Jurnal Ilmiah
ILMU KOMPUTER
Universitas Udayana

Vol. 6, No. 2, September 2013 **ISSN 1979-5661**

PENJADWALAN ALOKASI *JOB* BERBASIS METODA HEURISTIK PADA LINGKUNGAN GRID INDONESIAN EDUCATION GRID

I Nyoman Rudy Hendrawan, Waskitho Wibisono
Jurusan Teknik Informatika, Fakultas Teknologi Informasi,
Institut Teknologi Sepuluh Nopember (ITS), Surabaya, 60111
E-mail: rudyhendrawan11@mhs.if.its.ac.id, waswib@if.its.ac.id

ABSTRACT

Scheduling mechanism on a grid system became one of important factor in measuring performance of the grid system. Moreover, building a grid system which is used for experiment or research is time and cost consuming, and then a grid system simulation can be an alternative. In a previous study has been conducted by designing and simulating of the grid system which is called Indonesian Education Grid (IndoEdu-Grid). A random allocation job scheduling method was used in the previous study (the method is called UD-IndoEdu-Grid in the next section). Therefore, in this paper we proposed an allocation job scheduling based on heuristic method called TS-IndoEdu-Grid, and thus expected to improve the performance of the IndoEdu-Grid environment. Experiment results show that by using the TS-IndoEdu-Grid method could reduce makespan by 20% to UD-IndoEdu-Grid. In flowtime parameter the TS-IndoEdu-Grid method also produces 94% smaller than the UD-IndoEdu-Grid method.

Key words: IndoEdu-Grid, heuristic, TS-IndoEdu-Grid, UD-IndoEdu-Grid, makespan, flowtime.

ABSTRAK

Mekanisme penjadwalan pada sistem grid menjadi salah satu faktor penting dalam hal mengukur kinerja sistem grid. Selain itu membangun lingkungan grid yang dapat digunakan untuk eksperimen atau penelitian memerlukan waktu dan biaya yang sangat besar, sehingga penelitian dalam ranah simulasi dapat menjadi suatu alternatif. Pada penelitian sebelumnya telah dirancang dan disimulasikan teknologi grid yang bernama Indonesian Education Grid (IndoEdu-Grid). Pada penelitian tersebut digunakan metoda penjadwalan alokasi job secara random (pada bagian selanjutnya disebut UD-IndoEdu-Grid). Pada penelitian ini diajukan suatu metoda penjadwalan alokasi job berdasarkan metoda heuristik, yaitu TS-IndoEdu-Grid sehingga dapat meningkatkan kinerja lingkungan grid IndoEdu-Grid. Hasil pengujian menunjukkan bahwa dengan menggunakan metoda penjadwalan TS-IndoEdu-Grid dapat menimimalisasi nilai parameter makespan sebesar 20% dibanding dengan metoda UD-IndoEdu-Grid. Nilai parameter flowtime pada metodaTS-IndoEdu-Grid menghasilkan nilai 94% lebih kecil daripada nilai parameter flowtime yang dihasilkan pada metoda UD-IndoEdu-Grid.

Kata kunci: IndoEdu-Grid, heuristik, TS-IndoEdu-Grid, UD-IndoEdu-Grid, *makespan*, *flowtime*.

1. PENDAHULUAN

Pada sistem grid yang terhubung secara jaringan, mekanisme penjadwalan *job* menjadi suatu permasalahan tersendiri, karena lingkungan grid meliputi sistem komputasi yang heterogen dan dinamis, melibatkan berbagai spesifikasi perangkat keras, perangkat lunak, dan sistem administrasi (Ang *et al*., 2009). Penelitian oleh Xhafa dan Abraham, (2010), menjelaskan bahwa pada sebagian besar lingkungan grid, metoda penjadwalan *job* menjadi komponen yang paling penting. Kemajemukan dan tingginya dinamika yang terdapat di dalam lingkungan grid menyebabkan metoda penjadwalan *job* harus melibatkan banyak parameter untuk mendapatkan suatu hasil yang optimal. Tetapi pencapaian yang optimal sulit dicapai karena sumber daya grid digunakan untuk tujuan yang berbeda-beda, sehingga mekanisme penjadwalan *job* pada lingkungan grid dikatakan sebagai *computationally hard* (NP-*hard*) (Xhafa dan Abraham, 2010).

