NECTEC
Technical Journal
ISSN 1513-2145

วารสารวิชาการเนคเทค ปีที่ 2 ฉบับที่ 9 เดือน พฤศจิกายน 2543 – กุมภาพันธ์ 2544 Vol. II, No. 9, November 2000 – February 2001

Preface

Papers of SEAHPC2001

ECTI-21 NECTEC
National Electronics and Computer Technology Center
National Science and Technology Development Agency
สวทช. NSTDA

AMATA: Software Architecture and Implementation of High Availability Support for Beowulf Cluster

Jullawadee Maneesilp, Putchong Uthayopas
Parallel Research Group, CONSYL
Department of Computer Engineering, Faculty of Engineering,
Kasetsart University, Bangkok, Thailand
Phone: (662) 942-8555 Ext. 1416
Email: {g4265082, pu}@ku.ac.th

ABSTRACT -- High-availability support for Beowulf cluster becomes a critical factor in the acceptance of this platform for mission critical use in enterprise environment. A well defined and extensible HA software architecture is needed. This paper presents the proposed high-availability software architecture called AMATA. AMATA architecture clearly defines the software component and interaction for High Availability support. Many cases such as system software failure, registered user software error, hardware mal-function, and hardware overload can be detected and handle in a systematic way. Both discovery and recovery process can be added to provide an intelligent and automatic fault recovery process. Currently, a prototype implementation has been developed and the results obtained from that implementation have also been included.

KEY WORDS -- High availability, Beowulf clusters, Fault Tolerance,

1. Introduction and Motivation

Beowulf cluster [1] has been widely used as a scalable information server or large scientific parallel computer. Many important software and algorithm have been successful ported to this platform. Nevertheless, operating large Beowulf cluster system reliably is still a problem since most of the commodity off-the-shelf parts are not initially designed as an integral part of the highly reliable systems. PC components are less reliability than commercial server system. This problem is the main obstacle in using the cluster system for mission critical task in enterprise or industrial computing. In the commercial Unix marketplace, high availability [2] is today a key to selling server solution and virtually every Unix suppliers have their own HA software solution for customers. However, there is still a need for powerful open source software support that detects and recover from fault that allows users of Beowulf systems to construct a cost effective HA server solution for their computing needs.

In this paper, we present our high availability model and implementation for Beowulf cluster environment called AMATA. AMATA architecture define a well structure software architecture and interaction between software components for HA support in Beowulf systems. Under the framework of AMATA, software components can be built to detect major systems fault and recovery from that fault in a systematic way. Moreover, user can easily add some intelligence logic to the system to automate the detection and recovery processes.

The organization of this paper is as follows. Section 2 presents the discussion about some related works. Then, we explain briefly about our background technology that involves this work in section 3. Next in section 4 we will discuss about the design and implementation of AMATA system. Finally, we give the conclusion in section 5.

2. Background and Related Work

High-availability (HA) software can be classified into two main approaches. The first approach is to extend HA service in kernel level. Although, this approach can be very efficient, the portable implementation is not possible. Also, it is difficult to keep pace with the rapid kernel changes that happened with Linux kernel. Example of such work is Piranha [3] and Solaris-MC [4]. Solaris-MC is a prototype operating system for Solaris cluster that provides a single system image and high-availability by extending operating system abstraction across the cluster.

The second approach is to extend the high-availability service in user space. Not only does this approach has a higher portability, but also reduce the need to frequently release new kernel patches. One example is Keep alive project, which is based on VRRPd [5] implementation of VRRPv2. VRRP is a standard protocol that helps elect a master server on LAN when the old master server fail. However, VRRPd only support the high-availability in case of server fails. No solutions for user services or system services fail are provided yet. Linux FailSafe [6] is a community development effort lead by SuSE and SGI to port SGI IRIS FailSafe product to Linux. FailSafe provides a full suite of high-availability capabilities for Linux. These include full N-node cluster membership and quorum services application monitoring and failover-restart capabilities, with a set of GUI tools for administering and monitoring HA clusters. Unfortunately, only plug-in of Linux FailSafe is free. Hence, by delivering a fully available open source cluster, we can

stimulate the wide spread use of Beowulf clusters as a solution for mission critical IT needs.

3. AMATA Architecture

3.1 AMATA Structure

The proposed AMATA HA architecture is illustrated in Figure 1. AMATA architecture consists of the following components.

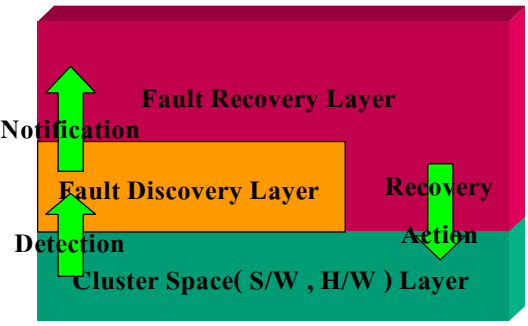


Figure 1. AMATA Architecture

Cluster Space Layer is an abstract view of cluster systems. This layer consists of 2 classed of objects, namely, ***software objects*** and ***hardware objects***. Examples of software objects are tasks, OS services, daemon, and user processes. On the other hand, hardware object is node, interface card, network, and hard disk.

Each object will have an attached component called ***object probe*** that monitors the object performance information and provides an interface for external object to access these information. Performance parameter of each object is represented by a set of counters such as CPU usage, numbers of packet send/receive and state. Currently, three types of object probe are available. Firstly, **Hardware object probe** is a module that connects with the hardware monitoring system to receive hardware information such as CPU utilization, memory usage and network traffic. Secondly, **System services object probe** is a module that monitors the system services on each node. Thirdly, **User service object probe** is used to monitor users applications.

Fault Discovery Layer is a layer receives the information from Object Probe, discover the potential fault and notify the recovery layer using event mechanism. The discovering process in this layer is based on a set of rules called *Discovery rule*. The role of theses rules is to transform a set of data received from cluster space into a set of answers. The execution of these rules takes place in part of the code called ***discovery module***.

Fault Recovery Layer this layer receive event form fault discovery layer to generate action to be taken and consists of recovery rule, recovery module and recovery action driver. The event that received from Event Service in KSIX[8,9] will pass to each event handler. We provide default action for each known event implemented with python. User can also extend their recovery rule easily by write python script follow by exist scripts.

3.2 Automatic Fault Detection and Recover Process

When any fault happened in the system, the process of automatic fault detection and recovery can be explained as a flowchart shown in Figure 2.

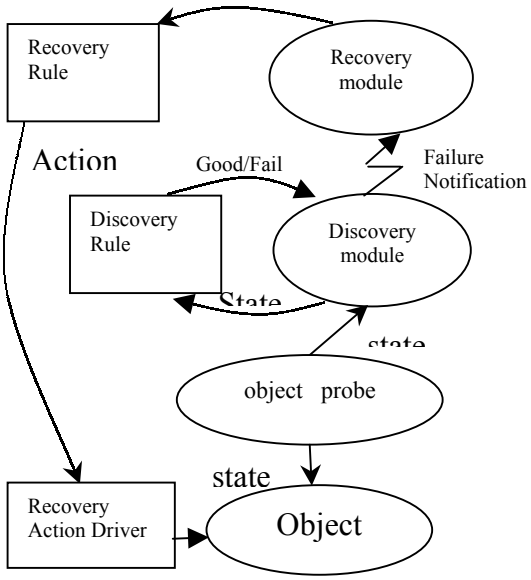


Figure 2. Recovery process

In Figure 2, the step started from system object in cluster space. Each object will be probed periodically to measure its performance state by discovery module. This state will be passed to internal discovery module, which transform it to a set of decision such as good or fail. If the failure is detected, a failure notification event will be sent to recover module. Recover module will determine an appropriate course of action using information embedded in the event and recovery rule. The action will be send to a code called recovery action driver which execute series of commands that automatically solved the problem for the user. In this process, users can add many complex and intelligent logic to system later to allow very automatic fix of the problems in cluster system.

3.3 AMATA Implementation

AMATA is implemented as a set of daemon and script that execute on each node in the system. The AMATA implementations rely on many services provided by KSIX [8] Middleware. After user boot this middleware, KSIX will automatically load AMATA as one of its services. The structure of this software system are as shown in Figure 2.

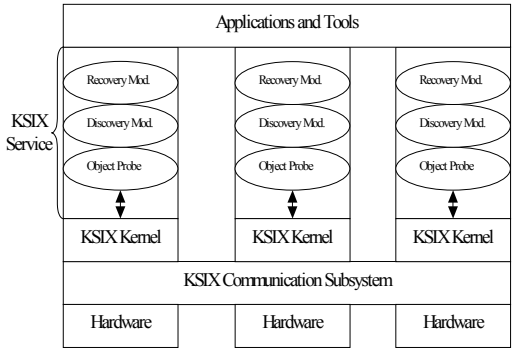


Figure 3. AMATA After KSIX boot

Object probe, discovery module and recovery module will use KSIX event service to communicate to each other. Moreover, KSIX feature called automatic restart process has been employ so that AMATA system will be automatically restarted when it fail.

One daemon called AMATA console, is used to log the notification from the fault recovery module. Once the notification has been accepted, the console daemon will start some predefined script corresponded to that event. This feature allows user to hook some logic to report error. For instance, having a script that send error message using the phone paging mechanism. User can also control this daemon by opening a socket connection to it and communicate with a simple command. The previously mentioned operation can be summarized as illustrated in Figure 3.

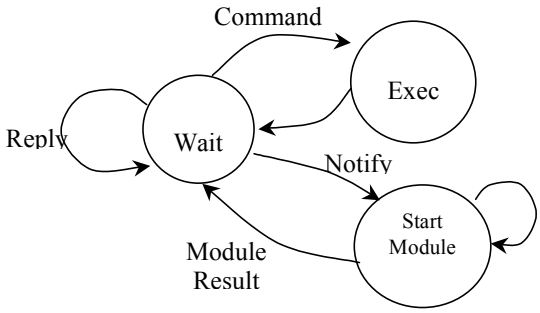


Figure 4. Console process diagram

4. Experimental Result

The cluster system used for the test consists of 4 Athlon 1 GHz and three Athlon 950 MHz with 512 Mbytes of memory per machines. These machines run Linux RedHat 6.2. They connected together with Myrinet Switch and Ethernet switch.

In the experiment, the purpose is to measure the time used to recovery from fault. In this test, we tested the system responds time in three cases. In each case, there are three services that are recovered from the artificial fault. For the first, case, we measure the recovery time when the system has no load. For the second test, a system is load with computation task. This computation load is generated from Linpack benchmark program at problem size 200. Finally, we test the system under I/O load condition. This I/O load has been simulated using the reading and writing to disk file. The elapse time is then measured starting from the termination time of the services until the time that system service has been fully recovered. The results are as reported in Table 2 and the CPU utilization has been reported in Table 3. From the results, we can see that the recover process has happened very fast. The implementation consume very low CPU usage which demonstrate that the implementation is quite efficient. The increase in I/O and CPU load do effect the recover speed but the clear result of the impact will need more study.

Table 1. Recovery Time

No. of services	No Load (sec)	CPU Load (sec)	I/O Load (sec)
1	3.960	12.782	6.840
2	4.061	19.056	17.760
3	4.794	19.458	16.249

Table 2. Percentage CPU of utilization

	No Load	CPU Load	I/O Load
%CPU of Utilization	1.5-3	0.5-1	1.5-2

5. Conclusion

The contribution of this work is mainly the new proposed concept well define architecture for HA on Linux cluster that provides a basis for real working implementation. In this model, we clearly define a modular and scalable, HA model that is easy to extend. The future work will include a better and broader implementation of each part, the addition of more intelligent logic that allows better and more automatic discover and recover of fault, the prediction of potential failure, and more study on performance and impact of external factors to the recovery process.

6. References

[1] T. Stering, D. J. Becker, D. Savarese, J. E. Dorband, U. A. Ranawake, and C. E. Packer, " Beowulf: A Parallel Workstation for Scientific Computation", In Proceedings of the International Conference on Parallel Processing 95, 1995.

[2] Harald, M, "Linux High Availability Howto", Free software Foundation, Inc., Boston, Massachusetts, USA. 1998.

[3] Kavin Railsback, "Linux Clustering in depth", Linux Magazine. August, 2000.

[4] Yousef A. Khaladi, Jose M Bernabeu, Vlada Matena, Ken Sherriff and Moti Thadanai, "Solaris MC: A multi computer OS", In the proceeding of USENIX 1996 Annual Technical Conference, San Diego, California, January 1996.

[5] Jerome Etienne, "High availability: vrrpd, usage, configuration and security", In the proceeding of Ottawa Linux Symposium 2001, Ottawa, Canada, July 24-28, 2001

[6] Lars Marowsky-Bree, "Linux FailSafe –High Availability for Linux", In the proceedings of Ottawa Linux Symposium 2000, Ottawa, Canada, July 19-22, 2000.

[7] Putchong Uthayopas, Jullawadee Maneesilp, Paricha Ingongnam, "SCMS: An Integrated Cluster Management Tool for Beowulf Cluster System", Proceedings of the International Conference on Parallel and Distributed Proceeding Techniques and Applications 2000 (PDPTA'2000) , Las Vegas, Nevada , USA , 26-28 June 2000.

[8] Thara Angskun, Putchong Uthayopas and Choopan Ratanpocha, "KSIX parallel programming environment for Beowulf Cluster", Technical Session Cluster Computing Technologies, Environments and Applications (CC-TEA), International Conference on Parallel and Distributed Proceeding Techniques and Applications 2000 (PDPTA'2000), Las Vegas, Nevada , USA , June 2000.

A STRUCTURAL STUDY OF POLYTHIOPHENE COUPLING THROUGH α AND β CARBONS: AN *AB INITIO* EVALUATION

Rolando V. Bantaculo[*a], *Arnold C. Alguno*[b], *Reynaldo M. Vequizo*[a], *Allen S. Dahili*[a], *Hitoshi Miyata*[c], *Edgar W. Ignacio*[d] *and Angelina M. Bacala*[a]

[a]*Department of Physics, MSU-IIT, 9200 Iligan City, Philippines*
[b]*College of Arts and Sciences, NORMISIST, Ampayon, 8600 Butuan City, Philippines*
[c]*Department of Physics, Niigata University, Ikarashi, Japan*
[d]*Department of Chemistry, MSU-IIT, 9200 Iligan City, Philippines*

ABSTRACT – **Detailed *ab initio* quantum mechanical calculations of a number of polythiophene oligomers are carried out to ascertain relative stability of structures bonding through α and β carbons. Energetics of dimers, trimers, tetramers, and pentamers with all possible linkages types are obtained from fully optimized geometries. This will determine the relative energy of α and β carbons linkages of the oligomers. Final energy of the oligomers is calculated using different ab-initio basis sets (3-21G and STO-3G) of the polythiophene geometry. Geometrical structures and energetics of thiophene oligomers are presented.**

KEY WORDS -- Polythiophene, α and β coupling, *ab initio,* basis set.

1. Introduction

Academic and industrial research groups around the world have shown great interest in conjugated polymers particularly polythiophene (PT) as an important class of electronic material [2]. Early studies had shown that these materials exhibit high electrical conductivities.

High conductivities, corrosion resistance, and low density are among their properties that are beginning to find applications in the fields of battery materials, electrochromic displays, electromagnetic shielding, sensor technology, non-linear optics, and molecular electronics [2, 3, 5].

Recently, a collaborative study between Niigata University of Japan and MSU-IIT of the Philippines conducted a study on the search of new semiconductor radiation detector by fabricating a radiation detector prepared through electrochemical synthesis using polythiophene doped with tetraflouroborate [4].

Silicon (Si) has been commonly used as radiation detectors [4]. It serves as vertex detectors of the interaction point and decay point of short-lived elementary particles in B-factory experiments in Kou Enerugi Kenkyushuo (KEK), Tsukuba, Japan and other major experiments in the USA and Europe. Since Si is very expensive to make into large silicon semiconductor detector, they tried to study other potent conducting polymers particularly polythiophene and polypyrrole as radiation sensors. The films produced by these polymers are strong, flat, thin, stable in the electron beam, easy to process, and entails very cheap fabrication cost compared to silicon.

Among these conjugated polymers, PT has been extensively studied because it is one of the most attractive intrinsic conductive polymers. It has good mechanical properties and environmental stability in both doped and pristine form [4]. PT films can be easily prepared through electropolymerization process. During the process coupling occurs primarily through the α carbon atoms of the heterocylic ring since these are the positions of highest unpaired electron π spin density and hence reactivity [3].

Theoretically, there are a small number of attempts to comapare α-α, α-β and β-β and they are carried out mostly in dimers of thiophene. Non α - α′ linkages (e.g. α-β′ and β-β′ couplings) can occur to variable extends, causing breaks in conjugation and hence, reduction in film conductivity. Such linkages are more profound in the later stages of electropolymerization where the unpaired electron π spin density of the α carbon atom of the oligomer approaches that of the α carbon atom. [3].

Like polypyrrole (PPy), the neutral polythiophene as observed in the IR in carbon ^{13} NMR spectra has shown that α-α′ carbon linkages predominate [1, 6]. Thus, it is assumed that the most probable coupling occurred during the electrochemical polymerization is α-α′ coupling [4]. This study aims to investigate the assumptions previously presented by experiments *via* computational analysis

* Corresponding author: tel.: +63-917-366-3423; fax: +63-63-221-4068; e-mail: rolando@physics.msuiit.edu.ph

employing *ab initio* method by calculating the final energy of the oligomers.

2. Computational Details

Intrinsic thiophene oligomers from dimer up to pentamer were investigated using *ab initio* quantum mechanical method. The *ab-initio* quantum mechanical method involves the molecular orbital calculations that employs Molecular Orbital (MO) methods based upon the Schroedinger hamiltonian expression for a multi-electron molecule (equation 1). This expression eludes exact solution, hence a variety of schemes have been made to obtain approximate solutions. For the hamiltonian H, a set of wavefunctions ψ exists that gives discrete energy solutions *E* for the molecular system. This is a classic eigenvector-eigenvalue problem, where the MO wavefunction eigenvectors correspond to the MO energy eigenvalues [11].

$$H\psi = E\psi \tag{1}$$

The Hamiltonian [12], H is the total energy operator for a system, and is written as the sum of the kinetic energy of all the componenents of the system and the internal potential energy. For a single molecule, the total Hamiltonian can be written as follows:

$$H = -\sum_{i}^{N}\frac{1}{2}\nabla_i^2 - \sum_{A}^{M}\sum_{i}^{N}\frac{Z_A}{r_{iA}} + \sum_{i>j}^{N}\frac{1}{r_{ij}} - \sum_{A}^{M}\frac{1}{2M_A}\nabla_A^2 + \sum_{A>B}^{M}\frac{Z_A Z_B}{r_{AB}} \tag{2}$$

The molecular spin orbitals $\chi_i(x)$ satisfy the eigenequation such that the Hartree Product wavefunctions are products of occupied spin orbitals, and thus an energy which is a sum of individual orbital energies, as

$$\Psi = \chi_i(x_1)\chi_j(x_2)\chi_k(x_3)\Lambda\ \chi_n(x_n) \tag{3}$$

$$\langle \Psi \mid H \mid \Psi \rangle = \varepsilon_i + \varepsilon_j + K + \varepsilon_\eta = E. \tag{4}$$

and the generalized wavefunction to give the N electron Slater determinant is,

$$\Psi = (N!)^{-1/2}\begin{vmatrix} \chi_i(x_1) & \chi_j(x_1) & K & \chi_\eta(x_1) \\ \chi_i(x_2) & \chi_j(x_2) & K & \chi_\eta(x_2) \\ M & M & O & M \\ \chi_i(x_N) & \chi_j(x_N) & K & \chi_\eta(x_N) \end{vmatrix} \tag{5}$$

Roothan's [13] contribution is to use a set of basis functions to expand the molecular wave function in terms of a set of basis functions to recast the integro-differential equations into a set of algebraic equations. These basis functions must span Hilbert space and be physically adequate. Thus the wave function looks like;

$$\vartheta_i = \sum c_{i\mu}\chi_\mu(\vec{r}_i) \tag{6}$$

$$\sum {c_{i\mu}}^2 = 1 \tag{7}$$

$$\vartheta_i\vartheta_j d\tau = 0(i \neq j); 1(i = j) \tag{8}$$

where φ_j are the basis functions and $c_{i\mu}$ are the expansion coefficients.

Now minimize the total energy with respect to the variational parameters $c_{i\nu}$ subject to orthonormality of the total wave function $\langle \chi_\mu \mid \chi_\nu \rangle = \delta_{\mu\nu}$. This constraint appears as Lagrange multipliers, written as $\varepsilon_{\mu\nu}$, in the minimization plus the expansion in terms on Hilbert Space basis functions, gives the Hartree-Fock-Roothan (HFR) algebraic equations to solve for the variational parameters $c_{i\nu}$, written as,

$$\sum_{\nu}(F_{\mu\nu} - \varepsilon_i S_{\mu\nu})c_{i\nu} = 0 \tag{9}$$

where the Fock matrix $F_{\mu\nu}$ and $D_{\mu\nu} = \sum c_{i\lambda}c_{i\sigma}$, is the density matrix and $S_{\mu\nu}$ is the overlap matrix which arises from the non-orthogonality of the basis functions [13].

In matrix form, the HFR equations are of the following form,

$$FC = SCE \tag{10}$$

where C is he expansion coefficient matrix $c_{i\nu}$ and E is the energy. To solve equation 10, transform to a standard eigen value problem, solve, and then transform back [13].

The types of integrals needed are, one electron integrals giving the overlap between different states, χ_μ and χ_ν,

$$S_{\mu\nu} = \int \chi_\mu^*(\vec{r})\chi_\mu(\vec{r}_i)d\vec{r}_i \tag{11}$$

one electron kinetic energy integrals,

$$T_{\mu\nu} = \int \chi_\mu^*(\vec{r}_1)[-\frac{1}{2}\nabla_1^2]\chi_\mu(\vec{r}_1)d\vec{r}_1 \tag{12}$$

coulomb attraction between a single electron and the nuclei,

$$V_{\mu\nu} = \int \chi_\mu^*(\vec{r}_1)[\sum\frac{Z_A}{r_{1A}}]\chi_\nu(\vec{r}_1)d\vec{r}_1 \tag{13}$$

and two electron integrals, one for the coulomb repulsion and one for the quantum mixing due to indistinguishability of particles.

$$V_{\mu\nu} \equiv (\mu\mu \mid \lambda\sigma) = \int \chi_\mu^*(\vec{r}_1)\chi_\nu^*(\vec{r}_2)\frac{1}{|\vec{r}_{12}|}\chi_\lambda(\vec{r}_1)\chi_\sigma(\vec{r}_2)d\vec{r}_1 d\vec{r}_2 \tag{14}$$

The system of HFR equations are solved iteratively and might be outlined as follows [13]:

(a) make an initial guess for c_i
(b) calculate $F_{\mu\nu}$ and $S_{\mu\nu}$
(c) solve HFR equations for ε_i and $c_{\mu\nu}$
(d) repeat steps (a)-(d) until ε_i and/or c_i converge.

The form of the wavefunction ψ varies with the level of approximation used. It is very common for the linear combination of atomic orbitals (LCAO) approximation to be used, such as all ψ's are made by combinations of Atomic Orbitals (AOs) from the constituent atoms of the molecule. The set of AOs used to make up the MOs is called the basis set. Linear combinations of the AOs give a number of MOs equal to the number of basis set orbitals, where the MO eigenvectors form an orthonormal set according to the equations (equations 6, 7, 8) [11].

However, the actual mathematical treatment is more general than this, and any set of appropriately defined functions may be used. The basis set is a mathematical description of the orbitals within a system (which in turn combine to approximate the total electronic wavefunction) used to perform the theoretical calculations. Standard basis sets for electronic structure calculations used linear combinations of gaussian functions equation

$$g(\alpha,\vec{r}) = cx^n y^m z^l e^{-\alpha r^2} \qquad (15)$$

to form the orbital equation [10]

$$\varphi_i = \sum_{\mu=1}^{N} c_{\mu i}\chi_\mu . \qquad (16)$$

For *ab initio* Molecular Orbital (MO) calculations, the minimal level of basis set (termed single-zeta) uses both core and valence AOs. For ease of computational integration, almost all modern *ab initio* computations approximate AOs as summations of gaussian type orbital (GTO) functions (equation 17). For higher-level work, complex basis sets have been devised, using two or more shells composed of summations of gaussian functions in order to simulate each occupied shell of an atom (and often even the higher-lying empty shells) [11]

$$\psi_{GTO} = \sum c_k e^{a_k r^2} . \qquad (17)$$

The rationale behind using complex basis sets for *ab-initio* computations is that *any* approximate set of MO eigenvectors ψ_i will yield a molecular energy that lies above the "*true*" energy. This is due to the variational theorem, which states that E_{approx} is greater than E_{actual} in (equation 1) for nonapproximate hamiltonian expressions H (in this case the nonrelativistic, time-independent hamiltonian is appropriate). The greater the flexibility of the basis set, the greater the flexibility in the approximate MOs ψ_i, and the closer E_{approx} will come to E_{actual}. The cost for this greater level of accuracy is an increase in the time required to run a computation, and an increased complexity in interpreting the final result. These time constraints can be very substantial for either medium to large molecules, or for large basis set computations. Therefore, *ab initio* theory is practically usable only for certain types of problems in materials chemistry, even with the present state of the art of fast programming algorithms and ever-faster computers to run them [11].

In this study, a minimum basis set of STO-3G was initially carried out. This minimal basis set contains the minimum number of basis functions needed for each atom [10]. It is used to fixed-size atomic-type orbitals with three gaussians primitives per basis function of this Slater-type-orbital that approximates with gaussian functions. Furthermore, a larger basis set was employed with split valence (3-21G) by increasing the number of basis sets per atom. It allows orbitals to change size, but not to change the shape of the oligomers. Larger basis sets more accurately approximate the orbitals by imposing fewer restrictions on the locations of the electrons in space [10].

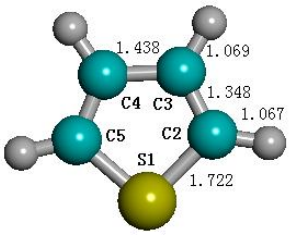
All calculations were initially calculated on semi-empirical method using HyperChem [7] suite of program and finally carried out using GAMESS [8] system of programs running at Sun Sparc station and Beowulf cluster. Molecular symmetry was applied throughout the entire program.

3. Results and Discussion

Geometries for thiophene monomer, dimers, trimers, tetramers and pentamers have been optimized at Hartree-Fock level using STO-3G and 3-21G basis sets. Some of the different types of coupling from dimer up to pentamer are shown in Figure 2. As observed, pure α-α′ linkages showed a planar conformation and linear chains, α-β′ bonding showed slightly linear but purely planar and β-β′ couplings showed a kink structure.

As shown in Figure 3, the coupling involving linear α-α′ structures manifested the lowest energy structure and linkages involving α-β′ and β-β′ in all oligomers showed a higher energy structure. However, some of the structures were almost energetically degenerate to the lowest energy, for example αα - αβ, βα - αα - αα and αα - αα - αβ - αα.

It turned out that the ground state of the resulting oligomers depend mainly on the number of α or β type terminating monomers but not on how they were ordered. For example, if one of the monomers was connected to the central one by its α carbon and the others through their β carbons, the other possible combinations were almost energetically degenerate.



