (1992) 1067-1072.

[13] R.W. Hockney, The communication challenge for MPP : Intel Paragon and Meiko CS-2, *Parallel Computing 20* (1994) 389-398.

Appendix

(1) Machine architecture of the NWT

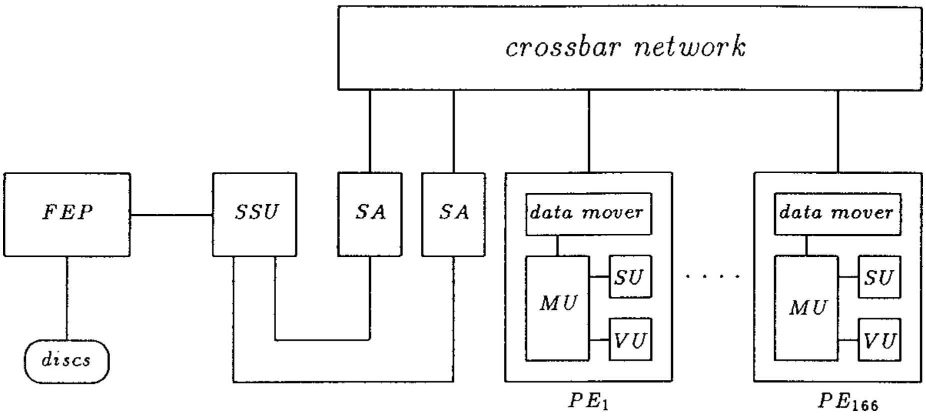
The machine architecture of the NWT is shown in Fig. A (a). Input and output are managed by two system administrators. Each processing element is a vector computer with pipelines of multiplicity 8. The addition, multiply, load and store pipelines can be operated simultaneously. The data transfer is managed by the data mover and can be done asynchronously through the crossbar network.

(2) NWT Fortran

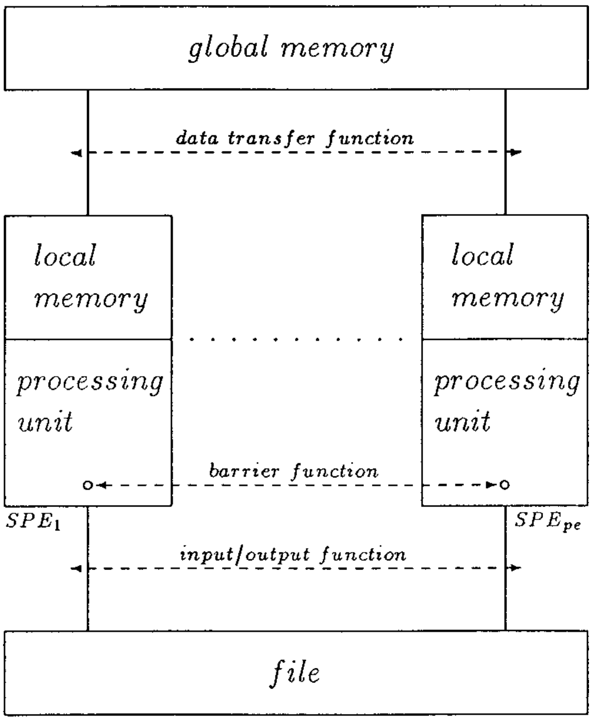
The NWT Fortran language specification is structured by adding the specification for the NWT Fortran compiler directives to the Fortran 77 language specification. Almost all of the compiler directives are comments in the Fortran 77 specification.

(3) Logical model of the NWT for programming

In order to ease programming, the logical model of the NWT assumed for programming in the NWT Fortran is a



(a) Hardware configuration.



(b) Logical model for programming.

Fig. A Architecture and logical model of the NWT (PE : processing element, SPE : selected PE5, MU : main memory unit, SU : scalar unit, VU : vector unit, SA : system administrator, SSU : system strage unit, FEP : front end processor).

hierarchical memory parallel computer system shown in Fig. A(b). A local memory (or space) is the memory specific to a processing element. The global memory (or space) is a virtual memory shared by the selected processing elememts, which is physically distributed across the selected processing elements.

(4) Synchronization

A barrier synchronization is performed for the selected processing elements that are running in parallel. After all the selected processing elements running in parallel reach the barrier point, the selected processing elements proceed over the barrier point together.

(5) Communication

Only the data declared as the global data are recognized as the data in the global memory. When the global data are referred or assigned by a processing element, the compiler and operating system judge where the data are, and transfer them into the local memory of the processing element referred, or into the global memory from the processing element assigned through the data mover and crossbar network.

(6) Reference

Also, about the above-mentioned things and so on, refer to the paper" H. Miyoshi, et al., Development and achievement of NAL Numerical Wind Tunnel(NWT) for CFD computations, *Proc. Supercomputing '94* (IEEE Computer Society Press, Washington, DC, 1994) 685-692 " .

TECHNICAL REPORT OF NATIONAL
AEROSPACE LABORATORY
TR-1362T

航空宇宙技術研究所報告1362T号（欧文）

平成10年10月発行

発行所　科学技術庁航空宇宙技術研究所
東京都調布市深大寺東町7－44－1
電話（0422）47－5911　〒182－8522
印刷所　株式会社　実業公報社
東京都千代田区九段北1－7－8

Published by
NATIONAL AEROSPACE LABORATORY
7－44－1 Jindaijihigashi－machi, Chōfu, Tokyo
JAPAN

Printed in Japan