# TUGAS AKHIR I

**ANALISIS PERFORMA *HOST OVERLOAD DETECTION* DI DALAM *DYNAMIC VM CONSOLIDATION***

**DENGAN WORKLOAD TPC-DS**

***PERFORMANCE ANALYSIS OF HOST OVERLOAD DETECTION IN DYNAMIC VM CONSOLIDATION***

***WITH TPC-DS WORKLOAD***

Diajukan untuk memenuhi salah satu syarat mengerjakan dan menempuh ujian

Tugas Akhir 2



Disusun oleh:

Nama : Wildanil Ghozi

NIM : A11.2011.05885

Program Studi : Teknik Informatika

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS DIAN NUSWANTORO**

**SEMARANG**

**2015**

# HALAMAN PERSETUJUAN

**TUGAS AKHIR I**

ANALISIS PERFORMA *HOST OVERLOAD DETECTION*

DI DALAM *DYNAMIC VM CONSOLIDATION*

DENGAN WORKLOAD TPC-DS

Dibuat oleh

Wildanil Ghozi

A11.2011.05885

Program Studi Teknik Informatika Strata 1

Fakultas Ilmu Komputer

Telah disetujui dan di uji pada tanggal 28 Juli 2015

Pembimbing

Guruh Fajar Shidik, M.Cs

# HALAMAN PENGESAHAN

**TUGAS AKHIR I**

ANALISIS PERFORMA *HOST OVERLOAD DETECTION*

DI DALAM *DYNAMIC VM CONSOLIDATION*

DENGAN WORKLOAD TPC-DS

Dibuat oleh

Wildanil Ghozi

A11.2011.05885

Program Studi Teknik Informatika Strata 1

Fakultas Ilmu Komputer

Telah dipertahankan di depan Tim Penguji pada tanggal 28 Juli 2015

Mengesahkan

|  |  |
| --- | --- |
| Pembimbing  Guruh Fajar Shidik, M.Cs | Penguji  Aisyatul Karima, S.Kom, MCS |

# RINGKASAN

*Cloud Computing* berperan dalam layanan komunikasi antar komputer yang dikombinasikan dengan internet berbasis server *virtual*. *Cloud computing* mengacu pada aplikasi, perangkat lunak dan perangkat keras yang disampaikan sebagai layanan melalui internet. *Cloud computing* mempermudah pengguna untuk saling berbagi sumber daya perangkat keras dan perangkat lunak.

Sebuah metode digunakan untuk meningkatkan pemanfaatan *resource* dan mengurangi konsumsi energi listrik di dalam *cloud computing* yaitu *Dynamic VM Consolidation. Dynamic VM consolidation* manfaatkan metode *live virtual machine (VM) migration*, yaitu teknik memindahkan seluruh sistem operasi dan aplikasi yang sedang berjalan dari mesin fisik satu ke mesin fisik lain untuk memaksimalkan penggunaan *resource* *host* fisik sehingga memungkinkan host lain dalam kondisi *idle* (tidak bekerja). *Host* dalam kondisi *idle* dinonaktifkan untuk mengurangi penggunaan energi.

Ada empat bagian pembahasan dari *dynamic VM consolidation* yaitu *host overload detection* untuk menentukan sebuah *host* dalam kondisi *overload*, *host underload detection* untuk menentukan sebuah *host* dalam kondisi *underload*, *vm selection* untuk menentukan *VM* yang hendak dipindahkan, dan *VM placement* untuk menentukan *host* tujuan *VM* dipindahkan.

Pada tugas akhir ini penulis bermaksud merancang dan membangun *server cloud computing* yang mampu menerapkan *dynamic VM consolidation* serta menganalisa performa algoritma *host overload detection* di dalamnya. Pada penelitian ini penulis menggunakan CentOS7 sebagai sistem operasi, OpenStack sebagai *cloud operating system* dan OpenStack Neat sebagai *framework dynamic VM consolidation*. Penulis menguji tiga algoritma *host overload detection* yaitu *static CPU utilization threshold* (THR), *local regression* (LR) dan *markov host overload detection* (MHOD). Workload yang berjalan pada *dynamic VM consolidation* dibangkitkan dari TPC-DS.