Bond angles
$\angle C_3C_2C_1$=112.383°
$\angle S_4C_3C_2$=111.929°
$\angle H_2C_1C_2$=123.692°
$\angle H_8C_3C_2$=127.093°

Figure 1. Structures of thiophene monomer optimized using 3-21G-basis set. Bond distances are given in angstrom (Å).

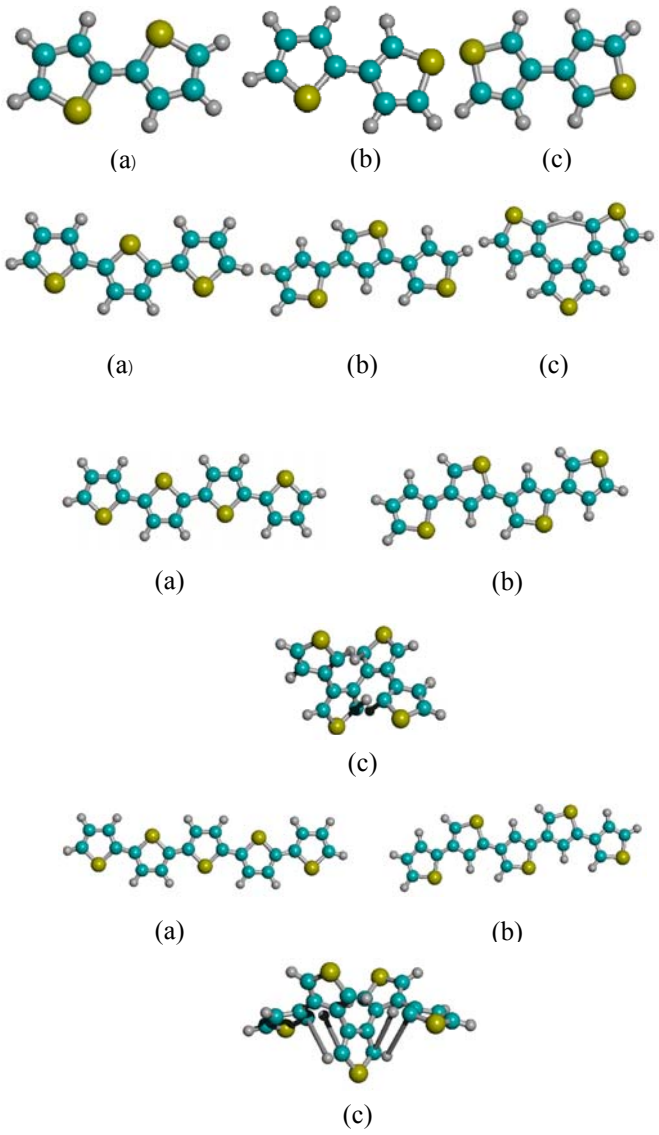


Figure 2. Optimized geometry structure of (a) αα-linear (b) αβ-planar (c) ββ-kink for dimers, trimers, tetramers, pentamers.

Table 1. Relative energies (in eV) of linear combination of dimer, trimer, tetramer and pentamer ground states.

Dimer

No.	Coupling Sequence	STO-3G	3-21G
1	αα	0.000	0.000
2	αβ	0.107	0.040
3	βα	0.108	0.040
4	ββ	0.237	0.088

Trimer

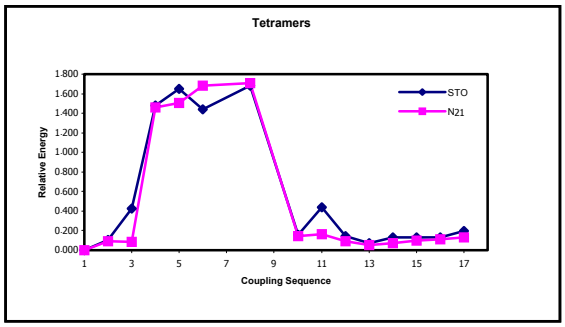
No.	Coupling Sequence	*STO-3G*	3-21G
1	αα - αα	0.000	0.000
2	αα - αβ	0.074	0.043
3	αα - βα	0.088	0.037
4	αα - ββ	0.155	0.062
5	αβ - αα	0.098	0.065
6	αβ - αβ	0.159	0.097
7	αβ - βα	1.024	1.738
8	αβ - ββ	1.041	1.669
9	βα - αα	0.071	0.017
10	βα - βα	0.159	0.097
11	βα - ββ	0.250	0.152
12	ββ - αα	0.244	0.171
13	ββ - ββ	0.418	0.067

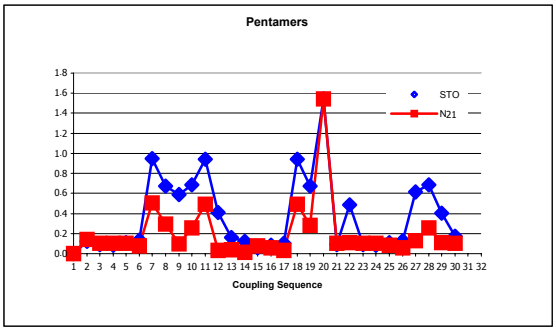
Tetramer

No.	Coupling Sequence	*STO-3G*	3-21G
1	αα - αα - αα - αα	0.000	0.000
2	αβ - αα - αα - αα	0.122	0.143
3	αα - αβ - αα - αα	0.089	0.105
4	αα - αα - αβ - αα	0.086	0.102
5	αβ - αβ - αα - αα	0.111	0.101
6	αβ - αβ - αβ - αβ	0.135	0.074
7	ββ - ββ - ββ - ββ	0.949	0.508
8	βα - ββ - ββ - ββ	0.671	0.297
9	ββ - βα - ββ - ββ	0.591	0.094
10	ββ - ββ - βα - ββ	0.688	0.259
11	ββ - ββ - ββ - βα	0.945	0.496
12	βα - βα - ββ - ββ	0.411	0.032
13	βα - βα - βα - ββ	0.161	0.037
14	βα - βα - βα - βα	0.125	0.010
15	βα - αα - αα - αα	0.060	0.079
16	βα - βα - αα - αα	0.082	0.055
17	βα - βα - βα - αα	0.104	0.033
18	αβ - ββ - ββ - ββ	0.945	0.495
19	αβ - αβ - ββ - ββ	0.671	0.283
20	αβ - αβ - αβ - ββ	1.543	1.543
21	αα - αα - αα - βα	0.087	0.105
22	ββ - ββ - ββ - αβ	0.487	0.112
23	αα - αα - βα - αα	0.089	0.105
24	αα - βα - αα - αα	0.086	0.102
25	αα - αα - βα - βα	0.110	0.082
26	αα - βα - βα - βα	0.129	0.057
27	ββ - ββ - αβ - ββ	0.613	0.126
28	ββ - αβ - ββ - ββ	0.688	0.259
29	ββ - ββ - αβ - αβ	0.401	0.108
30	ββ - αβ - αβ - αβ	0.172	0.101

Pentamer

No.	Coupling Sequence	*STO-3G*	3-21G
1	αα - αα - αα	0.000	0.000
2	αα - βα - αβ	0.103	0.093
3	αα - ββ - βα	0.427	0.083
4	αβ - βα - αβ	1.480	1.462
5	αβ - ββ - αα	1.650	1.505
6	αβ - βα - αα	1.440	1.682
7	αβ - ββ - αβ	1.680	1.711
8	αα - ββ - αα	0.156	0.144
9	αα - ββ - ββ	0.438	0.166
10	αβ - αβ - αβ	0.147	0.092
11	βα - αα - αβ	0.073	0.054
12	βα - αβ - αβ	0.128	0.072
13	βα - βα - αβ	0.133	0.098
14	ββ - αα - αα	0.130	0.112
15	ββ - αα - ββ	0.197	0.128





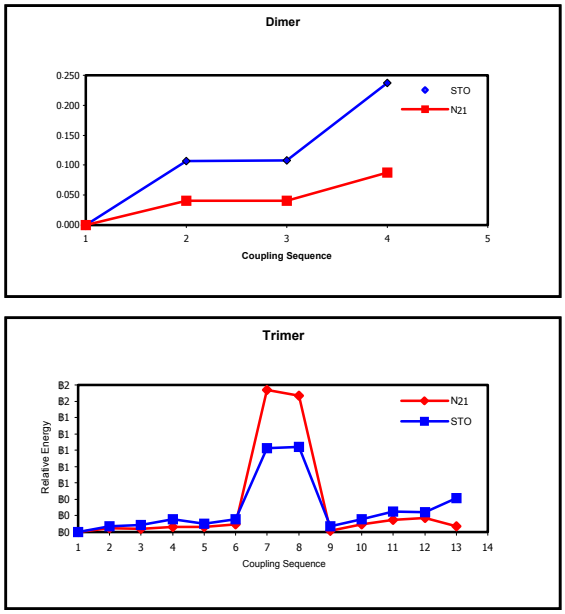


Figure 3. Plot of the relative energy vs coupling sequence of dimer, trimer, tetramer and pentamer

4. Conclusion

The addition of a thiophene to a polythiophene backbone can be achieved in a large number of ways depending on the position along the chain as well as the orientation of the monomer. The relative energies of the possible structures can then be predicted by counting the types of thiophene rings. Finally, it was observed that α - α′ coupling has the lowest final energy among the thiophene oligomers; thus it is the most stable coupling**.**

Acknowledgments

We thank the Commission on Higher Education – Mindanao Advanced Education Project (CHED-MAEP) for the financial support. Many thanks to Engr. Wilfredo Chung, Prof. Ronald Pascual and Mr. Marvin Fernandez for many fruitful discussions and suggestions. Likewise, thank is due to MSU-IIT Computing and Network Center for the generous allocation of computer resources. Thank is due to Mr. Dante D. Dinawanao for the many help extended.

References

[1] Xun-Zuan Yang and J. Chien, "Autooxidation and Stabilization of Undoped and Doped Polyacetylenes", *Journal of Polymer Science: Polymer Chemistry Edition*, Vol. 23, pp 856-878 (1995).

[2] A. Castillon, "Study on the New Organic Semiconductor Radiation Detector", *Materal Thesis*, MSU-IIT, Iligan Cty, Unpublished. (2000).

[3] J. Roncalia, "Conjugated Poly(thiophenes): Synthesis, Functionalizations, and Applications", *Chem Rev.* 1992, 97, pp 711-738 (1992).

[4] M. Yurtsener and E. Yurtsener." Structural Studies of Polypyrroles: An *Ab Initio* Evaluation of Bonding Through α and β carbons", *Synthetic Metals* 98 ,1999, pp 221-227 (1999).

[5] Kwon, O. and McKee, M.L. *J. Phys. Chem.* B **2000**, 104, 1686-1694.

[6] J. Roncalia, *Chem. Rev.* 1992, 42, 711-738.

[7] Hyperchem ™Release[3]. *Windows Molecular Modelling System*, Copyright ©1993, HyperCube, Inc. and Autodesk, Inc. Developed by HyperCube, Inc.

[8] GAMESS Version = 22 Nov 1995 from Iowa State University, M.W.Schmidt, K.K.Baldridge, J.A.Boatz, S.T.Elbert, M.S.Gordon, J.H.Jensen, S.Koseki, N.Matsunaga, K.A.Nguyen, S.J.Su, T.L.Windus, Together With M.Dupuis, J.A.Montgomer. *J. Comp.Chem.* 14, 1347-1363 (1993).

[9] P. Walters, M. Stahl, *BABEL Program (version 1.1)* Copyright ©1992, 93, 94, Dolota Research Group, Department of Chemistry, University of Arizona.

[10] Foresman, J. B. and Frisch, Æ., *Exploring Chemistry with Electronic Structure Methods, 2nd ed.* Gaussian, Inc., Pittsburgh, PA. 1995.

[11] http://www.chem.umass.edu /~lahti/ARTICLE /comp txt.com

[12] http://zopyros.ccqc.ga.edu/lec_top/hf/node1-5.html

[13] http://www.ncsc.org/training/materials/Software_ Release/foundations/qmnotes/SECTION001100000 00000000000

A STRUCTURAL STUDY OF POLYTHIOPHENE COUPLING THROUGH α AND β CARBONS: AN *AB INITIO* EVALUATION

Rolando V. Bantaculo[*a], *Arnold C. Alguno*[b], *Reynaldo M. Vequizo*[a], *Allen S. Dahili*[a], *Hitoshi Miyata*[c], *Edgar W. Ignacio*[d] *and Angelina M. Bacala*[a]

[a]*Department of Physics, MSU-IIT, 9200 Iligan City, Philippines*
[b]*College of Arts and Sciences, NORMISIST, Ampayon, 8600 Butuan City, Philippines*
[c]*Department of Physics, Niigata University, Ikarashi, Japan*
[d]*Department of Chemistry, MSU-IIT, 9200 Iligan City, Philippines*

ABSTRACT – **Detailed *ab initio* quantum mechanical calculations of a number of polythiophene oligomers are carried out to ascertain relative stability of structures bonding through α and β carbons. Energetics of dimers, trimers, tetramers, and pentamers with all possible linkages types are obtained from fully optimized geometries. This will determine the relative energy of α and β carbons linkages of the oligomers. Final energy of the oligomers is calculated using different ab-initio basis sets (3-21G and STO-3G) of the polythiophene geometry. Geometrical structures and energetics of thiophene oligomers are presented.**

KEY WORDS -- Polythiophene, α and β coupling, *ab initio,* basis set.

1. Introduction

Academic and industrial research groups around the world have shown great interest in conjugated polymers particularly polythiophene (PT) as an important class of electronic material [2]. Early studies had shown that these materials exhibit high electrical conductivities.

High conductivities, corrosion resistance, and low density are among their properties that are beginning to find applications in the fields of battery materials, electrochromic displays, electromagnetic shielding, sensor technology, non-linear optics, and molecular electronics [2, 3, 5].

Recently, a collaborative study between Niigata University of Japan and MSU-IIT of the Philippines conducted a study on the search of new semiconductor radiation detector by fabricating a radiation detector prepared through electrochemical synthesis using polythiophene doped with tetraflouroborate [4].

Silicon (Si) has been commonly used as radiation detectors [4]. It serves as vertex detectors of the interaction point and decay point of short-lived elementary particles in B-factory experiments in Kou Enerugi Kenkyushuo (KEK), Tsukuba, Japan and other major experiments in the USA and Europe. Since Si is very expensive to make into large silicon semiconductor detector, they tried to study other potent conducting polymers particularly polythiophene and polypyrrole as radiation sensors. The films produced by these polymers are strong, flat, thin, stable in the electron beam, easy to process, and entails very cheap fabrication cost compared to silicon.

Among these conjugated polymers, PT has been extensively studied because it is one of the most attractive intrinsic conductive polymers. It has good mechanical properties and environmental stability in both doped and pristine form [4]. PT films can be easily prepared through electropolymerization process. During the process coupling occurs primarily through the α carbon atoms of the heterocylic ring since these are the positions of highest unpaired electron π spin density and hence reactivity [3].

Theoretically, there are a small number of attempts to comapare α-α, α-β and β-β and they are carried out mostly in dimers of thiophene. Non α - α′ linkages (e.g. α-β′ and β-β′ couplings) can occur to variable extends, causing breaks in conjugation and hence, reduction in film conductivity. Such linkages are more profound in the later stages of electropolymerization where the unpaired electron π spin density of the α carbon atom of the oligomer approaches that of the α carbon atom. [3].

Like polypyrrole (PPy), the neutral polythiophene as observed in the IR in carbon ^{13} NMR spectra has shown that α-α′ carbon linkages predominate [1, 6]. Thus, it is assumed that the most probable coupling occurred during the electrochemical polymerization is α-α′ coupling [4]. This study aims to investigate the assumptions previously presented by experiments *via* computational analysis

* Corresponding author: tel.: +63-917-366-3423; fax: +63-63-221-4068; e-mail: rolando@physics.msuiit.edu.ph

employing *ab initio* method by calculating the final energy of the oligomers.

2. Computational Details

Intrinsic thiophene oligomers from dimer up to pentamer were investigated using *ab initio* quantum mechanical method. The *ab-initio* quantum mechanical method involves the molecular orbital calculations that employs Molecular Orbital (MO) methods based upon the Schroedinger hamiltonian expression for a multi-electron molecule (equation 1). This expression eludes exact solution, hence a variety of schemes have been made to obtain approximate solutions. For the hamiltonian H, a set of wavefunctions ψ exists that gives discrete energy solutions *E* for the molecular system. This is a classic eigenvector-eigenvalue problem, where the MO wavefunction eigenvectors correspond to the MO energy eigenvalues [11].

$$H\psi = E\psi \tag{1}$$

The Hamiltonian [12], H is the total energy operator for a system, and is written as the sum of the kinetic energy of all the componenents of the system and the internal potential energy. For a single molecule, the total Hamiltonian can be written as follows:

$$H = -\sum_{i}^{N}\frac{1}{2}\nabla_i^2 - \sum_{A}^{M}\sum_{i}^{N}\frac{Z_A}{r_{iA}} + \sum_{i>j}^{N}\frac{1}{r_{ij}} - \sum_{A}^{M}\frac{1}{2M_A}\nabla_A^2 + \sum_{A>B}^{M}\frac{Z_A Z_B}{r_{AB}} \tag{2}$$

The molecular spin orbitals $\chi_i(x)$ satisfy the eigenequation such that the Hartree Product wavefunctions are products of occupied spin orbitals, and thus an energy which is a sum of individual orbital energies, as

$$\Psi = \chi_i(x_1)\chi_j(x_2)\chi_k(x_3)\Lambda\ \chi_n(x_n) \tag{3}$$

$$\langle \Psi \mid H \mid \Psi \rangle = \varepsilon_i + \varepsilon_j + K + \varepsilon_\eta = E. \tag{4}$$

and the generalized wavefunction to give the N electron Slater determinant is,

$$\Psi = (N!)^{-1/2}\begin{vmatrix} \chi_i(x_1) & \chi_j(x_1) & K & \chi_\eta(x_1) \\ \chi_i(x_2) & \chi_j(x_2) & K & \chi_\eta(x_2) \\ M & M & O & M \\ \chi_i(x_N) & \chi_j(x_N) & K & \chi_\eta(x_N) \end{vmatrix} \tag{5}$$

Roothan's [13] contribution is to use a set of basis functions to expand the molecular wave function in terms of a set of basis functions to recast the integro-differential equations into a set of algebraic equations. These basis functions must span Hilbert space and be physically adequate. Thus the wave function looks like;

$$\vartheta_i = \sum c_{i\mu}\chi_\mu(\vec{r}_i) \tag{6}$$

$$\sum {c_{i\mu}}^2 = 1 \tag{7}$$

$$\vartheta_i\vartheta_j d\tau = 0(i \neq j); 1(i = j) \tag{8}$$

where φ_j are the basis functions and $c_{i\mu}$ are the expansion coefficients.

Now minimize the total energy with respect to the variational parameters $c_{i\nu}$ subject to orthonormality of the total wave function $\langle \chi_\mu \mid \chi_\nu \rangle = \delta_{\mu\nu}$. This constraint appears as Lagrange multipliers, written as $\varepsilon_{\mu\nu}$, in the minimization plus the expansion in terms on Hilbert Space basis functions, gives the Hartree-Fock-Roothan (HFR) algebraic equations to solve for the variational parameters $c_{i\nu}$, written as,

$$\sum_{\nu}(F_{\mu\nu} - \varepsilon_i S_{\mu\nu})c_{i\nu} = 0 \tag{9}$$

where the Fock matrix $F_{\mu\nu}$ and $D_{\mu\nu} = \sum c_{i\lambda}c_{i\sigma}$, is the density matrix and $S_{\mu\nu}$ is the overlap matrix which arises from the non-orthogonality of the basis functions [13].

In matrix form, the HFR equations are of the following form,

$$FC = SCE \tag{10}$$

where C is he expansion coefficient matrix $c_{i\nu}$ and E is the energy. To solve equation 10, transform to a standard eigen value problem, solve, and then transform back [13].

The types of integrals needed are, one electron integrals giving the overlap between different states, χ_μ and χ_ν,

$$S_{\mu\nu} = \int \chi_\mu^*(\vec{r})\chi_\mu(\vec{r}_i)d\vec{r}_i \tag{11}$$

one electron kinetic energy integrals,

$$T_{\mu\nu} = \int \chi_\mu^*(\vec{r}_1)[-\frac{1}{2}\nabla_1^2]\chi_\mu(\vec{r}_1)d\vec{r}_1 \tag{12}$$

coulomb attraction between a single electron and the nuclei,

$$V_{\mu\nu} = \int \chi_\mu^*(\vec{r}_1)[\sum\frac{Z_A}{r_{1A}}]\chi_\nu(\vec{r}_1)d\vec{r}_1 \tag{13}$$

and two electron integrals, one for the coulomb repulsion and one for the quantum mixing due to indistinguishability of particles.

$$V_{\mu\nu} \equiv (\mu\mu \mid \lambda\sigma) = \int \chi_\mu^*(\vec{r}_1)\chi_\nu^*(\vec{r}_2)\frac{1}{|\vec{r}_{12}|}\chi_\lambda(\vec{r}_1)\chi_\sigma(\vec{r}_2)d\vec{r}_1 d\vec{r}_2 \tag{14}$$

The system of HFR equations are solved iteratively and might be outlined as follows [13]:

(a) make an initial guess for c_i
(b) calculate $F_{\mu\nu}$ and $S_{\mu\nu}$
(c) solve HFR equations for ε_i and $c_{\mu\nu}$
(d) repeat steps (a)-(d) until ε_i and/or c_i converge.

The form of the wavefunction ψ varies with the level of approximation used. It is very common for the linear combination of atomic orbitals (LCAO) approximation to be used, such as all ψ's are made by combinations of Atomic Orbitals (AOs) from the constituent atoms of the molecule. The set of AOs used to make up the MOs is called the basis set. Linear combinations of the AOs give a number of MOs equal to the number of basis set orbitals, where the MO eigenvectors form an orthonormal set according to the equations (equations 6, 7, 8) [11].

However, the actual mathematical treatment is more general than this, and any set of appropriately defined functions may be used. The basis set is a mathematical description of the orbitals within a system (which in turn combine to approximate the total electronic wavefunction) used to perform the theoretical calculations. Standard basis sets for electronic structure calculations used linear combinations of gaussian functions equation

$$g(\alpha,\vec{r}) = cx^n y^m z^l e^{-\alpha r^2} \qquad (15)$$

to form the orbital equation [10]

$$\varphi_i = \sum_{\mu=1}^{N} c_{\mu i}\chi_\mu . \qquad (16)$$

For *ab initio* Molecular Orbital (MO) calculations, the minimal level of basis set (termed single-zeta) uses both core and valence AOs. For ease of computational integration, almost all modern *ab initio* computations approximate AOs as summations of gaussian type orbital (GTO) functions (equation 17). For higher-level work, complex basis sets have been devised, using two or more shells composed of summations of gaussian functions in order to simulate each occupied shell of an atom (and often even the higher-lying empty shells) [11]

$$\psi_{GTO} = \sum c_k e^{a_k r^2} . \qquad (17)$$

The rationale behind using complex basis sets for *ab-initio* computations is that *any* approximate set of MO eigenvectors ψ_i will yield a molecular energy that lies above the "*true*" energy. This is due to the variational theorem, which states that E_{approx} is greater than E_{actual} in (equation 1) for nonapproximate hamiltonian expressions H (in this case the nonrelativistic, time-independent hamiltonian is appropriate). The greater the flexibility of the basis set, the greater the flexibility in the approximate MOs ψ_i, and the closer E_{approx} will come to E_{actual}. The cost for this greater level of accuracy is an increase in the time required to run a computation, and an increased complexity in interpreting the final result. These time constraints can be very substantial for either medium to large molecules, or for large basis set computations. Therefore, *ab initio* theory is practically usable only for certain types of problems in materials chemistry, even with the present state of the art of fast programming algorithms and ever-faster computers to run them [11].

In this study, a minimum basis set of STO-3G was initially carried out. This minimal basis set contains the minimum number of basis functions needed for each atom [10]. It is used to fixed-size atomic-type orbitals with three gaussians primitives per basis function of this Slater-type-orbital that approximates with gaussian functions. Furthermore, a larger basis set was employed with split valence (3-21G) by increasing the number of basis sets per atom. It allows orbitals to change size, but not to change the shape of the oligomers. Larger basis sets more accurately approximate the orbitals by imposing fewer restrictions on the locations of the electrons in space [10].

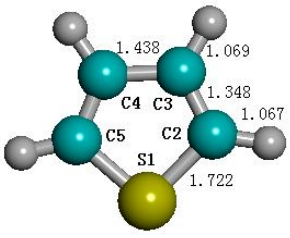
All calculations were initially calculated on semi-empirical method using HyperChem [7] suite of program and finally carried out using GAMESS [8] system of programs running at Sun Sparc station and Beowulf cluster. Molecular symmetry was applied throughout the entire program.

3. Results and Discussion

Geometries for thiophene monomer, dimers, trimers, tetramers and pentamers have been optimized at Hartree-Fock level using STO-3G and 3-21G basis sets. Some of the different types of coupling from dimer up to pentamer are shown in Figure 2. As observed, pure α-α′ linkages showed a planar conformation and linear chains, α-β′ bonding showed slightly linear but purely planar and β-β′ couplings showed a kink structure.

As shown in Figure 3, the coupling involving linear α-α′ structures manifested the lowest energy structure and linkages involving α-β′ and β-β′ in all oligomers showed a higher energy structure. However, some of the structures were almost energetically degenerate to the lowest energy, for example αα - αβ, βα - αα - αα and αα - αα - αβ - αα.

It turned out that the ground state of the resulting oligomers depend mainly on the number of α or β type terminating monomers but not on how they were ordered. For example, if one of the monomers was connected to the central one by its α carbon and the others through their β carbons, the other possible combinations were almost energetically degenerate.



Bond angles
$\angle C_3C_2C_1$=112.383°
$\angle S_4C_3C_2$=111.929°
$\angle H_2C_1C_2$=123.692°
$\angle H_8C_3C_2$=127.093°

Figure 1. Structures of thiophene monomer optimized using 3-21G-basis set. Bond distances are given in angstrom (Å).

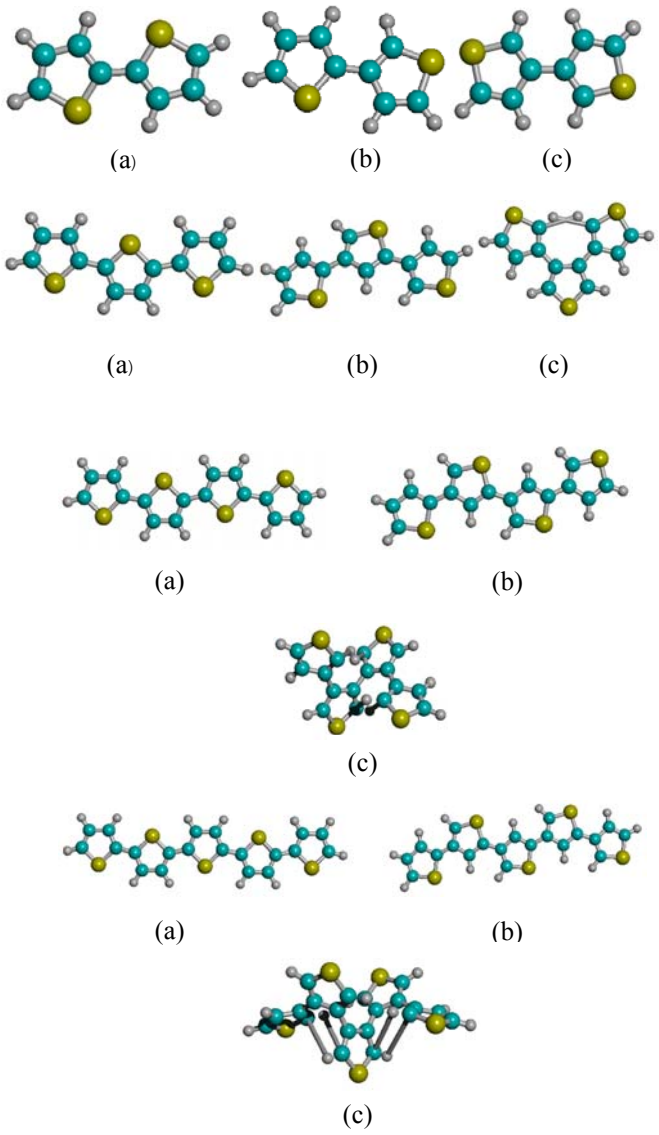


Figure 2. Optimized geometry structure of (a) αα-linear (b) αβ-planar (c) ββ-kink for dimers, trimers, tetramers, pentamers.

