An Improved Model for Predicting HPL Performance

Chau-Yi Chou, Hsi-Ya Chang, Shuen-Tai Wang, Kuo-Chan Huang[*], and Cherng-Yeu Shen

National Center for High-Performance Computing
[*] Department of Electronic Commerce, Hsing Kuo University, Taiwan

Abstract. In this paper, we propose an improved model for predicting HPL (High performance Linpack) performance. In order to accurately predict the maximal LINPACK performance we first divide the performance model into two parts: computational cost and message passing overhead. In the message passing overhead, we adopt Xu and Hwang's broadcast model instead of the point-to-point message passing model. HPL performance prediction is a multi-variables problem. In this proposed model we improved the existing model by introducing a weighting function to account for many effects such that the proposed model could more accurately predict the maximal LINPACK performance R_{max}. This improvement in prediction accuracy has been verified on a variety of architectures, including IA64 and IA32 CPUs in a Myrinet-based environment, as well as in Quadrics, Gigabits Ethernet and other network environments. Our improved model can help cluster users in estimating the maximal HPL performance of their systems.

1 Introduction

The continuous improvement in commodity hardware and software has made cluster systems the most popular alternative [1-5] for high performance computing for both academic institutions and industries.

In 1998, Pfister [5] estimated over 100,000 cluster systems were in use worldwide. In November 2006, more than 70% of machines on the 26th Top500 List were labeled as clusters [6]. Most of these clusters used HPL (High performance Linpack) to benchmark their system performance, in accordance with the requirement of the Top500 List.

HPL utilizes LU factorization with row partial pivoting to solve a dense linear system while using a two-dimensional block-cyclic data distribution for load balance and scalability. A number of analysis models [7, 8] have been developed for HPL performance prediction for different architectures. However, these models did not consider the effect of hardware overhead, such as cache misses, pipeline startups, memory load or store and floating point arithmetic. Most models adhere to Hockney's message passing model [9] in dealing with the message interchange overhead.

In this paper we propose an improved HPL performance prediction model where we use a weighting function to account for the hardware overhead on the computation side. On the communication side we adopt Xu and Hwang's broadcast model [10]. This improved model comes up with a closer prediction of the actual performance than the other models in the literature, after a series of experiments on the Myrinet-based, Gigabits Ethernet based, IA64- and IA32-based architectures.

2 HPL Algorithm and Performance Score Model

We first introduce the HPL algorithm in Section 2.1 and then the existing HPL performance prediction model from [7] in Sections 2.2.1-2.2.5. The improved model is discussed in Section 2.2.6. Here we list the definitions of the pertinent variables in Table 1.

Table 1. Definition of the variables

Variable	Definition
B	Block size
$N \times N$	Dimension of linear system
$P \times Q$	Two dimensional map of computational processors
α	Latency of Hockney's mode (point to point), constant
β	The reciprocal of throughput of Hockney's model (point to point), constant
α'	Latency of Xu and Hwang's model (MPI broadcast), function of (PQ)
β'	The reciprocal of throughput of Xu and Hwang's model (MPI broadcast), function of (PQ)
g_3	Floating-point operation rate of matrix-matrix operations
g_2	Floating-point operation rate of matrix-vector operations
γ_3	the approximate floating-point operations per second when the processor is performing matrix-matrix operations
$\gamma = w \times \gamma_3$	The real computational performance of HPL, not including message passing overhead. w is the weighting function in our proposed performance model

2.1 HPL Algorithm

The HPL algorithm is designed to solve a linear system by LU factorization with row partial pivoting. The data are first logically partitioned into $B \times B$ blocks, and then distributed onto a two-dimensional $P \times Q$ grid, according to the block-cyclic scheme to ensure load balance as well as scalability. The block size B is for the data distribution as well as for the computational granularity. The best B value is a function of the computation-to-communication performance ratio in a system. A smaller B performs better load balance from a data distribution point of view; but when it becomes too small, it may limit the computational performance because no data reuse occurs at the higher level of the memory hierarchy from a computational point of view. The recommended B value is between 32 and 256.