Maka Xhafa dan Abraham, (2010), merekomendasikan pendekatan dengan metoda heuristik dan meta-heuristik sebagai metoda penjadwalan *job* pada sistem grid. Penelitian oleh Xhafa dan Abraham, (2010), mengungkapkan beberapa alasan mengapa pendekatan dengan metoda heuristik dan meta-heuristik sesuai untuk penjadwalan *job* pada lingkungan grid yaitu, metoda heuristik tidak memerlukan solusi yang paling optimal karena solusi yang paling optimal hanya akan muncul pada satu waktu dan saat tertentu saja sedangkan proses penjadwalan *job* pada lingkungan grid akan berjalan selama lingkungan grid itu sendiri

masih bekerja. Selain itu dimungkinkan untuk mendapatkan solusi yang efisien pada waktu yang singkat, hal ini dikarenakan meta-heuristik mengabaikan mekanisme pencarian yang berkisar di satu titik saja (lokal optima), misalnya algoritma genetika dan algoritma *Tabu-search*. Sehingga berguna pada saat pengukuran untuk memperoleh nilai minimum dari suatu variabel pengukuran, misalnya *makespan* dan *flowtime*.

Permasalahan lain selain mekanisme penjadwalan *job* pada lingkungan grid adalah bagaimana membangun lingkungan grid yang sesuai dengan kondisi nyata dan dapat digunakan untuk eksperimen yang dilakukan berulang-ulang. Penelitian oleh Sulistio *et al*., (2005), mengungkapkan bahwa untuk melakukan hal tersebut diperlukan biaya yang besar dan waktu pembangunan yang lama. Oleh karena itu salah satu cara untuk mengatasi permasalahan tersebut yaitu dengan memecahkan permasalahan pada lingkungan grid dalam ranah simulasi.

Salah satu penelitian yang berada dalam ranah simulasi adalah peneltian oleh Nugroho dan Suhartanto, (2010); dan Suhartanto *et al*., (2012). Peneliti merancang dan mensimulasikan teknologi grid yang bernama *Indonesian Education Grid* (IndoEdu-Grid) yang digunakan untuk tujuan *e-learning* pada simulator GridSim. Pada penelitiannya, peneliti merancang suatu sumber daya grid yang tersebar di tiga puluh satu propinsi di Indonesia yang menghubungkan institusi perguruan tinggi melalui suatu jaringan yang disebut dengan INHERENT (*Indonesian Higher Education Networks*).

Fokus pada kedua penelitian Nugroho dan Suhartanto, (2010); dan Suhartanto *et al*., (2012), tersebut adalah melakukan penjadwalan paket jaringan, yaitu dengan metoda penjadwalan FIFO (*First in First out*) dan SCFQ (*Self-Clocked Fair Queuing*). Tetapi kedua penelitian tersebut tidak mempertimbangkan permasalahan pada penjadwalan *job* yang diproses di dalam lingkungan grid yang dirancang. Pada lingkungan grid yang dirancang tersebut mensimulasikan *job* yang yang dikirim secara *random* ke sumber daya grid yang dituju.

Oleh karena itu pada penelitian ini diajukan suatu metoda penjadwalan alokasi *job* sehingga dapat meningkatkan kinerja sistem grid dengan meminimalisasi *makespan* dan *flowtime* dari lingkungan grid berdasarkan studi kasus *Indonesian Education Grid* (IndoEdu-Grid). Metoda penjadwalan yang digunakan berdasarkan metoda heuristik yaitu dengan menggunakan algoritma *Tabu-search* yang disesuaikan dengan lingkungan grid IndoEdu-Grid (disebut TS-IndoEdu-Grid pada bagian selanjutnya). Penelitian sebelumnya oleh Xhafa *et al*., (2009), dengan menerapkan algoritma penjadwalan berdasarkan algoritma *Tabu-search* dapat meminimalisasi *makespan* dan *flowtime* pada lingkungan grid yang dibangun. Simulator yang digunakan pada penelitian ini adalah simulator GridSim versi 5.2 (Buyya, R., dan Murshed, M., 2002).

