Jurnal Teknologi Elektro, Universitas Mercu Buana
ISSN: 2086-9479

# KOMPUTASI PARALEL UNTUK PENGOLAHAN PRESTASI AKADEMIK MAHASISWA

Andri Lesmana Wanasurya
Magister Teknik Elektro
Universitas Katolik Indonesia Atma Jaya
Jakarta, Indonesia
andri.lesmana@atmajaya.ac.id

Maria Angela Kartawidjaja
Magister Teknik Elektro
Universitas Katolik Indonesia Atma Jaya
Jakarta, Indonesia
mariakw@atmajaya.ac.id

Sri Mulyanti
Teknik Elektro
Universitas Katolik Indonesia Atma Jaya
Jakarta, Indonesia
sri.mulyanti@atmajaya.ac.id

***Abstrak*— Pengolahan prestasi akademik mahasiswa pada institusi perguruan tinggi untuk memperoleh Indeks Prestasi Semester (IPS) dan Indeks Prestasi Kumulatif (IPK) seluruh mahasiswa bukanlah pekerjaan mudah yang dapat diselesaikan dalam waktu yang singkat. Kompleksitas pengolahan ini tergantung dari jumlah mahasiswa, jumlah mata kuliah, dan jumlah parameter penilaian yang sangat bervariasi. IPS dan IPK sangat penting bagi mahasiswa dalam proses akademik sehingga perhitungannya harus dilakukan secara akurat dan dalam waktu yang singkat. Dalam penelitian ini kami menggunakan komputasi paralel untuk menghitung prestasi akademik, menggunakan total lima buah komputer yang terhubung dalam jaringan dan memanfaatkan Message Passing Interface (MPI) untuk membentuk virtual parallel platform. Hasil dari pengujian menunjukkan kinerja komputasi paralel yang baik, terutama dengan jumlah mahasiswa yang cukup banyak. Dengan menggunakan empat prosesor, nilai speedup dapat mencapai 2,71 melalui konfigurasi empat jumlah proses paralel. Komputasi paralel dapat meningkatkan efisiensi waktu proses perhitungan sehingga penggunaan sumber daya komputasi lebih efektif dan efisien sesuai salah satu ciri teknologi hijau.**

***Kata Kunci*— *basis data; C#; IPK; IPS; komputasi paralel; mahasiswa; MPI.NET; prestasi akademik***

## I. PENDAHULUAN

Pada institusi perguruan tinggi, tiap semester dilakukan pengolahan data nilai mahasiswa, untuk mendapatkan Indeks Prestasi Semester (IPS) dan Indeks Prestasi Kumulatif (IPK). Kompleksitas pekerjaan untuk mengolah data ini tergantung dari jumlah mahasiswa, jumlah mata kuliah, dan jumlah parameter penilaian. Suatu mata kuliah memiliki minimal dua parameter penilaian, yaitu nilai Ujian Tengah Semester (UTS) dan nilai Ujian Akhir Semester (UAS). Selain itu ada parameter opsional, seperti nilai tugas, kuis dan proyek. Bobot masing-masing parameter ini ditentukan oleh dosen yang mengampu mata kuliah sesuai dengan peraturan akademik, dan ini berarti bobot penilaian tidak harus seragam. Dengan demikian jelaslah bahwa perhitungan IPS dan IPK mahasiswa bukanlah suatu pekerjaan mudah yang dapat diselesaikan dalam waktu yang singkat.

Saat ini pengolahan data nilai mahasiswa di institusi perguruan tinggi umumnya dilakukan dengan bantuan komputer. Makin banyak data yang diproses, waktu yang dibutuhkan untuk pengolahan juga akan makin besar. Nilai IPS dibutuhkan untuk menentukan jumlah Satuan Kredit Semester (SKS) yang boleh diambil seorang mahasiswa di semester berikutnya, dan nilai IPK dibutuhkan untuk menentukan peringkat prestasi seorang mahasiswa. Dengan demikian baik proses perhitungan nilai IPS ataupun IPK harus dilakukan dengan akurat dan dalam waktu yang singkat, atau dengan perkataan lain dibutuhkan suatu teknik pengolahan data dengan kinerja komputasi yang lebih tinggi dibandingkan dengan kinerja komputasi sekuensial yang umumnya digunakan selama ini.