Table 1. Relative energies (in eV) of linear combination of dimer, trimer, tetramer and pentamer ground states.

Dimer

No.	Coupling Sequence	STO-3G	3-21G
1	αα	0.000	0.000
2	αβ	0.107	0.040
3	βα	0.108	0.040
4	ββ	0.237	0.088

Trimer

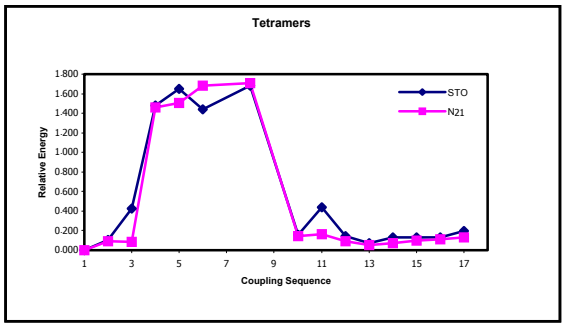
No.	Coupling Sequence	*STO-3G*	3-21G
1	αα - αα	0.000	0.000
2	αα - αβ	0.074	0.043
3	αα - βα	0.088	0.037
4	αα - ββ	0.155	0.062
5	αβ - αα	0.098	0.065
6	αβ - αβ	0.159	0.097
7	αβ - βα	1.024	1.738
8	αβ - ββ	1.041	1.669
9	βα - αα	0.071	0.017
10	βα - βα	0.159	0.097
11	βα - ββ	0.250	0.152
12	ββ - αα	0.244	0.171
13	ββ - ββ	0.418	0.067

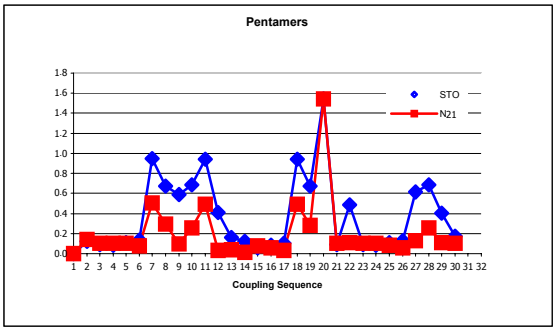
Tetramer

No.	Coupling Sequence	*STO-3G*	3-21G
1	αα - αα - αα - αα	0.000	0.000
2	αβ - αα - αα - αα	0.122	0.143
3	αα - αβ - αα - αα	0.089	0.105
4	αα - αα - αβ - αα	0.086	0.102
5	αβ - αβ - αα - αα	0.111	0.101
6	αβ - αβ - αβ - αβ	0.135	0.074
7	ββ - ββ - ββ - ββ	0.949	0.508
8	βα - ββ - ββ - ββ	0.671	0.297
9	ββ - βα - ββ - ββ	0.591	0.094
10	ββ - ββ - βα - ββ	0.688	0.259
11	ββ - ββ - ββ - βα	0.945	0.496
12	βα - βα - ββ - ββ	0.411	0.032
13	βα - βα - βα - ββ	0.161	0.037
14	βα - βα - βα - βα	0.125	0.010
15	βα - αα - αα - αα	0.060	0.079
16	βα - βα - αα - αα	0.082	0.055
17	βα - βα - βα - αα	0.104	0.033
18	αβ - ββ - ββ - ββ	0.945	0.495
19	αβ - αβ - ββ - ββ	0.671	0.283
20	αβ - αβ - αβ - ββ	1.543	1.543
21	αα - αα - αα - βα	0.087	0.105
22	ββ - ββ - ββ - αβ	0.487	0.112
23	αα - αα - βα - αα	0.089	0.105
24	αα - βα - αα - αα	0.086	0.102
25	αα - αα - βα - βα	0.110	0.082
26	αα - βα - βα - βα	0.129	0.057
27	ββ - ββ - αβ - ββ	0.613	0.126
28	ββ - αβ - ββ - ββ	0.688	0.259
29	ββ - ββ - αβ - αβ	0.401	0.108
30	ββ - αβ - αβ - αβ	0.172	0.101

Pentamer

No.	Coupling Sequence	*STO-3G*	3-21G
1	αα - αα - αα	0.000	0.000
2	αα - βα - αβ	0.103	0.093
3	αα - ββ - βα	0.427	0.083
4	αβ - βα - αβ	1.480	1.462
5	αβ - ββ - αα	1.650	1.505
6	αβ - βα - αα	1.440	1.682
7	αβ - ββ - αβ	1.680	1.711
8	αα - ββ - αα	0.156	0.144
9	αα - ββ - ββ	0.438	0.166
10	αβ - αβ - αβ	0.147	0.092
11	βα - αα - αβ	0.073	0.054
12	βα - αβ - αβ	0.128	0.072
13	βα - βα - αβ	0.133	0.098
14	ββ - αα - αα	0.130	0.112
15	ββ - αα - ββ	0.197	0.128





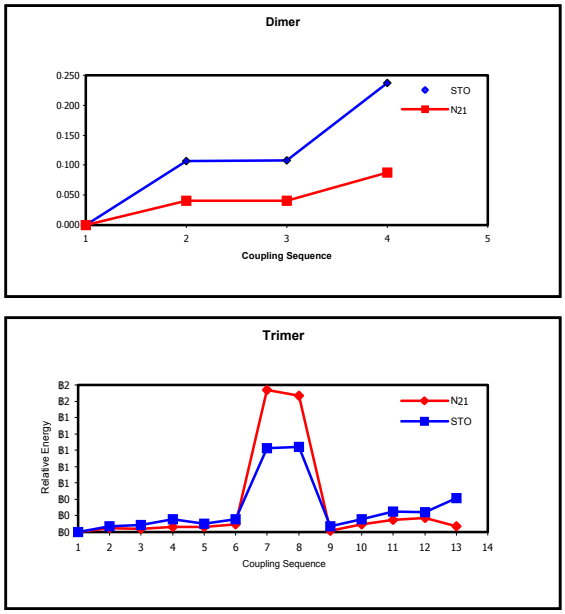


Figure 3. Plot of the relative energy vs coupling sequence of dimer, trimer, tetramer and pentamer

4. Conclusion

The addition of a thiophene to a polythiophene backbone can be achieved in a large number of ways depending on the position along the chain as well as the orientation of the monomer. The relative energies of the possible structures can then be predicted by counting the types of thiophene rings. Finally, it was observed that α - α′ coupling has the lowest final energy among the thiophene oligomers; thus it is the most stable coupling**.**

Acknowledgments

We thank the Commission on Higher Education – Mindanao Advanced Education Project (CHED-MAEP) for the financial support. Many thanks to Engr. Wilfredo Chung, Prof. Ronald Pascual and Mr. Marvin Fernandez for many fruitful discussions and suggestions. Likewise, thank is due to MSU-IIT Computing and Network Center for the generous allocation of computer resources. Thank is due to Mr. Dante D. Dinawanao for the many help extended.

References

[1] Xun-Zuan Yang and J. Chien, "Autooxidation and Stabilization of Undoped and Doped Polyacetylenes", *Journal of Polymer Science: Polymer Chemistry Edition*, Vol. 23, pp 856-878 (1995).

[2] A. Castillon, "Study on the New Organic Semiconductor Radiation Detector", *Materal Thesis*, MSU-IIT, Iligan Cty, Unpublished. (2000).

[3] J. Roncalia, "Conjugated Poly(thiophenes): Synthesis, Functionalizations, and Applications", *Chem Rev.* 1992, 97, pp 711-738 (1992).

[4] M. Yurtsener and E. Yurtsener." Structural Studies of Polypyrroles: An *Ab Initio* Evaluation of Bonding Through α and β carbons", *Synthetic Metals* 98 ,1999, pp 221-227 (1999).

[5] Kwon, O. and McKee, M.L. *J. Phys. Chem.* B **2000**, 104, 1686-1694.

[6] J. Roncalia, *Chem. Rev.* 1992, 42, 711-738.

[7] Hyperchem ™Release[3]. *Windows Molecular Modelling System*, Copyright ©1993, HyperCube, Inc. and Autodesk, Inc. Developed by HyperCube, Inc.

[8] GAMESS Version = 22 Nov 1995 from Iowa State University, M.W.Schmidt, K.K.Baldridge, J.A.Boatz, S.T.Elbert, M.S.Gordon, J.H.Jensen, S.Koseki, N.Matsunaga, K.A.Nguyen, S.J.Su, T.L.Windus, Together With M.Dupuis, J.A.Montgomer. *J. Comp.Chem.* 14, 1347-1363 (1993).

[9] P. Walters, M. Stahl, *BABEL Program (version 1.1)* Copyright ©1992, 93, 94, Dolota Research Group, Department of Chemistry, University of Arizona.

[10] Foresman, J. B. and Frisch, Æ., *Exploring Chemistry with Electronic Structure Methods, 2nd ed.* Gaussian, Inc., Pittsburgh, PA. 1995.

[11] http://www.chem.umass.edu /~lahti/ARTICLE /comp txt.com

[12] http://zopyros.ccqc.ga.edu/lec_top/hf/node1-5.html

[13] http://www.ncsc.org/training/materials/Software_ Release/foundations/qmnotes/SECTION001100000 00000000000

High Performance Computing for Compressible Turbulent Flow

Ekachai Juntasaro[1], Putchong Uthayopas[2],
Boonlue Sawatmongkhon[1] and Khongthep Boonmee[2]

[1]Computational Fluid Dynamics Laboratory (CFD Lab),
School of Mechanical Engineering, Institute of Engineering,
Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand.
Email: junta@ccs.sut.ac.th, Phone: (66 44) 224410 – 2, Fax: (66 44) 224411

[2]Parallel Research Group (PRG),
Department of Computer Engineering, Faculty of Engineering,
Kasetsart University, Bangkok 10900, Thailand.
Email: pu@ku.ac.th, Phone: (66 2) 9428555 ext 1416, Fax: (66 2) 5796245

ABSTRACT -- The aim of the present research and development work is to develop the computer program to simulate the steady two-dimensional compressible turbulent flow. The finite volume method is used to numerically solve the flow governing equations. The Navier-Stokes equations are solved for the velocity field and the SIMPLE algorithm is used to adjust the velocity field to satisfy the conservation law of mass. Since all the variables are stored at the center of each control volume, the Rhie-Chow interpolation is used to avoid the decoupling between the velocity and the pressure. The corrected velocity field is used to solve the $k-$ and $\varepsilon-$equations. The eddy viscosity, that represents the influence of turbulence on the mean flow field, can then be calculated from those values of k and ε obtained. The energy equation is solved for the temperature field. The effects of temperature and pressure on the fluid density are taken into account via the equation of state. The boundary layer on a flat plate is employed as a test case because it is one of the standard benchmark problems for the validation of CFD software. The sequential-computing solver is first used to obtain the computed results. It is found that the computed results are in good agreement with the experimental data at subsonic speed. The parallel-computing solver is also implemented here and tested against the sequential-computing one. It is found that the parallel program can run faster than the sequential one up to 2.55 times for the best case. Furthermore, the governing equations are solved on the structured and body-fitted coordinates so that this computer program can be developed further for the simulation of flow over or inside any object of complex geometry in the future.

KEY WORDS Compressible Turbulence Flow, High Performance Computing, Parallel Solver

1. Introduction

Fluid flow involves many advanced applications in science, engineering and technology. Understanding of the flow behavior is therefore important for the design and development of scientific and engineering innovations. In fluid dynamics, the flow behavior is governed by the continuity equation, the Navier-Stokes equations, the energy equation and the equation of state. For compressible flow, where the free-stream Mach number is higher than 0.3, the effects of temperature variation on fluid properties are so large that the fluid properties must be treated as variable.

To study turbulent flow, the continuity, Navier-Stokes, energy and state equations can be solved directly by any numerical method called Direct Numerical Simulation (DNS). However, the simulation requires the large number of grid points, volumes, elements or other form of sub-domains to capture the characteristics of turbulent flow. Therefore, the computation requires a supercomputer that have a large storage to store all essential data and good computing power to run the program as fast as possible. At present, the supercomputer can only provide the solution for the turbulent flow at low Reynolds number with simple geometries. In other words, the turbulent flow in engineering applications cannot practically be predicted and studied by this approach. In general, the turbulent flow is predicted and studied on the basis of mean quantities. By this way, the continuity, Navier-Stokes, energy and state equations are essentially time-averaged using the density-weighted technique. This technique gives rise to some extra unknown terms which need to be properly modeled. Turbulence models have been developed and widely used with success over a wide range of engineering applications.

The present work is aimed to develop the computer program for the simulation of steady two-dimensional compressible turbulent flow using the two-equation turbulence model. However, the recent emergence of Beowulf cluster computing technology, which is the use of commodity PC

and high speed network to build a cost effective supercomputer, has created a lot of interest around the world. The potential speed increases by parallelizing the CFD program to run on this plat form using portable standard programming such as MPI can be immense. Hence, there is a need to explore such technique to reduce the computation time and to increase productivity gained. Part of this work is performed according to that goal.

2. Governing Equations

Compressible flow is governed by the continuity, Navier-Stokes, energy and state equations where all the fluid properties are variable. For turbulent compressible flow, these governing equations are essentially time-averaged using the density-weighting technique and the resulting solution is the mean quantities. This technique gives rise to the extra unknown terms which cause a closure problem. This problem can be solved using an appropriate turbulence model. For steady two-dimensional mean flow, the governing equations with the turbulence model can be expressed in terms of tensor notation as follows:

2.1 Continuity Equation

$$\frac{\partial}{\partial x_j}(\bar{\rho}\tilde{u}_j) = 0$$

where $\bar{\rho}$ is the fluid density and $\tilde{u}_j$ is the flow velocity.

2.2 Navier-Stokes Equations

$$\frac{\partial}{\partial x_j}(\bar{\rho}\tilde{u}_j\tilde{u}_i) = \frac{\partial}{\partial x_j}(\bar{t}_{ij} + \tau_{ij}) - \frac{\partial \bar{P}}{\partial x_i}$$

where $\bar{P}$ is the pressure, and $\bar{t}_{ij}$ and τ_{ij} are the laminar- and turbulent-flow stresses respectively with the following definitions:

$$\bar{t}_{ij} = \mu\left[\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) - \frac{2}{3}\delta_{ij}\frac{\partial \tilde{u}_k}{\partial x_k}\right]$$

where μ is the fluid viscosity and δ_{ij} is the Kronecker delta:

$\delta_{ij} = 0$ for $i \neq j$ and $\delta_{ij} = 1$ for $i = j$, and

$$\tau_{ij} = \mu_t\left[\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) - \frac{2}{3}\delta_{ij}\frac{\partial \tilde{u}_k}{\partial x_k}\right] - \frac{2}{3}\delta_{ij}\bar{\rho}k$$

where μ_t is the eddy viscosity and k is the kinetic energy of turbulence.

2.3 Two-Equation Turbulence Model of Launder and Sharma (1974)

$$\mu_t = \bar{\rho}c_\mu f_\mu \frac{k^2}{\varepsilon}$$

where c_μ is the model constant, f_μ is the damping function, and ε is the dissipation rate of k. The transport equations for k and ε are modelled as follows:

$$\frac{\partial}{\partial x_j}(\bar{\rho}\tilde{u}_j k) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + \tau_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} - \bar{\rho}\varepsilon + \bar{\rho}D$$

where σ_k is the model constant and D is the extra term. For compressible turbulent flow, ε is split into two parts: ε_s (Solenoidal dissipation rate of k) and ε_d (Dilatation dissipation rate of k). Thus,

$$\varepsilon = \varepsilon_s + \varepsilon_d$$

Sarkar, Erlebacher, Hussaini & Kreiss (1991) have proposed that

$$\varepsilon_d = M_t^2\varepsilon_s$$

where M_t is the turbulent Mach number defined as

$$M_t^2 = 2\frac{k}{a^2}$$

with a is the speed of sound. ε_s is calculated from the following equation:

$$\frac{\partial}{\partial x_j}(\bar{\rho}\tilde{u}_j\varepsilon_s) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon_s}{\partial x_j}\right] + c_{\varepsilon 1}f_{\varepsilon 1}\frac{\varepsilon_s}{k}\tau_{ij}\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) - \bar{\rho}c_{\varepsilon 2}f_{\varepsilon 2}\frac{\varepsilon_s^2}{k} - \frac{4}{3}\bar{\rho}\varepsilon_s\frac{\partial \tilde{u}_k}{\partial x_k} + \bar{\rho}E$$

where (σ_ε , $c_{\varepsilon1}$, $c_{\varepsilon2}$) are the model constants, ($f_{\varepsilon1}$, $f_{\varepsilon2}$) are the damping functions, and E is the extra term.

For the $k-\varepsilon$ turbulence model of Launder and Sharma (1974), the model constants, damping functions and extra terms are provided as follows:

$$c_\mu = 0.09, \quad \sigma_k = 1.0, \quad \sigma_\varepsilon = 1.3, \quad c_{\varepsilon1} = 1.44, \quad c_{\varepsilon2} = 1.92,$$

$$f_\mu = \exp\left[\frac{-3.4}{\left(1+\frac{R_t}{50}\right)^2}\right], \qquad f_{\varepsilon1} = 1.0,$$

$$f_{\varepsilon2} = 1 - 0.3\exp\left(-R_t^2\right),$$

$$D = -2\frac{\mu}{\bar{\rho}}\left(\frac{\partial\sqrt{k}}{\partial x_i}\right)^2, \quad \text{and} \quad E = 2\frac{\mu}{\bar{\rho}}\frac{\mu_t}{\bar{\rho}}\left(\frac{\partial^2\tilde{u}_i}{\partial x_j \partial x_k}\right)^2$$

where $R_t = \dfrac{\bar{\rho}k^2}{\mu\varepsilon}$.

2.4 Energy Equation

$$\frac{\partial}{\partial x_j}(\bar{\rho}\tilde{u}_j\tilde{e}_T) = \frac{\partial}{\partial x_j}\left[\left(\frac{k_T}{c_v}+\frac{\mu_t}{Pr_t}\right)\frac{\partial\tilde{e}_T}{\partial x_j}\right] + \frac{\partial}{\partial x_j}\left[\tilde{u}_i\left(\bar{t}_{ij}+\tau_{ij}\right)\right] - \frac{\partial}{\partial x_j}\left(\tilde{u}_j\bar{P}\right) + \frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] - \frac{\partial}{\partial x_j}\left[\left(\frac{k_T}{c_v}+\frac{\mu_t}{Pr_t}\right)\frac{\partial}{\partial x_j}(K+k)\right]$$

where k_T is the thermal conductivity, c_v is the specific heat at constant volume, Pr_t is the turbulent Prandtl number taken as 0.91, and $\tilde{e}_t$ is the total energy which is defined as $\tilde{e}_t = \tilde{e} + K + k$

where $\tilde{e}$ is the internal energy ($\tilde{e} = c_v\tilde{T}$ where $\tilde{T}$ is the temperature) and K is the kinetic energy of the mean flow, i.e. $K = 0.5(\tilde{u}^2 + \tilde{v}^2)$.

2.5 Equation of State

$$\bar{P} = (\gamma-1)\bar{\rho}\left(\tilde{e}_t - K - k\right)$$

where γ is the specific heat ratio.

The fluid properties, μ and k_T, of compressible flow can be influenced by the variation of the temperature so that they must be defined in terms of temperature as the following relations:

2.6 Sutherland's Law

$$\mu = \mu_\infty\left(\frac{\tilde{T}}{T_\infty}\right)^{3/2}\frac{T_\infty+110}{\tilde{T}+110}$$

where the subscript ∞ denotes the value at free-stream.

2.7 Prandtl Number

$$Pr = \frac{\mu c_p}{k_T}$$

where c_p is the specific heat at constant pressure.

3. Numerical Method

The finite volume method is used to numerically solve the governing equations which can be written in a general form as follows:

$$\frac{\partial}{\partial x_i}\left(\rho\bar{u}_i\phi\right) = \frac{\partial}{\partial x_i}\left(\Gamma\frac{\partial\phi}{\partial x_i}\right) + S^\phi$$

where ϕ is the general dependent variable, Γ is the effective diffusion coefficient, and S^ϕ is the source/sink term of ϕ. To be able to simulate the internal flow with variable cross-sectional area and the external flow past an object of complex shape, the general form of the governing equations is essentially transformed from the physical domain

(x, y) into the computational domain (ξ, η) as the following equation:

$$\frac{\partial}{\partial\xi}(\rho U\phi)+\frac{\partial}{\partial\eta}(\rho V\phi)=\frac{\partial}{\partial\xi}\left[\frac{\Gamma}{J}\left(\alpha\frac{\partial\phi}{\partial\xi}-\beta\frac{\partial\phi}{\partial\eta}\right)\right]+\frac{\partial}{\partial\eta}\left[\frac{\Gamma}{J}\left(\gamma\frac{\partial\phi}{\partial\eta}-\beta\frac{\partial\phi}{\partial\xi}\right)\right]+JS^{\phi}$$

where

$$U=\bar{u}\frac{\partial y}{\partial\eta}-\bar{v}\frac{\partial x}{\partial\eta},\quad V=\bar{v}\frac{\partial x}{\partial\xi}-\bar{u}\frac{\partial y}{\partial\xi},$$

$$\alpha=\left(\frac{\partial x}{\partial\eta}\right)^2+\left(\frac{\partial y}{\partial\eta}\right)^2,\quad \beta=\frac{\partial x}{\partial\xi}\frac{\partial x}{\partial\eta}+\frac{\partial y}{\partial\xi}\frac{\partial y}{\partial\eta},$$

$$\gamma=\left(\frac{\partial x}{\partial\xi}\right)^2+\left(\frac{\partial y}{\partial\xi}\right)^2,\ \text{and}\ J=\frac{\partial x}{\partial\xi}\frac{\partial y}{\partial\eta}-\frac{\partial y}{\partial\xi}\frac{\partial x}{\partial\eta}.$$

Using the finite volume method, the computational domain is divided into a number of control volumes. The transformed equations can be integrated as follows:

$$[(\rho U\Delta\eta)\phi]_w^e+[(\rho V\Delta\xi)\phi]_s^n=\left[\frac{\Gamma\Delta\eta}{J}\left(\alpha\frac{\partial\phi}{\partial\xi}-\beta\frac{\partial\phi}{\partial\eta}\right)\right]_w^e+\left[\frac{\Gamma\Delta\xi}{J}\left(\gamma\frac{\partial\phi}{\partial\eta}-\beta\frac{\partial\phi}{\partial\xi}\right)\right]_s^n+(J\Delta\xi\Delta\eta)\bar{S}_P^{\phi}$$

where $\bar{S}_P^{\phi}$ is the mean value of S^{ϕ} at the center P of each control volume, and (e, w, n, s) are the east, west, north and south faces of each control volume. The convection terms are approximated by the first-order upwind differencing scheme and the diffusion terms are estimated by the second-order central differencing scheme. Therefore, the standard form of the finite volume equation can be obtained as

$$A_P^{\phi}\phi_P=A_E^{\phi}\phi_E+A_W^{\phi}\phi_W+A_N^{\phi}\phi_N+A_S^{\phi}\phi_S+b^{\phi}$$

where

$$A_E^{\phi}=\left(\frac{\Gamma}{J}\alpha\frac{\Delta\eta}{\Delta\xi}\right)_e+\max[0,-(\rho U\Delta\eta)_e],$$

$$A_W^{\phi}=\left(\frac{\Gamma}{J}\alpha\frac{\Delta\eta}{\Delta\xi}\right)_w+\max[0,(\rho U\Delta\eta)_w],$$

$$A_N^{\phi}=\left(\frac{\Gamma}{J}\gamma\frac{\Delta\xi}{\Delta\eta}\right)_n+\max[0,-(\rho V\Delta\xi)_n],$$

$$A_S^{\phi}=\left(\frac{\Gamma}{J}\gamma\frac{\Delta\xi}{\Delta\eta}\right)_s+\max[0,(\rho V\Delta\xi)_s],$$

$A_P^{\phi}=A_E^{\phi}+A_W^{\phi}+A_N^{\phi}+A_S^{\phi}$, and

$$b^{\phi}=(J\Delta\xi\Delta\eta)\bar{S}_P^{\phi}-\left[\frac{\Gamma\Delta\eta}{J}\left(\beta\frac{\partial\phi}{\partial\eta}\right)\right]_w^e-\left[\frac{\Gamma\Delta\xi}{J}\left(\beta\frac{\partial\phi}{\partial\xi}\right)\right]_s^n.$$

The standard SIMPLE algorithm is employed here to satisfy the conservation law of mass. The continuity equation is not solved directly with other governing equations. The p'-equation is solved instead to obtain the pressure correction p' and its value is used to correct the values of pressure and velocities to satisfy the conservation law of mass. The p'-equation can be written in a standard form as follows:

$$A_P^p p'_P=A_E^p p'_E+A_W^p p'_W+A_N^p p'_N+A_S^p p'_S+m_P$$

where

$$A_E^p=\left(\rho B\frac{\Delta\eta}{\Delta\xi}\right)_e\quad A_W^p=\left(\rho B\frac{\Delta\eta}{\Delta\xi}\right)_w,$$

$$A_N^p=\left(\rho C\frac{\Delta\xi}{\Delta\eta}\right)_n,\ A_S^p=\left(\rho C\frac{\Delta\xi}{\Delta\eta}\right)_s,$$

$A_P^p=A_E^p+A_W^p+A_N^p+A_S^p$, and

$$m_P=(\rho U^*\Delta\eta)_e-(\rho U^*\Delta\eta)_w+(\rho V^*\Delta\xi)_n-(\rho V^*\Delta\xi)_s.$$

U^*, V^* are calculated from the resulting velocities of the Navier-Stokes equations, whereas

$$B=B^u\frac{\partial y}{\partial\eta}-B^v\frac{\partial x}{\partial\eta},\ \text{and}\ C=C^v\frac{\partial x}{\partial\xi}-C^u\frac{\partial y}{\partial\xi}$$

where

$$B^u = -\frac{\Delta\xi\Delta\eta}{A_P^u}\frac{\partial y}{\partial \eta}, \qquad B^v = \frac{\Delta\xi\Delta\eta}{A_P^v}\frac{\partial x}{\partial \eta},$$

$$C^u = \frac{\Delta\xi\Delta\eta}{A_P^u}\frac{\partial y}{\partial \xi}, \text{ and } C^v = -\frac{\Delta\xi\Delta\eta}{A_P^v}\frac{\partial x}{\partial \xi}.$$

In general, the standard SIMPLE algorithm is implemented on the staggered grid system to prevent the decoupling between the velocity and the pressure. However, the staggered grid system is technically rather complicated for programming and requires a large amount of computer storage. This drawback becomes obvious when the computer program is developed further for real-world applications. The collocated grid system is employed in this work so that all the variables are stored at the center of each control volume. The problem of velocity-pressure decoupling is solved by the Rhie-Chow interpolation where (U_e^*, U_w^*, V_n^*, V_s^*) are calculated from the appropriate pressure gradient.

