OPTIMASI DAYA DATA CENTER CLOUD COMPUTING PADA WORKLOAD HIGH PERFORMANCE COMPUTING (HPC) DENGAN SCHEDULING PREDIKTIF SECARA REALTIME

Amirullah[1)], Royyana Muslim Ijtihadie[2)], dan Hudan Studiawan[3)]
[1, 2, 3)]Teknik Informatika, Institut Teknologi Sepuluh Nopember
Jl. Teknik Kimia, Gedung Teknik Informatika, Kampus ITS Sukolilo, Surabaya, 60111
e-mail: amir.pnl08@gmail.com[1)], roy@if.its.ac.id[2)], hudan@if.its.ac.id[3)]

ABSTRAK

Tantangan terbesar yang muncul pada data center cloud computing adalah meningkatnya biaya konsumsi daya. Pengembangan data center akan bertolak belakang dengan penghematan daya, semakin tinggi performa sebuah data center, maka semakin tinggi pula konsumsi energi yang dibutuhkan, hal ini disebabkan oleh kebutuhan jumlah server ataupun hardware pada data center yang semakin meningkat.

Data center cloud computing yang berbasis High Performance Computing (HPC) merupakan sebuah teknologi yang dibangun dari kumpulan server dalam jumlah besar untuk menjamin ketersediaan tinggi dari sebuah cloud computing, namun sebenarnya beberapa server tersebut hanya direncanakan untuk beban puncak yang jarang atau tidak pernah terjadi. Ketika beban pada titik terendah, maka server tersebut akan berada dalam kondisi idle.

Optimasi daya dengan DNS (Dynamics Shutdown) dengan memanfaatkan kondisi beban rendah server dapat menjadi solusi yang tepat untuk mengurangi konsumsi daya pada data center. Namun jika optimasi tersebut dilakukan dengan konvensional dan hanya berdasarkan data realtime, maka kemungkinan besar akan berpengaruh terhadap performa data center.

Optimasi yang dilakukan pada penelitian ini adalah dengan metode prediksi menggunakan moving average untuk menentukan penjadwalan DNS. Hasil pengujian dengan komputer virtual menunjukkan bahwa dengan metode prediksi dapat mengurangi konsumsi daya sebesar 1,14 Watt dibandingkan dengan metode konvensional.

***Kata Kunci**: Cloud computing, Data Center, Optimasi Daya, Dynamics Shutdown, Prediksi, RealTime, moving average, penjadwalan.*

ABSTRACT

The biggest challenge that emerged in the data center cloud computing is increasing costs for power consumption. The higher performance of a data center, the higher the energy consumption required, this was due to the needs of the number of servers or hardware in the data centers are growing.

Data center cloud computing-based High Performance Computing (HPC) is a platform built on very large number of servers to ensure high availability of cloud services. The workload is likely to be happening at specific periods of time, instead of all time. Therefore, when the load is at its lowest point, the server will be in the idle condition.

Implementing DNS (Dynamics Shutdown) at appropriate times could be a solution for reducting power consumption in the data center. However, if the optimization is done only on the basis of conventional and real time data, it will most likely affect the performance of the data center.

In this study, optimization is done by conducting prediction using moving average method to determine the schedule of DNS. Results of testing with virtual computer shows that the prediction methods can reduce the power consumption of 1.14 Watts compared to conventional methods.

***Keywords**: Cloud computing, Data Center, Power optimization, Dynamics Shutdown, Prediction, RealTime, moving average, scheduling.*

I. PENDAHULUAN

PADA beberapa tahun terakhir, perkembangan layanan *cloud computing* mengalami peningkatan yang sangat drastis, hal ini disebabkan karena adanya keterlibatan data *center* dan paradigma komputasi paralel. *Cloud computing* dan data *center* tidak dapat dipisahkan. Data *center* merupakan fasilitas untuk menempatkan sistem komputer dan komponen-komponen yang terkait [1]. Pengoperasian data *center* yang tersebar di wilayah yang luas memerlukan pertimbangan seberapa besar konsumsi energi terhadap total biaya pengoperasian dari data *center*.

Masalah utama dari infrastruktur *cloud* bukan hanya dari segi biaya yang mahal akan tetapi juga kurang ramah lingkungan. Biaya pemakaian energi yang tinggi kemudian emisi karbon yang dihasilkan akibat tingginya kebutuhan akan energi listrik baik untuk tujuan yang berhubungan dengan komputasi ataupun untuk tujuan

pendukung operasional dari data *center*. Para penyedia layanan infrastruktur *cloud* perlu mengukur agar margin keuntungan layanan *cloud* tidak tereduksi oleh tingginya biaya pemakaian energi listrik [2].

Tantangan terbesar yang muncul pada data *center* adalah meningkatnya biaya konsumsi untuk daya. Seperti yang dijelaskan oleh Filani dalam papernya, dalam dekade terakhir biaya untuk daya dan pendingin data *center* telah meningkat sebesar 400% dan kecenderungannya akan terus meningkat. Pada beberapa kasus, konsumsi daya listrik memakan porsi 40-50% dari keseluruhan biaya operasional dari data *center* [3].