Salah satu cara untuk meningkatkan kinerja komputasi adalah dengan menggunakan komputasi paralel yang memungkinkan sejumlah komputer bekerja secara bersama-sama untuk mencari solusi atas suatu masalah.

## II. PENELITIAN SEBELUMNYA

Penelitian komputasi paralel menggunakan MPI yang sudah pernah dilakukan di antaranya adalah untuk implementasi pengolahan paralel pola Williamson Array dalam Pembangunan Matriks skew-Hadamard [1], dan untuk menghitung nilai akhir mata kuliah Metode Numerik di Jurusan Matematika Universitas Andalas [2]. Kedua penelitian tersebut berbeda dengan penelitian ini. Penelitian paralel untuk pola

Williamson Array dalam Pembangunan Matriks skew-Hadamard dan perhitungan nilai akhir mata kuliah Metode Numerik yang disebutkan di atas menggunakan MPI dengan bahasa pemrograman C, sedangkan penelitian ini menggunakan MPI.NET dengan bahasa pemrograman C#. C# adalah bahasa pemrograman dalam platform Microsoft .NET untuk komputasi berkinerja tinggi dengan fitur modern, object oriented programming (OOP), dan mendukung pengembangan aplikasi yang dapat terhubung melalui jaringan. Selain itu, penelitian ini menggunakan Relational Database Management System (RDBMS) untuk penyimpanan datanya dan hal ini tidak ditemukan dalam kedua penelitian terdahulu.

## III. Tujuan Dan Manfaat Penelitian

Pada penelitian ini dilakukan simulasi proses perhitungan IPS dan IPK mahasiswa dengan menggunakan sejumlah komputer yang dihubungkan sedemikian rupa sehingga membentuk suatu platform paralel, dengan tujuan untuk mempersingkat waktu komputasi dalam pengolahan prestasi akademik mahasiswa.

Penelitian akan menghasilkan suatu perangkat lunak simulasi pengolahan prestasi akademik mahasiswa. Model simulasi yang dihasilkan dapat digunakan untuk membantu unit pengolah data akademik mahasiswa dalam melakukan pengolahan prestasi akademik mahasiswa, baik prestasi per semester (IPS) maupun prestasi kumulatif (IPK). Selain itu, hasil penelitian ini dapat dimanfaatkan untuk penelitian dengan data yang lebih kompleks, dengan jumlah dan ragam data yang lebih bervariasi.

## IV. Studi Pustaka

Suatu sistem paralel didefinisikan sebagai suatu kombinasi dari algoritma paralel dan arsitektur paralel yang digunakan untuk eksekusi progam paralel. Oleh karena itu kinerja suatu sistem paralel seyogyanya dievaluasi berdasarkan algoritma dan arsitekturnya.

### *A. Arsitektur Komputer Paralel*

Berdasarkan aliran instruksi dan aliran data, menurut Flynn, komputer dapat dikategorikan ke dalam 4 (empat) kelompok, yaitu Single-Instruction Stream, Single Data Stream (SISD), Single Instruction Stream, Multiple Data Stream (SIMD), Multiple Instruction Stream, Single Data Stream (MISD), dan Multiple Instruction Stream, Multiple Data Stream (MIMD) [3][4]. Sistem komputer paralel termasuk dalam kelompok MIMD. Kelompok ini memiliki sejumlah prosesor yang masing-masing beroperasi pada sejumlah data.