In the current work, the boundary layer on a flat plate is chosen as a test case and the implementation of the SIMPLE algorithm and the Rhie-Chow interpolation must be slightly changed to suit the flow problem. Physically, the pressure field of this flow is constant and presumably known throughout the flow domain. Therefore, the pressure correction p' obtained is used to correct the velocities only, not to correct the pressure, because the pressure itself is already known and constant. Moreover, the Rhie-Chow interpolation is simplified to the linear interpolation of (U_e^*, U_w^*, V_n^*, V_s^*) between grid nodes without any effect of pressure gradient.

The algorithm for the simulation of turbulent compressible flow can be summarized as follows:

(1) Start the computation with an initial guess of velocities, pressure correction, turbulence kinetic energy, dissipation rate of turbulence kinetic energy, temperature, density and viscosity
(2) Calculate the Navier-Stokes equations for the velocities
(3) Calculate the p'-equation for the pressure correction
(4) Correct the velocities by the pressure correction
(5) Calculate the k-equation for the turbulence kinetic energy
(6) Calculate the ε-equation for the dissipation rate of turbulence kinetic energy
(7) Calculate the energy equation for the total energy, and hence temperature
(8) Calculate the density from the equation of state, then the viscosity from Sutherland's law, and the thermal conductivity from the definition of Prandtl number
(9) Repeat from step (2) until the solution converges

4. Development of Parallel Computer Program

Parallel computing is a technique of partitioning long computation tasks into many sub-tasks that execute concurrently on multiple computers. Many useful techniques and algorithms can be found in parallel computing text such as [14][15]. In general, to partition a sequential program to run on cluster system, this program has to be analyzed using the profiling program called "gprof" to discover the most compute intensive part. Below is some example captured from the result of gprof profiling.

```
%  cumulative  self          self     total
 time  seconds  seconds   calls  us/call  us/call  name
 30.81   678.05   678.05 2224080000     0.30     0.30
NavierStokes::Diff(int, int, double **, char, char)

 28.87  1313.32   635.27     1500 423513.33 661914.16
NavierStokes::EnergyEquation(void)

 11.15  1558.71   245.39     1500 163593.33 359517.83
NavierStokes::Momentum(double **, double **)

  5.63  1682.62   123.91      800 154887.50 326566.29
NavierStokes::epslEquation(void)

  4.75  1787.12   104.50     1500 69666.67 456301.14
NavierStokes::SIMPLE(void)
```

The obtained results show that "Diff" method are the most using and calling in this program. However, Diff is a common subroutine that is called from other method. It turns out that "EnergyEquation" method call "Diff" method more than other method. Another long computation method is "Momentum" method, but "Momentum" method is called from "SIMPLE". Thus the decision are made to parallelize the "EnergyEquation" ,"Momentum" and "SIMPLE" method. But some methods have relation with three methods. Thus, we must to parallelize them. The main code consists of two levels of iteration. For the first iteration, program computes the one equation until variable DeltaOverallResidual less than 10^{-12}. Within this loop has eight methods as SIMPLE, UpdateUV, OneEquation, UpdateMu_tOneEq, EnergyEquation, EquationOfState, SutherlandLaw and CalThermalConductivity. Every 50 iterations will compute DeltaOverallResidual for using check condition in iteration. When program finish first iteration will prepare initialized data for second iteration. Second iteration, program will compute by use two-equation method until DeltaOverallResidual less than 10^{-14}. Within loop have five methods as SIMPLE, UpdateUV, kEquation, epslEquation, UpdateMu_tTwoEq, EnergyEquation, EquationOfState, SutherlandLaw and CalThermalConductivity. The pseudo codes of two main loops are as follows.

```
do{
        ++iteration;
        Turbulent.SIMPLE();
        Turbulent.UpdateUV();
        Turbulent.OneEquation();
        Turbulent.UpdateMu_tOneEq();
        Turbulent.EnergyEquation();
        Turbulent.EquationOfState();
        Turbulent.SutherlandLaw();
        Turbulent.CalThermalConductivity();
        if( iteration > 0  &&  iteration%50 == 0 ){
                Turbulent.CalOverallResidual();
                OverallResidualNew =
Turbulent.GetOverallResidual();
                DeltaOverallResidual = fabs(
                OverallResidualNew - OverallResidualOld
                );
                OverallResidualOld =
OverallResidualNew;
                printf("\n%6d  OE %.2e ",iteration,
DeltaOverallResidual);
                Turbulent.Display();
        }
}while( DeltaOverallResidual > pow(10.0, -12.0) );
printf("\n\tSwitch to Two-Equations");
do{
        ++iteration;
        Turbulent.SIMPLE();
        Turbulent.UpdateUV();
        Turbulent.kEquation();
        Turbulent.epslEquation();
        Turbulent.UpdateMu_tTwoEq();
        Turbulent.EnergyEquation();
        Turbulent.EquationOfState();
        Turbulent.SutherlandLaw();
        Turbulent.CalThermalConductivity();
        if( iteration > 0  &&  iteration%50 == 0 ){
                Turbulent.CalOverallResidual();
                OverallResidualNew =
Turbulent.GetOverallResidual();
                DeltaOverallResidual = fabs(
                OverallResidualNew - OverallResidualOld
                );
                OverallResidualOld =
OverallResidualNew;
                printf("\n%6d  OE %.2e ",iteration,
DeltaOverallResidual);
                Turbulent.Display();
        }
}while( DeltaOverallResidual > pow(10.0, -14.0) );
```

In order to parallelize this code, we divide the data into several parts using the column base partitioning approach. The important issue involved in data partitioning is to distribute the data to all nodes in a balanced fashion. A simple way of partitioning data can be used effectively here. For this program, the number of column is divided by number of computing node. If there is any column left, number of column in each node will be increases by one column. Thus, each node will have the same amount of data number to be processes.

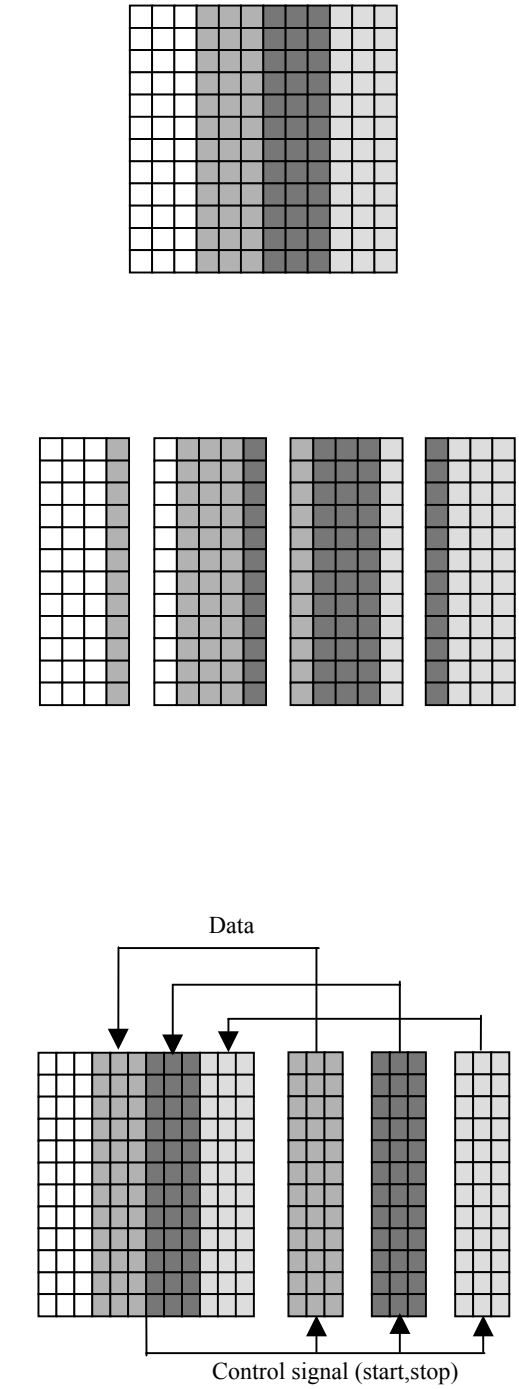


Figure 3. Communication Model

4.1 Communication Model

For our approach, we divide the computing node into two types: master node and slave node. The master node controls the processing step of parallel task. It send signal to slave node to start the computing process. Every 50 iterations, master node will receive the error data that will be used to compute the overall error. When overall error is lower than the minimum value required, master node will send signal to slave node to stop the computing process. Master node also computes the result using data that it has. In contrast, the main function of slave nodes is to compute the data that belong to them. When they receive the start signal from master node, slave node will start the computing process. When they receive stop signal from master node, they will stop the computing process and send result to master node.

4.2 Exchanging Data

The parallelization of this code has been done using MPI standard and MPICH [16][17] implementation from Argonne National Laboratory. For this program, there are many exchange of boundary data between nodes. This is done using MPI::COMM_WORLD.Send and MPI::COMM_WORLD.Receive primitive in MPI. The example statements excerpt from the code are as shown below.

```
void SendRight(int var_id,NavierStokes *Turbulent,double
*buffer,int rank)
{
        long number;
        Turbulent->GetArr(var_id,buffer,&number,Turbulent->
        endxCV-6,Turbulent-> endxCV);
        MPI::COMM_WORLD.Send(&number,1,MPI::LONG,ra
        nk+1,NUMTAG+100+rank+1);
        MPI::COMM_WORLD.Send(buffer,number,
        MPI::DOUBLE,rank+1,NUMTAG+100+rank+1);
}

void ReceiveLeft(int var_id,NavierStokes *Turbulent,double
*buffer,int rank)
{
        MPI::Status status;
        long number;
        MPI::COMM_WORLD.Recv(&number,1,MPI::LONG,
        rank-1,NUMTAG+100+rank,status);
        MPI::COMM_WORLD.Recv(buffer,number,
        MPI::DOUBLE,rank-1, NUMTAG+100+rank ,status);
        Turbulent->SetArr(var_id,buffer,Turbulent->startxCV-
        7,Turbulent->startxCV-1);
}
```

In this computation, we assume that each computing node is ordered from left to right. The node that has rank less than other will be located on the left side. The first node locates on the left side of second node and so on. Next step is to identify the communication pattern by locating the variables that need to be updated. In the first loop consist of 8 method, every method modify the value of variables. Example of this is the SIMPLE method that computes the value of u, v, pCrtn, uCrtn and vCrtn. Thus, each node must exchange value of u, v, pCrtn, uCrtn and vCrtn. In the second loop, it contains 9 methods that update the value of variables. Thus, each node must exchange this data properly.

5. Results and Discussion

Computations are conducted for laminar and turbulent compressible flows, and input data are summarized in Table 1.

Table 1. Input Data

Parameter	Compressible Laminar Flow	Compressible Turbulent Flow
ξ_{max}	151	151
η_{max}	151	151
Re_L	2,000,000	19,500,000
M_∞	0.4, 0.6, 0.8	0.824
T_∞ (K)	300	300
P_∞ (Pa)	101,325.0	110,995.5
T_W (K)	300	Adiabatic Recovery Temperature
Relaxation Factor	0.5	0.5

where ξ_{max} and η_{max} are the numbers of grid lines used on the computational domain, Re_L is the Reynolds number based on the length of the flat plate and the free-stream velocity, M_∞ is the free-stream Mach number, T_∞ is the free-stream temperature, P_∞ is the free-stream pressure, T_W is the wall temperature and the relaxation factor is used to stabilize the numerical scheme used.

5.1 Laminar Compressible Flow

Figure 4 shows the velocity distributions in which the numerical solutions are compared with the analytical solutions at three free-stream Mach numbers. The definitions of the normalized cross-stream distance and stream wise velocity are $y/\sqrt{x\mu_\infty/\rho_\infty u_\infty}$ and u/u_∞ respectively. It is found that the numerical solutions are in very good agreement with the analytical solutions at all free-stream Mach numbers considered. For subsonic flow where the free-stream Mach number is as high as 0.8, the velocity distribution is not influenced by the Mach number. Physically, the compressibility effect is so little that its effect does not appear on the velocity distribution of the flow.

Figure 5 illustrates the temperature distributions where the numerical solutions are compared with the analytical solutions at three free-stream Mach numbers. The definition of the normalized temperarure is T/T_∞ whereas the normalized cross-stream distance has the same definition as in Figure 4. The numerical solutions compared well with the

analytical solutions at all three free-stream Mach numbers. The difference between the numerical solution and the analytical solution is larger as the Mach number is higher. The maximum temperature is higher as the Mach number increases, that is, from about 0.5% at Mach 0.4 to around 2.5% at Mach 0.8.

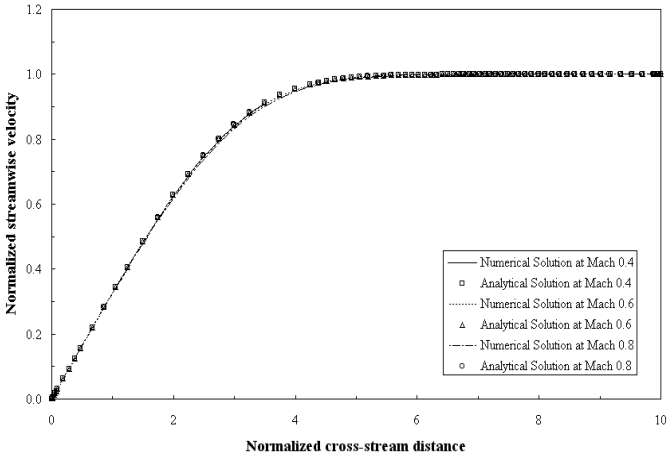


Figure 4. Velocity distributions of the laminar boundary layer on a flat plate

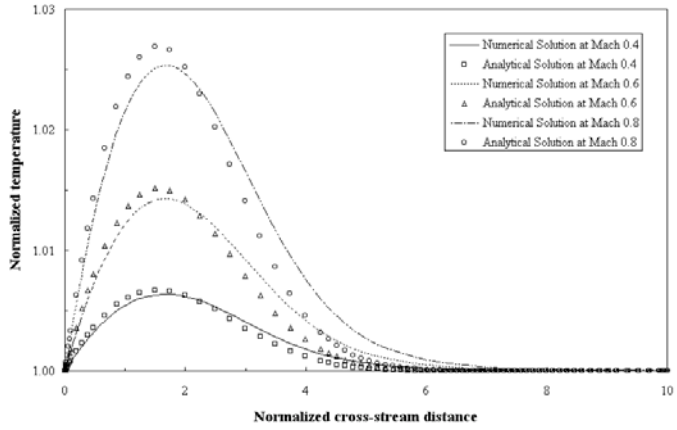


Figure 5. Temperature distributions of the laminar boundary layer on a flat plate

5.2 Turbulent Compressible Flow

Figures 6(a) and 6(b) show the velocity distributions of the turbulent boundary layer on a flat plate at Mach 0.824 where the numerical solution is compared with the law of the wall in Figure 6(a) while the experimental data of Motallebi (1994) is compared with the law of the wall in Figure 6(b). It is found that both the numerical solution and the Motallebi data are in good agreement with the law of the wall in a log-linear region where $5 < \ln(y\rho_w u_\tau / \mu_w) < 8$. In both figures, u^* is the transformed velocity, which is defined by the van Driest transformation as follows:

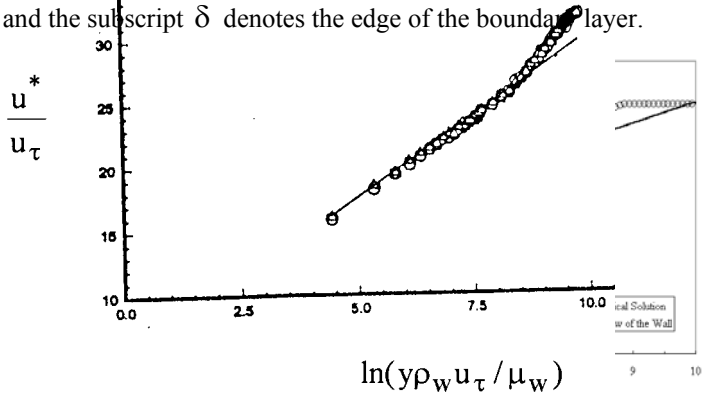
$$u^* = \frac{\tilde{u}_\delta}{b}\sin^{-1}\left[\frac{2b^2\frac{\tilde{u}}{\tilde{u}_\delta} - a}{\sqrt{a^2+4b^2}}\right]$$

where

$$a = \frac{\tilde{T}_\delta}{\tilde{T}_w}\left[1 + r\frac{\gamma-1}{2}M_\delta^2\right] - 1, \text{ and}$$

$$b^2 = r\frac{\gamma-1}{2}M_\delta^2\frac{\tilde{T}_\delta}{\tilde{T}_w}$$

with r is the recovery factor ($r = 0.89$ for turbulent flow) and the subscript δ denotes the edge of the boundary layer.



Figures 7(a) and 7(b) show the comparisons of the numerical solution and the experimental data of Motallebi (1994) with the following Maise and McDonald correlation respectively:

$$\frac{u_\delta^* - u^*}{u_\tau} = -2.5 \ln \frac{y}{\delta} + 1.25\left[1 + \cos\left(\pi \frac{y}{\delta}\right)\right]$$

where δ is the boundary layer thickness and u_τ is the friction velocity, i.e. $u_\tau = \sqrt{\tau_w / \bar{\rho}_w}$. It is found that both the numerical solution and the Motallebi data compare very well with this correlation.

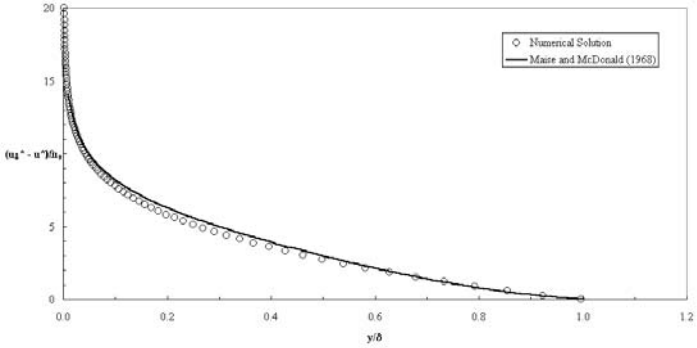


Figure 7(a) Velocity distribution of the turbulent boundary layer on a flat plate; Numerical solution of the present work

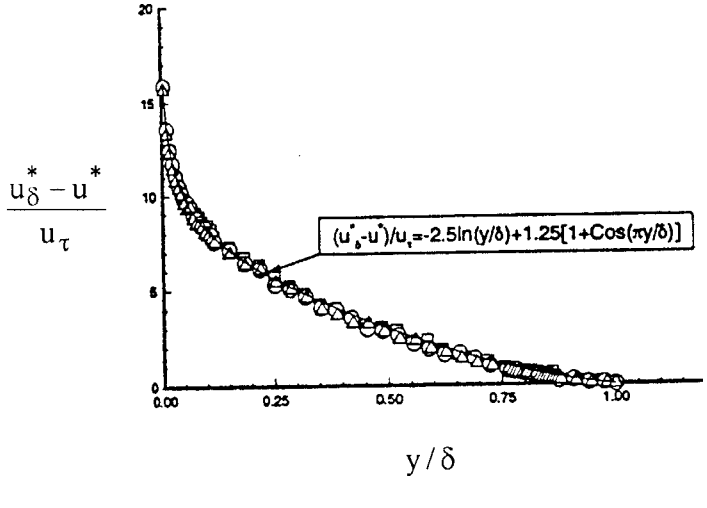


Figure 7(b) Velocity distribution of the turbulent boundary layer on a flat plate; Symbol for the experimental data of Motallebi (1994); Line for the Maise and McDonald correlation

Figures 8(a) and 8(b) show the comparisons of the numerical solution and the experimental data of Motallebi (1994) with the following Fernholz and Finley correlation respectively:

$$\frac{u_\delta^* - u^*}{u_\tau} = -4.7 \ln \frac{y}{\Delta^*} - 6.74$$

where

$$\Delta^* = \delta \int_0^1 \left(\frac{u_\delta^* - u^*}{u_\tau} \right) d\left(\frac{y}{\delta}\right)$$

It is found that the numerical solution and the Motallebi data are reasonably well compared with the Fernholz and Finley correlation.

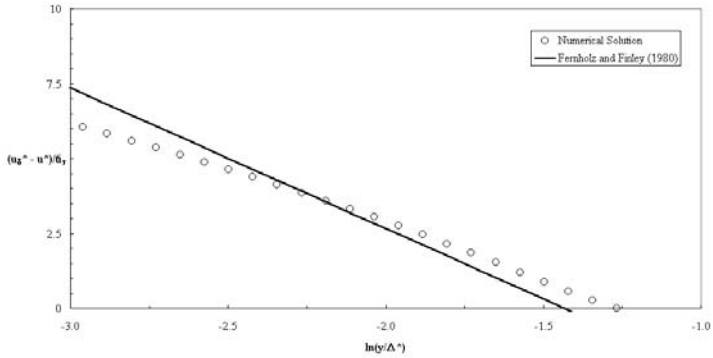


Figure 8(a) Velocity distribution of the turbulent boundary layer on a flat plate; Numerical solution of the present work

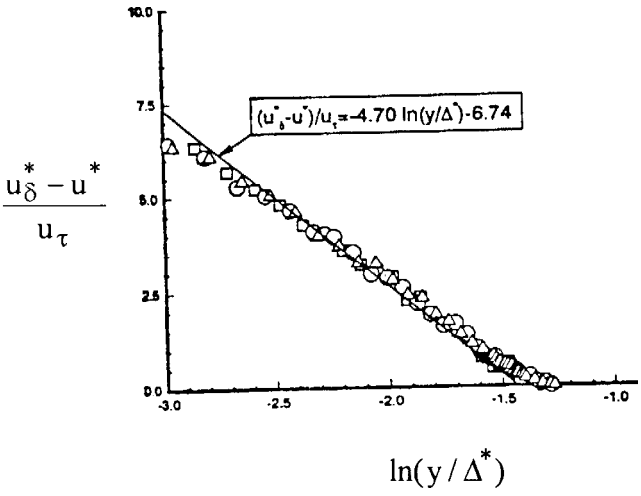


Figure 8(b) Velocity distribution of the turbulent boundary layer on a flat plate; Line for the Fernholz and Finley correlation

5.3 Performance of Parallel Computer Program

To evaluate the performance of the system, parallel program has been tested on AMATA Beowulf system. This system consists of:

- 4 Athlon 950 MHz, 256 Mbytes RAM and 20 Gbytes Hard disk
- 4 Athlon 1GHz , 256 Mbytes RAM and 20 Gbytes Hard disk
- Fast Ethernet Switch Interconnection between nodes

First, the test has been conduct by running sequential program to measure the runtime. Then, parallel program has been run on 2, 4, and 8 nodes consequently. The runtime of parallel code has also been measured. The test has been repeated several times for several problem sizes. The results obtained are as depicted in Table 2. Also, the speedup curve has been plotted and illustrated in Figure 9.

Table 2. Runtime results of the experiment

Test No.	Number of Grids	Sequential Runtime (seconds)	Parallel Runtime			Speedup		
			2 nodes	4 nodes	8 nodes	2 nodes	4 nodes	8 nodes
1	151*151	1065	1061.32	1206.02	1021.90	1.00	0.88	1.04
2	201*251	1856.6	1192.39	967.71	1236.52	1.56	1.92	1.50
3	251*151	3088	2142.85	1401.94	1373.27	1.44	2.20	2.25
4	301*151	5403	3230.65	2558.63	2506.05	1.67	2.11	2.16
5	351*151	7208	4645.10	2825.21	3161.77	1.55	2.55	2.28

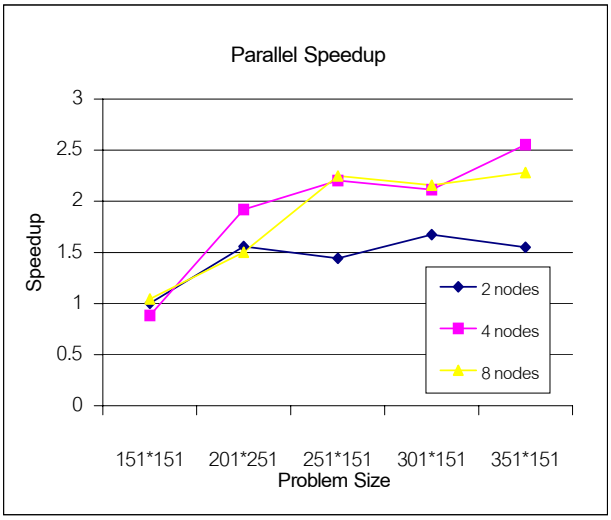


Figure 9. Plot of speed up results

From Figure 9, the parallel speed up shows this parallel algorithm receives the maximum speed when it run on 4 nodes. When number of computing node increase, speed up doesn't increase too. Because speed up of 8 computing nodes almost equal with speed up of 4 computing node. The maximum speed up obtained is as high as 2.55 times of the sequential execution speed.

However, as problem size increases, speed up will increase with it. But speed up doesn't increase follow to number of computing nodes. Thus this algorithm will improve computation/communication ratio for good performance.

6. Conclusions

Both laminar and turbulent compressible flows are simulated in the present work. The flow is considered at subsonic speed where the free-stream Mach number is as high as approximately 0.8. The numerical scheme of the computer program is validated using the laminar compressible boundary layer on a flat plate as a test case. It is found that the computer program is capable of simulating the laminar compressible boundary layers accurately at three free-stream Mach number considered. The turbulent compressible boundary layer on a flat plate at Mach 0.824 is used as a test case to validate the performance of the two-equation turbulence model of Launder and Sharma. It is found that the computer program can accurately simulate the turbulent compressible boundary layer at subsonic speed. The parallel implementation exhibits some moderate speedup. So, one of the most important future work is to analyze the performance of this parallel code and find out how to increases the speedup of this application.

7. Acknowledgements

Part of the present work of the first and third authors are financially supported from SUT Research Fund 55/2542. This support is greatly appreciated. We also would like to thank Dr.Varangrat Juntasaro for sharing her knowledge on the physics of compressible turbulent flow, and the modeling and numerical techniques. Some of the equipment used for performance assessment, especially the Athlon based cluster system, are supported by AMD Far East Inc. Part of the support is also from KURDI SRU Grant.

References

[1] Fernholz, H.H., and Finley, P.J. (1980) "A Critical Commentary on Mean Flow Data for Two-Dimensional Compressible Turbulent Boundary Layers," AGARDograph 253.

[2] Gibson, M.M., Jones, W.P., and Whitelaw, J.H. (1992) "Turbulence Models for Computational Fluid Dynamics," Course Lecture Notes, 18-20 November 1992, Department of Mechanical Engineering, Imperial College of Science, Technology and Medicine.

[3] Gosman, A.D., and Issa, R.I. (1992) "Computational Fluid Dynamics," Post Experience Course, 16-18 November 1992, Department of Mechanical

Engineering, Imperial College of Science, Technology and Medicine.

[4] Gropp, W., Lusk, E., and Thakur, R. (1999) "Using MPI: Portable Parallel Programming with the Message-Passing Interface, 2nd edition", MIT Press.