At a given iteration of the main loop, each panel factorization occurs in one column of processes because of the Cartesian property of the distribution scheme. Once the panel factorization has been computed, this panel of columns is broadcast to the other process columns. The update of the trailing sub-matrix by the last panel in the look-ahead pipe is made in two phases. First, the pivots must be applied to form the current row panel U. U should then be solved by the upper triangle of the column panel. Finally U needs to be broadcast to each process row so that the local rank-B update can take place.

2.2 Performance Score Model

2.2.1 Assumption and Definition

Let the communication time to transfer L length of double precision messages be $T_c = \alpha+\beta L$, where$\alpha$and$\beta$are latency and the reciprocal of maximum bandwidth, respectively. Both $\alpha$and$\beta$are constants. Also, g_1, g_2 and g_3 are defined as the times needed for performing one floating point of the vector-vector, matrix-vector and matrix-matrix operations, respectively. With the definitions behind us, we may proceed to solve an $N\times N$ linear system.

2.2.2 Panel Factorization and Broadcast

Let us consider an $I\times J$ panel distributed over a P-process column. The execution time for panel factorization and broadcast can be approximated by:

$$T_{pfact}(I, J) = (I/P - J/3)\, J^2\, g_3 + J \ln(P)(\alpha + 2\beta J) + \alpha + \beta I\, J / P \tag{1}$$

2.2.3 Trailing Sub-matrix Update

Let's consider the update phase of an $I\times I$ trailing sub-matrix distributed on a $P\times Q$ process grid. From a computational point of view, one has to (triangular) solve I right-hand sides and to perform a local rank-J update of this trailing sub-matrix. Thus, the execution time for the update operation can be approximated by:

$$T_{update}(I, J) = g_3\, (I\, J^2/Q + 2\, I^2\, J\, /P/Q) + \alpha(\ln(P)+P-1) + 3\beta I\, J\, /Q. \tag{2}$$

2.2.4 Backward Substitution

The number of floating point operations performed during the backward substitution is given by $N^2/P/Q$. Then, the execution time of the backward substitution can be approximated by:

$$T_{backs}(N, B) = g_2\, N^2/(PQ) + N\,(\alpha/ B + 2\beta). \tag{3}$$

2.2.5 The Original HPL Performance Model

The total execution time *T* is given by:

$$
\begin{aligned}
T &= \sum_{k=0,\ B,\ 2B,\cdots}^{N} \left[T_{pfact}(N-k,B)+T_{update}(N-k-B,B)\right]+T_{backs}(N,B) \\
&= g_3\left\{\frac{2}{3PQ}N^3+\left(\frac{1}{2P}+\frac{1}{2Q}-\frac{1}{PQ}\right)BN^2+\left(\frac{2}{PQ}-\frac{1}{Q}-\frac{1}{3}\right)B^3\right\}+g_2\left\{\frac{N^2}{PQ}\right\} \\
&+\alpha\left\{N\left[\frac{(B+1)ln(P)+P+1}{B}\right]+B\,ln(P)+log(P)+P\right\} \\
&+\beta\left\{\left(\frac{3P+Q}{2PQ}\right)N^2+\left(\frac{1}{2P}+2ln(P)-\frac{3}{2Q}+2\right)BN+\left(2ln(P)-\frac{3}{Q}\right)B^2\right\}
\end{aligned}
\tag{4}
$$

The algorithm totally perform $2N^3/3 + 3\,N^2/2$ of floating point operations, Then, the performance score, hereinafter called $R_{est\_original}$, becomes:

$$
\begin{aligned}
R_{est\_original} &= \frac{2N^3/3+2N^2/2}{T} \\
&= \left\langle \frac{2N^3}{3}+\frac{3N^2}{2}\right\rangle \Big/ \left\langle
\begin{array}{l}
g_3\left\{\frac{2}{3PQ}N^3+\left(\frac{1}{2P}+\frac{1}{2Q}-\frac{1}{PQ}\right)BN^2+\left(\frac{2}{PQ}-\frac{1}{Q}-\frac{1}{3}\right)B^3\right\}+g_2\left\{\frac{N^2}{PQ}\right\} \\
+\alpha\left\{N\left[\frac{(B+1)log(P)+P+1}{B}\right]+B\,ln(P)+ln(P)+P\right\} \\
+\beta\left\{\left(\frac{3P+Q}{2PQ}\right)N^2+\left(\frac{1}{2P}+2ln(P)-\frac{3}{2Q}+2\right)BN+\left(2ln(P)-\frac{3}{Q}\right)B^2\right\}
\end{array}
\right\rangle
\end{aligned}
\tag{5}
$$