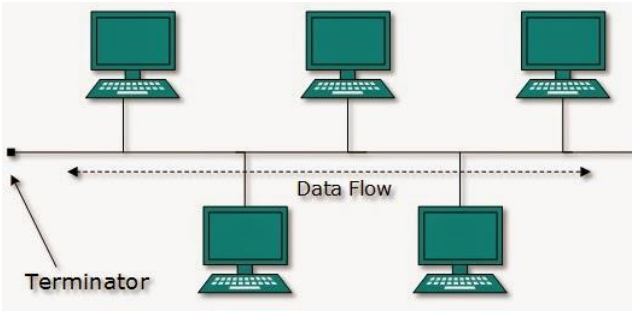
Berdasarkan ruang pengalamatan, sistem komputer dalam kelompok MIMD ini dapat dibagi menjadi sistem multiprosesor dan sistem multikomputer. Sebuah sistem multiprosesor adalah sistem komputer paralel yang didasarkan pada pemakaian memori tunggal secara bersama-sama, sedangkan sistem multikomputer adalah sistem komputer paralel dengan setiap CPU memiliki memorinya sendiri dan independen.

Dalam penelitian ini digunakan sistem komputer multikomputer yang memiliki memori terpisah, sehingga dibutuhkan saluran interkoneksi guna menghubungkan komputer-komputer tersebut.

### *B. Topologi Jaringan Komputer*

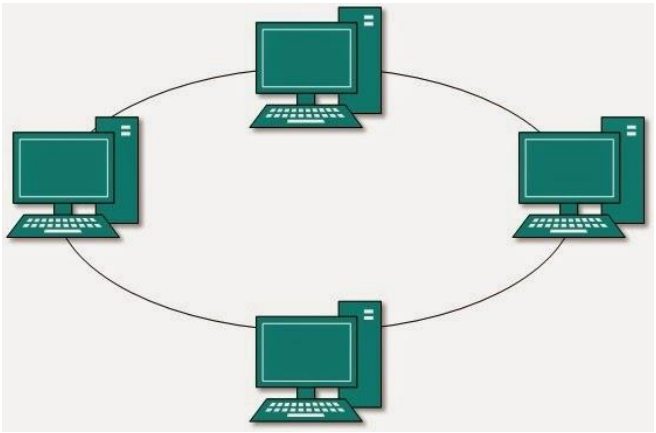
Saluran interkoneksi untuk menghubungkan komputer yang satu dengan komputer lainnya dapat memiliki berbagai topologi, misalnya topologi bus, topologi ring, topologi star, topologi mesh, dan topologi tree [5]. Topologi bus, ring dan star ditunjukkan pada Gambar 1, 2, dan 3.

Topologi bus bisa dikatakan sebagai topologi yang paling sederhana dibandingkan dengan topologi lainnya. Topologi bus hanya menggunakan sebuah kabel jenis coaxial di sepanjang node (komputer) dan pada umumnya ujung kabel coaxial tersebut biasanya diberikan konektor T sebagai kabel end to end.



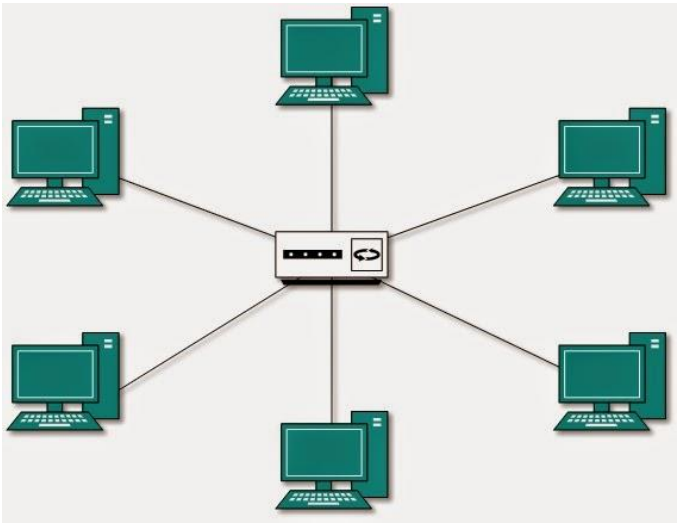

Gambar 1. Topologi bus.