Pengembangan pada data *center* akan memiliki kebutuhan untuk menambah server baru agar dapat memenuhi lonjakan akses yang semakin tinggi. Namun berdasarkan survei terakhir oleh Filani, faktor penghambat utama dalam pengembangan data *center*, senilai 56% adalah berdasarkan konsumsi daya dan pendinginan [3]. Dengan kondisi tersebut, maka data *center* saat ini menghadapi dua masalah besar yakni bagaimana mengembangkan layanan baru tetapi dengan konsekuensi konsumsi daya untuk komputasi dan pendinginan terus meningkat. Jika kecenderungan ini terus terjadi, maka kemampuan data *center* untuk menambah layanan baru akan terhambat.

Untuk mengatasi hal ini, pengelola data *center* memiliki beberapa pilihan diantaranya sebagai berikut [3]:

1. Menambah kapasitas daya dan pendingin
2. Membangun data *center* baru
3. Melakukan pengelolaan energi yang memaksimalkan kapasitas yang ada.

Dua pilihan awal akan sangat mahal karena melibatkan belanja modal dan instalasi baru. Maka pilihan ketigalah yang paling memungkinkan untuk mengatasi dua hal tersebut di atas.

Ada beberapa mekanisme optimasi daya yang dapat dilakukan untuk memaksimalkan kapasitas server komputasi yang ada, pertama adalah dengan mengurangi daya pada setiap server sesuai dengan kebutuhan dan yang ke dua mematikan (shutdown) server yang dalam kondisi *idle* sehingga konsumsi energi bisa ditekan pada kondisi minimal.

Telah ada penelitian yang berfokus pada dua hal tersebut yaitu mengurangi daya dan mematikan (shutdown) daya pada server yang tidak diperlukan. Hasil dari beberapa penelitian menunjukkan bahwa penerapan dua mekanisme tersebut dapat menghasilkan tingkat optimasi yang sangat baik. Dalam penelitian [4] optimasi dengan cara mengurangi konsumsi daya pada CPU dilakukan dengan skema *Dynamic Voltage and Frequency Scaling* (DVFS) telah menunjukkan penghematan energi CPU hingga mencapai 80%. Penelitian [5] telah mengkaji perbandingan optimasi daya antara DVFS dengan DVFS + DNS (Dynamic Shutdown) hasilnya menunjukkan bahwa penghematan yang lebih baik adalah dengan menggunakan DVFS + DNS.

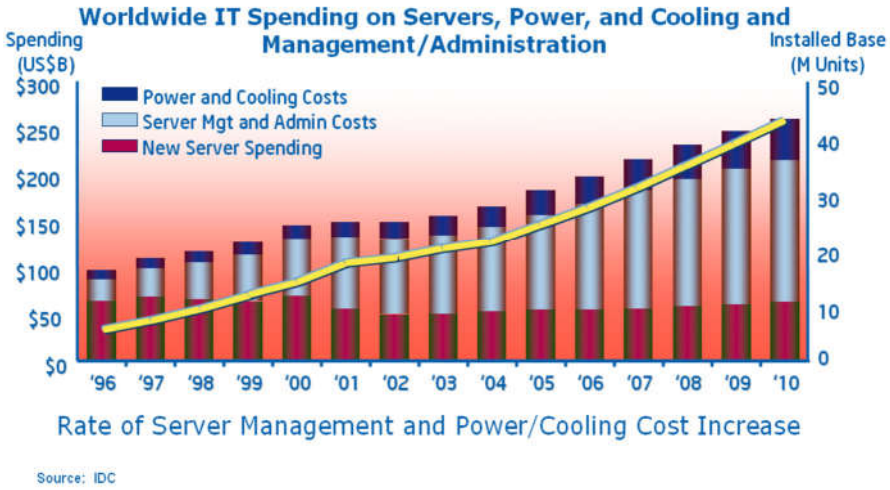
DVFS dan DNS merupakan skema optimasi berdasarkan deteksi beban secara dinamis atau *realtime* sesuai kondisi beban kerja pada server[6], artinya saat akses ataupun beban kerja pada server dalam keadaan rendah, sangat rendah atau bahkan tidak ada beban sama sekali, maka sistem akan mengurangi konsumsi daya pada server tersebut sehingga kunsumsi daya mendekati titik terendah. Ketika server berada pada kondisi *idle*, maka sistem akan melakukan shutdown otomatis pada server tersebut sehingga konsumsi daya dapat diminimalkan.

Namun masalahnya adalah jika beban kembali melonjak secara tiba-tiba dan saat itu juga sistem membutuhkan tambahan komputasi dari server yang telah dikurangi kemampuannya atau telah di-*shutdown*, maka sistem akan membutuhkan waktu beberapa menit agar dapat menjadikan sistem kembali pada performa normal ataupun optimal. Semakin lama durasi waktu yang dibutuhkan, maka semakin lama pula performa server dapat dikembalikan, sehingga performa server secara keseluruhan juga akan ikut terganggu. Dan bagaimana jika ini terjadi pada ratusan bahkan ribuan server yang ada dalam data *center*, maka ini sangat berpengaruh pada kinerja keseluruhan sistem *cloud computing* yang telah dibangun.

Ada hal yang perlu diperhatikan untuk memaksimalkan skema tersebut, yaitu bagaimana melakukan penjadwalan sehingga proses DNS (*Dynamics Shutdown*) tersebut dapat dilakukan tanpa mengurangi performa pada kinerja data *center*. Penjadwalan yang kurang tepat dalam membaca atau mendeteksi fluktuasi akses pada setiap server dapat menyebabkan layanan data *center cloud* menjadi bermasalah.