For a very large *N*, we need only to consider the dominant term in g_3, α, and β. Then, Eq.(5) becomes:

$$
R_{est\_original} = \frac{1}{\dfrac{g_3}{PQ}+\dfrac{3\alpha[(B+1)ln(P)+P]}{2\,B\,N^2}+\dfrac{3\beta(3P+Q)}{4\,N\,P\,Q}}
\tag{6}
$$

2.2.6 Our HPL Performance Model

Wang and co-workers [8] defined a new variationγ_3 as the approximate floating point operations per second when the processor is performing matrix-matrix operations. Then, $\gamma_3 = \dfrac{1}{g_3}$.

Now, we propose a weighting function *w* to include overheads such as cache misses, pipeline startups, and memory load or store. This weighting function *w* will be taken as the ratio of the time for matrix multiplication to the total HPL execution time on a single processor; and $0 \leq w \leq 1$. Next, we define a new variable $\gamma = w \times \gamma_3$ to represent the approximate floating point operations per second for the total HPL solution.

The parameters representing the communication overhead, α and β in Eq.(5) and Eq.(6), are based on Hockey's model; that is, they are constants. However, in our proposed model, we will adopt Xu and Hwang's model to account for the communication overhead. The communication time to transfer L length of double precision messages is then $T_c = \alpha' + \beta' L$, where α' and β' are latency and the reciprocal of maximum bandwidth, respectively. Now, both α' and β' are functions of the total number of processors (PQ). Therefore, the performance score of our modified HPL performance model, hereinafter call $R_{est\_modified}$, becomes:

For small size cluster, $R_{est\_modified} =$

$$\left\langle \frac{2N^3}{3} + \frac{3N^2}{2} \right\rangle / \left\langle \begin{array}{l} \frac{1}{\gamma}\left\{ \frac{2}{3PQ}N^3 + \left(\frac{1}{2P} + \frac{1}{2Q} - \frac{1}{PQ} \right)BN^2 + \left(\frac{2}{PQ} - \frac{1}{Q} - \frac{1}{3} \right)B^3 \right\} + g_2\left\{ \frac{N^2}{PQ} \right\} \\ + \alpha'\left\{ N\left[\frac{(B+1)log(P) + P + 1}{B} \right] + B\,log(P) + log(P) + P \right\} \\ + \beta'\left\{ \left(\frac{3P+Q}{2PQ} \right)N^2 + \left(\frac{1}{2P} + 2\,log(P) - \frac{3}{2Q} + 2 \right)BN + \left(2\,log(P) - \frac{3}{Q} \right)B^2 \right\} \end{array} \right\rangle \tag{7}$$

For large cluster,

$$R_{est\_modified} = \frac{1}{\frac{1}{PQ\gamma} + \frac{3\alpha'[(B+1)log(P) + P]}{2N^2B} + \frac{3\beta'(3P+Q)}{4NPQ}} \tag{8}$$

The denominator of Eq. (8) consists of three terms. The first term dominates the performance of the system if communication overhead is not considered, with the best score being $PQ\gamma$. The second and the third terms account for the communication overhead resulting from discrete computing, while α' and β' depend on the latency and bandwidth of the network for MPI collective message, respectively. In general, when the size of a cluster system increases, so do the influences of α' and β'.