Performa *dynamic VM consolidation* dapat dievaluasi dengan menghitung gabungan waktu *idle* semua *host* atau disebut dengan *aggregated idle time fraction* (AITF) dan menghitung gabungan waktu *overload* semua *host* atau *aggregated over load time fraction* (AOTF). Semakin kecil nilai AITF dan AOTF berarti *dynamic VM consolidation* semakin baik.

Kesimpulan diambil dengan membandingkan nilai AOTF dan AITF dari *dynamic VM consolidation* yang menggunakan ketiga algoritma *host overload detection*. Penulis mengharapkan akan mampu menyimpulkan dan menentukan algoritma terbaik dari ketiga algoritma *host overload detection* yang diujikan.

# DAFTAR ISI

[TUGAS AKHIR I i](#_Toc426554821)

[HALAMAN PERSETUJUAN ii](#_Toc426554822)

[HALAMAN PENGESAHAN iii](#_Toc426554823)

[RINGKASAN iv](#_Toc426554824)

[DAFTAR ISI v](#_Toc426554825)

[DAFTAR GAMBAR vii](#_Toc426554826)

[DAFTAR TABEL viii](#_Toc426554827)

[BAB I PENDAHULUAN 1](#_Toc426554828)

[1.1. Latar Belakang Masalah 1](#_Toc426554829)

[1.2. Rumusan Masalah 4](#_Toc426554830)

[1.3. Batasan Masalah 5](#_Toc426554831)

[1.4. Tujuan Penelitian 6](#_Toc426554832)

[1.5. Manfaat Penelitian 6](#_Toc426554833)

[BAB II TINJAUAN PUSTAKA 8](#_Toc426554834)

[2.1. Tinjauan Studi 8](#_Toc426554835)

[2.2. Tinjauan Pustaka 10](#_Toc426554836)

[2.3. Kerangka Pemikiran 21](#_Toc426554837)

[BAB III METODE PENELITIAN 22](#_Toc426554838)

[3.1. Instrumen Penelitian 22](#_Toc426554839)

[3.2. Prosedur Pengambilan Data 25](#_Toc426554840)

[3.3. Metode yang Diusulkan 25](#_Toc426554841)

[3.4. Pengujian dan Analisis 28](#_Toc426554842)

[Jadwal Penyusunan Tugas Akhir 29](#_Toc426554843)

[DAFTAR PUSTAKA 30](#_Toc426554844)

# DAFTAR GAMBAR

[Gambar 2.3 Kerangka Pemikiran 21](#_Toc426554930)

[Gambar 3.1 Model Penelitian 25](#_Toc426554931)

[Gambar 3.2 Topologi Jaringan 27](#_Toc426554932)

[Gambar 3.3 Skenario Pengujian 28](#_Toc426554933)

# DAFTAR TABEL

[Tabel 2.1 Daftar Penelitian 9](#_Toc425638337)

[Tabel 3.1 Jadwal Penyusunan Tugas Akhir 29](#_Toc425638338)

# BAB I PENDAHULUAN

## Latar Belakang Masalah

Teknologi informasi dan komunikasi meruapakan teknologi yang digunakan untuk mengelola data sehingga menghasilkan informasi yang memiliki manfaat. Komputer merupakan alat yang digunakan oleh manusia dalam mengelola data, sedangkan proses pengelolaannya disebut dengan komputasi. Dalam teknologi informasi dan komunikasi dikenal istilah jaringan komputer yang berarti dua komputer atau lebih yang saling terhubung untuk saling berbagi informasi. Teknologi ini memungkinkan adanya sistem komputasi terpusat dimana satu atau beberapa komputer menyediakan lyanan tertentu untuk komputer lain, disebut dengan client-*server*. Setiap server biasanya menyediakan layanan khusus seperti web server, database server, file server dan sebagaianya yang bisa diakses dan digunakan oleh *client*.

*Workload* meruapakan beban kerja yang dikerjakan oleh sebuah komputer dalam satu waktu. Karakteristik *workload* pada setiap komputer berbeda-beda berdasarkan layanan dan analisa data. Biasanya analisa data meliputi *business intelligence*, *machine learning*, *bio-informatics*, dan *ad hoc analysis*[1]. Berbagai macam karakteristik *workload* bisa dibangkitkan dengan menggunakan *benchmarks tools*.