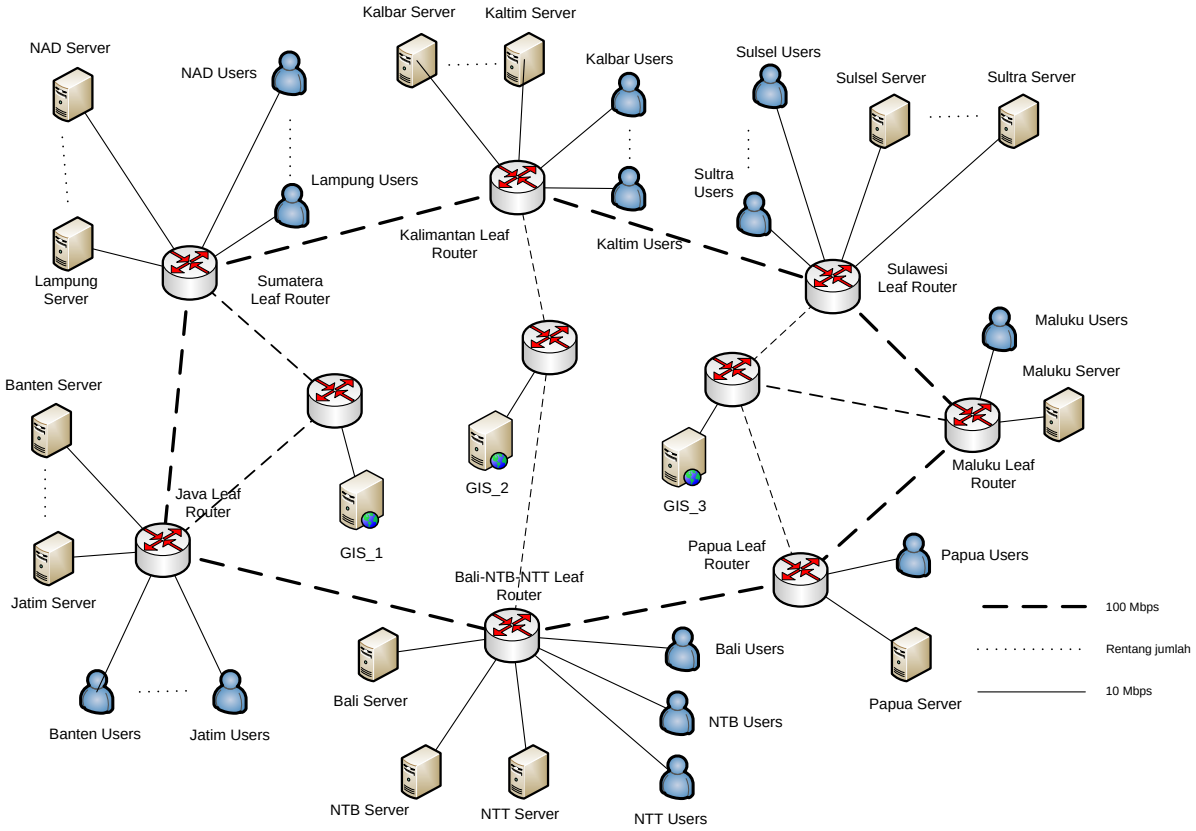
2. PERANCANGAN SIMULASI

Pada tahap ini dirancang entitas-entitas yang diperlukan pada saat tahap simulasi berlangsung. Entitas-entitas tersebut adalah entitas topologi jaringan IndoEdu-Grid, entitas *JobSubmission System*, dan entitas Penjadwal (*Scheduler*).