3 Comparative Analysis of Different Models on Various Clusters

We now proceed to analyze the HPL performance on three different cluster systems, i.e., the Formosa Cluster [11], the Triton Cluster [12], and Dawning 4000A [13]. The Formosa cluster is equipped with IA32 CPUs and in a Gigabit Ethernet environment. The Triton Cluster uses the IA64 CPUs with Quadrics interconnection network [14]. The Dawning 4000A is a cluster of IA64 CPUs with Myrinet [15] network environment. Details of the systems are described in Sections 3.1, 3.2, and 3.3.

3.1 NCHC Formosa PC Cluster

This PC Cluster was built by the National Center for High-Performance Computing (NCHC) in September 2003. Our team had diligently optimized the system, specifically the network drive, the MTU, two network interface cards with two different private subnets, and with unused services turned off. It was the 135th on the 22th Top500 List in November 2003, and it was then the fastest computer system in Taiwan [11].

The system utilizes IBM X335 servers with Intel Xeon 2.8GHz dual processors. There are 300 CPUs connected together by a Gigabit Ethernet network. We adopted Debain 3.0 (kernel 2.6.0) operating system (OS), Intel compile 8.0 compiler, LAM/MPI 7.0.6 [16], and GOTO BLAS [17].

To compare Eq.(7) with Eq.(5), we need to first decide the parameters in these two equations. We apply the *DGEMM* function in HPL; that is, matrix multiplication of double precision random numbers of HPL, to compute the floating-point operations per second of matrix multiplication, shown in figure 1. From figure 1, we obtain$\gamma_3 = 4.6$ *GFLOPS*. Similarly, $1/g_2 = 633$ *MFLOPS*.

Next, we determine the value of the weighting function, w, by adding a timing merit of matrix multiplication in HPL software and enabling the option: `-DHPL_DETAILED_TIMING`. The output is shown as figure 2, and then $w = 516.42 / 586.77 = 0.88$.

In our previous research [18], we obtain $\alpha = 51.8\mu s$, $\beta = 0.011\mu s$, $\alpha' = 81.3154\,ln(PQ) - 63.81$, and $\beta' = 0.0193\,ln(PQ) - 0.0085$. Both α' and β' are inμs.

Table 2 lists the performance scores in *GFLOPS* of the measured R_{max} value and the $R_{est\text{-}original}$ using Eq. (5), and $R_{est\text{-}modified}$ using Eq. (7) on 4, 6, and 8 processors. It demonstrates that $R_{est\text{-}modified}$ is indeed closer to R_{max} than $R_{est\text{-}original}$.

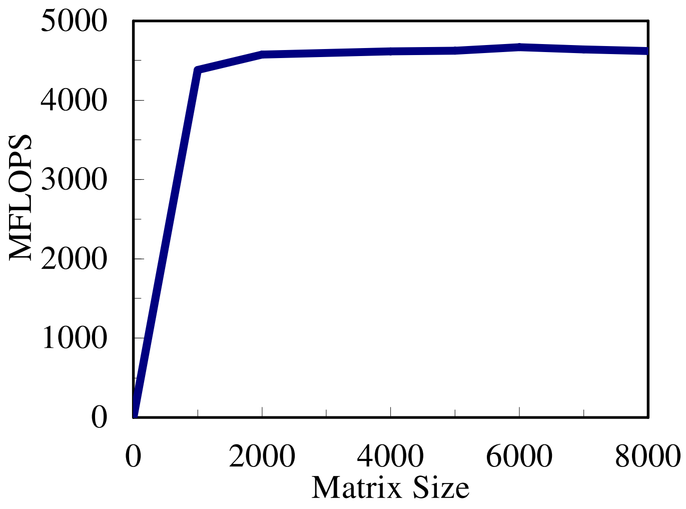

Fig. 1. MFLOPS vs. Matrix size on the Formosa Cluster

```
    T/V              N     NB     P     Q              Time           Gflops
W00L2L88  15840 88  1  1     586.77  4.516e+00
--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV--VVV-
Max aggregated wall time HPL_DGEMM. . :| 516.42|
Max aggregated wall time rfact. . . : 17.06
+ Max aggregated wall time pfact . . : 17.06
+ Max aggregated wall time mxswp . . : 0.27
Max aggregated wall time update  . . : 569.30
+ Max aggregated wall time laswp . . : 9.71
Max aggregated wall time up tr sv  . : 0.41
```