Seiring dengan perkembangan teknologi, kebutuhan manusia akan informasi juga terus meningkat sehingga mengakibatkan meningkatnya komputasi yang dilakukan oleh manusia. Hal tersebut diikuti dengan meningkatnya *resource* yang dibutuhkan untuk komputasi. Untuk memaksimalkan dan memudahkan pengelolaan *resource* yang ada dibutuhkan teknologi *virtual machine*. Teknologi ini memungkinkan untuk menempatkan beberapa server ke dalam satu *host* fisik sebagai *virtual machine*[2]. Konsep *virtual machine* pertama kali dikembangkan oleh IBM untuk menyediakan akses interaktif bersamaan ke komputer mainframe[3].

Baru-baru ini muncul paradigma *cloud computing* dengan memeanfaatkan teknologi virtualisasi dan menyediakan *on-demand resource* melalui internet dengan sistem sewa[2]. *Cloud computing* telah menjadi isu populer dengan kemajuan saat ini di teknologi jaringan[4]. *Cloud computing* memanfaatkan teknologi virtualisasi dalam konsep kerjanya. Dalam *cloud computing*, sebuah sistem terdistribusi mengelola sekumpulan *resource* dan menyediakannya sebagai layanan sesuai permintaan sebagai *resource* komputasi untuk diremote melalui jaringan. *Resource* ini kemungkinan terdistribusi dan beraneka ragam[5].

Dalam mengelola *resource cloud computing* yang besar, dibutuhkan energi listrik yang besar sehingga mengakibatkan biaya operasional tingi dan melepaskan emisi karbon dioksida (CO2) ke lingkungan[6]. Diperkirakan pada tahun 2008 konsumsi energi untuk *cloud data center* di seluruh dunia dari tahun 2005 sampai 2010 meningkat sebesar 56% dan akan tercat antara 1,1% hingga 1,5% dari penggunaan energi listrik global pada tahun 2010 dan jumlahnya akan terus meningkat setiap tahun[7].

Tingginya konsumsi energi pada *cloud computing* telah menjadi perhatian dalam beberapa tahun terahir. Metode yang digunakan untuk meningkatkan pemanfaatan *resource* dan mengurangi konsumsi energi listrik yaitu *dynamic VM consolidation,* memanfaatkan metode *live virtual machine (VM) migration* yang mampu mengirim virtual machine antar *physical server* dengan *downtime* mendekati nol[8]. *Live VM* *migration* adalah teknik migrasi atau memindahkan seluruh sistem operasi dan aplikasi yang sedang berjalan dari mesin fisik satu ke mesin fisik lain untuk menciptakan efisiensi energi, ketersediaan tinggi dari *load balancing* dari *physical server* di tengah *cloud data. Live VM migration* berjalan tanpa mengganggu aplikasi yang berjalan di atasnya[9]. Dengan *live VM migration,* di sisi lain *host* asli (mesin fisik) dapat dinonaktifkan ketika migrasi selesai [10].

*Dynamic VM consolidation* mampu memaksimalkan penggunaan *resource* dengan memigrasikan *virtual machine* dari *host* yang mengalami kondisi *overload* ke *host* lain untuk menjaga *quality of service* atau dari *host* yang mengalami *underload* untuk mengurangi *host* *busy*. *Host* dalam kondisi *idle* (tidak bekerja) dinonaktifkan untuk mengurangi penggunaan energi. Performa *dynamic VM consolidation* dapat dievaluasi dengan menghitung gabungan waktu *idle* semua *host* atau disebut dengan *aggregated idle time fraction* (AITF) dan menghitung gabungan waktu *overload* semua *host* atau *aggregated over load time fraction* (AOTF). Semakin kecil nilai AITF dan AOTF berarti *dynamic VM consolidation* semakin baik.

A. Beloglazov dan R. Buyya (2014) mengatakan bahwa ada empat permasalahan pada *dynamic VM consolidation* yaitu *host overload detection* untuk menentukan apakah sebuah *host* mengalami kondisi *overload* atau tidak, *host underload detection* untuk menentukan apakah sebuah *host* mengalami kondisi *underload* atau tidak, *VM selection* untuk menentukan VM yang akan dimigrasikan dari *host* yang mengalami *underload* atau *overload,* *VM placement* menentukan *host* tujuan untuk menempatkan VM yang sebelumnya telah dipilih [11].