2.1. Perancangan Topologi Jaringan IndoEdu-Grid

Entitas topologi jaringan dirancang berdasarkan topologi jaringan yang pada penelitian sebelumnya dirancang oleh Suhartanto *et al*., (2012) yaitu, *Indonesian Education Grid* (IndoEdu-Grid) yang ditunjukkan pada Gambar 1. Pada topologi ini, mesin sumber daya grid dikelompokkan berdasarkan propinsi-propinsi yang ada di Indonesia, dengan *leaf router* sebagai penghubung utama antar pulau-pulau utama, misalnya Pulau Sumatera (Sumatera *Leaf Router*), Pulau Kalimantan (Kalimatan *Leaf Router*), Pulau Sulawesi (Sulawesi *Leaf Router*), Kepulauan Maluku (Maluku *Leaf Router*), Pulau Irian (Papua *Leaf Router*), Kepulauan Bali-NTB-NTT (Bali-NTB-NTT *Leaf Router*), dan Pulau Jawa (*Java Leaf Router*). Setiap propinsi diasumsikan memiliki satu *server* dan satu pengguna (*user*).

Setiap mesin sumber daya grid pada Pulau Sumatera dan Jawa terhubung dengan *server* GIS_1 (*Grid Information Service*), setiap mesin sumber daya grid pada Pulau Bali-NTB-NTT dan Kalimantan terhubung dengan *server* GIS_2, dan setiap mesin sumber daya grid pada pulau Sulawesi, Maluku, dan Papua terhubung dengan *server* GIS_3. *Server* GIS berfungsi sebagai penyimpan informasi tentang mesin-mesin sumber daya grid yang berada didaftarnya. *Bandwidth* jaringan yang berasal dari pengguna (*user*) dan dari sumber daya grid ke *leaf router* (ditunjukkan oleh garis lurus pada Gambar 1) diasumsikan sebesar 10 Mbps, dan *bandwidth* jaringan yang menghubungkan setiap *leaf router* (ditunjukkan oleh garis putus-putus pada Gambar 1) sebesar 100 Mbps, asumsi dibuat berdasarkan penelitian sebelumnya oleh (Nugroho, I.B., dan Suhartanto, H., 2010).

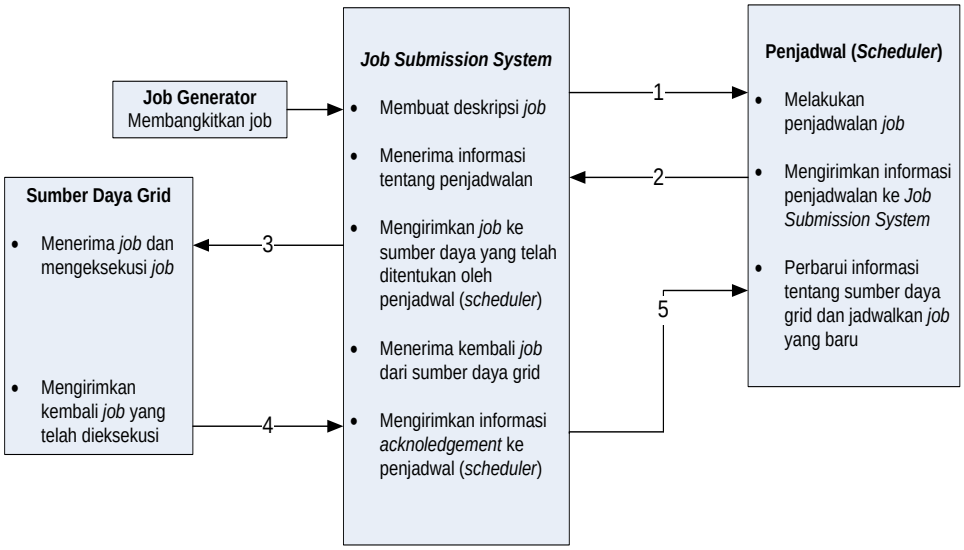


Gambar 1. Topologi Jaringan Indonesian Education Grid (IndoEdu-Grid)