[5] Gropp, W., Lusk, E., and Thakur, R., (1999) "Using MPI-2: Advanced Features of the Message-Passing Interface", MIT Press.

[6] Hinze, J.O. (1975) "Turbulence," 2nd edition, McGraw-Hill.

[7] Hutchings, B., and Iannuzzelli, R. (1987) "Benchmark Problems for Fluid Dynamics Codes," Mechanical Engineering, June, pp. 54-58.

[8] Juntasaro, E., and Sawatmongkhon, B. (1999) "Compressible Laminar Flow towards a Numerical Wind Tunnel," Proceedings of the 13th National Mechanical Engineering Conference, 2-3 December 1999, Royal Cliff Beach Resort Hotel, South Pattaya, Cholburi, Thailand, Vol. 1, pp. 132-137.

[9] Juntasaro, E., Uthayopas, P., Sawatmongkhon, B., and Boonmee, K. (2001) "High Performance Computing for Steady Two-Dimensional Turbulent Flow," Proceeding of the 5th Annual National Symposium on Computational Science and Engineering, 18-20 June 2001, Sofitel Central Plaza, Bangkok, Thailand, pp. 126-140.

[10] Karki, K.C., and Patankar, S.V. (1989) "Pressure Based Calculation Procedure for Viscous Flows at All Speeds in Arbitrary Configurations," AIAA Journal, Vol. 27, No. 9, pp. 1167-1174.

[11] Kumar, V., Grama, A., Gupta, A., and Karypis, G. (1994) "Introduction Parallel Computing Design and Analysis of Algorithms", Benjamin/Cummings.

[12] Lang, N.J., and Shih, T.H. (1991) "A Critical Comparison of Two-Equation Turbulence Models," NASA Technical Memorandum 105237.

[13] Launder, B.E., and Sharma, B.I. (1974) "Application of the Energy-Dissipation Model of Turbulence to the Calculation of a Flow near a Spinning Disk," Letters in Heat and Mass Transfer, Vol. 1, pp. 131-138.

[14] Maise, G., and McDonald, H. (1968) "Mixing Length and Kinematic Eddy Viscosity in a Compressible Boundary Layer," AIAA Journal, Vol. 6, No. 1, pp. 73-80.

[15] Motallebi, F. (1994) "Mean Flow Study of Two-Dimensional Subsonic Turbulent Boundary Layers," AIAA Journal, Vol. 32, No. 11, pp. 2153-2161.

[16] Motallebi, F. (1996) "Reynolds Number Effects on the Prediction of Mean Flow Data for Adiabatic 2-D Compressible Boundary Layers," Aeronautical Journal, Vol. 100, No. 992, pp. 53-59.

[17] Patankar, S.V. (1980) "Numerical Heat Transfer and Fluid Flow," Hemisphere.

[18] Patel, V.C., Rodi, W., and Scheuerer, G. (1985) "Turbulence Models for Near-Wall and Low Reynolds Number Flows: A Review," AIAA Journal, Vol. 23, No. 9, pp. 1308-1319.

[19] Rhie, C.M., and Chow, W.L. (1983) "Numerical Study of the Turbulent Flow past an Airfoil with Trailing Edge Separation," AIAA Journal, Vol. 21, No. 11, pp. 1525-1532.

[20] Sarkar, S., Erlebacher, G., Hussaini, M.Y., and Kreiss, H.O. (1991) "The Analysis and Modelling of Dilatational Terms in Compressible Turbulence," Journal of Fluid Mechanics, Vol. 227, pp. 473-493.

[21] Schlichting, H. (1979) "Boundary-Layer Theory," 7th edition, McGraw-Hill.

[22] Varangrat, S. (1999) "Computational Study of Compressible Flow in an S-Shaped Duct," Ph.D. Thesis, Department of Mechanical Engineering, Imperial College, University of London, U.K.

[23] Versteeg, H.K., and Malalasekera, W. (1995) "An Introduction to Computational Fluid Dynamics: The Finite Volume Method," Longman Scientific & Technical.

[24] White, F.M. (1991) "Viscous Fluid Flow," 2nd edition, McGraw-Hill.

[25] Wilcox, D.C. (1993) "Turbulence Modeling," DCW Industries.

[26] Wilkinson, B., and Allen, M. (1999) "Parallel Programming", Prentice-Hall.

Blurring of Shifts in the Multi-Shift *QR* Algorithm: Numerical Experiments using UBASIC

Roden Jason A. David
Mathematics Department
Ateneo de Manila University

ABSTRACT – The multi-shift *QR* algorithm for approximating the eigenvalues of a full matrix is known to have convergence problems if the number of shifts used in one iteration is large. The mechanism by which the values of the shifts are being transmitted from one bulge matrix to another has been discovered. In the presence of round-off errors, however, the values of the shifts are blurred in certain bulge matrices causing the *QR* algorithm to miss the true eigenvalues of the matrix. In this paper, we give the maximum number of shifts that can be used in one iteration to keep the values of the shifts from blurring. We use the UBASIC language, and specify the minimum level of precision that maintains well-focused shifts.

KEY WORDS – eigenvalues, *QR* algorithm, Schur upper triangular form, bulge-chase technique, Hessenberg form, blurring of shifts

1. The Problem and its Background

Many mathematical softwares approximate the eigenvalues of a square matrix using the *QR* algorithm. The *QR* algorithm is an iterative algorithm that approximates the Schur upper triangular form of a matrix, and the eigenvalues of the matrix emerge along the main diagonal of the Schur form. Provided that exact arithmetic is used and a convergent shift strategy is found , the QR algorithm converges quadratically for square matrices in general, and cubically for normal matrices in particular [5].

Most implementation of the *QR* algorithm use the bulge-chase technique introduced by Francis [3] in 1961. With this technique, the matrix is first reduced to upper Hessenberg form. Two iterations of the shifted *QR* algorithm are effectively performed in one *QR step*. This is done implicitly using Householder reflections that create a bulge in the matrix. The matrix is reduced back to upper Hessenberg form by chasing the bulge down the matrix. This implicit implementation of two *QR* steps in one iteration came to be known as the *double-step QR algorithm*.

In 1989, Bai and Demmel [1] generalized the double-step *QR* algorithm into the *multi-shift QR* algorithm. An arbitrary number of iterations is performed simultaneously in one multi-shift iteration. Again the same bulge chasing technique, generalized to accommodate any arbitrary number of shifts, performs this implicitly.

While multi-shift *QR* algorithm looked promising, in 1991 Dubrulle [2] presented a number of numerical experiments that show that the multi-shift *QR* algorithm does not perform very well if the number of shifts used in one multi-shift step is taken too large.

In an attempt to shed light on this convergence problem, Watkins [4] was able to identify the mechanism by which the values of the shifts are transmitted from one bulge matrix to another. His main result was that the finite eigenvalues of the matrix pencil B-μN are precisely the values of the shifts that were used in the multi-shift step. Here B is the bulge matrix and N is the Jordan nilpotent matrix of the same order as B.

Further, Watkins has detected a phenomenon called *blurring of shifts*, where the values of the shifts are incorrectly transmitted form one bulge matrix to another. Round-off errors is the apparent cause. It turns out that the characteristic polynomial of the matrix pencil B-μN, the finite zeroes of which are the shifts, is highly sensitive to perturbations of its coefficients. Since the values of the shifts are incorrectly transmitted during the bulge chase, the *QR* algorithm misses the true eigenvalues of the matrix.

2. Objectives and Methodology

In the light of the observations of Watkins [4], this study investigates the phenomenon of the blurring of shifts using extended precision arithmetic. We determine the maximum empirical number of shifts that can be used in one multi-shift step and its corresponding level of precision that keeps the shifts from blurring. Three types of matrices were studied with orders between 8 and 15 (very small). All computations were done in UBASIC allocating a maximum of 250 words per variable. All internal computations were done in 542 words per variable.

For each matrix of a given order, a set of shifts was used in the multi-shift QR algorithm and the bulge matrices were extracted from the iteration matrix. For each bulge matrix B_i, the finite zeroes of the characteristic polynomial of B_i-μN were computed and compared with the original values of the shifts. We note the maximum number *m* of shifts that can be used without exhibiting the blurring of shifts phenomenon and the minimum number of words used for the fractional part that can make this possible. This is determined as follows: beginning with a reasonably high number of worfor the fractional part, we continue to increase the number of shifts until the algorithm exhibits blurring. Having determined *m*, the maximum number of shifts, we begin to

decrease the words allocated for the fractional part until the algorithm begins to fail.

The three types of matrices studied were: random matrices with integer entries taken from the set {0,1,2,…,99}, the Hilbert matrix H=(h_{ij}) where h_{ij}=1/(i+j-1), and the tridiagonal matrix tridiag(-1,2,-1) which arises in finite-difference approximation of the second derivative.

3. Experimental Results

The tables below summarize our findings. The column labeled "Max *m*" gives the maximum number of shifts in one multi-shift iteration that were attained without exhibiting shift blurring. The column labeled "Min words" gives the corresponding minimum words per variable that can be used to attain the well-focused shifts.

Random matrix (0-99):

Size	Max *m*	Min Words
8	6	10
9	7	15
10	6	15
11	6	10
12	9	20
13	5	8
14	5	8
15	7	10

Hilbert matrix:

Size	Max *m*	Min Words
8	6	14
9	7	20
10	4	18
11	5	24
12	4	25
13	4	28
14	4	30
15	4	34

Tridiagonal:

Size	Max *m*	Min Words
8	6	6
9	7	12
10	7	10
11	6	10
12	6	12
13	5	6
14	5	6
15	5	6

We also made two observations that were not reported in the table. Firstly, some iterations start to exhibit shift blurring during the middle part of the multi-step bulge chase. For instance, with a random matrix of order 14, m=6, and 10 words for the fractional part, bulges B_0 and B_1 have well-focused shifts, while B_2 exhibits blurring. Secondly, some shift blurring become focused as the iteration progresses, but the error accumulates that the computed eigenvalues are too far from the actual values. Watkins [4] has also presented this second observation as an open question.

4. Conclusions and Recommendations

We summarize below the trends observed:

1. For matrices of order n > 12, the maximum m χ n/2. This is generally observed for matrices of higher order.

2. Hilbert matrices require greater precision for the fractional part.

3. Tridiagonal matrices require fewer words for the fractional part.

4. Some iterations exhibit blurring during the middle of the bulge chase, and some blurring become focused as the iteration progresses.

We recommend as possible future work a search for a closed formula for the maximum number of shifts in one multi-shift iteration. In the light of conclusions 1 and 2, this closed formula might be a function of the order and the condition number of the matrix. A closed formula for the minimum number of words that can be used to prevent the shifts from blurring may also be found.

References

[1] Z. Bai and J. Demmel, "On a Block Implementation of the Hessenberg Multi-Shift QR Iteration ," *Internat. J. High Speed Computing*, 1 (1989), 97-112.

[2] A. A. Dubrulle, "The Multi-Shift QR Algorithm –Is It Worth the Trouble?", *Tech. Report G320-3558x*, IBM Corp., Palo Alto, 1991.

[3] J. G. F. Francis, "The QR Transformation, parts I and II," *Computer J.* 4 (1961), 265-272, 332-345.

[4] D. S. Watkins, "Transmission of Shifts and Shift Blurring in the QR Algorithm," *Linear Algebra Appl.*, 241-243 (1996), 877-896.

[5] D. S. Watkins, and L.Elsner, "Convergence of Algorithms of Decomposition Type for the Eigenvalue Problem," *Linear Algebra Appl.,*143 (1991),19-47.

Building GraphBased Symmetric Cluster

Felix P. Muga II
Mathematics Department
Ateneo de Manila University
Quezon City, Philippines
fpmuga@admu.edu.ph
Rafael P. Salda˜na
High Performance Computing and Networking Laboratory
Ateneo de Manila University
Quezon City, Philippines
raf@admu.edu.ph
William Emmanuel S. Yu
High Performance Computing and Networking Laboratory
Ateneo de Manila University
Quezon City, Philippines
wyy@admu.edu.ph

ABSTRACT -- In August 2000 the High Performance Computing Research Group (AHPC) of the Ateneo de Manila University built an 8-node Beowulf- class computer designed for computational science applications. As more researchers and students in the University are trained in cluster computing, the need for building a better cluster arises. This year the AHPC proposes to build a large-scale graph-based symmetric cluster. The proposed high performance computing system will be a symmetric cluster with a single-switch latency and f lat networking neighborhood topology. The proposed design also features minimized cost and maximized bandwidth. This presentation will deal with mathematical and computational aspects of graph-based clusters, and design considerations for a large-scale symmetric cluster with a single-switch latency.

KEYWORDS -- symmetric, balanced and flat network neighborhoods, isomorphic graphs, bisection bandwidth, pairwise node bandwidth, regular graphs, switch latency, cluster computing, beowulf, parallel computing

1. INTRODUCTION

Cluster computing is becoming an accepted form of supercomputing. In universities, government institutions and commercial companies, there is a growth in the cluster installation base. In the international scene, there is a race to build the biggest and the fastest clusters.

The popularity of cluster computing is growing among scientific computing and research communities. It is also expanding in the commercial sector, and a large number of very large scale clusters are being deployed. However, according to Amdahl's Law, the speedup of a system is limited by the speedup of a single component in such a system. This is true in the case of cluster computing. It is not simply a case of adding compute nodes to the cluster to make it perform better. Other factors, such as interconnection network, will cause a performance bottleneck.

To improve the performance of a supercomputing cluster, it is important to eliminate bottlenecks. Limitations in network switch sizes, latencies and other network devices do not make this task easier. The use of alternative neighborhood networks can help answer these network limitations.

In August 2000 the High Performance Computing Research Group (AHPC) of the Ateneo de Manila University built an 8-node Beowulf- class computer designed for computational science applications[10,11]. As more researchers and students in the University are trained in cluster computing, the need for building a better cluster arises. This year the AHPC proposes to build a large-scale graph-based symmetric cluster. The proposed high performance computing system will be a symmetric cluster with a single-switch latency and flat networking neighborhood topology. The proposed design also features minimized cost and maximized bandwidth. This presentation will deal with mathematical and computational aspects of graph-based clusters, and design considerations for a large-scale symmetric cluster with a single-switch latency.

2. THE NETWORK PROBLEM

Parallel computation is typically composed of tasks that are parallel and some tasks that are not. Parallel tasks are those tasks that can be accomplished simultaneously with or without active communication. Serial tasks are those tasks that have to be completed one after another in a proper sequence. Serial tasks are usually irreducible and are treated as fixed computational overhead and at times it can expand. Parallel tasks have ideal completion times like $1/N$ where N is the number of parallel tasks undertaken at the same time[5]. Parallel task, however, require a communications overhead between tasks. All of these are made formal in Equation 2 referred to as Amdahl's Law and quantitatively corrected in books in parallel computation[5].

$$\frac{R(P)}{R(1)} = \frac{(T_s + T_p)}{(T_s + (\frac{T_p}{P}))} \quad (1)$$

$$\frac{R(P)}{R(1)} = \frac{(T_s + T_p)}{(T_s + (P * T_{is}) + (\frac{T_p}{P}) + T_{ip})} \quad (2)$$

This law strictly limits the amount of speedup that can be attained from a paralleled program. However, this equation does not consider some other factors such as T_{is} or the average serial time which includes time delays due to Inter-process communications, setup, initialization and other. Another factor is T_{ip} which is the average parallel time spend by each processor performing tasks like initialization, setup and even idle time. With these in mind the more realistic form of Amdahl's Law is shown in Equation 2.2.

It can be seen that the speedup of the entire system is severely limited by this law. The communications overhead while is part of the computation can severely limit the speedup gained. Thus, a large number of cluster nodes cannot be justified if the network will simply reduce its benefits.

3. AGILA NETWORK DESIGN – A GRAPH BASED CLUSTER

We are proposing to enlarge the Athlon Beowulf cluster of the Ateneo High Performance Computing Group known as AGILA from the present set-up of 15 nodes to 256 nodes. Using a 4-way motherboard, our new cluster system will have a total of 1024 processors.

The topology of the cluster system we are proposing is based on 16 copies of a circulant graph of order 16 with jump sizes $\pm 1, \pm 4, \pm 7, 8$. The vertices of the graph are labelled 0, 1, 2,…, 15 such that vertex u is adjacent to the vertex $u \pm 1$, $u \pm 4$, $u \pm 7$, $u + 8$ (the sum is taken under modulo t).

The next sections discuss the theoretical basis of our proposed topology.

3.1 A GraphBased -Cluster

Let G be a graph with vertices 0, 1, 2, . . . , t - 1 and let G_0, G_1, . . . , G_{m-1} be m copies of G. Then the vertices of G_i are labelled as i, $m + i$, $2m + i$, . . . , $tm - m + i$ and the order of G_i is t. (Note that *order* of a graph means the number of vertices of a graph.)

Consider a cluster whose compute nodes are the vertices of the m copies of G. Let us partition the mt compute nodes of the cluster into t subnets such that each subnet C_k consists of the nodes km, $km + 1$, $km + 2$, . . . , $km + m - 1$. We connect the nodes of the cluster to t network switches S_0, S_1, . . . , S_{k-1} using the *node-to-switch connection procedure* given below.

Procedure NTS-1

Let u be a compute node belonging to subnet C_k. Then

NTS-1. connect node u to switch S_k.

NTS-2. connect node u to S_{k_1}, S_{k_2}, . . . , $S_{k_{r-1}}$, if k is adjacent to

vertice $k_1, k_2, \ldots, k_{r-1}$ in G.

Let us denote the cluster based on m copies of a graph G of order t and connected using the node-to-switch connection procedure by $C_G(m, R, t)$ where R is the set $\{r_0 + 1, r_1 + 1, \ldots, r_{t-1} + 1\}$ and each r_k denotes the degree of vertex k in G. If $r_0 = r_1 = \ldots, r_{t-1}$ G is called a regular graph. An example of a regular graph is the circulant graph $G(t; \pm s_0, \pm s_1, \ldots, \pm s_r)$ such that the vertices are labelled as 0, 1, . . . , t - 1 and each vertex v is adjacent to vertices $v \pm s_0, v \pm s_1, \ldots, v \pm s_d$ where addition is taken under modulo t. If t is even, then $\frac{t}{2} \equiv -\frac{t}{2} \pmod{t}$. Hence, if $s_d = \frac{t}{2}$ when t is even, the regularity of the circulant graph is odd. Consider the following example.

NTS-1 accounts for 1 network switch for each node, while NTS-2 accounts for r - 1 switches for each node. Hence, each node in the cluster is attached to r network switches. Therefore, there are r NICs installed in each node.

Theorem 1 *Let G be a graph of order t and regularity r - 1. Suppose that the cluster $C_G(m, R, t)$ uses s-port switches. Then $m \leq \left\lfloor \frac{s}{r} \right\rfloor$, i.e ., each subnet can have at most $\left\lfloor \frac{s}{r} \right\rfloor$ compute nodes.*

Proof. The total available number of ports is st. If m copies of the regular graph G are to be used, then the cluster uses mrt ports.

Hence, $mrt \le st \Leftrightarrow mr \le s \Leftrightarrow m \le \frac{s}{r}$.

Since m is a positive integer, $m \le \left\lfloor \frac{s}{r} \right\rfloor$.

Example 1

In Fig. 1, the cluster is based on 6 copies of $G(8; \pm 1, 4)$. This is a 3-regular circulant graph and has 8 vertices. Since NTS-1 connects a node to one switch and NTS-2 connects the node to three other switches, each node in the cluster needs 4 1-port NICs or 1 4-port NICs. If the cluster uses 24-port switches, then it must have $\left\lfloor \frac{24}{4} \right\rfloor = 6$ copies of the base graph. Hence, the cluster has $8 \times 6 = 48$ compute nodes partitioned into 8 components $C_0, C_1, \ldots, C_7$ where each component C_k has the six nodes km, $km + 1$, $km + (m - 1)$. The total number of installed NICs is $4 \times 48 = 192$. The cluster needs 8 network switches labelled as $S_0, S_1, \ldots, S_7$.

A node symmetric cluster or network has no distinguishable node. The "view" of the rest of the network cluster is the same from any node. Rings, fully connected networks, and hypercubes are all node symmetric network. This property is similar to that of the *vertex-transitive graph*. Hence, a cluster C is *node-symmetric* (or *vertex-transitive*) if there exists an *automorphism* ϕ from the cluster's node set $V(C)$ onto itself. If a cluster is node-symmetric we simply call it as a symmetric cluster.

Theorem 2 *If G is a vertex-transitive graph, then the cluster $C_G(m, R, t)$ is symmetric.*

Proof. Let G be a vertex-transitive graph of order t. Define a mapping φ from the vertex set of $C_G(m, R, t)$) onto itself by $\varphi(km + i) = \phi(k)m + i$ where ϕ is an automorphism of of the vertex set of G onto itself. Clearly, φ is an automorphism from the vertex set of $C_G(m, R, t)$ onto itself. Therefore, the cluster $C_G(m, R, t)$ generated by the vertex-transitive graph G is symmetric.

Theorem 3 *If G has diameter* 1 *or* 2*, then every pair of nodes in $C_G(m, R, t)$ has a common switch. Therefore, $C_G(m, R, t)$ has the FNN topology.*

Proof. Let $u = k_1m + i_1$ and $v = k_2m + i_2$ be two distinct compute nodes in the $C_G(m, R, t)$. If $k_1 = k_2$, then the two nodes belong to C_k. Hence, they are joined by switch S_k. Suppose that $k_1 \neq k_2$. If k_1 and k_2 are adjacent in G, then u and v are joined by two switches. If k_1 and k_2 are not adjacent (only when diameter is not 1), then they have a common neighbor since G has diameter 2. Hence, u and v are joined by a switch. Therefore, $C_G(m, R, t)$ has the FNN topology.

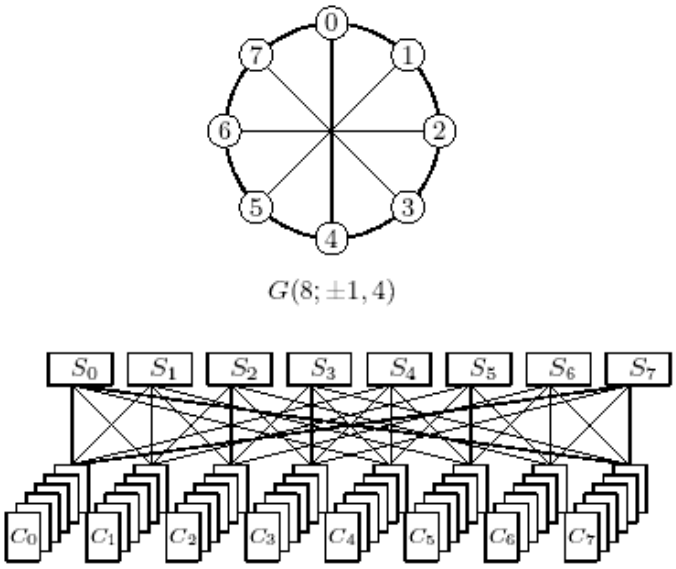


Figure 1. A cluster based on 6 copies of $G(8; \pm 1, 4)$ cluster $C_G(m, R, t)$

4. A PROPOSED GIGANTIC CLUSTER DESIGN

In this section, we shall show a design of a symmetric FNN cluster with about 1000 processors. The base graph is a circulant graph with 16 vertices and jump sizes $\pm 1, \pm 4, \pm 7, 8$. See the graph in Fig. 2.

Since the base graph has $t = 16$ vertices, the cluster needs 16 switches. Also, the base graph is 7 regular. It follows that each compute node needs two 4-port NICs or $r = 8$. On a 128-port 100 Mbps Fast-Ethernet switches, the number of compute nodes per subnet is at most $\left\lfloor \frac{128}{8} \right\rfloor = 16$. Hence, $m = 16$. Each subnet corresponds to a network switch. Thus, the $C_G(16, 8, 16)$ cluster has $16 \times 16 = 256$ compute nodes. Using 4-way motherboards, the $C_G(16, 8, 16)$ cluster can have at most $256 \times 4 = 1{,}024$ processors.

4.1 Bisection Bandwidth of the GraphBased Cluster

A communication link between two nodes in a cluster using the *FNN* topology is the connection from one node to a switch and the connection from the switch to the other node. The number of communication links between two distinct nodes is defined as the *pairwise bandwidth* of the given pair.

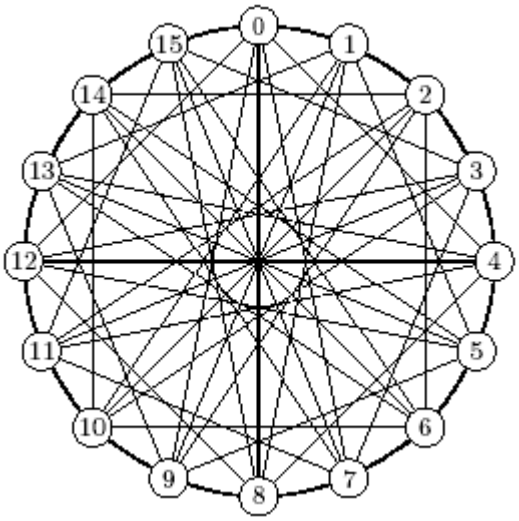


Figure 2. Circulant G (16; ± 1, ± 4, ± 7, 8)

The minimum number of communication links that must be removed to partition a network into two equal halves is defined as the *bisection bandwidth* of *G* is denoted by *BBW* (*G*). This type of partition is known as the *bisection* of the *G*.

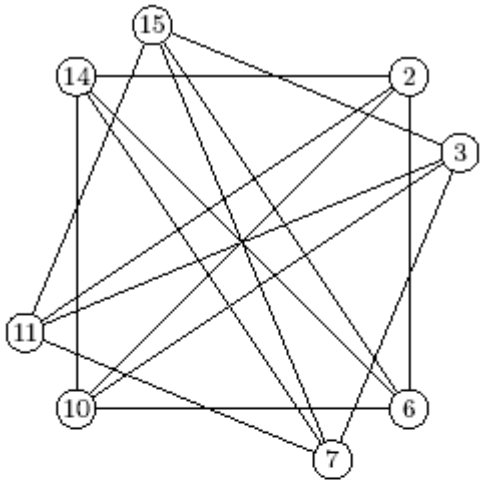
The bisection bandwidth of a *tree* is one, since any partition cuts across only one communication links. The bisection bandwidth of a *hypercube* of order 2^d (number of nodes) is 2^{d-1}, since at least 2^{d-1} communication links must cross any partition of a hypercube into two subcubes. The bisection bandwidth of a complete graph of order p is $p^2 / 4$, if p is even, or $(p^2 - 1) / 4$, if p is odd.

Switches are the *de-facto* standard component for a cluster system. Switches are used to connect different nodes in such a way that each node is given a guaranteed bandwidth. A switch S_k with the nodes in component C_k determines a subnet of nodes of the graph-based cluster. Switch S_k is associated with vertex k in G. If two vertices k_1 and k_2 are adjacent in G, then the subnets determined by S_{k_1} and S_{k_2} are also "adjacent", because they share the same nodes in C_{k_1} and C_{k_2}. Hence, If the edge that is incident to k_1 and k_2 is removed, then the communication links between the nodes in C_{k_1} and S_{k_2} and between the nodes in C_{k_2} and S_{k_1} are removed. There are $2m$ such links. Hence, we have the following theorem.

Theorem 4 *The bisection bandwidth of* C_G (*m, r, t*) *is* $2m \times BBW(G)$.