Pada *dynamic VM consolidation* sangat penting untuk menentukan *host* berada dalam kondisi *overload* atau tidak. Karena jika kapasitas *resource* benar-benar dimanfaatkan, sangat mungkin bahwa aplikasi yang berjalan akan kehabisan *resource* dan terjadi penurunan performa[12]. Terdapat beberapa algoritma *host overload detection* yaitu *static CPU utilization threshold* (THR), *local regression* (LR) dan *Markov overload detection* (MHOD). Performa algoritma *host overload detection* telah dievaluasi dengan diimplementasikan pada sebuah *cloud data center* yang mendukung *dynamic VM consolidation.* Pada evaluasi tersebutdiberikan *workload* dengan karakteristik heterogen yang diambil dari sebuah aplikasi *workload trace*[8]. Namun saat ini belum ditemukan publikasi penelitian yang mengevaluasi performa algoritma *host overload detection* dengan *workload* pendukung keputusan.

Sistem pendukung keputusan digunakan oleh perusahaan untuk membantu dalam menentukan kebijakan-kebijakan perusahaan. Karakteristik *workload* pendukung keputusan atau *dessicion support* yaitu penggunaan *resource* *processor* tinggi, *memmory* rendah dan *input/output* tinggi. *Workload* *dessicion support* bisa dibangkitkan dengan menggunakan benchmark tools TPC-DS. TCP-DS merupakan dessicion support benchmark saat ini dikembangkan oleh *Transaction Processing Performance Council*. merupakan model dari sistem pendukung keputusan yang difungsikan untuk ritail dan *supplier* produk. Mendukung skema yang berisi informasi bisnis vital seperti pelanggan, pesanan dan data produk.

Dari penelitian di atas, peneliti termotifasi untuk melakukan analisis performa algoritma *host overload detection* pada *dynamic VM consolidation* dengan *workload* TPC-DS. Analisis tersebut dilakukan secara langsung pada *server* *local* *cloud computing*. Hasil yang ingin dicapai adalah mengetahui performa *aggregated idle time fraction* (AITF) dan *aggregated overload time fraction* (AOTF) dari masing-masing algoritma *host overload detection* serta membandingkannya untuk mengetahui algoritma terbaik. Oleh karena itu penulis mengambil judul “Analisis Performa *Host Overload Detection* di dalam *Dynamic VM Consolidation* dengan *Workload* TPC-DS”.

## Rumusan Masalah

Berdasarkan uraian diatas, rumusan masalah dalam penelitian ini adalah di dalam *dynamic VM consolidation* terdapat metode *overload detection*. Menurut Anton Beloglazov setidaknya terdapat tiga metode *overload detection* yaitu *static CPU utilization threshold* (THR), *local regression* (LR), *Markov host overload detection* (MHOD). Akan tetapi belum diketahui performa (AOTF dan AITF) ketiga metode tersebut ketika diberikan *workload* TPC-DS. Sehingga masalah dalam penelitian ini adalah :

1. Bagaimana membangun infrastruktur *cloud computing* yang mampu menerapkan *dynamic VM consolidation*?
2. Bagaimana performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *static CPU utilization threshold* (THR)?
3. Bagaimana performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *local regression* (LR)?
4. Bagaimana performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *Markov host overload detection* (MHOD)?
5. Manakah metode *overload detection* yang menghasilkan performa terbaik?