2.2. Perancangan *JobSubmission System* dan Penjadwal (*Scheduler*)

Entitas *JobSubmission System* berfungsi sebagai pembangkit *job-job* yang dibuat dan bertugas mendistribusikan *job* ke seluruh mesin sumber daya grid. Entitas Penjadwal (*Scheduler*) bertugas melakukan penjadwalan alokasi *job* berdasarkan informasi tentang *job* dan informasi tentang mesin sumber daya grid yang tersedia. Kedua entitas ini akan saling berkomunikasi pada saat simulasi berlangsung, komunikasi ini melibatkan mesin sumber daya grid sebagai tempat dimana *job* akan dieksekusi selanjutnya. Berikut adalah skema komunikasi antara *JobSubmission System*, Penjadwal (*Scheduler*), dan mesin sumber daya grid (Klusacek *et al*., 2008).

Berikut adalah tahapan interaksi simulasi sistem grid. Tahap pertama, *job submission system* mengirimkan deskripsi *job* yang akan dijadwalkan. Penjadwal (*Scheduler*) menggunakan daftar sumber daya grid yang tersedia dan beberapa parameter lainnya seperti jumlah prosesor dan kecepatan prosesor (dalam MIPS – *Million Instructinos per Second*). Sistem penjadwalan ini bersifat terpusat (sentralisasi), sehingga penjadwal (*Scheduler*) memiliki informasi tentang seluruh sumber daya yang ada serta akses ke seluruh sumber daya. Tahap kedua, penjadwal (*Scheduler*) menanggapi peristiwa dari *job submission system* dengan menyediakan informasi tentang sumber daya grid mana yang akan mengeksekusi *job*. Komunikasi antara *job submission system* dan penjadwal (*Scheduler*) adalah asinkron (*asynchronous*), oleh karena itu *job submission system* tidak akan menunggu tanggapan dari penjadwal (*Scheduler*) untuk mengirimkan deskripsi *job* baru yang berasal dari *job generator*. Ketiga, *job* dikirimkan ke sumber daya grid oleh *job submission system* berdasarkan beban kerja sumber daya grid yang akan dituju. Kemudian *job* akan dieksekusi oleh sumber daya grid. Tahap keempat, setelah *job* selesai dieksekusi, *job* akan dikirimkan kembali oleh sumber daya grid ke *job submission system*. Tahap terakhir yaitu tahap kelima, pada saat *job* sampai, *job submission system* akan mengirimkan pesan *acknowledgement* ke penjadwal (*Scheduler*). Kemudian penjadwal (*Scheduler*) akan memperbarui informasi tentang beban kerja sumber daya grid. Hal ini dilakukan untuk mendukung tentang perencanaan penjadwalan *job* selanjutnya, sehingga mencegah sumber daya grid berada dalam keadaan *idle*.



Gambar 2. Skema Komunikasi Komponen Sistem Grid (Klusacek *et al*., 2008)

1.1. Kriteria Optimasi

Kriteria optimasi yang dicari adalah meminimalisasi nilai parameter *makespan* dan nilai *flowtime*. *Makespan* adalah waktu terbesar penyelesaian suatu *job* dari seluruh rangkaian *job* ada di suatu mesin (Zomaya, 2001) dan *flowtime* adalah waktu total penyelesaian seluruh *job* (Grosan *et al*., 2007). Menurut penelitian oleh Xhafa dan Abraham (2010), *makespan* adalah suatu indikator produktifitas dari suatu sistem grid, semakin kecil nilai *makespan* berarti suatu penjadwal (*Scheduler*) dapat melakukan penjadwalan *job* secara efisien. Sedangkan *flowtime* mengacu pada waktu respon penjadwal (*Scheduler*) terhadap *job* yang dikirimkan oleh pengguna (*user*), sehingga meminimalisasi nilai *flowtime* berarti mengurangi rata-rata waktu respon dari sistem grid. Sehingga untuk menghitung nilai *makespan* dan nilai *flowtime* diperlukan waktu penyelesain *job*/*actual completion time* (ACT) (Xhafa *et al*., 2009); (Xhafa dan Abraham, 2010).