The bisection width of Circulant *G* (16; ± 1, ± 4, ± 7, 8) requires at least 16 edges to partition the graph into two equal halves. See Fig. 3.

Hence the (bidirectional) bisection bandwidth of C_G (16, 8, 16) is $2 \times 16 \times 16 \times 200$ Mbps = 102.4 Gbps edges.



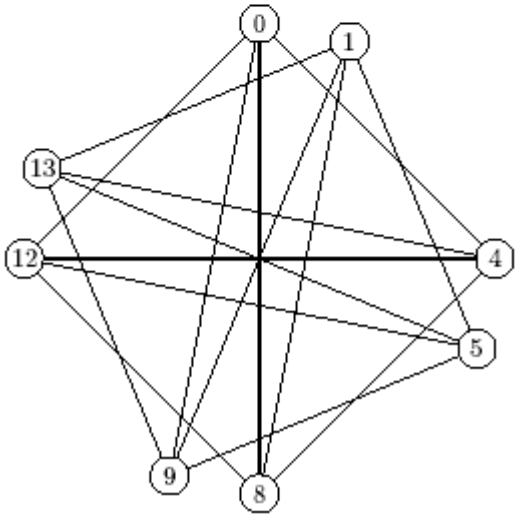


Figure 3. A Bisection of G (16; ± 1, ± 4, ± 7,8)

Theorem 5 *Suppose that* t *is the total number of switches used,* s *be the total number ports used per switch and the cluster has* n *compute nodes. Then the average pairwise links between two compute nodes is*

$$\frac{\binom{s}{2} \times t}{\binom{n}{2}}.$$

Proof. Since the cluster has *t* available switches with *s* available ports per switch, it follows that the total possible links of the cluster is $\binom{s}{2} \times t$. The cluster has n compute nodes. Hence,

the total number of pairs of compute nodes is $\binom{n}{2}$. Therefore, the average number of links between two nodes is

$$\frac{\binom{s}{2} \times t}{\binom{n}{2}}.$$

The average number of links of the C_G (16, 8, 16) cluster is $\frac{\binom{128}{2} \times 16}{\binom{256}{2}}$ = 3.98431373 or an average bidirectional pairwise bandwidth of 796.862746 Mbps.

5. CONCLUSION

In this paper we have discussed the theoretical basis of our proposed topology for building graph-based symmetric clusters with single switch latency.

6. ACKNOWLEDGMENT

This project received partial funding from the Philippine Commission on Higher Education Center of Excellence (CHED-COE) Grant of the Ateneo de Manila University.

References

[1] Ahmad, I., “Gigantic Clusters: Where Are They Going?” IEEE Concurrency, 83-85 (April-June 2000).

[2] Becker, D., Sterling, T., Savarese, D., Fryxell, B., and Olson, K., "Communication Overhead for Space Science Applications on the Beowulf Parallel Workstation." High Performance and Distributed Computing, 1995.

[3] Cand R., Dumas, E., Menel, F., Linux Kernel Book. (England, John-Wiley & Sons Ltd), (1997).

[4] Dietz, H. G. and Mattox T. I., "Compiler Techniques For Flat Neighborhood Networks," 13th International Workshop on Languages and Compilers for Parallel Computing(LCPC00). New York, August 11, 2000.

[5] Foster, I., Online Book: Designing Parallel Programs. Viewable online at http://www-unix.mcs.anl.gov/dbpp/.

[6] Katz, D., "Beowulf Applications and User Experiences." Seventh International Symposium on High Performance Distributed Computing. (1998).

[7] Muga, F.P. II, “Maximal Order of 3- and 5-Regular Circulant Graphs”, Matimyas Matematika, (1999).

[8] Muga, F.P. II, “Designing Clusters Using Scalable, Symmetric, Balanced and Flat Neighborhood Networks”, submitted for presentation.

[9] Reschke, C., Sterling, T., Ridge,D., Savarse, D., Beer, D., Merkey P., "A Design Study of Alternative Network Topologies for the Beowulf Parallel Workstation." Proceedings of the 5th IEEE International Symposium on High Performance Distributed Computing 1996.

[10] Salda˜na, R.,J. Garcia, F. Muga II, W. Yu, "Development of a Beowulf-Class High Performance Computing System for Computational Science Applications." Proceedings of the 18th National Physics Congress, Palawan, Philippines2000.

[11] Salda˜na, R.,"Project AGILA: the Ateneo High Performance Computing System." Proceedings of the First Philippine Computing Science Congress, Manila, Philippines 2000.

The Development of Myrinet Driver for DP Low Latency Message Passing System

Theewara Vorakosit, Putchong Uthayopas
Parallel Research Group, CONSYL
Department of Computer Engineering,
Faculty of Engineering, Kasetsart University
Bangkok, Thailand.
Email: g4465018@ku.ac.th, pu@ku.ac.th

ABSTRACT -- Low-latency communication system is crucial for the performance of parallel application on Beowulf cluster systems since its reduced the overhead of important operation such as barrier synchronization. This paper presents our work on the design and development of Myrinet driver for DP, low latency communication system in cluster environment. This driver allows user to exploit DP capability on fast message transmission on Myrinet network.

This driver is implemented as a loadable kernel module, which can be load or unload without modifying or recompiling the OS kernel. The performance has been measure and clearly shows the good improvement over traditional UDP transmission.

KEYWORDS -- Cluster System, Communication Latency Reduction, Myrinet, Linux Driver

1. Introduction and Related work

High bandwidth, low latency network communication subsystem is essential for cluster system since many important synchronization depends on the fast transmission of short massages. By reducing the latency involved, parallel message passing applications can gain much higher performance on cluster system. Traditional Generic communication protocol such as TCP/IP is too complex for cluster systems since they are not designed to be a "local" protocol in such system. Hence, many communication subsystems are developed to be use in cluster systems. The examples of such works are Directed Point[7], Active Message[2], Fast Message[3], U-Net[4] and Virtual Interface Architecture[5].

DP is one of the implementation that seems to be very interesting in many aspects. This includes a well-protected kernel level implementation, fast and low overhead system call, well define and low overhead architecture. But the problem is that current DP implementation only supports FastEthernet, share memory, and ATM.

One of the most used network switch technology is Myrinet from Myricom. Myrinet is a robust, scalable, and high performance high bandwidth network technology. Myrinet has many useful features to use in cluster such as high bandwidth, ANSI standard, and scalability. Myrinet is fully programmable at the NIC level. This facts is a motivation for the development of a Myrinet driver for DP system on Beowulf cluster so that users can fully exploit the performance of Myrinet-base Beowulf cluster with DP technology.

The design of driver aims at achieving a low latency for short message and high bandwidth for large message. To achieve our goal, the complexity of original Myrinet driver from Myricom [6] has been reduced and other functionality require for the driver such as registering to DP, source route configuration, memory map utility are added. The developing Myrinet driver is still a challenging work due to the programming complexity in kernel level. Hence, this is also the motivation for the selection of GNU/Linux system as an implementation platform since GNU/Linux is a free software and all kernel source code is available.

2. Background

Directed Point (DP) is a communication subsystem for parallel computing that comes from the research project at University of Hong Kong. The DP communication subsystem employs a high-level abstraction to express the interprocess communication in a cluster. In DP model, each node in the cluster is assigned a logical identity called *Node ID (NID)*. Each endpoint of the directed edge is labeled with a unique Directed Pont ID (DPID). Each program can use an association of 3 tuples {*local DPID, peer Node ID, peer DPID*} to identify a communication channel.

The DP subsystem consists of three layers, namely, *application programming interface (API) layer, service layer,* and *network interface layer*. The DP *API layer* provides a way to use DP system. The DP *service layer* is the core of the DP subsystem that provides services for message delivery. This layer is responsible for the delivering of messages from user space to network hardware level. It also helps deliver

the incoming packets to the target DP end point. The DP *network interface layer* provides an interface for DP *service layer* to interact with the network hardware. Figure 1 illustrates the DP system architecture.

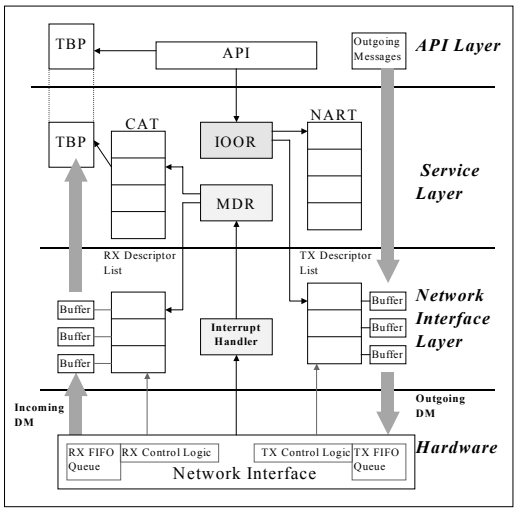


Figure 1. The architecture of DP communication subsystem

Myrinet is a switching technology that is widely used to interconnection for high-performance cluster systems. Some features of Myrinet are:

- Full-duplex 2+2 Gigabit/second links, switch ports, and interface ports.
- Flow control, error control, and "heartbeat" continuity monitoring on every link.
- Low-latency, cut-through, crossbar switches, with monitoring for high-availability applications.
- Scalable to tens of thousands of hosts, with network-bisection data rates in Terabits per second, and can also provide alternative communication paths between hosts.
- Host interfaces has build-in microcontroller called LANai that execute a control program to interact directly with host processes ("OS bypass") for low-latency communication, and directly with the network to send, receive, and buffer packets.
- Support any topology and protocol.
- Conform to American National Standard ANSI/VITA 26-1998

Myrinet card has a memory space of 16 MB. LANai memory is between address 0 to 0x800000. The block diagram of Myrinet card is shown in Figure 2.

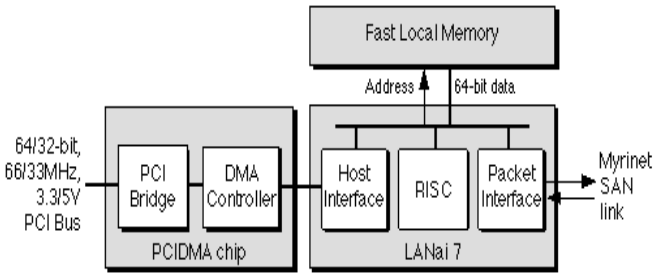


Figure 2. The block diagram of Myrinet NIC

3. Implementation of Myrinet Driver for DP

Myrinet driver is designed to work s a network layer of DP. The driver consists of 2 parts: the LANai control program and Linux Host Driver. LANai control program is a program that execute on LANai processor on Myrinet board. The Linux host driver is a driver execute in Linux Kernel. The Linux host driver and LANai control program communicate using hardware level share memory.

In LANai, the memory available is a 2 MB SRAM (expandable to 8 MB). The LANai memory is divided into 7 sections as follows:

1. Myrinet control program region for Myrinet control program (MCP). The size of this program in our driver is about 256 kB. At the end of this region, it is a base stack pointer. We have to move base stack pointer to this address in order to use the rest of LANai memory. This region if from 0 to 0x3ffff.
2. Blank region that acts as a guard between MCP and other regions. This region is ranging from address 0x40000 to 0x4ffff.
3. Command region. This region allows host and LANai to write and read the commands and status codes such as sending command, busy flags and so more. This region ranges from address 0x50000 to 0x5ffff.
4. DMA control block region. Myrinet NIC contains a DMA controller. The controller uses chains of control blocks stored in LANai memory to initiate DMA-mastering operations. There are 2 chains, one for sending and another one for receiving. This region is located at the address 0x60000 to 0x6ffff.
5. Source route table region. This region is used to maintain the source route table. System administrator has to configure source route table for each node statically. When sending a packet, LANai will search for a source route for target host from DP header. This region is from 0x70000 to 0x7ffff. In this version, the driver supports 6 bytes source route. That means cluster can span to maximum of 6 switches or a few hundred nodes. This table can contain up to 21845 hosts.
6. Send buffer. This region is used to store the outgoing packet to be sent. Only one packet can be stored in this region at a time. MCP supports up to 65536 bytes of

data. However, the current version of DP supports only 1500 bytes. This region is ranging from address 0x9000 to 0x9ffff.

7. Receive buffer. This region is used to store the incoming packet. This region ranges from address 0x10000 to 0x10ffff.

For the driver to work, the LANai controller code has been developed using C language. This C code is the compiled to LANai machine code using cross compiler available from Myricom. The LANai is programmed logic is a state machine. Figure 3 shows the flowchart that explains the LANai controller code logic.

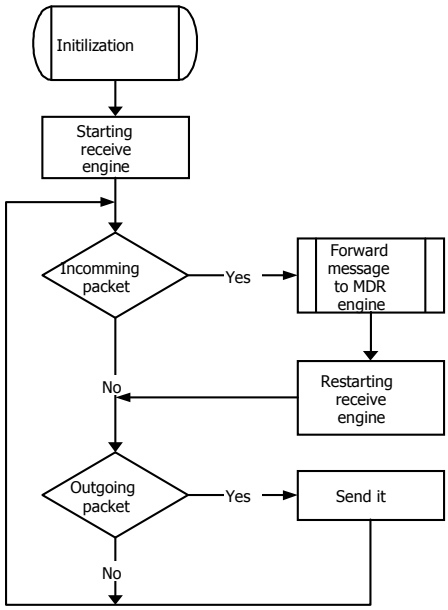


Figure 3. Flow chart for LANai Send Receive Operation

For the Linux host driver, the driver consists of 4 main functions: sending function, receiving function, memory-mapped function, and source route configuration and registration function. The operations of each function can be briefly summarized as follows:

- Sending function responsible for the sending of outgoing packet. This module is called directly by DP service layer. The transmission of a message works as follow:
  1. Driver checks whether LANai is free.
  2. Driver creates DP header in kernel memory.
  3. Copy the header and data to LANai memory.
  4. Wait until the send operation in LANai completed.
- Receiving function receives the incoming packet. LANai will receive all incoming packet at all time. If there is an incoming packet, the routine works the following way:
  1. LANai checks that there is an incoming packet and whether the incoming packet is a DP packet or not.
  2. LANai set the size of incoming packet in DMA control block and raises the interrupt to host driver.
  3. Driver triggers DMA transfer of incoming packet into host memory.
  4. Driver calls DP service layer to dispatch incoming packet.
- Memory-mapped function provides the mapping of all 16MB of LANai memory space into a part of the user space. To use this function, user have to create a character device file with predefine major number and minor number to be 0. The function is used to debug Myrinet.
- Source route configuration function provides a way to set source route for each node. Source route is required for the sending of packets to other node. In order to keep the overhead low, the source route table are configure statically. Source route configuration is registered in a file /proc/net/myri_route. User can configure source route using cat command to /proc/net/myri_route. This file can also be read.

The driver itself is implemented as a loadable kernel module. Hence, the driver can be loaded or unloaded from memory without modifying or recompiling the kernel. In order to support the listed functionally, driver composes of 7 major modules, which are:

- Initialization module – this module responsible for locating Myrinet NIC from a system using PCI function called in Linux and gets the pointer to configuration space if a Myrinet NIC is located. Next, the memory in Myrinet NIC is mapped as a part of kernel memory, so kernel can directly access the card. Finally, the routine will initialize Myrinet hardware, clear, and check all content of LANai memory.
- Loading module – this module load MCP program to LANai memory. MCP code is programmed in C language, which is compiled by LANai cross compiler to LANai machine code. This machine code is then converted into C array definition in order to simplify the loading task. LANai executable file can be converted to C array using "*gendat"* utility from Myricom and then included into a driver source code. Loading task is simplified to be only the copy of array content to LANai memory.
- Sending module. This module is a major part of this driver. Sending module is called by DP service region to send a packet to another node.
- Interrupt service routine. This routine is another major part of the driver. This routine is called when LANai interrupts host. LANai will interrupt host only when an incoming packet is a DP packet. This routine also help copy the data from LANai memory to DP service layer buffer.
- Character device and /proc Routine. This routine provides a convenient way to directly access the NIC. User can directly access a whole NIC as though the NIC is in user space using *mmap* system call as well as generic read/write system call. /proc provide a

convenient way to configure source route for that node. This module will create /proc/net/myri_route.

- Post-initialization module. The main function of this module is to register components such as interrupt service routine to a kernel. After finish the registration process, this routine will start the LANai operation.
- Clean up module. The driver can be unloaded from kernel using the *rmmod* command. The *rmmod* command will call clean up module to free all resources that are allocated during the work. Hence, this module has to free all resources before the driver are unloaded from kernel. The resource cleaned are things such as memory, interrupt line, /proc and so on.

4. Performance

The driver has been tested using 2 nodes from a Beowulf cluster called AMATA. These two system has the following setup: 1 GHz Athlon processor with 512 MB RAM connect through Myrinet switch, Linux Kernel 2.2.16, and Myrinet card model: M2M-PCI64A-2-50973 with LANai version of 7.2 and 2 MB board memory.

A ping-pong program has been developed to measure the round-trip time for message size ranging from 8 bytes to 1024 bytes. The comparison has been made between the newly developed driver and the performance of code with UDP over FastEthernet, UDP over Myrinet, and GM driver over myrinet. The results are as shown in Table 1 and the graph are plotted as illustrated in Figure 4. The time given in Table 1 is in the unit of Microsecond.

Table 1. Comparison of UDP over Fast Ethernet, Myrinet and DP over Myrinet

Message size	UDP Fast Ethernet	UDP Myrinet	DP Myrinet	GM Myrinet
1	52.3	62.8	26.25	15.99
2	47.8	47.8	26.75	15.98
4	48.5	47.8	27	16.02
8	48.0	47.0	27	16.02
16	46.8	47.0	27	16.02
32	48.5	48.0	27.25	15.96
64	51.8	49.0	28.25	16.78
128	53.5	52.5	29.75	19.98
256	58.0	61.8	36.5	30.18
512	68.5	68.3	40.25	38.42
1024	88.3	88.5	51.5	55.16

For the new driver, the graph in Figure 4 clearly shows a much lower latency compared to usual UDP over both Myrinet and Fast Ethernet. When compared with GM driver from Myricom, for small message size, GM driver is a little bit faster (about 10 Microsecond) that our DP driver. The DP driver has slightly lower latency when message size is larger than 512 bytes. The time different may cause by many factors. One of the most important factor is how fine tune the code has been done. The logical step to further fine tune the code is to profile them in more detail and look at the code tuning at assembly language level.

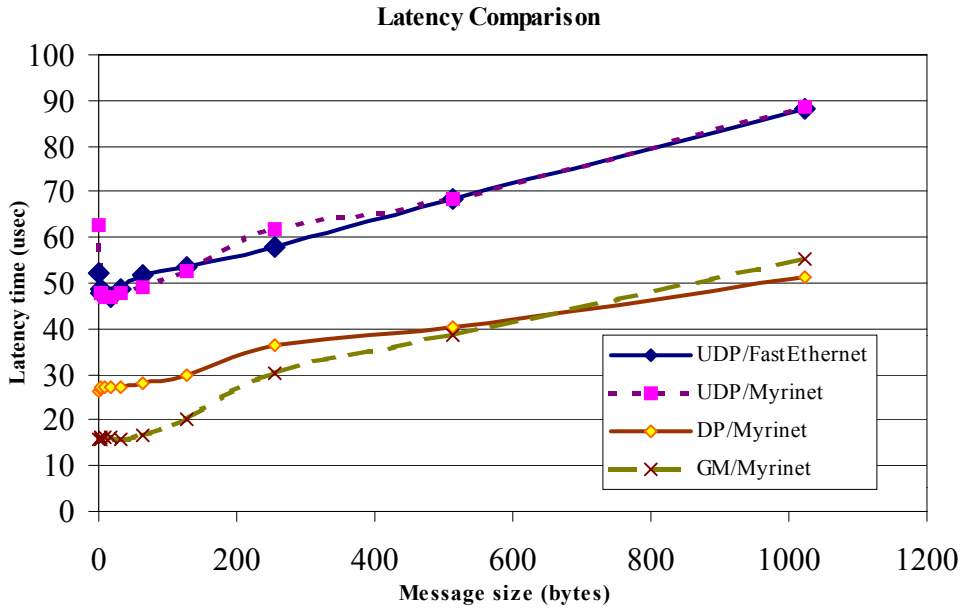


Figure 4. Performance comparison between Myrinet over DP and UDP

5. Conclusion and Future Work

In this paper, an implementation of Myrinet driver for DP system has been discussed with the experimental results. The techniques used in developing the driver have been explained. From the experimental results, the newly developed driver shown a satisfactory reduction of message latency although, there seems to be slight problem that slow down the Myrinet hardware transmission.

In the future, this driver will be used as a communication subsystem under a planned Pico-MPI that will be developed later. This implementation aims to explore the full optimize of message passing implementation from user space level to kernel level which has a high potential to generate fast and compact experimental MPI implementation.

6. Acknowledgement

This research has been partly supported by KU research and Development Institute under SRU grant. Most of the equipment used in the developed is sponsored by AMD Far East Inc.

7. References

[1] Chun-Ming Lee, Anthony Tam, and Cho-Li Wang, *Directed Point: An Efficient Communication Subsystem for Cluster Computing,* The University of Hong Kong, 1997

[2] T. von Eichken, D. E. Culler, S. C. goldstein and K. E. Schauser, Active Message: a Mechanism for Integrated Communication and Computation," *The 19th Int'l on Computer Architecture, 1992*

[3] S. Palin, M. Lauria and A. Chien, "High Performance Messages (FM) for Myrinet," *Supercomputing,* 1995

[4] Thorsten von Eicken, Anindya Basu, Vineet Buch, and Werner Vogels, *U-Net: A user-level network interface for parallel and distributed computing*. In Proceedings of the 15th ACM Symposium on Operating Systems Principles, December 1995. Available from

http://www.cs.cornell.edu/Info/Projects/ATM/sosp.ps

[5] D. Dunning and G. Regnier. *The virtual interface architecture*. InHot Interconnects V, pages 47--58, 1997

[6] Myricom Inc, *M2M PCI64A Product Specification*, http://www.myri.com/myrinet/PCI64/m2m-pci64a.html, July 2000

[7] Myricom Inc, *PCI64 Programmer's Documentation*,

http://www.myri.com/myrinet/PCI64/programming.html, July 2000

Ethernet-Based Interconnections for Massively Parallel Clusters[1]

Vara Varavithya and Thongchai Thepuatrakul
Department of Electrical Engineering
King Mongkut's Institute of Technology North Bangkok
1518 Piboonsongkram Rd., Bangsue Bangkok 10800, Thailand.
E-mail: vara@hpc.ee.kmitnb.ac.th

ABSTRACT -- Large production volume of the devices results in very low equipment cost based on Ethernet Technology. Advances mix-signal VLSI chip, DSP and analog signal, lead to two order of magnitude of bandwidth improvement in the Ethernet Network. The performance of the Ethernet technology is further enhanced by the development of Ethernet switches in which the aggregated bandwidth is much larger than the broadcast hub. The large scale clusters interconnect using Ethernet is considered as a economical solution. The Ethernet topologies for massively parallel clusters are discussed in the paper. Based on classical network topologies, we proposed Stack Ring and Stack Mesh topologies for the large systems. The stack of Ethernet switch is considered as a single lump node and connected using ring and mesh topologies. A set of the processors are assigned to perform the forwarding tasks. The topology definitions are defined and the IP assignment algorithms for both ring and mesh networks are presented. The effects of forwarding overhead is evaluated and HPL benchmark was tested on the system.

Keywords -- Cluster of workstations, network topology, Ethernet, high performance computing.

1. Introduction

A parallel high performance computing platform is made more accessible by interconnecting a group of workstations via a high speed interconnection network [1, 2, 3]. Examples of applications that can benefit from a cluster of computers, include computational fluid dynamics, weather forecast, bioinformatics, transaction computing, and Internet information servers. A Beowulf-class cluster [4] adopts commodity products, both hardware and software, to construct high performance parallel systems. It has been estimated in [5] that there are currently more than 100,000 clusters around the globe. The improvement in microprocessor and network technologies further drives the realization of these clusters.

A network with low latency and high bandwidth is required to sustain high performance in multicomputers [6]. These systems require that the network latency is in order of a few micro seconds and bandwidth is in order of a few Gbps. System area networks (SANs) [7, 8, 9, 10], are designed to transfer information at very high data rate in a relatively short distance. Although SAN offers very low latency and high bandwidth, the price of the SAN is expensive. Currently, the cost of a single SAN network interface card can exceed the cost of the computing node itself. ATM technology is another candidate as cluster interconnect. The cost of ATM equipments is still relatively high. Comprehensive treatments on interconnection networks for multicomputers is presented in [11].

The *Ethernet* technology is a strong candidate when the cost is considered as an important design requirement. Because of commodity products has very large production volumes the price of Ethernet devices is relatively low. The bandwidth of Ethernet technology has evolved from 10Mbps, 100Mbps, and 1Gbps where 10Gbps Ethernet is around the conner. The performance of current generation Ethernet network is lower than that of system area networks due to hardware speed and heavy communication library [12]. Higher bandwidth in the next generation hardware and supporting some of the communication protocol in the network interface cards will reduce the performance gap between the Ethernet and SANs. However, Ethernet-connected clusters have proven to be suitable for computational intensive applications and have been widely implemented [13].

Carrier Sense Multiple Access (CSMA), adopted in Ethernet, is a contention-based protocol in which network performance is severely degraded in high volume traffic. The collision problem is partially solved using hardware switch [14] at the data link level. *Ethernet Switch* has more aggregation bandwidth compared to the broadcast bus where, with no output port conflict, multiple communication messages can be exchanged in parallel. Most of the Ethernet devices today are shipped as switch-based devices. The contention Ethernet hub is obsolete from the market. In [15], multiple network interface units are implemented in the computing nodes to increase aggregate bandwidth. The channel bonding technique was proposed to provide alternative paths from the source to the destination

where the computing nodes are connected in the mesh-like topology.

This paper investigates the Ethernet topologies for large scale cluster interconnections that take advantage of contemporary consumer switching devices. There are several interesting issues in exploring Ethernet technology as an interconnection in closely connected cluster. A certain class of Ethernet switches has stackable capability. A set of Ethernet switches can be connected together using special backplane connection cables. The number of ports is multiply increased using this technique without performance degradation. We consider a group of stackable switches as a basic building block. The scalability of the Ethernetconnected cluster is therefore limited by the number of ports in one stack of switches, typically 80-120 ports. An Ethernet network for a large cluster requires more complicate details of implementation to maintain performance. Two classical network topologies, ring and mesh, are applied to the stacks of Ethernet Switches. We propose the *Stack-Ring* (SR) and the *Stack-Mesh* (SM) as interconnection topologies for Ethernet-connected massively parallel clusters. The SR/SM interconnects groups of *stackable switches* in ring/mesh with wrap around links. All destinations are being covered by assigning routing tasks to the nodes in a distributed manner. Beowulf-class clusters can be configured using regular IP addressing and forwarding schemes. The forward selection and routing setup schemes are presented for the proposed topologies. The communication overheads associated with the proposed topology were measured and its scalability is evaluated. The performance results of the real applications based on HPL benchmark were tested and compared.

This paper is organized as follows. Section 2 presents preliminaries. The proposed Ethernet topologies are discussed in Section 3. The network parameters and performance are compared in Section 4. Concluding remarks are drawn in Section 5.