Penelitian mengenai penjadwalan pada optimasi daya telah dilakukan pada beberapa paper seperti [6] menggunakan metode penjadwalan *realtime*, pada penelitian [7] menggunakan *scheduling parallel realtime* sebagai metode penjadwalannya. Namun dalam beberapa penelitian tersebut belum membahas metode penjadwalan berdasarkan metode prediksi maupun kombinasi metode prediksi dengan *realtime*. Penelitian mengenai metode prediksi atau prediktif pada *cloud computing* biasanya sering dilakukan pada penelitian *auto scaling*.Seperti pada[8] dan[9] telah membuat penelitian tentang *auto scaling* pada virtual untuk memprediksi *workload*, namun belum ada penelitian untuk penjadwalan optimasi daya menggunakan model prediksi dan kombinasi antara prediksi dan *realtime*.

Dalam penelitian ini akan dilakukan metode optimasi daya pada server dengan skema *Dynamics Shutdown* (DNS) [10]. Untuk penjadwalan akan digunakan kombinasi metode prediksi dan *realtime*, sehingga proses



Gambar 1. Struktur Pembiayaan Data Center [3]

optimasi dapat dilakukan dengan tepat tanpa mempengaruhi kinerja atau performa dari sistem data *center cloud computing*.

Metode penjadwalan pada penelitian ini akan digunakan pendekatan metode atau algoritma *moving average* yang kemudian akan dikombinasikan dengan *realtime*. Metode *moving average* menggunakan nilai rataan sejumlah data masa lalu dengan menentukan jumlah periode tertentu untuk menentukan nilai prediksi yang akan datang [11]. Penelitian metode *moving average* banyak digunakan untuk memprediksi fluktuasi saham seperti pada penelitian [12]. Sebagaimana kita ketahui, saham memiliki fluktuasi yang sangat dinamis, namun dengan metode prediksi seperti penelitian [12] menunjukkan hasil yang sangat baik dalam mendeteksi perkembangan saham. Dalam penelitian *cloud computing* metode *moving average* digunakan sebagai metode prediksi pada autoscaling[13]. Penelitian lain pada [14] telah menganalisa bagaimana performa avaibility prediksi *auto scaling cloud computing* dengan menggunakan *moving average*. Dari berbagai literatur belum ditemukan metode kombinasi antara *moving average* dan *realtime* digunakan untuk optimasi daya menggunakan DNS. Namun dari berbagai penelitian tersebut memiliki kesamaan dengan penelitian ini, maka jelas sekali bahwa metode *moving average* yang dikombinasikan dengan *realtime* memungkinkan dapat diterapkan pada optimasi daya dengan skema DNS.

Hipotesis yang akan didapatkan dengan menggunakan metode prediksi *moving average* dari penelitian ini adalah dapat melakukan optimasi daya pada *cloud computing* tanpa mempengaruhi performa dari layanan tersebut.

Dalam penelitian ini akan memuat beberapa pembahasan yaitu tinjauan pustaka, metodologi, uji coba dan evaluasi, serta kesimpulan dan saran.

II. TINJAUAN PUSTAKA

A. Optimasi Daya

Masalah utama dari pesatnya perkembangan komputasi, baik itu komputasi individual maupun komputasi sekelas data *center provider cloud computing* adalah meningkatnya konsumsi energi. Gambar 1 menunjukkan bahwa daya dan pendinginan pada data *center* cenderung memiliki biaya yang terus meningkat seiring dengan perkembangan waktu.

Untuk mengatasi hal ini, pengelola data *center* memiliki beberapa pilihan diantaranya sebagai berikut [3]:

1. Menambah kapasitas daya dan pendingin
2. Membangun data *center* baru
3. Melakukan pengelolaan energi yang memaksimalkan kapasitas yang ada.

Dua pilihan awal akan sangat mahal karena melibatkan belanja modal dan instalasi baru. Maka pilihan ketigalah yang paling memungkinkan untuk mengatasi dua hal tersebut di atas.

TABEL I
PERBANDINGAN SKEMA EFISIENSI ENERGI [5]

Parameter	Power Consumtion (kWh)			
	No energy-saving	**DVFS**	**DNS**	**DVFS+DNS**
Data center	503.4	486.1 (96%)	186.7 (37%)	179.4 (35%)
Servers	351	304.5 (97%)	138.4 (39%)	132.4 (37%)
Switches	152.4	145.6 (95%)	48.3 (32%)	47 (31%)
Energy cost/year	$441k	$435k	$163.5k	$157k

B. *Dynamics Shutdown* (DNS)

Dalam sebuah penelitian menjelaskan bahwa server yang dalam kondisi *idle* akan tetap mengkonsumsi energi sebesar 66% dari kapasitas penuhnya[5], maka pada mekanisme DNS, skema penghematan dilakukan adalah dengan cara mematikan server yang dalam kondisi *idle* sehingga konsumsi energi bisa ditekan pada kondisi minimal.

Pemanfaatan DNS oleh [15] telah membahas penghematan energi pada *Storage Area Networks* (SAN) mencapai 35,6 %. Sedangkan dalam penelitian [5] telah membahas perbandingan DVFS, DNS, dan DVFS+DNS seperti yang ditunjukkan padaTabel I.