Fig. 2. The output of HPL

Table 2. Comparison of two Performance Scores in *GFLOPS* on 4-, 6-, and 8- CPUs on the Formosa Cluster

No. of Procs		R_{max}	$R_{est\text{-}original}$	$R_{est\text{-}modified}$
4	Score	16.06	17.00	15.79
	error	--	6 %	2 %
6	Score	23.47	26.98	23.47
	error	--	15%	0%
8	Score	31.51	35.96	31.02
	error	--	14%	2%

Note: R_{max} is the maximal LINPACK performance achieved.

We reported a measured R_{max} = 0.9975 *TFLOPS* to the Top500 List in October 2003. R_{max}, as defined in the Top500 List, represents the maximal LINPACK performance achieved where B = 88, N = 188000, P = 12, and Q = 25.

Table 3 lists the performance scores in *TFLOPS* of the measured R_{max} value and the $R_{est\text{-}original}$ using Eq. (6), and $R_{est\text{-}modified}$ using Eq. (8). It demonstrates that $R_{est\text{-}modified}$ of 1.05 is indeed closer to R_{max} of 0.9975.

Table 3. Comparison of two Performance Scores in *TFLOPS* on 300 CPUs on the Formosa Cluster

	R_{max}	$R_{est\text{-}original}$	$R_{est\text{-}modified}$
Score	0.9975	1.35	1.05
error	--	35 %	5 %

Note: R_{max} is the maximal LINPACK performance achieved.

3.2 NCHC Triton Cluster

This Cluster was built by NCHC in March 2005 and is currently the fastest computer system in Taiwan [12]. The system contains 384 Intel Itanium 2 1.5GHz processors (192 HP Integrity rx2600 servers) connected together by a Quadrics interconnection network, with a RedHat AS3.0 operating system and Intel compile 8.1, HP MLIB v.19B, and HP MPI v2.01 software.

As in Section 3.1, we must first determine the parameters in Eqs. (6) and (8). With a sequential static analysis and curve fitting, we obtainα= 2.48μs, α' = 20.55μs, β= 0.0040μs and β' = 0.010665μs.

R_{max} = 2.03 was measured and reported to the Top500 List with the following parameters B = 72, N = 25500, P = 12, and Q = 32.

By the *DGEMM* function in HPL, we plot figure 3 and obtainγ_3 of 5.88 *GFLOPS*.

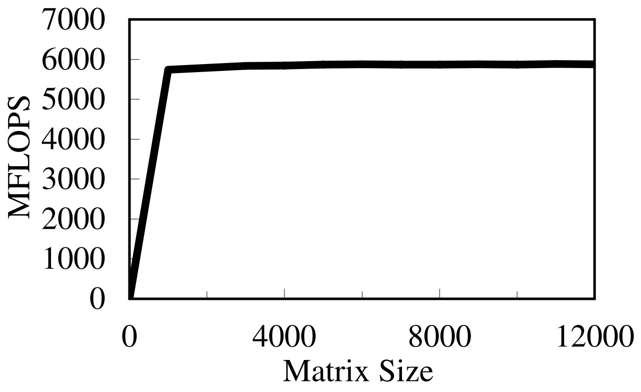

Fig. 3. MFLOPS vs. Matrix size multiplication on the Triton Cluster

Following the similar procedure in Section 3.1 gives the weighting factor w of 0.93.

Table 4 lists the performance scores of the measured R_{max} value and the scores using Eq. (6) and Eq. (8) for the Triton Cluster. It is clear that $R_{est\text{-}modified}$ yields a score of 2.07, a much better prediction than $R_{est\text{-}original}$ of 2.25 using the original model.

Table 4. Comparison of two Performance Scores in *TFLOPS* on Triton Cluster

	R_{max}	$R_{est\text{-}original}$	$R_{est\text{-}modified}$
Score	2.03	2.25	2.07
error	-	11 %	2 %

3.3 Dawning 4000A

This cluster system was ranked 10th in the 23rd Top500 List in November 2003. It contains 2560 AMD Opterons running at 2.2 *GHz* connected together by a Myrinet network. Parameters used on Eqs. (6) and (8) are: R_{max} = 8.061 *TFLOPS* and N = 728400 from the Top500 List. P = 40 and Q = 64 are assumed.