Topologi *ring* merupakan salah satu topologi jaringan yang menghubungkan satu komputer dengan komputer lainnya dalam suatu rangkaian melingkar, mirip bentuk sebuah cincin. Biasanya topologi ini hanya menggunakan *Local Area Network* (LAN) untuk menghubungkan komputer satu dengan komputer lainnya.



Gambar 2. Topologi *ring*.

Topologi *star* atau bintang merupakan salah satu bentuk topologi jaringan yang biasanya menggunakan *switch/hub* untuk menghubungkan satu komputer dengan komputer lainnya. Pada penelitian ini, seluruh komputer yang digunakan dalam pengujian saling terhubung dalam jaringan komputer dengan topologi *star*.



Gambar 3. Topologi *star*.

### C. Speedup

Ada beberapa metrik yang lazim digunakan untuk mengukur kinerja komputasi paralel, misalnya waktu eksekusi, peningkatan kecepatan atau yang biasa disebut sebagai *speedup*, efisiensi prosesor dan memori, serta biaya. Salah satu kinerja yang umum digunakan adalah *speedup* [6].

Secara umum s*peedup* dapat dikategorikan ke dalam dua bentuk, *absolute speedup* dan *relative speedup*. *Absolute speedup* mengukur peningkatan kinerja suatu program paralel dibandingkan terhadap program sekuensial yang memiliki waktu eksekusi paling singkat. Oleh karena program sekuensial yang paling cepat untuk suatu masalah belum tentu merupakan program sekuensial yang paling cepat untuk masalah lain, maka sebagai program sekuensial diambil program aplikasi yang umum digunakan untuk memecahkan masalah. *Relative speedup* mengukur peningkatan kinerja yang diperoleh dengan menjalankan suatu program paralel pada sejumlah prosesor dibandingkan dengan program paralel yang sama yang dijalankan pada satu prosesor. Oleh karena penelitian ini bertujuan untuk mengukur seberapa besar peningkatan kinerja yang dapat diperoleh dengan menggunakan sejumlah prosesor yang bekerja bersama-sama, adalah wajar untuk menggunakan *relative speedup* sebagai metrik.

Secara matematis *relative speedup* didefinisikan sebagai:

$$Sp = \frac{T_1}{T_p} \qquad (1)$$

dengan $T_1$ dan $T_p$ masing-masing adalah waktu eksekusi program paralel pada satu dan pada $p$ prosesor.

### D. Message Passing Interface

*Message Passing Interface* **(**MPI) adalah salah satu protokol komunikasi yang digunakan dalam pemrograman pada komputer paralel. Komunikasi dengan MPI dapat dilakukan baik secara *point-to-point* ataupun secara kolektif (*broadcast*). Tujuan menggunakan MPI adalah memberikan kinerja yang tinggi, *scalable* dan mudah dimigrasikan dari *platform* komputasi yang satu ke *platform* komputasi yang lain. MPI merupakan protokol yang banyak digunakan dalam komputasi berkinerja tinggi [7][8][9].

MPI.NET adalah implementasi MPI untuk lingkungan pengembangan berbasis Microsoft .NET. MPI.NET menyediakan dukungan untuk seluruh bahasa pemrograman berbasis .NET, terutama bahasa pemrograman C# [10]. C# merupakan salah satu bahasa pemrograman untuk komputasi berkinerja tinggi (*High-Performance Computing*) [11] dengan fitur modern dan kompatibilitas yang baik dengan berbagai RDBMS.

## V. METODE PENELITIAN

Penelitian diawali dengan pemahaman lebih lanjut mengenai komputasi paralel dan cara membangun *virtual platform* menggunakan perangkat lunak yang dinamakan MPI, kemudian ditentukan jumlah prosesor dan topologi interkoneksi yang akan digunakan. Tahap berikut adalah

membuat interkoneksi antar-komputer dan membentuk *virtual platform*. Kemudian ditentukan cara pendistribusian data ke semua prosesor dalam sistem, sehingga beban kerja pada masing-masing prosesor seimbang, dengan harapan semua prosesor akan dapat menyelesaikan kerjanya dalam waktu yang relatif sama.