Pada Tabel I menunjukkan bahwa penghematan energi dengan menggunakan skema DNS hanya menghabiskan biaya $163.5k dari pada yang tidak menggunakan skema sebesar $441 dan dengan skema DVFS adalah $435k. dan akan lebih bagus lagi jika DNS dikombinasi dengan DVFS yaitu hanya $157k. Ini artinya skema DNS menunjukkan hasil yang sangat bagus dalam hal optimasi konsumsi daya.

C. *Moving average*

Moving average merupakan salah satu jenis metode prediksi berdasarkan *time* series atau keturutan waktu kuantitatif dalam teori peramalan. Metode *moving average* menggunakan nilai di masa lalu yang digunakan sebagai acuan dalam melakukan prediksi di masa depan. Secara umum tujuan dari jenis peramalan *time* series adalah menemukan pola dalam deret historis dari suatu data dan mengeskploitasinya untuk dijadikan pola masa depan.

Data *time series* seringkali mengandung ketidakteraturan yang akan menyebabkan prediksi yang beragam. Untuk menghilangkan efek yang tidak diinginkan dari ketidakteraturan ini, metode *moving average* mengambil beberapa nilai yang sedang diamati, memberikan rataan, dan menggunakannya untuk memprediksi nilai untuk periode waktu yang akan datang [11]. Semakin banyak jumlah pengamatan yang dilakukan, maka pengaruh metode *moving average* akan lebih baik. Meningkatkan jumlah observasi akan menghasilkan nilai peramalan yang lebih baik karena ia cenderung meminimalkan efek-efek pergerakan tidak biasa yang muncul pada data. Persamaan (1) adalah persamaan matematis single *moving average*[14].

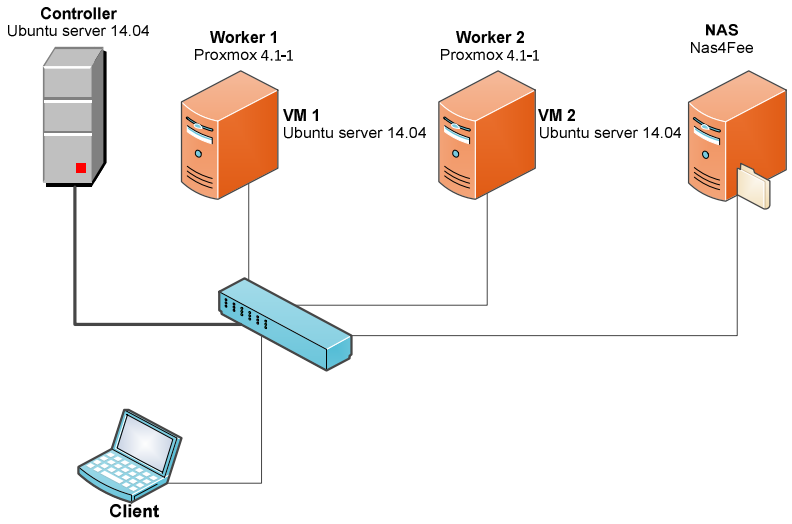
$$M_t = F_{t+1} = \frac{Y_t + Y_{t-1} + Y_{t-2} + \cdots + Y_{t-N+1}}{N} \quad (1)$$

dimana:
M_t adalah *Moving average* untuk periode *t*
F_{t+1} adalah prediksi *usage* CPU pada periode *t* + 1
N adalah total jumlah periode rataan.
Y_t adalah nilai *real* pada periode *t*

III. METODOLOGI

A. Desain sistem

Arsitektur *cloud computing* yang diusulkan terdiri dari *cluster* server. Dalam penelitian ini akan digunakan beberapa server seperti yang terlihat pada rancangan Gambar 2. Di antaranya terdiri dari satu server *controller,* dua server *worker.* Pengujian sistem yang diusulkan pada penelitian ini hanya berfokus pada lingkungan server, artinya penelitian ini tidak akan memasukkan *core network*, *link state*, dll sebagai mana *data center cloud computing*. Namun sistem yang diusulkan ini akan didesain semirip mungkin dengan *data center cloud computing*, sistem *cloud computing*pada pengujian ini akan memanfaatkan *framework* yang telah tersedia sebagai virtualisasi.



Gambar 2. Desain Sistem

Pelaksanaan uji coba sistem dilakukan pada virtual komputer dengan memanfaatkan VMware Workstation [16]. Ada beberapa komputer server virtual yang dipersiapkan dalam VMware Workstation, diantaranya adalah satu komputer sebagai server controller dan dua komputer sebagai server *worker*, satu server sebagai NAS serta ada satu komputer client. Penjelasan istilah yang digunakan pada desain system dapat dilihat pada Tabel II.