2. Preliminaries

A cluster is a collection of workstations interconnecting via an interconnection network. Figure 1 (a) shows the general architecture of a cluster. A computing node is an autonomous computer which has its own processor(s), memory, hard drive, network interface, shown in Figure 1 (b). Communication between computing nodes is accomplished by passing messages. The performance of these computers (nodes) varies from the PCs to the high-end workstations. Due to high aggregation bandwidth and lower cost, the Ethernet switch is usually adopted as a core network devices. The developments of parallel applications can be efficiently achieved using parallel programming development platforms such as PVM [2] and MPI [16]. The system softwares that support networking and parallel programming need to be install on all the nodes. Another important service provided by the operating system is the network file system (NFS) that transparently services the file system to all the computing nodes.

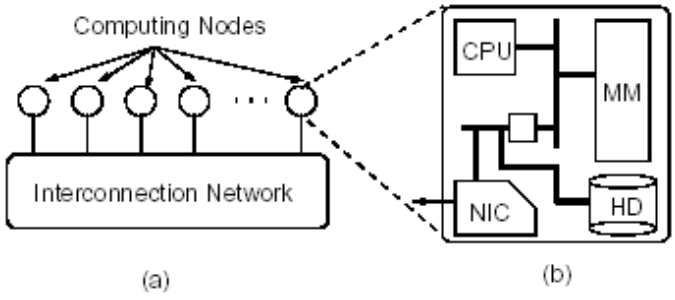


Figure 1. Cluster architecture: (a) Clusterarchitecture, (b) Computing node architecture

Relevant terminologies and network parameters are defined as follows. A *cluster interconnection network*, G is a strongly connected graph, $G(V,C)$, where V represents the set of vertices and C represents a set of physical links connecting the nodes. A set of vertices V in the cluster is divided into two sets, a set of *nodes* (computers) N_c and a set of *switches* N_s. A *node degree* I_c is the number of network interface at each node and a switches degree I_s is the number of ports at each switch. The switch backplane has peak bandwidth of *SP*. The maximum number of switches in a single stack is equal to SK_n. A group of switches in a single stack is considered as a lump switching device called *Stack Switch Box* (SSB). The maximum number of ports in one SSB is therefore equal to $I_s \times SK_n$.

The *network diameter* D is a shortest distance between any two remotest nodes in the network. Each link has *b* bit-per-second bandwidth. *Bisection Bandwidth BW* is the amount of information that can flow between two equal halves of the nodes in the network [11]. The performance metrics for cluster interconnects include *communication latency* and *message throughput*. The communication latency is the time elapsed between the initiation of the message and the reception of the entire message at the destination. The message throughput is the number of the messages delivered per unit time.

3. Ethernet Topologies

In this section, we present several Ethernetconnected topologies as interconnection for massively parallel clusters. For the completeness of the paper, some materials from [17] are reiterated.

3.1 Star Topology

In a small cluster, the nodes can be interconnected using a single Ethernets switch, as shown in Figure 2 (a) where there is only one switch delay between any nodes. The switch entity can be either single switch or a group of stackable switches. A message is first forwarded from the source node to the switch and then from the switch to the destination node. Only one network interface (I_c = 1) is required at each node. The maximum number of nodes N_c in the cluster is therefore limited by the number of switch ports I_s, typically 12-36 ports.

The Ethernet switches usually offer fully connection backplane where, in absent of output port conflict, all messages can be concurrently forwarded without contention.

We can increase bandwidth of the system using additional switches and network interfaces. The cluster can be configured such that the traffic is distributed among these interfaces. As shown in Figure 2 (b), each node is equipped with two network interfaces. The traffic can be distributed by equally dividing the nodes that generate messages to each interface or by differentiating traffic into subclasses. Two important classes of traffic in the cluster are intra-node communication and NFS communication. Separate intra-node and NFS communication enhances performance of the system especially for applications with a lot of file activities [18]. The number of interfaces I_c is limited by the number of I/O slots at the node, typically 4 to 6 PCI slots.

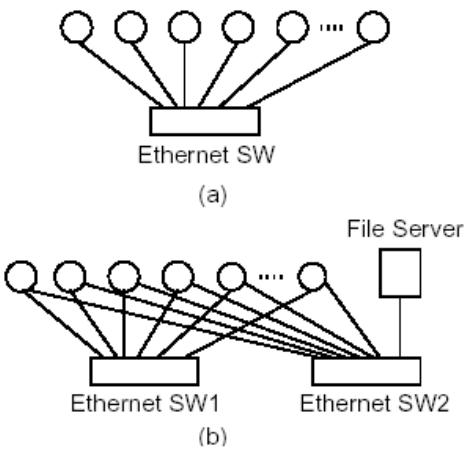


Figure 2. Cluster interconnection: (a) Simple cluster connection, (b) Separate network for NFS.

A certain type of Ethernet switches has stackable capability. The stackable switch has a backplane connection for interconnecting a group of switches together. In some technologies, the backplane bandwidth of an SSB still able to handle full connectivity. In current technology, four to seven switches can be stacked together. The number of nodes in a single SSB cluster is therefore in range of one hundred nodes.

3.2 Tree Topology

The size of the cluster can be exponentially scaled up by interconnecting switches according to the tree topology. The routing between the nodes is accomplished by switch learning. The 3-level tree network is presented in Figure 3 (a). In *M*-level tree, the nodes are connected to the (*M* - 1)-level switches. At least one port in the (*M* - 1)-level switch is used to connect upward to the root.

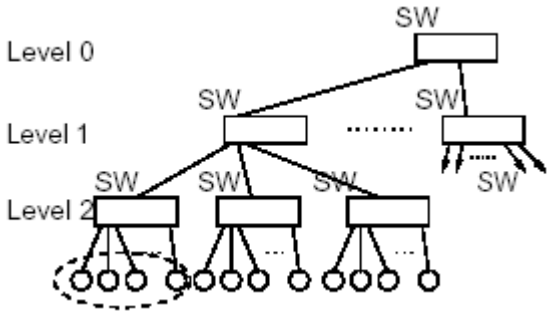


Figure 3. Tree topology.

The tree topology can provide connection up to very large size cluster. The maximum number of nodes in the system can be represented by $N_c = (I_s - 1)^{(M)} + (I_s - 1)^{(M-1)}$ nodes which is astronomically large even for a small number of tree levels. The tree topology is not suitable for communication intensive applications since links that connected toward the root node become bottlenecks in the communication paths between nodes in the different switches. Some switch technologies allow a set of links to be bundled. Using bundle links, a cluster can be build according to fat tree topology which alleviates the bottleneck problems. However, the number of bundled links is bounded by two to four links.

3.3 Stack-Ring Topology

As previously mentioned, the scalability of both single-hop and tree topologies is limited. Based on the ring topology, we propose a *Stack-Ring topology* (SR) to build a large scale Ethernet-connected cluster. Figure 4 shows the 5-stage SR network. In the SR topology, the computing nodes responsible for not only executing applications but also forwarding messages. The SR network has S_t stages. Two network interfaces are implemented at each node labeled as East (E) interface and West (W) interface. A single SSB belongs to one stage. The SSB i ($i \in (0, \ldots, S_t - 1)$) services communication requests from the nodes at the stage i and $|i-1| \bmod S_t$. The dashed oval in Figure 4 show the group of nodes serviced by the SSB in the stage 1. Each stage has the maximum of N_r nodes which is equal to $\frac{I_s \times SK_n}{2}$. The node r at the stage i is labeled as (r, i) where $0 \le r \le (N_r - 1)$ and $0 \le i \le (S_t - 1)$.

The traffic is divided into two classes, intrastage traffic and inter-stage traffic. The intra-stage traffic is the communication between the nodes within the same SSB, accomplished via the hardware Ethernet switch. The inter-stage traffic is the communication between the nodes that do not have direct data link level connections. For interstage communication, the nodes are responsible for forwarding parts of messages to their destinations. The message forwarding process is performed in software. Since the software routing incurs higher overhead compared to hardware routing, the shortest path from the

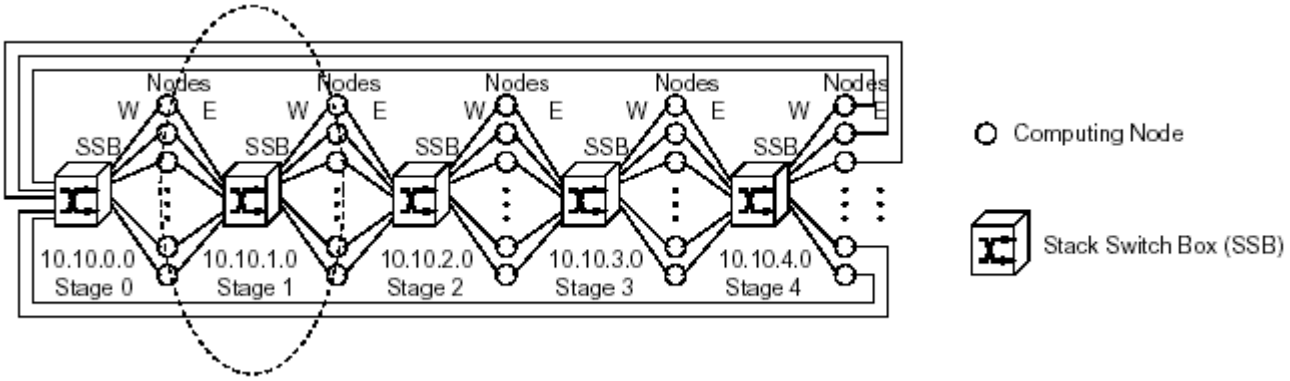


Figure 4. Stack Ring topology (SR): The topology consists of 5 stages with wrap around communication links. Each computing node has two network interface called East channel and West Channels.

source to the destination is selected using the algorithm shown in Figure 5.

The algorithm selects the interface that leads the destination with the minimum number of hops. The relative distance from the source to the destination is compared to provide routing information. For example, consider the nodes in the stage 1, Figure 4, the interface Wis selected for destinations belong to stage 0 and 4 and the interface E is selected for destinations in the stage 2.

Assuming that TCP/IP protocol is adopted in communication subsystem in the cluster, every node (x, i) is assigned an IP address to each interface. The nodes within the same stage are in the same subnetwork with IP:[10.10.*i*.*x*] for the interface A and IP:{[10.10.(*i* - 1).*x*], $i \neq 0$ and [10.10.(*St* - 1).*x*], $i = 0$} for the interface W. Hence, this IP configuration can support up to 127 nodes at each stage which covers available ports in the stack switch configurations. The routing table at each node is assigned according to information obtained from the interface selection algorithm. The load balancing of forwarding tasks is managed by the following scheme. The x^{th} node restricts the x^{th} nodes of the next/previous stage as its gateways to route messages to the nodes in other stages. Therefore the forwarding tasks are evenly distributed among all the nodes.

The table 1 shows an example of routing table assignment for the nodes in the stage 1.

Table 1. IP Forwarding table in the SR network.

Network	Interface	Gateway
10.10.0.0	W	10.10.0.x
10.10.1.0	E	N/A
10.10.2.0	E	10.10.1.x
10.10.3.0	W	10.10.0.x

3.4 Stack-Mesh Topology

The scalability of the SR topology is limited by the network diameter. The communication delay incurred in software forwarding process through the ring can degrade the performance if the message has to pass a large number of stages. Several popular topologies are classified as *orthogonal topology*. The nodes in the orthogonal networks can be arranged in the orthogonal n-dimensional space [11]. Due to their scalable properties, the orthogonal n-dimensional topologies are the basic topologies used in most contemporary multicomputers. Two important orthogonal n-dimensional topologies are the *n-dimensional mesh* and *k-ary n-cube* topologies and are defined as follows [6]:

Definition 1: An *n-dimensional mesh network* is defined as an interconnection network that has $k_0 \times k_1 \times k_2 \times \ . \ . \ . \ \times k_{n-1}$ nodes where k_i is the network radix of dimension i and n is the network dimension. The particular node is identified by the position in each dimension which can be represented by vector $(x_1, x_2, x_3, \ . \ . \ . \ , x_n)$. Two nodes, $(x_1, x_2, x_3, \ . \ . \ . \ , x_n)$ and $(y_1, y_2, y_3, \ . \ . \ . \ , y_n)$ are neighbors to each other if and only if there exists an i such that $x_i = y_i + 1$, and $x_j = y_j$ for all $i \neq j$.

Definition 2: An *k-ary n-cube network* is de- fined as an interconnection network that has n dimensions having k nodes in each dimension. The particular node in *k-ary n-cube* is identified by the position in each dimension which can be represented by vector $(x_1, x_2, x_3, \ . \ . \ . \ , x_n)$. Two nodes, $(x_1, x_2, x_3, \ . \ . \ . \ , x_n)$ and $(y_1, y_2, y_3, \ . \ . \ . \ , y_n)$ are neighbors to each other if and only if there exists an i such that $x_i = (y_i + 1) \ mod \ k$, and $x_j = y_j$ for all $i \neq j$. There are wraparound channels in the k-ary n-cube, with are not present in the ndimensional mesh networks. If $k = 2$, then every node has n neighbors. If $k > 2$, then every node has $2n$ neighbors.

```
nterface Selection Algorithm for SR
nputs:
. Address of the source node (R_s, X_s)
. Address of the destination node (R_d, X_d).
Output:
Selected Interface
Procedure:

egin
  if(X_s ≤ S_L/2)
    if((X_s ≤ X_d) and ((X_s + S_L/2) > X_d)
      Select interface E;
      return;
    else
      Select interface W;
      return;
  endif
  if(X_s > S_L/2)
    if((X_s > X_d) and ((X_s − S_L/2) ≤ X_d)
      Select interface W;
      return;
    else
      Select interface E;
      return;
  endif
nd
```

Figure 5. An interface selection algorithm for Stack-Ring topology.

The SM topology conforms to the *k-ary 2-cube network* since the *k-ary n-cube* network is regular and symmetric since all the nodes are identical compared to formal definition of the Mesh network. The node degree in the mesh network depends on its location. The utilization of channels in the center area of the mesh is higher than the channels near the edges. Figure 6 shows the four by four Stack-Mesh Topology (SM). The computing node has four network interfaces connected to the nearby SSBs. Interfaces are labeled as *N, E, S*, and *W*, according to their directions. The computing nodes in the systems is addressed as (*x, i, j*). *x* represents the node's rank in the group. The node group is surrounded by the dash line in the Figure 6. The location of the group of nodes in the network is specified by (*i, j*).

There are two issues need to be considered in the configuration of SM network, addressing and distribution of forwarding task. Assuming that the cluster adopts IP protocol in the message passing communication, the IP subnetwork is assigned to each node group. There are several approaches in assigning the IP addresses to SM network. We selects a simple method to ease of configuration. The node (*x, i, j*) is assigned the IP address of [10, *j, i, x*]. Using this IP addressing scheme, the SM network can scale up to 256× 256 mesh.

The forwarding table can be filled up using the algorithm shown in Figure 7. The shortest paths from the source to the destinations are selected. The message follows the dimension-order path in which the *X* dimension is traversed first and followed by the *Y* dimension. Similar to SR network, the forwarding tasks are distributed among the node. The gateways for the node [*x, x, x, i*] are configured according to the scheme in the Figure 7 with the IP address of [*y, y, y, i*] . The number of entries in the forwarding table is equal to the product of *N* and *M*.

4. Performance Comparisons

Both hardware switches and software forwarding are adopted in both SR and SM networks. To have better understanding on the overhead incurred in software forwarding, we have evaluated several communication subsystem performance tests on different network configurations using Netpipe. The maximum cluster size based on current technology is estimated for each topology. The HPL benchmark were tested and their results are discussed in the last part of the performance comparisons.

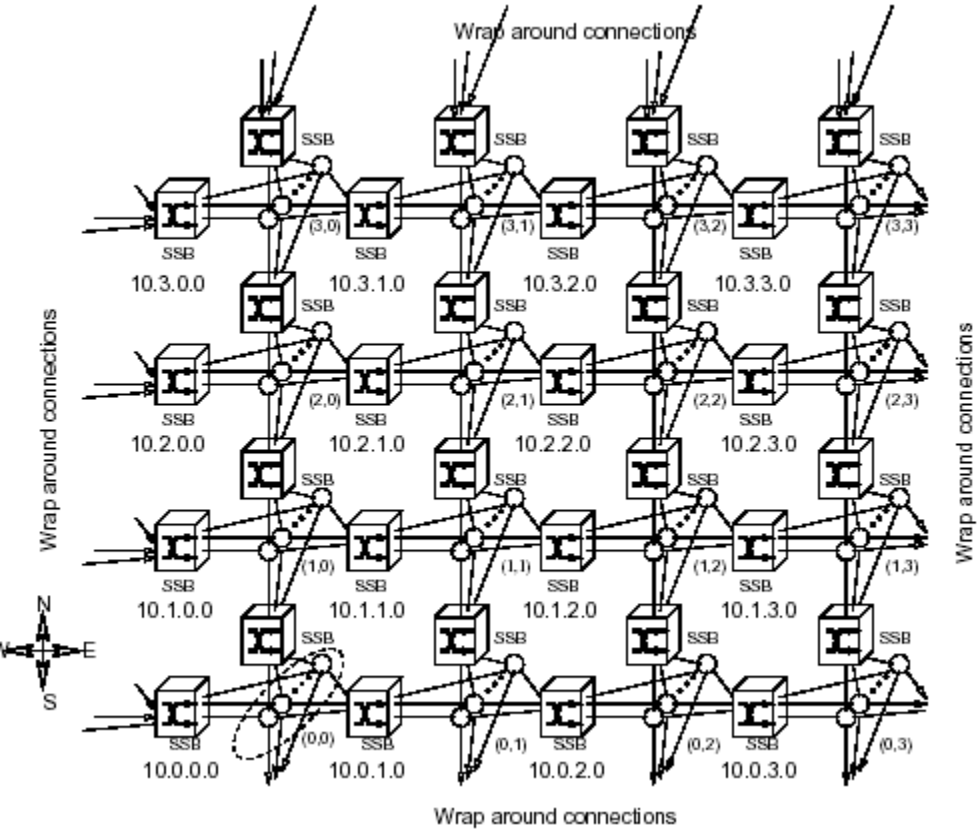


Figure 6. 4-by-4 Stack-Mesh topology for large scale clusters.

4.1 Software Forwarding Overheads

The network performance of several network configurations was measured using NetPIPE [19]. A set of Pentium-II 400MHz with 384MB of memory running LINUX Kernel version 2.2.9-27mdk and the Intel EtherExpress 520T switches were used in the experiments. The MPI routines were called to send/receive messages between a pair of nodes.

The differences in the transfer time of hardware-based and software-based are reflected in Figure 8. For the block size of 1 Kilo-bit, the transfer delays for direct connection, hardware Ethernet switch, software forwarding are measured as 146 μs, 151 μs, 208 μs, respectively. The hardware switching is 1.37 time faster than the software forwarding. In the Ethernet throughput graph, Figure 9 (a), the throughput of the hardware switch are very close to the direct connection (cross wire). The performance of two switch in series is slightly lower than the single switch. This is because the switching delay is small compared to the software overhead incurred during packet initiation and reception.

The throughput of software forwarding is 10-20% less than the hardware switch for the medium block sizes. The throughput of the communication further decrease when two-pairs of nodes send messages simultaneously. Similar trend is observed in the Ethernet signature graph, in Figure 9 (b). This is because, in software forwarding, the message is passing through two switches and one software forwarding. The message is first sent from the node to the switch and then is to the gateway node. The gateway node performs forwarding functions and then sends message to the second switch which is in turn forwarding the message to the destination node. From the results of the experiment, communication performance degrades considerably using software forwarding. The intergroup communication in both SR and SM networks should be minimized.

4.2 Scalability

We studied the scalable performance of Ethernet technology by estimating the maximum cluster size for different topologies. The 24-port fast Ethernet switch is considered as the network device building block. We make an assumption that the maximum number of switches in the same stack is equal to five and at most four network interfaces can be installed at each node. The maximum sizes of the star, tree, SR, SM topologies are estimated as follow.

Interface Selection Algorithm for $N \times M$ network
Inputs:
1. Address of the source node (R_s, X_s, Y_s)
2. Address of the destination node (R_d, X_d, Y_d)
Output:
Selected Interface
Procedure:

```
begin
    // Step 1: Forward in X dimension;
    if(X_s < X_d)
        if((X_d - X_s) < N/2)
            Select interface E;
            return;
        else
            Select interface W;
            return;
    if(X_s > X_d)
        if((X_s - X_d) < N/2)
            Select interface W;
            return;
        else
            Select interface E;
            return;
    // Step 2: Forward in Y dimension;
    if(Y_s < Y_d)
        if((Y_d - Y_s) < N/2)
            Select interface N;
            return;
        else
            Select interface S;
    if(Y_s > Y_d)
        if((Y_s - Y_d) < N/2)
            Select interface N;
            return;
        else
            Select interface S;
            return;
end
```

Figure 7. An interface selection algorithm for Stack-Mesh topology.

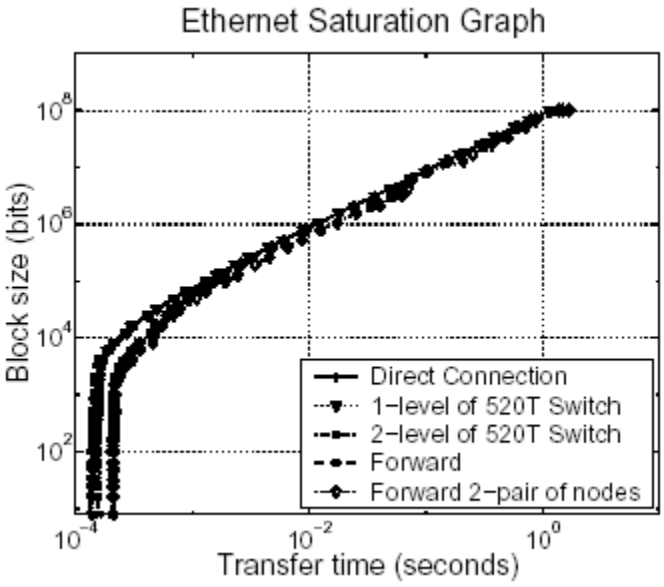


Figure 8. Network performance comparison: Block size versus transfer time.

Table 2. Summarize the scalability of the proposed Ethernet topologies for massively parallel cluster.

Topology	# of Switches	# of Nodes	Software Forwarding	Bisection Bandwidth
Star	5	120	none	1.2 Gbps
Tree	61	1416	none	1.2 Gbps
SR	25	300	1	12 Gbps
SM	160	960	1	48 Gbps

For the star topology, the maximum number of nodes connected to the cluster is equal to the product of the number of port I_s (24 ports) and the number of switch in the stack (5), 120 (24×5) nodes in our case. The bisection bandwidth BW can be represented as $min(I_s / 2 * b, SP) \times \#ofNI_s$. Under parameters considered, the value of $I_s / 2 * b$ is 1.2 Gbps.

For the two-level tree topology, the connection between the root node and the level one comprises of two bundled links to increase bandwidth to 200Mbps. The interconnection network consists of a single switch as the root and sixty switches in the level 1 (twelve groups of five switches). The total of 1416 nodes can be connected to the cluster as leave nodes with *min* (1.2*Gbps*,*SP*) bisection bandwidth. Although a large number of nodes can be connected to the tree topology, available bisection bandwidth BW located at the root switch is not well balanced with the number of nodes.

A 5-stage SR network consists of 25 switches (5 stackable switches at each stage) and the total of 300 nodes (60 nodes per stages). The peak bisection bandwidth is 12Gbps ($60 \times 2 \times 100M$). The network diameter D is 4 hops. A message has at most one forward operation between any pair of source and destination. While the communication delay in the SR topology is higher due to software forwarding, the aggregated bisection bandwidth of the proposed topology significantly higher by a factor of 10 and communication locality of intra-stage nodes can benefit from the directly connected hardware switch. We believe that as performance of microprocessor continuing to increase and the adoption of a fast processor in the network interface, the software forwarding overhead will decrease in the near future.

The 4×4 SM networks consists of 32 SSBs which are comprised of 160 switches. The number of computing nodes in a single group is equal to 60 nodes therefore the total number of nodes is $60 \times 16 = 960$ nodes. The peak bisection bandwidth is 48Gbps. Each group of nodes has 6Gbps bandwidth. The network has 4 groups in one dimension. Therefore the total of $6 \times 4 = 24$Gbps bisection bandwidth. With the wrap around channel, the bisection bandwidth is doubled to 48 Gbps At most one forward operation is required between any pair of source and destination. The scalability results are summarized in Table 2.

4.3 HPL Benchmark Experiments

To investigate the validity of the proposed topologies, HPL-A portable implementation of the High Performance Linpack

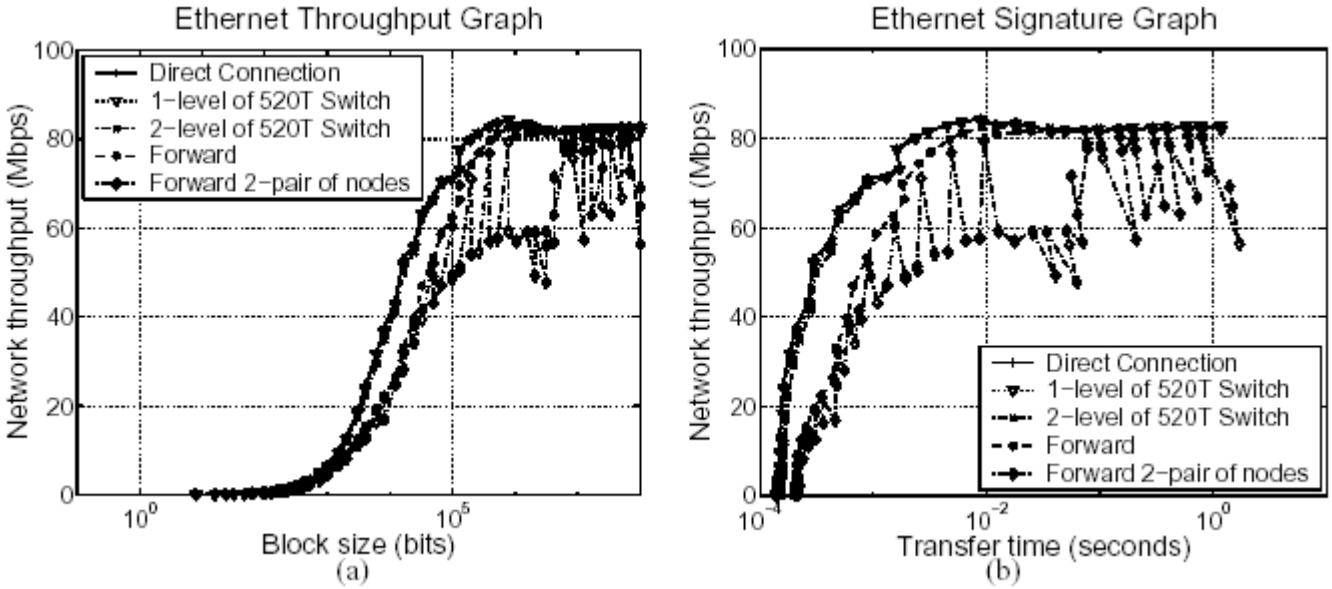


Figure 9. Network performance comparison (a) Network throughput versus message block size, (b) Network throughput versus transfer time. The test results show the performance of the hardware switching is 1.37 times and the software forwarding. The software forwarding shows higher communication overhead at the same block size.

benchmark for distributed- Memory computers was tested on different network topologies. Due to limited resources, the clusters consists only eight nodes. The same computer configurations were used. The HPL results cannot reflect the scalability of the proposed system since the number of nodes is small but we test the HPL to show the implementation of the proposed topology. The following parameters are selected: problem size = 10000, Block size NB = 64, and $P \times Q = 1 \times 8$. The performance results are shown in Table 3.