Jika mesin m adalah mesin sumber daya grid dan j_n adalah rangkaian *job* dimana n = {1, 2, 3, …, n}, maka definisi nilai *makespan* berdasarkan definisi Xhafa dan Abraham, (2010), adalah pada persamaan berikut:

$$makespan = \max\{ACT_{[j][m]},\ j = 1,2,3,....,n\} \quad (1)$$

Dimana ACT[j][m] adalah *Actual Completion Time job* j pada mesin m. Nilai *Actual Completion Time* (ACT) dipengaruhi oleh waktu kesiapan (*ready times*) dan waktu yang diharapkan untuk menyelesaikan suatu *job* dari suatu mesin m (WPj_m). Sehingga nilai *Actual completion time* pada mesin m diekspresikan pada persamaan sebagai berikut (Xhafa *et al*., 2009); (Xhafa dan Abraham, 2010):

$$ACT[m] = rt[m] + WPj_m \quad .... (2)$$

Variabel *rt* adalah *ready times* dari mesin m dan WPj_m adalah waktu yang diharapkan untuk menyelesaikan *job* dari suatu mesin m. Nilai waktu yang diharapkan untuk menyelesaian suatu *job* dihasilkan dari kecepatan prosesor (MIPS) dibagi beban kerja *job* (MI), dapat diekspresikan dengan persamaan berikut:

$$WPj = \frac{k}{b} \quad ..... (3)$$

Dimana nilai parameter *k* adalah kepatan prosesor dalam satuan MIPS dan nilai parameter b adalah beban kerja *job* dalam satuan MI. Nilai parameter *flowtime* adalah waktu total penyelesaian seluruh *job* (Grosan, 2007). Sehingga nilai *flowtime* juga dipengaruhi oleh nilai actual completion time (ACT) suatu *job* j pada mesin m, maka *flowtime* dapat didefinisikan dengan persamaan berikut (Xhafa *et al*., 2009); (Xhafa dan Abraham, 2010):

$$flowtime = \sum ACT_{[j][m]} \quad ..... (4)$$

1.2. Skenario Uji Coba

Pada bagian Pendahuluan telah disebutkan bahwa pada penelitian sebelumnya, (Nugroho, I.B., dan Suhartanto, H., 2010)(Suhartanto, H., Nugroho, I.B., dan Herdiani, A., 2012) dilakukan penjadwalan alokasi *job* secara random. Sehingga pada penelitian ini penjadwalan alokasi *job* secara random diasumsikan penjadwalan dengan metode pendistribusian *job* secara normal (*uniform*). Hal ini berarti *job* memiliki kemungkinan pengalokasian ke semua mesin sumber daya grid yang hampir (approximately) merata. (Sumber: http://

docs.oracle.com/javase/6/docs/api/java/util/Random.html). Mekanisme pendistribusian ini disebut dengan UD-IndoEdu-Grid pada bagian selanjutnya.

Pengujian dilakukan dengan membandingkan kinerja sistem grid yang menggunakan metoda penjadwalan TS-IndoEdu-Grid dengan metoda UD-IndoEdu-Grid. Kinerja sistem grid diukur berdasarkan parameter *makespan* dan *flowtime* yang didapatkan pada saat simulasi berlangsung. Jumlah *job* yang dibuat sebanyak 1000 *job*, dimana jumlah *job* tersebut tetap pada saat simulasi berlangsung. Jumlah mesin sumber daya grid sebanyak 31 dimana jumlah tersebut sama dengan jumlah propinsi di Indonesia berdasarkan Gambar 1. Beban *job* eksekusi sebesar 1000, 2000, 3000, 4000, dan 5000 MI (*Million Instructions*) dimana *job* tersebut juga memiliki ukuran sebesar 10 *Mega Byte*.