We choose an B *of* 240 from reference [19], assuming identical behavior to the AMD Opterons running at 1.6 *GHz* found in the literature (AMD 2.2 *GHz* Opteron were used in the Dawning 4000A) andγ_3 = 4.4 × 0.918 = 4.0392 *GFLOPS* [17].

The message passing overhead is assumed to be similar to the Gunawan and Cai's results [20] with a Linux platform with 64bit 66 MHz PCI; then $\alpha = 14.08\mu s$, $\alpha' = 259.79\mu s$, $\beta = 0.009\mu s$ and $\beta' = 0.11\mu s$.

Assuming that the behavior of HPL on the Dawning 4000 was similar to that of reference [19], we then calculate the weighting function w to be 0.9. The prediction results $R_{est\text{-}original}$ and $R_{est\text{-}modified}$ are listed in Table 6. Again, our improved model gives an error of 4 % versus 27 % if we use the original model.

Table 6. Comparison of two Performance Scores in *TFLOPS* on the Dawning 4000A

	R_{max}	$R_{est\text{-}original}$	$R_{est\text{-}modified}$
Score	8.061	10.28	8.417
error	-	27 %	4 %

4 Prediction of R_{max} on SIRAYA

The maximal LINPACK performance achieved R_{max} in the Top500 List depends on network communication overhead, BLAS, motherboard, PCI system, size and bandwidth of main memory, compiler, MPI-middleware. In Sections 3.1-3.3, our improved model of Eq. (8) has resulted in a better correlation with R_{max} in all three clusters: the Formosa, the Triton, and the Dawning 4000A clusters. It should be noted on the first two clusters we use the actually measured parameters, and in the cases of the last, only "estimated" parameters are used. We believe once the parameters for the last become available, the prediction results should be even more accurate.

The authors of HPL suggest that the problem size N should be about 80% of the total amount of memory in reference[7]; that is $N = 0.8 \times N_{\max}$, where $N_{\max} = \text{SQRT}(TM/8)$ is the allowable maximum problem size, TM is total memory size, reserving 20% of the total memory for system kernel overhead. In our experience, the problem sizes of the IA32-based cluster, Formosa, is quite near $N_{\max}$, and may be larger than the suggested values. On the other hand, the problem sizes for the two IA64-based platforms--both Triton and Dawning--are smaller than the suggested, where $N = 0.58 \times N_{\max}$ for the Triton and $N = 0.46 \times N_{\max}$ for the Dawning 4000A, because the IA64 based clusters need to save large memory for system kernel overhead [6].

SIRAYA is a high-performance Beowulf cluster located within the Southern Business Unit of NCHC. The cluster was designed and constructed by the 'HPC Cluster Group' at NCHC for computational science applications.

The computing nodes in SIRAYA are 80 IBM eSeries e326 in 1U cases mounted in three racks. Each IBM eSeries e326 has two AMD Opteron 275 DualCore processors running at 2.2 *GHz* with 1 *MB* of L2 cache, 4 *GB* of DDR400 registered ECC SDRAM. This means SIRAYA has 320 cores. All computers are connected together in a star topology to six stackable Nortel BayStack 5510-48T 10/100/1000 *Mbps* switches.

Based on above elaboration, we use the following parameters to predict the maximal performance score on SIRAYA. $N = 0.5 \times N_{\max} = 10^5$, $B = 240$, $w = 0.9$, $\gamma_3 = 4.0392$

GFLOPS from section 3.4, $\alpha' = 405.24\mu s$, and $\beta' = 0.10283\mu s$ from section 3.1. Then, $R_{est\text{-}modified}$ of 835.6 *GFLOPS* using Eq. (8) is very close to R_{max} of 848.2 *GFLOPS.*

Next phase, we will upgrade the system to 8 *GB* RAM for each node and fat-tree high performance network. Moreover, the system will be increased sixteen nodes. Then, the parameters become $N = 1.5 \times 10^5$. Therefore, we predict the maximal performance score on SIRAYA will be 1.37 *TFLOPS* after upgrade at the second phase.