Dalam tahap perancangan ditentukan struktur data dalam tabel pada RDBMS dan ditentukan pula jenis perangkat lunak yang akan dibuat. Perangkat lunak dibuat dengan menggunakan *Integrated Development Environment* (IDE) Microsoft Visual Studio versi 2008. *Framework* .NET yang akan digunakan adalah versi 2.0 dan program ditulis dalam bahasa pemrograman C#. RDBMS yang digunakan adalah Microsoft SQL Server versi 2008.

Tahap pengujian dilakukan melalui simulasi model pengujian. Data uji yang digunakan disesuaikan dengan data untuk menghitung prestasi akademik mahasiswa. Pengujian menggunakan data fiktif yang dibentuk di awal pengujian sebanyak 8156 mahasiswa dan rata-rata telah menempuh tujuh sampai delapan semester perkuliahan. Pengujian dilakukan dengan menggunakan satu komputer sebagai *server* basis data dan empat komputer lainnya untuk menjalankan proses perhitungan. Dalam pengujian diukur besarnya waktu yang dibutuhkan untuk menghitung prestasi akademik mahasiswa menggunakan satu prosesor, dan menggunakan dua, tiga, dan empat prosesor dengan jumlah proses paralel yang berbeda, yaitu satu, dua, tiga, empat, dan delapan. Dari hasil pengukuran dapat dihitung peningkatan kinerja komputasi paralel dengan menggunakan *relative speedup* sebagai metrik. Tahap terakhir adalah melakukan analisis dari hasil pengujian.

## VI. HASIL DAN ANALISA

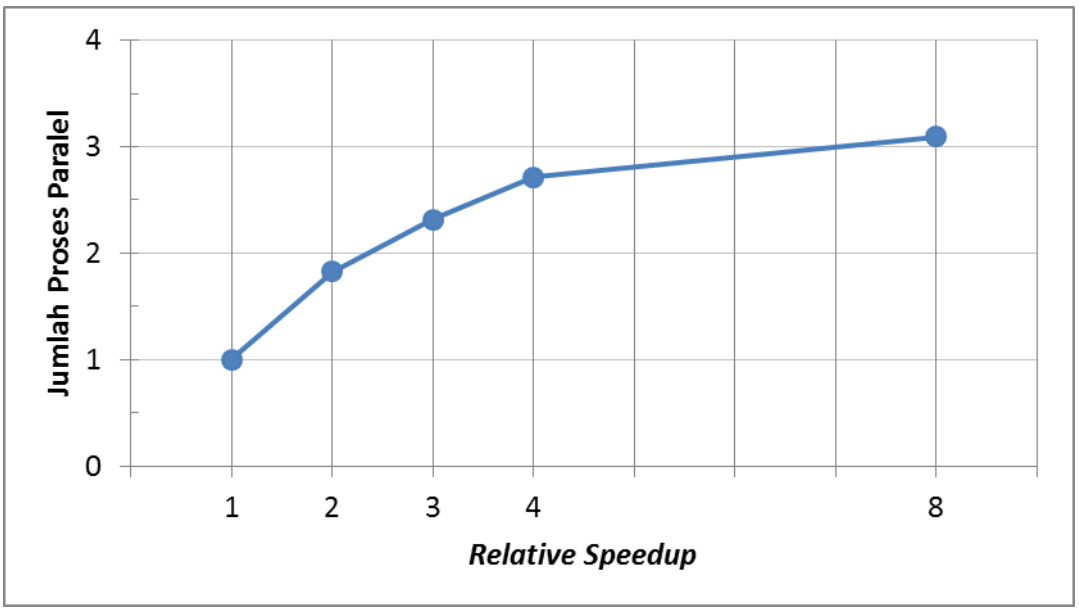
Lima simulasi model pengujian telah dijalankan dalam tahap pengujian, yaitu:

1. Model pertama menggunakan dua buah komputer, yaitu satu komputer sebagai *server* basis data dan satu komputer lainnya untuk menjalankan proses perhitungan. Perhitungan pada model ini hanya menggunakan satu prosesor dengan total jumlah proses paralel hanya satu. Sesuai persamaan matematis *relative speedup*, waktu proses rata-rata dari model pertama ini digunakan sebagai nilai $T_1$.
2. Model kedua menggunakan tiga buah komputer, yaitu satu komputer sebagai *server* basis data dan dua komputer lainnya untuk menjalankan proses perhitungan. Perhitungan pada model ini menggunakan dua prosesor dengan total jumlah proses paralel dua.
3. Model ketiga menggunakan empat buah komputer, yaitu satu komputer sebagai *server* basis data dan tiga komputer lainnya untuk menjalankan proses perhitungan. Perhitungan pada model ini menggunakan tiga prosesor dengan total jumlah proses paralel tiga.
4. Model keempat menggunakan lima buah komputer, yaitu satu komputer sebagai *server* basis data dan empat komputer lainnya untuk menjalankan proses perhitungan. Perhitungan pada model ini menggunakan empat prosesor dengan total jumlah proses paralel empat.
5. Model kelima sama dengan model keempat namun total jumlah proses paralel yang dijalankan adalah delapan sehingga masing-masing prosesor menjalankan dua proses paralel.

Untuk masing-masing model dilakukan pengujian sebanyak tiga kali sehingga total pengujian yang dilakukan sebanyak lima belas kali. Hasil pengujian berupa waktu proses rata-rata dan hasil perhitungan *relative speedup* ditunjukkan pada Tabel 1. Perbandingan nilai *relative speedup* terhadap jumlah proses paralel ditunjukkan pada Gambar 4.

Tabel 1. Hasil pengujian dan *relative speedup*.

| **Model Pengujian** | **Jumlah Komputer Total** | **Jumlah Komputer Paralel** | **Jumlah Proses Paralel** | **Waktu Proses Rata-Rata (menit)** | ***Relative Speedup*** |
|---|---|---|---|---|---|
| 1 | 2 | 1 | 1 | 34.87 | 1.00 |
| 2 | 3 | 2 | 2 | 19.09 | 1.83 |
| 3 | 4 | 3 | 3 | 15.05 | 2.32 |
| 4 | 5 | 4 | 4 | 12.86 | 2.71 |
| 5 | 5 | 4 | 8 | 11.28 | 3.09 |




Gambar 4. Perbandingan *relative speedup* terhadap jumlah proses paralel.

Contoh hasil perhitungan nilai UTS dan UAS pada salah satu mahasiswa dalam pengujian ditunjukkan pada Gambar 5.

Nilai *relative speedup* berdasarkan perbandingan waktu proses rata-rata pada percobaan model pertama terhadap model kedua, ketiga, keempat, dan kelima. Dari hasil pengujian dapat dilihat peningkatan nilai *relative speedup* ketika jumlah komputer paralel / prosesor, dan jumlah proses paralel ditambahkan. Peningkatan ini dikarenakan proses komputasi yang dijalankan secara bersamaan dan data yang dapat diproses dalam satu waktu semakin banyak.

## VII. KESIMPULAN

Dari penelitian ini dapat disimpulkan:

- Nilai *relative speedup* yang lebih tinggi mencerminkan kinerja komputasi yang lebih baik dan waktu komputasi lebih singkat.
- Komputasi paralel mampu memberikan kinerja komputasi yang lebih tinggi dibandingkan dengan kinerja komputasi sekuensial dalam pengolahan prestasi akademik mahasiswa.