Tabel 1. Parameter-parameter simulasi

No	Parameter	Spesifikasi
1	Jumlah Sumber Daya Grid	31
2	Jumlah Mesin	31
3	Jumlah Pengguna	31
4	Jumlah *Job*	1000
5	Jumlah Prosesor per Mesin	2 (dalam satuan PE – *Processing Element*)
6	Kecepatan Prosesor per Prosesor	300 MIPS (dalam *Million Instructions per Seconds*)
7	Beban Eksekusi *Job* (*job length*)	1000, 2000, 3000, 4000, 5.000 (dalam MI - *Million Instruction*)
8	Ukuran *Job* (*job file size*)	10 (dalam *Mega Byte*)

1. EVALUASI KINERJA

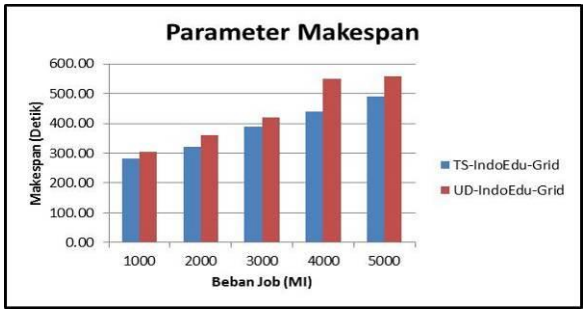
Pada Tabel 2. dibawah adalah hasil pengujian parameter *makespan* dan *flowtime* pada metoda TS-IndoEdu-Grid dan metoda UD-IndoEdu-Grid. Pengujian dilakukan berdasarkan spesifikasi pada Tabel 1, dimana jumlah *job* sebanyak 1000, beban eksekusi *job* sebesar 1000 sampai 5000, dan ukuran *job* sebesar 10 *Mega Byte*.

Pada Tabel 2 terlihat bahwa nilai parameter *makespan* dan *flowtime* pada metoda TS-IndoEdu-Grid selalu lebih kecil daripada nilai parameter *makespan* dan *flowtime* pada metoda UD-IndoEdu-Grid. Perbedaan terbesar nilai *makespan* pada kedua metoda adalah pada beban eksekusi *job* 4000 MI, dimana perbedaan nilai *makespan* pada kedua metoda tersebut mencapai 20%. Pada beban eksekusi 1000 MI perbedaan nilai *makespan* mencapai 7,2%, pada beban eksekusi *job* 2000 MI meningkat sebesar 11,2%, pada beban eksekusi *job* 3000 MI terjadi penurunan sebesar 7,7%, sedangkan pada beban eksekusi *job* terbesar yaitu 5000 MI terjadi penurunan kembali sebesar 11,9%. Sehingga peningkatan nilai parameter *makespan* dari beban eksekusi *job* terkecil yaitu 1000 MI sampai beban eksekusi *job* terbesar yaitu sebesar 5000 MI mencapai 42,5% pada metoda TS-IndoEdu-Grid dan sebesar 45,4% pada metoda UD-IndoEdu-Grid. Grafik nilai *makespan* pada kedua metoda terdapat pada Gambar 2.

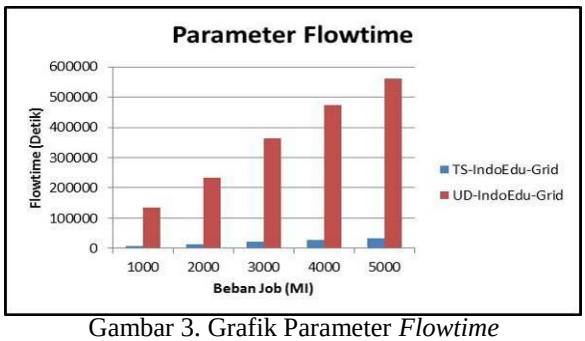
Gambar 3. adalah grafik nilai parameter *flowtime* pada metoda TS-IndoEdu-Grid dan UD-IndoEdu-Grid. Terlihat pada grafik bahwa nilai parameter *flowtime* pada metoda TS-IndoEdu-Grid lebih kecil dibandingkan dengan nilai parameter *flowtime* pada metoda UD-IndoEdu-Grid. Perbedaan nilai parameter ini hingga mencapai 94% pada seluruh beban eksekusi *job*.