B. Prediksi Menggunakan *Moving average*

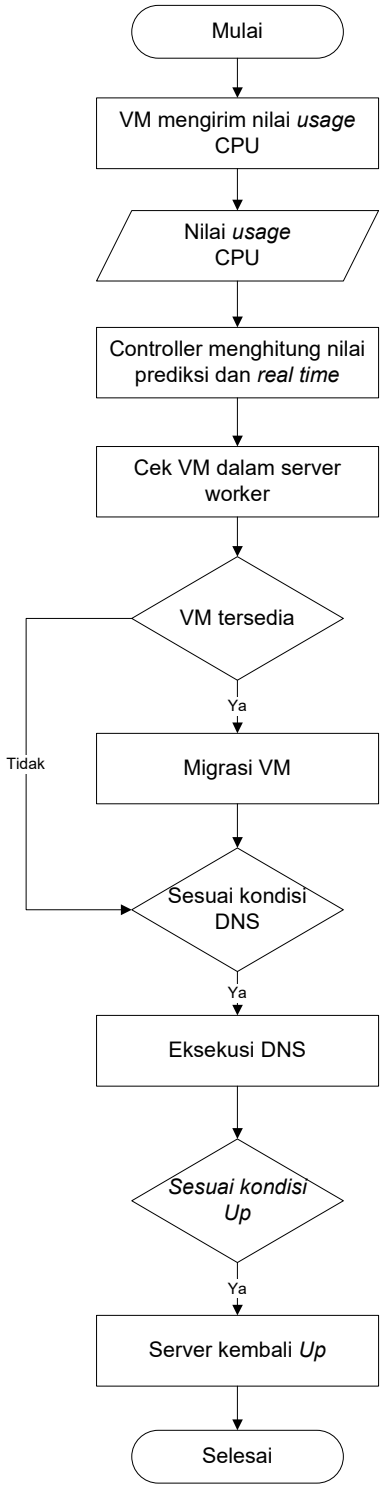
Controller akan menjalankan fungsi *moving average,* dimana akan mengolah setiap data yang dikirimkan oleh server VM dan *worker*. *Moving average* akan bekerja secara *realtime*, setiap data *usage* CPU yang masuk akan langsung dihitung untuk memprediksi nilai berikutnya. Nilai *usage* CPU yang dihitung adalah nilai rata-rata perlima menit, dan selanjutnya akan diprediksi nilai *usage* CPU pada lima menit berikutnya. Nilai *usage* CPU yang dijadikan sebagai data *forecast* atau prediksi adalah nilai *usage* CPU dari server VM, bukan nilai *usage* CPU server *worker*. Hal ini dikarenakan *usage* CPU dari server VM tidak mengalami perlakuan shutdown, sehingga nilai *usage* CPU dapat dikirimkan secara terus menurus tanpa henti walaupun server *worker* telah dimatikan.

TABEL II
ISTILAH YANG DIGUNAKAN DALAM DESAIN SISTEM

No	Istilah	Keterangan
1	Contoller	Merupakan server pengatur yang akan bertindak untuk memanajemen sistem yang dibangun. *Controller* menerima kiriman data *usage*CPU dari*worker*, dan VM serta mengolah kiriman data tersebut.*Controller* akan mengirimkan perintah pada *worker* yang berupa migrasi VM, eksekusi DNS, atau *Up* kembali. Pada *controller* juga akan menampilkan kondisi dari semua server, diantaranya *usage* CPU, konsumsi daya, *Up time*, jadwal migrasi atau DNS. Sistem operasi yang digunakan pada Controller adalah Ubuntu Server 14.04.
2	Worker	Server*worker* diinstal Linux Proxmox [17] sebagai sistem operasi sekaligus sebagai *framework cloud computing.*Setiap server *worker* akan menjalankan satu server VM. Semua *worker* dapat membaca kondisi server yaitu membaca *workload* penggunaan CPU (*usage* CPU) dan list jumlah VM yang aktif.
3	VM	Server VM adalah server yang berada dalam *worker*. Server VM dapat berpindah (migrasi) dari server *worker* 1 ke server *worker* 2 ataupun sebaliknya, sesuai intruksi penjadwalan yang telah dibuat oleh server controller. VM dapat membaca kondisi server VM itu sendiri dan mengirimkan ke controller.
4	NAS	NAS yang digunakan adalah Nas4Free [18]. Penggunaan NAS bertujuan untuk memudahkan migrasi VM antar server*worker*.
5	Client	Komputer *client* akan diinstal aplikasi Httperf untuk membangkitkan data dan mengirim/mengakses ke VM yang akan mengakibatkan fluktuasi *usage* CPU pada VM dan *worker.Client* juga digunakan untuk memantau ketersedian sistem yang dibangun. Parameter yang akan dilihat untuk memeriksa ketersedian sistem adalah *response time.*

C. Optimasi Daya

Optimasi daya yang akan dilakukan pada penelitian ini adalah dengan mempertimbangkan performa sistem tetap berada pada keaadaan optimal, sehingga penghematan konsumsi daya pada server data *center* yang dilakukan pada DNS dapat dilakukan dengan bantuan prediksi *usage* CPU seperti yang telah dijelaskan sebelumnya. Eksekusi DNS akan bekerja sesuai nilai *threshold* yang telah ditentukan. Sebelum DNS dijalankan, maka sistem akan menjalankan migrasi VM ke server*worker* yang yang lain terlebih dahulu. Selanjutnya server kembali dalam kondisi *Up* ketika *threshold* DNS sudah tidak memenuhi lagi. Diagram alir optimasi daya dapat dilihat pada Gambar 3.



Gambar 3.Flowchart Optimasi Daya

IV. UJI COBA DAN EVALUASI

A. Distribusi Beban

Distribusi beban pada pengujian penelitian ini dibangkitkan dengan mengenerate trafik secara berkala dan terus menerus dengan bantuan *tool* Httperf [19] pada lingkungan Linux. Namun sebelum trafik dibangkitkan dengan *tool* Httperf, maka terlebih dahulu akan ditentukan pemodelan trafik dengan mengunakan distribusi Poisson.