Table 3. HPL performance results.

Topology	Times	Gflops
Star	436.78	1.527
Tree	438.09	1.522
SR	5785.75	0.1153
SR with extra node	455.14	1.465

In the star topology, eight nodes are connected to a single Ethernet switch. The performance of the start topology is the best for the eight-node system. For the tree topology, two sets of four nodes are connected to two level-1 switches. The level-1 switches have a single link connected to the level-0 switches. The SR topology consists of four stages. Each stage consists of two computing nodes. The performance of the SR topology is very poor. The HPL benchmark has considerable communication activities. The nodes in the SR topology have to process the forwarding tasks.

The degradation in performance results from the forwarding overhead and the context switch overhead. However, the HPL results for the 8- node system is not a fair comparison. The SR topology is designed for the system that the star topology cannot accommodate. In the larger system, the intra-stage traffic can communicate through hardware. The number of nodes at each stage is equal to 60 which can perform a certain amount of task. The larger number of nodes means better distribution of forwarding task of interstage traffic. Also the large problems are usually divided into multiple levels of hierarchies. The communication between subproblem is less.

To relieve the forwarding overhead, the extra node is added to each stage to perform the forwarding task or act as a gateway. The system consists of 12 nodes, 8 nodes for computation and 4 nodes for forwarding. The performance is improved by an order of magnitude. As the network interface technology and microprocessor technology continues to evolve, the software forwarding overhead will decrease and slowly migrate to hardware level.

5. Concluding Remarks

The Ethernet network topologies for large scale clusters were studied. The contemporary Ethernet switches can be stacked together and have large backplane bandwidth. We proposed the SR and SM topologies for medium to large size clusters

implemented with stackable Ethernet switches. In SR network, the groups of stack switches are connected in series with wraparound links. The mesh network with wraparound links is used as a basis of the SM topology.

The software forwarding overheads were measured and compared to the hardware overhead. The results show that the hardware switch is faster than the software forwarding (adopted in the proposed topology) by a factor of 1.37. The aggregate bisection bandwidth of the SR network is more than other topologies by a factor of ten. The forwarding tasks are distributed among all nodes. The SR/SM networks of 300/960 nodes with 12/48Gbps bisection bandwidth are shown. The realization of the SR networks was tested using HPL benchmark. For the small cluster, the results from HPL benchmarks in SR network is very poor. However, we believe that applications run in a large cluster, the subproblems are classified into several levels. The communication intensive parts of the task are allocated to the same stage. Therefore the interstage traffic can be reduced. The proposed approach is a promising interconnect solution for a cluster-based supercomputer not only as the main interconnection but also the back-up network of the high speed network for better system availability.

References

[1] T. Agerwala, J. L. Martin, J. H. Mirza, D. C. Sadler, D.M. Dias, and M. Snir, "SP2 system architecture," *IBM Systems Journal*, vol. 34, no. 2, pp. 152–184, 1995.

[2] J. Dongarra, G. Geist, R. Manchek, and V. Sunderam, "Integrated PVM framework supports heterogeneous network computing," *Computers in Physics*, April 1993.

[3] M. Litzkow, M. Livny, and M. W. Mutka, "Condor–a hunter of idel workstations," in *Proceedings of the Eighth International Conference of Distributed Computing Systems*, pp. 104–111, 1988.

[4] D. J. Becker, T. Sterling, D. Savarese, J. E. Dorband, U. A. Ranawak, and C. V. Packer, "Beowulf: A parallel workstation for scientific computation," in *International Conference on Parallel Processing*, 1995.

[5] I. Foster and C. Kesselman, eds., *The Grid: Blueprint for a Future Computing Infrastructure*. Morgan Kaufmann, 1999.

[6] P. Mohapatra, "Wormhole routing techniques for directly connected multicomputer systems," *To appear in the ACM Computing Survey*.

[7] M. D. Schroeder, A. D. Birrell, M. Burrows, H. Murray, R. M. Needham, T. L. Rodeheffer, E. H. Satterthwaite, and C. P. Thacker, "Autonet: A high-speed self-configuring local area network using point-to-point links," tech. rep., SRC Research Report 59, 1990.

[8] C. B. Stunkel, D. G. Shea, and B. Abali, *et al*, "The SP2 high-performance switch," *IBM System Journal*, vol. 34, pp. 185–204, February 1995.

[9] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawlk, C. L. S. ad J. N. Seizovic, and W. Su, "Myrinet: A Gigabit-per-second local area network," *IEEE Micro*, vol. 15, pp. 29–36, February 1994.

[10] R. W. Horst, "TNet: A reliable system area network," *IEEE Micro*, vol. 15, pp. 37–45, February 1995.

[11] J. Duato, S. Yalmachili, and L. M. Ni, *Interconnection networks: An Engineering Approach*. Los Alamitos, California: IEEE Computer Society, 1997.

[12] V. Karamcheti and A. A. Chien, "Software overhead in messaging layers: Where does the time go?," in *Proceedings of ASPLOS-VI*, (San Jose, California), October 1994.

[13] T. Sterling, "How to build a beowulf: Assembling, programming, and using a clustered pc do-it-yourself supercomputer," in *Supercomputing*, (San Jose), 1997.

[14] T. W. Giogis, "29 switching hub save the bandwidth," *BYTE*, pp. 162–169, July 1995. [15] C. Reschke, T. Sterling, and D. Ridge, "A design study of alternative network topologies for the beowulf parallel workstation," in *High Performance and Distributed Computing*, 1996.

[16] "MPI: Message passing interface forum." Message passing interface forum, http://www.mpi-forum.org, 1994.

[17] V. Varavithya and T. Thepuatrakul, "High bandwidth ethernet topology for large scale clusters," in *EECON 23*, CMU, December 2000.

[18] V. Varavithya, P. Lousangfa, and C. Ngamphiw, "Effects of network configurations on performance of beowulf-class clusters," in *EECON 23*, CMU, December 2000.

[19] Q. O. Snell, A. R. Mikler, and J. L. Gustafson, "Netpipe: A network protocol independent performance evaluator," in *IASTED Conference*, (http://www.scl.ameslab.gov/netpipe/).

AB INITIO AND DENSITY FUNCTIONAL STUDIES OF POLYTHIOPHENE ENERGY BAND GAP

*Arnold C. Alguno[*a], Wilfredo C. Chung[a], Rolando V. Bantaculo[b], Reynaldo M. Vequizo[b], Hitoshi Miyata[c], Edgar W. Ignacio[*d] and Angelina M. Bacala[*b]*

[a]*College of Arts and Sciences, NORMISIST, Ampayon, Butuan City, Philippines*
[b]*Department of Physics, MSU-IIT, Iligan City, Philippines*
[c]*Department of Physics, Niigata University, Ikarashi, Japan*
[d]*Department of Chemistry, MSU-IIT, Iligan City, Philippines*

ABSTRACT – The energy band gap of intrinsic thiophene monomer and dimer were calculated using Hartree-Fock (HF) and density functional theory (DFT) methods employing various combinations of exchange and correlation functionals with electron core potential (ECP) split valence basis sets. HF overestimates band gap by up to 6.10 eV for thiophene monomers and dimers. DFT calculations with hybrid functionals (B3LYP and B3P86) give an excellent results (4.06 eV and 4.11 eV) which are in good agreement with the experimental energy band gap (4.05 eV).

KEYWORDS -- Energy band gap; density functional theory; *ab initio*; HOMO-LUMO; thiophene.

1. Introduction

Due to its chemical stability, high conductivity upon doping, and their non-linear optical properties, polythiophene is among the widely studied conjugated organic polymers, experimentally and theoretically [1]. During the recent years systematic efforts were aimed at investigating the molecular and electronic structure of thiophene oligomers and its derivatives [2, 3, 4, 5, 6, 7].

The energy gap between valence and conduction band of polymer is related to the lowest allowed energy of its monomer units and to the bandwidth resulting from the overlap between the monomer orbitals as shown in Figure 1 [2]. The energy band gaps obtained from band structure calculations for solids are analogous to Highest Occupied Molecular Orbital (HOMO) – Lowest Unoccupied Molecular Orbital (LUMO) energy differences in molecules.

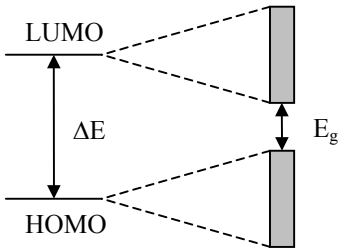


Figure 1. Relationship between HOMO-LUMO levels of finite and band gaps of the infinite system.

To design a low band gap polymer, it is desirable to start with monomer units with small excitation energies. A prior idea on the estimate is often useful. One way to obtain excitation energies is to calculate the energy of the ground and excited state explicitly and to take the energy difference. The commonly accepted structure of polythiophene is a linear chain of monomers α - α′ (2, 5) bounded by carbon [3] as shown in Figure 2.

An initial estimate of the band gap can be carried out using density functional theory (DFT). DFT is very attractive in calculations involving finite system because even the lowest level of DFT – the Local Spin Density Approximation (LSDA) – includes some electron correlation. This is extremely important in the design of conducting organic polymers which have an extended π system. Although the LSDA eigenvalue differences underestimate the band gap compared to experiment, the shift is almost vertical and very systematic [2].

DFT method had successfully been used to study band gaps of conjugated organic polymers where the HOMO/LUMO difference provide good estimate of the excitation energy. While there is some controversy surrounding the interpretation of DFT orbitals energies, we find that HOMO/LUMO energy difference offers a very good estimate of band gaps. It should be noted that the HOMO/LUMO energy difference at *ab initio* level does not closely relate to excitation energies due to the absence of orbital relaxation effects [8].

2. Computational Details

Initial geometries were optimized at Hartree-Fock (HF) level of theory and further reoptimized using DFT methods to include correlation corrections. In this study, an exchange

functional which was proposed by Becke [11a] in 1988 using a gradient-corrected correlation functional of Lee, Yang and Parr [11b, 11c] was employed. Hybrid functional are also used, the Becke's three-parameter functional (B3) [11d] which defines the exchange functional as the linear combination of Hartree-Fock, local and gradient-corrected exchange terms. The B3 hybrid functional was used in combination with the correlation functionals of Lee, Yang and Parr and non-local correlation expression provided by the Perdew 86 (P86) [11e].

The type of basis set is that of Stephens/Basch/Krauss ECP split valence (CEP-31G), augmented with polarization functions on heavy atoms (CEP-31G*), diffused function (CEP-31+G) and polarized functions (6-31G**). These basis sets were employed because some previous calculations [6] suggest that their results are in good agreement with the experimental values of the energy band gap of different polymeric system.

The vibrational frequency calculations were carried out to characterize the stationary points. Symmetry constraints were applied whenever possible.

All calculations were performed using the Gausian '94 [9] and GAMESS [10] suites of quantum chemistry programs running under Beowulf cluster, SunSparc station, and DEC alpha machines.

3. Results and Discussion

3.1 Geometries

The optimized results using hybrid DFT functional (B3LYP/CEP-31G*) are shown in Figure 2. This is the stable geometry of thiophene monomer and dimer. With the HF/6-31G* and B3LYP/CEP-31G*, the bond length are up to ±0.025Å and ±0.046Å, respectively when compared to experimental values. The bond angles agree very closely (to within ±0.33^{o}) with experiment. Pure and hybrid functionals geometries are almost identical. Compared to HF theory, DFT yields longer C=C double bonds. Thus, at HF, π-electrons is more localized. This is most likely due to the neglect of electron correlation. For geometry calculations, HF revealed more accurate estimates, indicating that it may not be necessary to perform DFT calculations to obtain good geometries in this case. Complete documentation of geometries is listed in Table 1.

3.2 Energy Band Gaps

Spectroscopic data for organic π-systems are usually determined either in solution or in the solid state (crystal or thin film). Since our calculations are for isolated molecules in the gas phase, we have attempted to compare our calculation to experimental results in solution.

Table 2 summarizes the energy band gap of intrinsic thiophene monomers and dimers at HF, BLYP, B3LYP and B3P86 with various combinations of basis sets. As expected, the RHF energy band gap of intrinsic thiophene oligomers overestimates the excitation energy because of the absence of correlation contribution [16]. Little improvement of the energy band gap is obtained in applying a higher level basis set; the absolute error is 5.51-6.10 eV. It is expected that the percent error decrease with increasing length of polymers at this level of theory [9]. The pure DFT functional (BLYP) underestimate the energy band gap by up to 1.77 eV compared to the experimental excitation energy. Hybrid DFT method (B3LYP and B3P86) yielded HOMO-LUMO energy difference which are in good agreement with the experiment. The DFT (B3LYP and B3P86) energy band gaps for thiophene dimer give an error of only up to 0.22 eV and 0.17 eV, respectively. The two hybrid functionals lead to almost identical results. Applying more comprehensive basis ddid not significantly improve the energy band gap of thiophene oligomer. This study shows that by using hybrid DFT functionals, a substantial improvement in the excitation energy can be obtained. The same observation was also reported previously by other workers [1, 2] using different polymeric system.

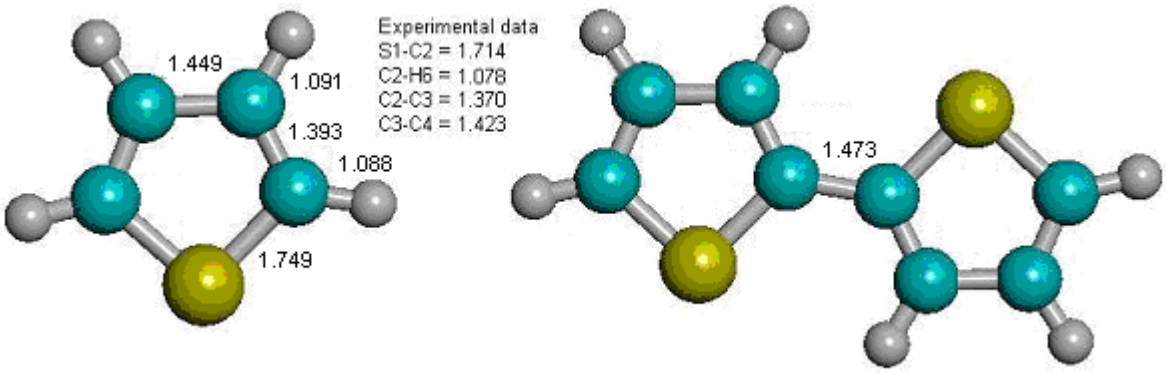


Figure 2. Structures of thiophene (a) monomer (b) dimer. Calculated bond distances are given in angstrom (Å) at B3LYP/CEP-31G.*

Table 1. Optimized structure of thiophene oligomers at 6-31G (HF) and CEP-31G* (DFT).*

Parameter	RHF	error	BLYP	error	B3LYP	error	B3P86	error	Expt.
Monomer									
Bond length									
$C_2 - H_6$	1.074	0.004	1.095	0.017	1.088	0.010	1.088	0.010	1.078
$S_1 - C_2$	1.726	0.012	1.764	0.050	1.749	0.035	1.737	0.023	1.714
$C_4 = C_5$	1.345	0.025	1.404	0.034	1.393	0.023	1.391	0.021	1.370
$C_3 - C_4$	1.437	0.014	1.455	0.032	1.449	0.026	1.443	0.020	1.423
Bond angle									
$S_1C_2C_3$	111.83	0.33	111.43	0.07	111.50	0.000	111.51	0.01	111.50
$C_2C_3C_4$	112.52	0.13	112.63	0.02	112.51	0.140	112.39	026	112.65
$C_4C_3H_7$	124.11	0.12	124.21	0.02	124.23	0.000	124.32	0.09	124.23
Dimer									
Bond length									
$S_1 - C_5$	1.736	0.012	1.766	0.042	1.750	0.026	1.739	0.015	1.724
$C_5 - C_2$	1.481	0.017	1.476	0.012	1.473	0.009	1.468	0.004	1.464
$C_4 = C_5$	1.367	0.014	1.403	0.050	1.391	0.038	1.390	0.037	1.353
$C_3 - C_4$	1.449	0.018	1.448	0.017	1.443	0.012	1.437	0.006	1.431
$C_4 - C_3$	1.375	0.019	1.414	0.058	1.402	0.046	1.339	0.043	1.356

Bond distances and bond angles are given in Å and degrees, respectively.

Table 2. HOMO-LUMO energy differences (energy band gap in eV) at various levels of theory.

	Monomer				Dimer			
	6-31G	*6-31+G*	*6-31G**	*6-31G***	*6-31G*	*6-31+G*	*6-31G**	*6-31G***
RHF	12.6066	11.4289	12.6888	12.6700	9.9676	9.5622	9.7031	10.1500
	CEP-31G	*CEP-31+G*	*CEP-31G**	*6-31G***	*CEP-31G*	*CEP-31+G*	*CEP-31G**	*6-31G***
BLYP	4.2298	4.2325	4.2415	4.4400	2.6741	2.2771	2.6790	2.8000
B3LYP	5.8426	5.8265	5.8570	6.1200	4.0491	4.0417	4.0572	4.2700
B3P86	5.9177	5.9153	5.9357	6.1800	4.1008	4.1049	4.1092	4.2200
Expt'l. value	5.2300[a]				4.0500[b]			

[a]ref. [18], [b]ref. [19]

4. Concluding Remarks

We have shown that a substantial improvement of DFT energy band gaps can be achieved with hybrid DFT functionals. Vertical excitation energies of thiophene oligomers were approximately reproduced with ±0.22 eV of the corresponding experimental results (solution phase).

Polythiophene presents special problems in computational modeling because of its extended molecular system. As a consequence, their properties may arise from secondary and tertiary structure effects, as well as from the primary microstructure. Their size and structure complexity causes the difficulty, even without worrying about inter chain-packing effect.

Energy band gap calculated using hybrid DFT functionals yielded estimates of excitations energies which are in good agreement with experimental value. The use of DFT hybrid functionals therefore will lead to a significant improvement of the energy band gap relative to those computed by HF methods. This study also indicated that the presences of polarized functions on heavy atoms as well as the corresponding diffused functions are crucial in the computations of the energy band gap of thiophene oligomers.

We hope that this contribution will stimulate the computational modeling community for the discussion of intrinsic polythiophenes as well as other conducting polymers. It is further suggested that longer thiophene oligomer chains up to 11 units or even longer with higher basis sets will be used so that a more accurate energy band gap could be predicted though this requires a very high computational cost.

Acknowledgments

We thank Prof. Ronald and Mr. Marvin Fernandez for many fruitful discussions. Likewise, thank is due to Mr. Allen S. Dahili for many helpful assistance. This study was conducted through the scholarship grant supported by the Commission on Higher Education – Mindanao Advanced Education Project (CHED-MAEP).

Reference

[1] Salzner, U., Lagowski, J. B., Pickup, P. G. and Poirier, A., *Synth. Met.* 96 (1998) 177-189.

[2] Salzner, U., Lagowski, J. B., Pickup, P. G. and Poirier, A., *J. Comp. Chem.* 18, 15, 1943-1953, 1997.

[3] Salzner, U., Lagowski, J. B., Pickup, P. G. and Poirier, A., *Synth. Met.* 98 (1999) 221-227.

[4] Nakanishi, H, *et al. J. Org. Chem.* 1998, 63, 8632-8633.

[5] Lathti, P. M., Obrzut, J., Karasz, F. E., *Macromolecules*, 1987, 20, 2023.

[6] Smith, James Richard, *Electrosynthesis of Novel Polyheterocycles*, Ph.D. Dissertation, Unpublished, School of Pharmacy & Biomedical Sciences, University of Portsmouth, Portsmouth, UK, 1995.

[7] Irle, Stephan and Lischka, Hans. *J. Chem. Phys.* 103 (4) 1995.

[8] Foresman, J. B. and Frisch, Æ., *Exploring Chemistry with Electronic Structure Methods, 2nd ed.* Gaussian, Inc., Pittsburgh, PA. 1995.

[9] *Gaussian 94, Revision B.2.* M. J. Frisch, G. W. Trucks, H. B. Schlegel, P. M. W. Gill, B. G. Johnson, M. A. Robb, J. R. Cheeseman, T. Keith, G. A. Petersson, J. A. Montgomery, K. Raghavachari, M. A. Al-Laham, V. G. Zakrzewski, J. V. Ortiz, J. B. Foresman, C. Y. Peng, P. Y. Ayala, W. Chen, M. W. Wong, J. L. Andres, E. S. Replogle, R. Gomperts, R. L. Martin, D. J. Fox, J. S. Binkley, D. J. Defrees, J. Baker, J. P. Stewart, M. Head-Gordon, C. Gonzalez, and J. A. Pople, Gaussian, Inc., Pittsburgh PA, 1995.

[10] GAMESS Version = 22 Nov 1995 from Iowa State University, M.W.Schmidt, K.K.Baldridge, J.A.Boatz, S.T.Elbert, M.S.Gordon, J.H.Jensen, S.Koseki, N.Matsunaga, K.A.Nguyen, S.J.Su, T.L.Windus, Together With M.Dupuis, J.A.Montgomer. *J. Comp. Chem.* 14, 1347-1363 (1993).

[11] (a) A. D. Becke, *Phys. Rev*., A38, 3098 (1988). (b) C. Lee, W. Yang and R. G. Parr, *Phys. Rev.*, B37, 785-789 (1988). (c) B. Miehlich, A. Savin, H. Stoll and H. Preuss, *Chem. Rev. Lett*, 157, 200 (1989). (d) A. D. Becke, *J. Chem. Phys*. 98, 5648-5652 (1993). (e) J. P. Perdew, *Phys. Rev.* B33, 8822-8824 (1986).

[12] Slater, J.C., Quantum Theory of Molecules and solids, McGraw-Hill: New York, 1974, Vol. 4.

[13] Vosko, S.H., Wilk, L., Nusair, M., Can, *J. Phys.* 1980, 58, 1200.

[14] Becke, A.D., *J. Chem. Phys.* 1993, 98, 1372.

[15] P. Walters, M. Stahl, *BABEL Program (version 1.1)* Copyright ©1992, 93, 94, Dolota Research Group, Department of Chemistry, University of Arizona.

[16] Hyperchem ™ Release 3. *Windows Molecular Modelling System*, Copyright ©1993, Hypercube, Inc. and Autodesk, Inc. Developed by Hypertube, Inc.

[17] R.M. Dreiler and E.K.U. Gross, *Density Functional Theory*, Springer, Berlin, 1990.

[18] Simmons, W.W., *Handbook of Ultraviolet Spectra*; Sdler Res. Lab.: Philadelphia.

[19] Colditz, R., Grebner, D., Helbig, M., Rentsch, S., *Chem. Phys.*1995, 201, 309.

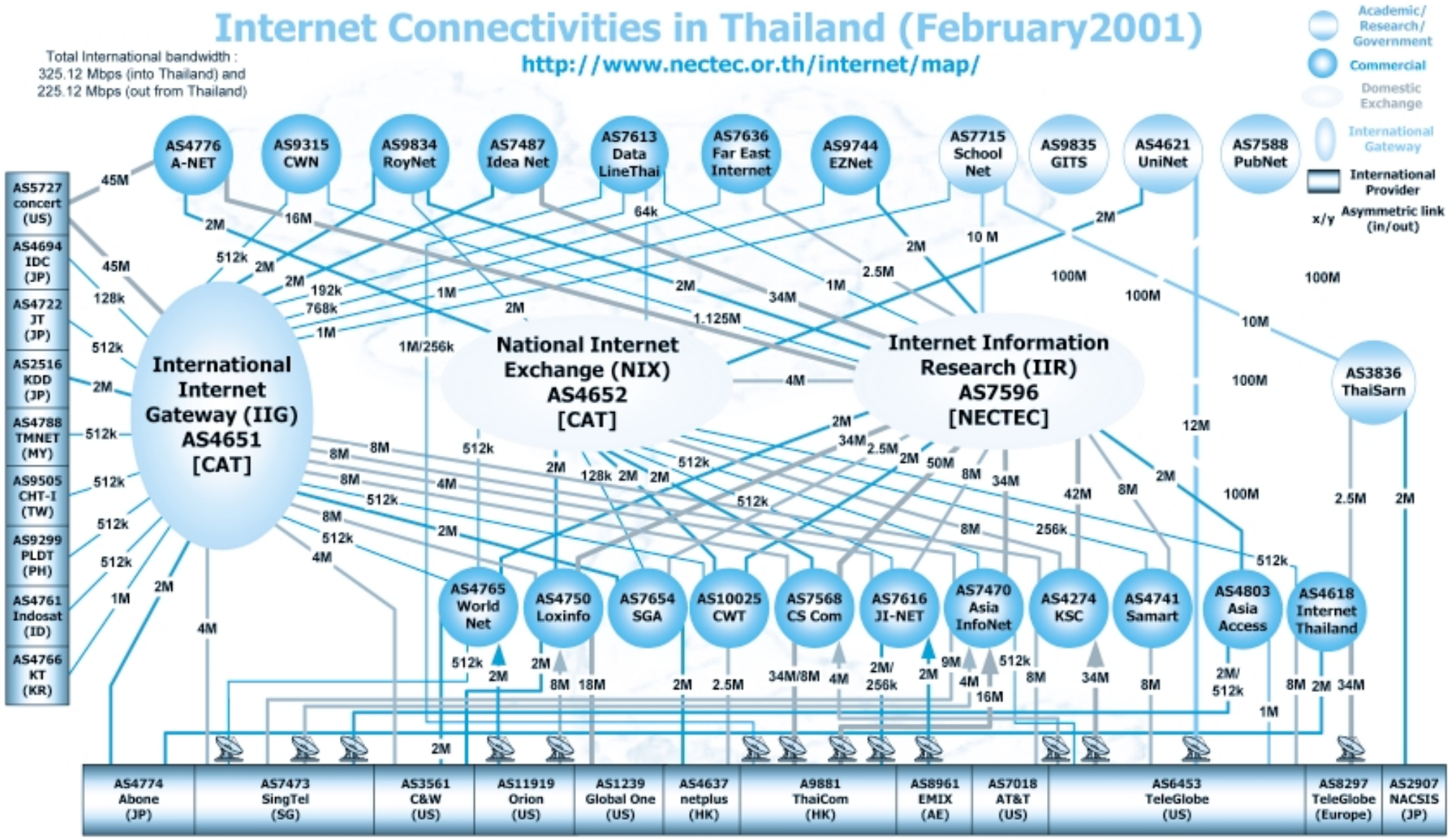
[20] O. Kwong and M.L. McKee, *J. Phys. Chem.* B2000, 104, 1686-1694.

[21] Theoretical Studies of electronic Spectra of Organic Molecules, Roos, B.O., Fülscher, M., Malmqvist, P.-Å., Merchàn, M., Serrano-Andrès, L., in *Quantum Mechanical Structure Calculations with Chemical Accuracy*, Langhoff, S.R., Ed., Kluwer Academic Publishers: Boston, 1995; pp 357-438.

Internet Connectivities in Thailand (February2001)

http://www.nectec.or.th/internet/map/

Total International bandwidth :
325.12 Mbps (into Thailand) and
225.12 Mbps (out from Thailand)



DISCLAIMER

Chart Date: 2001-02-02

This chart is designed, maintained and copyrighted by Primas Taechashong, Kittiya Sringamphong and Thaweesak Koanantakool NTL, NECTEC. All rights reserved. The information contained in this chart is based on actual measurements and estimation. We welcome update information, but reserve the rights to verify the accuracy of the given information. Please contact us at **netadmin@ntl.nectec.or.th**. For authoritative information please contact Communications Authority of Thailand.

ไทยสาร ThaiSam