Tabel 2. Hasil Pengujian pada Skenario Jumlah *Job* 1000

Beban Eksekusi *Job* (MI)	TS-IndoEdu-Grid		UD-IndoEdu-Grid	
	Makespan (Detik)	*Flowtime* (Detik)	*Makespan* (Detik)	*Flowtime* (Detik)
1000	282.14	8000	304.10	134336
2000	320.89	14000	361.17	232736
3000	387.95	22000	420.29	362340
4000	439.89	28000	550.10	474488
5000	490.89	34000	557.17	561408



Gambar 2. Grafik Parameter *Makespan*



Gambar 3. Grafik Parameter *Flowtime*

2. KESIMPULAN

Berdasarkan hasil pengujian terlihat bahwa metoda TS-IndoEdu-Grid dapat meminimalisasi nilai parameter *makespan* dan *flowtime* dibandingkan metoda UD-IndoEdu-Grid pada skenario jumlah 1000, beban eksekusi *job* 1000 MI sampai 5000 MI, dan ukuran *job* sebesar 10 *Mega Byte*. Sehingga berdasarkan pengujian tersebut dapat disimpulkan bahwa metoda TS-IndoEdu-Grid mampu meningkatkan kinerja lingkungan grid Indonesian Education Grid yang telah dirancang pada penelitian sebelumnya berdasarkan parameter *makespan* dan *flowtime*.

3. DAFTAR PUSTAKA

Ang, T.F., Ng, W.K., Ling, T.C., Por, L.Y., dan Liew, C.S. (2009), "A Bandwidth-Aware *Job* Grouping-Based Scheduling on Grid Environment." *Information Technology Journal* (Asian Network for Scientific Information). Vol. 8, No. 3, hal. 372 - 377.

Buyya, R., dan Murshed, M., (2002), "GridSim: A Toolkit for The Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing." *Concurrency and Computation: Practice and Experience*. John Wiley & Sons, Ltd., hal. 1175 - 1220.

Grosan, C., Abraham, A., dan Helvik, B., (2007), "Multiobjective Evolutionary Algorithms for Scheduling *Job*s on Computational Grids.", *DIS International Conference - Applied Computing*, Salamanca, Nuno Guimares and Pedro Isaias.

Klusacek, D., Matyska, L., dan Rudova, H., (2008), "Alea - Grid Scheduling Simulation Environment.", *7th International Conference Parallel Processing and Applied Mathematics*, Gdansk, Polandia, Springer Berlin Heidelberg, hal. 1029-1038.

Nugroho, I.B., dan Suhartanto, H., (2010), "Design and Simulation of Indonesian Education Grid Topology Using Gridsim Toolkit." *Asian Journal of Information Technology*, *Medwell Journals*, Vol. 9, hal. 263 - 271.

Suhartanto, H., Nugroho, I.B., dan Herdiani, A., (2012), "Province Based Design and Simulation of Indonesian Education Grid Topology." *International Journal of Computer Science Issues*, Vol. 9, No. 1, hal. 142 - 147.

Sulistio, A., Poduvaly, G., Buyya, R., dan Thamy, C. K. (2005), "Constructing A Grid Simulation with Differentiated Network Service Using GridSim." *6th International Conference on Internet Computing (ICOMP 2005).* Vol 6, Las Vegas.

Xhafa, F., Carretero, J., Dorronsoro, B., dan Alba, E., (2009), "A Tabu Search Algorithm for Scheduling Independent *Job*s in Computational Grids", *Computing and Informatics*, Vol. 28, hal. 1001 – 1014.

Xhafa, F., dan Abraham, A., (2010), "Computational Models and Heuristic Methods for Grid Scheduling Problems." *Future Generation Computer System*, Elsevier, hal. 608 - 621.

Zomaya, A. Y., (2001), "Observations on Using Genetic Algorithms for Dynamic Load-Balancing.", *IEEE Transactions on Parallel and Distributed Systems*, IEEE.

Oracle., *docs.oracle.com*, http://docs.oracle.com/javase/6/docs/api/java/util/Random.html, di akses pada 7 Juli 2013