Pada Tabel III memperlihatkan nilai beban dengan pemodelan distribusi Poisson. Nilai X adalah nilai sampel dimulai dari 0 sampai dengan 11. Nilai *mean* 7 mengindentifikasi bahwa kenaikan dengan nilai tertinggi akan terjadi pada sampel ke-7 yaitu kira-kira pada menit ke-30. Nilai-nilai probabilitas yang dihasilkan bernilai di bawah 1. Oleh karena itu, nilai tersebut harus dinaikkan untuk mengasilkan nilai request yang cukup untuk membangkitkan beban *request* dengan Httperf. Angka 950 dipilih sebagai perkalian dengan nilai probabilitas yang dihasilkan sebelumnya, hal ini disebabkan karena dengan perkalian nilai 950 menghasilkan *request* maksimal yang dirasa cukup, yaitu menghasilkan *request* maksimal 141,553.

Selanjutnya setiap sampel akan ditentukan total jumlah request selama lima menit. Dalam Tabel III terlihat-bahwa ada 12 sampel *request* ataupun koneksi, masing-masing sampel *request* akan berlangsung selama 5 menit, dimulai dari menit ke-00 smpai dengan menit ke-55. Nilai *request* perdetik setiap lima menit awal akan berbeda dengan nilai *request* perdetik pada lima menit kedua, dan begitu seterusnya. Sebagai contoh pada menit ke-00, *request* akan berlangsung dari menit ke-00 sampai menit ke-05 dan memiliki nilai 1 *request* perdetik, sehingga total *request* yang dijalankan selama lima menit tersebut adalah 300 *request*.

Uji coba pada penelitian ini dilakukan dengan dua skenario beban. Skenario pertama adalah pada pola peningkatan beban, dan skenario kedua adalah pada pola penurunan beban.oleh karena itu, nilai yang telah dihasilkan menggunakan pemodelan distribusi Poisson tersebut akan dibagi menjadi dua pola yaitu pola beban meningkat dan pola beban menurun. Distribusi Poisson untuk beban meningkat diperlihatkan padaTabel IV, sedangkan distribusi Poisson untuk beban menurun diperlihatkan pada Tabel V.

Tabel IVmenunjukkan bahwa uji coba pada peningkatan beban dilakukan selama 30 menit dari menit ke-15 sampai dengan menit ke-45 dan pada Tabel V uji coba pada penurunan beban dilakukan selama 30 menit pula yaitu dari menit ke-45 sampai dengan menit ke-15.

Pada Tabel IV dan Tabel V, nilai sampel X untuk pola beban meningkat dimulai dari 3 sampai dengan 9, sedangkan pada pola beban menurun dimulai dari 9 sampai dengan 11 dan 0 sampai dengan 3. Untuk nilai mean, trafik, trafik perdetik dan jumlah trafik dalam 5 menit akan mengikuti berdasarkan hasil perhitungan pada distribusi Poisson sebelumnya yaitu seperti yang telah diperlihatkan pada Tabel III.

Selanjutnya, dari nilai yang telah dibangkitkan dengan distribusi Poisson tersebut pada kedua pola beban tersebut akan digunakan untuk meng-*generate* trafik dengan *tool* Httperf. Pengujian akan dilakukan pada dua metode, yang pertama adalah dengan metode prediksi dan yang kedua adalah dengan metode konvensional, kemudian akan dibandingkan hasil pengujian antara kedua metode tersebut.

TABEL III
DATA PEMODELAN TRAFIK DENGAN DISTRIBUSI POISSON

No	Time Menit	X	Probabilitas mean 7,0	Trafik x 950	Trafik Perdetik (request)	Jumlah Trafik dalam 5 menit (request)
1	00	0	0,0009	0,866	1	300
2	05	1	0,0064	6,064	6	1800
3	10	2	0,0223	21,224	21	6300
4	15	3	0,0521	49,523	50	15000
5	20	4	0,0912	86,665	87	26100
6	25	5	0,1277	121,331	121	36300
7	30	6	0,1490	141,553	142	42600
8	35	7	0,1490	141,553	142	42600
9	40	8	0,1304	123,859	124	37200
10	45	9	0,1014	96,334	96	28800
11	50	10	0,0710	67,434	67	20100
12	55	11	0,0452	42,913	43	12900

TABEL IV
BEBAN MENINGKAT DENGAN DISTRIBUSI POISSON

`	Time Menit	X	Probabilitas mean 7,0	Trafik x 950	Trafik Perdetik (request)	Jumlah Trafik dalam 5 menit (request)
1	15	3	0,0521	49,523	50	15000
2	20	4	0,0912	86,665	87	26100
3	25	5	0,1277	121,331	121	36300
4	30	6	0,1490	141,553	142	42600
5	35	7	0,1490	141,553	142	42600
6	40	8	0,1304	123,859	124	37200
7	45	9	0,1014	96,334	96	28800

TABEL V
BEBAN MENURUN DENGAN DISTRIBUSI POISSON

No	Time Menit	X	Probabilitas mean 7,0	Trafik x 950	Trafik Perdetik (request)	Jumlah Trafik dalam 5 menit (request)
1	45	9	0,1014	96,334	96	28800
2	50	10	0,0710	67,434	67	20100
3	55	11	0,0452	42,913	43	12900
4	00	0	0,0009	0,866	1	300
5	05	1	0,0064	6,064	6	1800
6	10	2	0,0223	21,224	21	6300
7	15	3	0,0521	49,523	50	15000