| | Tahun | Semester | MhsID | Status | SKSSemester | NilaiGradeHasilSemester | SKSTotal | NilaiGradeHasilTotal | IPS | IPK |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2012 | 1 | 11242145 | A | 17 | 64.4 | 17 | 64.4 | 3.78... | 3.78... |
| 2 | 2012 | 2 | 11242145 | A | 19 | 70.7 | 36 | 135.1 | 3.72... | 3.75... |
| 3 | 2012 | 3 | 11242145 | A | 5 | 19.4 | 41 | 154.5 | 3.88 | 3.76... |
| 4 | 2013 | 1 | 11242145 | A | 19 | 58.5 | 60 | 213 | 3.07... | 3.55 |
| 5 | 2013 | 2 | 11242145 | A | 22 | 61.3 | 82 | 274.3 | 2.78... | 3.34... |
| 6 | 2014 | 1 | 11242145 | A | 21 | 76.5 | 103 | 350.8 | 3.64... | 3.40... |
| 7 | 2014 | 2 | 11242145 | A | 22 | 84.6 | 125 | 435.4 | 3.84... | 3.48... |
| 8 | 2015 | 1 | 11242145 | A | 15 | 60 | 140 | 495.4 | 4 | 3.53... |

Gambar 5. Contoh hasil perhitungan UTS dan UAS salah satu mahasiswa.

- Peningkatan nilai *relative speedup* sejalan dengan penambahan jumlah komputer paralel / prosesor dan jumlah proses paralel. Penambahan jumlah proses paralel tanpa menambah jumlah komputer paralel / prosesor tidak memberikan peningkatan nilai *relative speedup* yang signifikan, seperti pada model pengujian kelima.

Penelitian ini masih memungkinkan untuk dikembangkan dengan jumlah data uji, variasi model pengujian, dan komputer paralel / prosesor yang lebih banyak. Kemungkinan pengembangan lain adalah dengan menerapkan komputasi paralel pada bidang selain pengolahan prestasi akademik mahasiswa, misalnya perhitungan simulasi perubahan suku bunga bagi kredit usaha, perhitungan depresiasi nilai aset perusahaan, dsb.

## UCAPAN TERIMA KASIH

Penelitian ini didanai oleh Hibah Penelitian Kompetitif Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) Universitas Katolik Indonesia Atma Jaya tahun 2016. Penulis mengucapkan terima kasih kepada tim editorial Jurnal Teknologi Elektro atas dipublikasikannya penelitian ini.

## DAFTAR PUSTAKA

[1] Suharini, Y. S. and Indriasari, M., Implementasi Pengolahan Paralel Pola Williamson Array dalam Pembangunan Matriks skew-Hadamard, Jurnal IPTEK 9 (1): 1-7, 2014.

[2] Bahri, S., Pemograman Paralel Untuk Menghitung Nilai Akhir Mata Kuliah Metode Numerik di Jurusan Matematika Unand, Working Paper, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Andalas, 2010, unpublished.

[3] Hwang, K., Advanced Computer Architecture: Parallelism, Scalability, Programmability, New York: McGraw-Hill, 1993.

[4] Hennessy, J. L. and Patterson, D. A., Computer Architecture: A Quantitative Approach, 5th ed., San Francisco: Morgan Kaufmann, 2012.

[5] Wilkinson, B. and Allen, M., Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers, 2nd ed, New Jersey: Prentice Hall, 2005.

[6] Culler, D. et al., Parallel Computer Architecture: A Hardware/Software Approach, San Francisco: Morgan Kaufmann, 1997.

[7] Sayatan, S. et al., High Performance and Scalable MPI over InfiniBand with Reduced memory Usage: An In depth Performance Analysis, Proceedings of the 2006 ACM/IEEE conference on Supercomputing, 2006.

[8] Message Passing Interface Forum. MPI: A Message Passing Interface Standard, 1994.

[9] Message Passing Interface Forum. MPI: Extension to the Message Passing Interface, 1997.

[10] http://www.crest.iu.edu/research/mpi.net, Tanpa tahun, "MPI.NET: High-Performance C# Library for Message Passing", http://www.crest.iu.edu/research/mpi.net, diakses tanggal 11-02-2016.

[11] Willcock, J. et al., Using MPI with C# and the Common Language Infrastructure, Indiana University Computer Science Department Technical Report 570, 2002.