5 Conclusion

Building on Wang's HPL performance model, we propose an improved HPL performance prediction models. Four existing clusters are used for comparing the prediction results. One of them is IA32 system and the other three are IA64 systems. The intercommunication media used in these four clusters are Myrinet, Quadrics, and Gigabit Ethernet network. In all cases, our improved model shows consistently better predictions than those using the existing model.

Our improved HPL performance prediction model would be a great help for those who wish to better understand their systems. It helps reduce the time for trial-and-error runs; it provides a user in scientific computing with useful information in predicting the performance and scalability of his own program as well.

References

1. Sterling, T., Becker, D., Savarese, D., et al.: BEOWULF: A Parallel Workstation for Scientific Computation. Proc. Of the 1995 International Conf. On Parallel Processing (1995)
2. Sterling, T., Savarese,D., Becker, D., et al.: Communication Overhead for Space Science Applications on the Beowulf Parallel Workstation. Proc. of 4th IEEE Symposium on High Performance Distributed Computing (1995)
3. Reschke, C., Sterling T. and Ridge, D.: A Design Study of Alternative Network Topologies for the Beowulf Parallel Workstation. Proceedings of the 5th IEEE Symposium on High Performance and Distributed Computing (1996)
4. Ridge, D., Becker, D. and Merkey, P.: Beowulf: Harnessing the Power of Parallelism in a Pile-of-PCs. Proceedings of IEEE Aerospace (1997)
5. Pfister, G. F.: In Search of Clusters. Prentice-Hall, Inc. (1998)
6. Top 500 List, http://www.top500.org
7. HPL Web site, http://www.netlib.org/benchmark/hpl/
8. Wang, P., Turner, G., Lauer, D., Allen, M., Simms, S., Hart, D., Papakhian, M. and Stewart, C.: LINPACK Performance on a Geographically Distributed Linux Cluster. 18th International Parallel and Distributed Processing Symposium (IPDPS'04), Santa Fe, New Mexico (2004)
9. Hockney, R. W.: The Communication Challenge for MPP: Intel Paragon and Meiko CS-2. Parallel Computing 20 (1994) 389-398
10. Xu, Z. and Hwang, K.: Modeling Communication Overhead: MPI and MPL Performance on the IBM SP2. IEEE Parallel & Distributed Technology 4(1) (1996) 9-23

11. NCHC Formosa PC Cluster Home Page, http://formosa.nchc.org.tw
12. NCHC Triton Cluster Home Page, http://www/english/pcCluster.php
13. Zhang, W., Chen, M. and Fan, J. : HPL Performance Prevision to Intending System Improvement. Second International Symposium on Parallel and Distributed Processing and Applications (2004)
14. Boden, N. J., et al.: Myrinet: A Giga-bit-per-second Local-area Network. IEEE micro (1995)
15. Burns, G.., Daoud, R. and Vaigl, J.: LAM:An Open Cluster Environment for MPI. Proceedings of Supercomputing Symposium'94 (1994) 379-386
16. Petrini, F., et al. : Performance Evaluation of the Quadrics Interconnection Network. Cluster Computing (2003)
17. GOTO library, http://www.cs.utexas.edu/users/kgoto
18. Chou, Chau-Yi, Chang, His-Ya, Wang, Shuen-Tai, Tcheng, Shou-Cheng: Modeling Message-Passing overhead on NCHC Formosa PC Cluster. GPC 2006, LNCS 3947 (2006) 299 – 307
19. Zhang, W., Fan, J. and Chen, M. : Efficient Determination of Block Size NB for Parallel Linpack Test. The 16th IASTED International Conference on Parallel and Distributed Computing and Systems (2004)
20. Gunawan, T. and Cai, W.: Performance Analysis of a Myrinet-Based Cluster. Cluster Computing 6 (2003) 229-313