TABEL VI
PERBANDINGAN HASIL UJI COBA KETIKA BEBAN MENINGKAT

Metode pengujian	Rata-rata Usage CPU (%)				Response *time* (ms)			Daya (watt)
	VM1	VM2	worker1	worker2	VM1	VM2	Total response *time*	
Prediksi	29,88	29,59	31,01	33,18	50,4	67,2	117,6	13,25
Konvensional	23,84	22,53	29,23	28,26	52,8	62,1	114,9	12,78

TABEL VII
PERBANDINGAN HASIL UJI COBA KETIKA BEBAN MENURUN

Metode pengujian	Rata-rata Usage CPU (%)				Response *time* (ms)			Daya (watt)
	VM1	VM2	worker1	worker2	VM1	VM2	Total response *time*	
Prediksi	19,02	19,25	42,13	19,46	24,4	21,9	46,3	11,57
Konvensional	22,92	20,9	29,39	28,67	23,5	24,2	47,7	13,18

TABEL VIII
KONSUMSI DAYA DAN RESPONSE TIME PADA DUA METODE PENGUJIAN

Metode pengujian	Response *time* rata-rata (ms)	Daya (Watt)
Prediksi	81,95	24,82
Konvensional	81,3	25,96
Selisih/penghematan	**+0,65 ms**	**1,14**

A. Perbandingan Hasil Uji Coba

Pengujian *response time* dan konsumsi daya dilakukan. Selanjutnya akan dibandingkan pengujian dengan metode prediksi dan dengan metode konvensional.

Ada dua skenario utama pengujian yang dilakukan, yaitu pada saat beban meningkat dan pada saat beban menurun. Pada Tabel VI menunjukkan perbandingan hasil uji coba ketika beban meningkat, hasil dibandingkan antara sistem pengujian yang menggunakan metode prediksi dan metode konvensional. Konsumsi daya yang digunakan oleh sistem dengan prediksi memiliki nilai yang lebih besar daripada sistem yang di uji cobakan dengan konvensional. Nilai konsumsi daya dengan metode prediksi adalah 13,25 Watt sedangkan nilai konsumsi daya pada sistem konvensional adalah 12,78 Watt. Perlu diketahui bahwa dari beberapa percobaan, menulis menganalisa bahwa yang menyebabkan konsumsi daya sedikit lebih besar pada metode prediksi adalah karena pada metode prediksi melakukan *up* (menghidupkan) server *worker* secara otomatis, sehingga sistem juga akan membutuhkan konsumsi daya tambahan. Jika dihitung selisihnya adalah hanya 0,47 Watt. Hal ini justru berbeda dengan konsumsi daya yang digunakan ketika beban mengalami penurunan seperti yang tertera pada Tabel VII. Ketika beban mengalami penurunan, maka konsumsi daya dapat dihemat lebih besar daripada selisih penggunaan daya pada beban meningkat.

Pada Tabel VII menunjukkan bahwa penghematan dapat dilakukan dengan metode prediksi pada saat beban menurun. Ini merupakan hal yang diharapkan sesuai hipotesa sebelum melakukan pengujian. Konsumsi daya dengan metode prediksi pada saat beban menurun ini adalah sebesar 11,57 Watt sedangkan pada sistem konvensional adalah 13,18 Watt. Ini artinya sistem dengan metode prediksi dapat menghemat konsumsi daya sebesar 1,61 Watt. Dan jika dikurangi dengan konsumsi daya pada saat beban meningkat yang memiliki kenaikan sebesar 0,47 Watt, maka total konsumsi energi yang dapat dihemat dengan metode prediksi ini adalah sebesar 1,14 Watt. Memang angka ini terlihat kecil, namun ini merupakan pencapaian yang sangat bagus mengingat uji coba hanya dilakukan pada lingkungan virtual dan hanya menggunakan laptop. Yang perlu diperhatikan adalah bahwa jika keberhasilan sistem dengan menggunakan metode prediksi ini dapat diaplikasikan pada server data *center cloud computing* yang sesungguhnya. Menurut penulis angka penghematan konsumsi daya yang dapat dioptimasi pada server data *center cloud computing* yang sebenarnya akan jauh lebih besar dan signifikan mengingat pada setiap server *cloud computing* memiliki konsumsi daya yang jauh lebih tinggi daripada konsumsi daya pada laptop sederhana.

Hal lain yang perlu dilihat adalah mengenai total jumlah response *time* yang terjadi pada sistem dengan metode prediksi memiliki nilai yang lebih rendah daripada sistem konvensional, ini artinya bahwa layanan dengan penghematan konsumsi daya yang menggunakan metode prediksi tetap dapat memiliki ketersedian tinggi. Ini dapat dilihat dari angka yang ditunjukkan pada Tabel VI dan Tabel VII. Hanya saja pada pengujian sistem dengan metode prediksi saat beban meningkat memiliki response *time* yang lebih besar daripada nilai *response time* dengan sistem konvensional, namun itu hanya selisih 2,7 ms.

Dari kedua metode pengujian dengan dua pola beban yang diuji coba, maka selisih konsumsi daya yang dapat dihemat dengan mengunakan metode prediksi adalah sebesar 1,14 Watt (25,96-24,82), ini merupakan selisih antara konsumsi daya dengan metode prediksi dan metode konvensional. Nilai rata-rata *response time* dari kedua pola dengan metode prediksi tersebut adalah 81,95. Namun jika dilihat nilai *response time* keseluruhan yaitu pada pola naik dan turun, maka nilai *response time* rata-rata pada sistem yang dibangun ini mengalami sedikit kenaikan yaitu sebesar rata-rata 0,65 ms seperti terlihat pada Tabel VIII.

V. KESIMPULAN

Berdasarkan hasil penelitian, beberapa hal yang dapat disimpulkan bahwa optimasi dayadengan metode prediksi pada sistem ini dapat dilakukan pada migrasi VM dan DNS, dari hasil pengujian menunjukkan bahwa dengan metode prediksi dapat mengurangi konsumsi daya sebesar 1,14 Watt dibandingkan dengan metode konvensional.Konsumsi daya pada pola beban meningkat adalah 13,25 Watt, dan pada pola beban menurun adalah 11,57 Watt.Hasil *forecast* atau prediksi *usage* CPU pada sistem ini dapat dilakukan menggunakan *moving average.*

Adapun saran untuk pengembangan penelitian ini adalah penggunan prediksi atau *forecast* pada *usage* CPU dapat dikembangkan dengan penambahan parameter pada *moving average* dengan mempertimbangkan pola data awal. Selain itu, pengujian dapat dikembangkan dengan beban data yang lebih bervariasi.

DAFTAR PUSTAKA

[1] I. Sofana, *Cloud Computing Teori dan Praktik*. Bandung: Informatika, 2012.

[2] M. Fathurahman, T. Telekomunikasi, J. T. Elektro, and P. N. Jakarta, "Efisiensi Kinerja Pengelolaan Energi pada Arsitektur Data Center Komputasi Awan Menggunakan Greencloud," *J. Ilm. Elit. ELEKTRO*, vol. 3, no. 1, pp. 6–14, 2012.

[3] D. Filani, J. He, S. Gao, M. Rajappa, A. Kumar, P. Shah, and R. Nagappan, "Dynamic Data Center Power Management Trends, Issues, and Solutions," *Intel Technol. J.*, vol. 12, no. 01, pp. 59–67, 2008.

[4] K. C. K. Choi, R. Soma, and M. Pedram, "Dynamic Voltage and Frequency Scaling Based on Workload Decomposition," in *Proceedings of the 2004 International Symposium on Low Power Electronics and Design*, 2004.

[5] D. Kliazovich, P. Bouvry, and S. U. Khan, "GreenCloud : A Packet-level Simulator of Energy- aware Cloud Computing Data Centers," *J. Supercomput.*, vol. 62, no. 3, pp. 1263–1283, 2012.

[6] W. Y. Lee, "Energy-saving DVFS Scheduling of Multiple Periodic Real-time Tasks on Multi-core Processors," in *Proceedings - IEEE International Symposium on Distributed Simulation and Real-Time Applications, DS-RT*, 2009, pp. 216–223.

[7] K. Lakshmanan, S. Kato, and R. (Raj) Rajkumar, "Scheduling Parallel Real-Time Tasks on Multi-core Processors," *2010 31st IEEE Real-Time Syst. Symp.*, pp. 259–268, 2010.

[8] J. Yang, C. Liu, Y. Shang, Z. Mao, and J. Chen, "Workload Predicting-Based Automatic Scaling in Service Clouds," 2013.

[9] N. Roy, A. Dubey, and A. Gokhale, "Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting," *2011 IEEE 4th Int. Conf. Cloud Comput.*, pp. 500–507, Jul. 2011.

[10] E. L. Sueur and G. Heiser, "Dynamic voltage and frequency scaling: the laws of diminishing returns," in *Proceedings of the 2010 international conference on Power aware computing and systems*, 2010, pp. 1–8.

[11] R. A. Yaffee and M. McGee, *Introduction to Time Series Analysis and Forecasting: With Applications of SAS and SPSS*. New York: Academic Press, 2000.

[12] I. Zulkarnain, "Akurasi Grafik Main Chart Dalam Prediksi Harga Saham Harian : Kasus The Winnest Dan The Losest," *J. Ilm. STIE MDP*, vol. 1, no. 2, pp. 74–83, 2012.

[13] T. Lorido-Botran, J. Miguel-Alonso, and J. A. Lozano, "Auto-scaling Techniques for Elastic Applications in Cloud Environments," 2012.

[14] P. Wahyu W, T. B. Harsono, and N. Annggis S, "Analisis Performansi Sistem Dari Implementasi Auto-Scaling pada Cloud Computing Dengan Menggunakan Predictive System Simple Moving Average," 2014, pp. 1–7.

[15] M. Ikram, Q. U. A. Babar, Z. Anwar, and A. W. Malik, "GSAN: Green cloud-simulation for storage area networks," *Proc. - 11th Int. Conf. Front. Inf. Technol. FIT 2013*, pp. 265–270, 2013.

[16] "http://www.vmware.com/id.html," 2016. [Online]. Available: http://www.vmware.com/id.html. [Accessed: 20-Jul-2016].

[17] "https://www.proxmox.com/," 2016. [Online]. Available: https://www.proxmox.com/. [Accessed: 20-Jul-2016].

[18] "http://www.nas4free.org/," 2016. [Online]. Available: http://www.nas4free.org/. [Accessed: 20-Jul-2016].

[19] "https://github.com/httperf/httperf," 2016. [Online]. Available: https://github.com/httperf/httperf. [Accessed: 20-Jul-2016].