## Batasan Masalah

Pada penelitian ini, diperlukan batasan permasalahan yang dikerjakan agar fokus terhadap bidang yang akan diteliti. Batasan yang digunakan dalam penelitian ini adalah sebagai berikut:

1. *Server cloud* menggunakan server fisik dan *virtual*.
2. *Sistem* operasi yang digunakan pada server adalah sistem operasi *CentOS7* dan menggunakan OpenStack sebagai *Cloud operating system* serta OpenStack neat sebagai *Framework dynamic VM consolidation.*
3. *Analisis* yang di bahas hanya mencakup performa algoritma *overload detection* di dalam *dynamic VM consolidation*.
4. *Workload* yang digunakan adalah *dessicion support* yang dibangkitkan dengan menggunakan TPC-DS.
5. Metode *overload detection* yang digunakan yaitu *static CPU utilization threshold* (THR), *local regression* (LR) dan *markov host overload detection* (MHOD).
6. Metode *underload detection* yang digunakan yaitu *static CPU utilization threshold* (THR).
7. Metode *VM selection* yang digunakan yaitu *minimum migration time* (MMT).
8. Metode *VM placement* yang digunakan yaitu *power aware bestfit decreasing* (PABD).
9. *Parameter* yang digunakan peneliti yaitu *aggregated idle time fraction* (AITF), *aggregated overload time fraction* (AOTF).

## Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini ialah :

1. Membangun infrastruktur *cloud computing* yang mampu menerapkan *dynamic VM consolidation*.
2. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *static CPU utilization threshold* (THR).
3. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *local regression* (LR).
4. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *markov host overload detection* (MHOD).
5. Menentukan metode *overload detection* yang menghasilkan performa terbaik.

## Manfaat Penelitian

Adapun manfaat yang diharapkan dapat diperoleh dari penelitian ini adalah :

1. Bagi Akademik
   1. Menambah literatur kepustakaan di perpustakaan Universitas Dian Nuswantoro yang dapat digunakan sebagai gambaran atau petunjuk dalam membuat Laporan Tugas Akhir.
   2. Sebagai tolak ukur kemampuan mahasiswa serta sarana untuk mengukur sejauh mana pemahaman dan penguasaan materi kuliah yang diberikan dalam menuntut ilmu di Universitas Dian Nuswantoro.
   3. Dapat dijadikan bahan informasi dan bahan referensi serta kerangka acuan bagi pihak yang membutuhkan.
2. Bagi Masyarakat
3. Memperkenalkan metode mengoptimalkan penggunaan *resource* dan mengurangi konsumsi energi pada *cloud data center*.
4. Memperkenalkan penggunaan *Openstack* sebagai *cloud operating system.*
5. Memperkenalkan infrastruktur *cloud computing* yang mampu menerapkan *dynamic VM consolidation.*
6. Bagi Peneliti
7. Untuk menambah pengetahuan penulis dibidang dunia komputer khususnya mengenai bagaimana menganalisa perfoma *dynamic VM consolidation* dengan beban kerja pendukung keputusan.
8. Sebagai bahan acuan atau penelitian pendahuluan untuk penelitian selanjutnya.
9. Dengan menyusun Laporan Tugas Akhir ini, penulis dapat mengembangkan ilmu tentang jaringan komputer yang telah diberikan pada masa kuliah.

# BAB II TINJAUAN PUSTAKA

## Tinjauan Studi

Peneilitan yang berhubungan dengan topik yang penulis angkat salah satunya yang berjudul “*Energy Efficient Allocation of Virtual Machines in Cloud Data Centers*” oleh Anton Beloglazov dan Rajkumar Buyya (2010). Penelitian tersebut membahas tentang managemen alokasi *resource* untuk virtualisasi di *cloud data center*, berfokus pada pengujian algoritma *virtual machine selection* menggunakan CloudSim toolkit. Pengujian dilakukan dengan menghitung konsumsi energi dari algoritma alokasi *resource* statis yaitu *Non Power Aware* (NPA) dan DVFS kemudian dibandingkan dengan Algoritma alokasi *resource* dinamis yaitu Single Tracehold (ST) dan Minimization of Migration (MM). Hasil dari pengujian tersebut algoritma MM menjadi yang terbaik dalam menghemat energi.

Penelitian lain yang dilakukan oleh Anton Beloglazov dan Rajkumar Buyya berjudul “*OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds*” tahun 2014. Penelitian ini memperkenalkan arsitektur dan implementasi OpenStack Neat yang merupakan framework *dynamic VM consolidation* yang berjalan pada OpenStack. OpenStack Neat menggunakan metode *live VM migration* untuk memindahkan *VM* dari *host* yang *overload* untuk menjaga performa serta dari *host* yang mengalami *underload* untuk mengurangi penggunaan *resource*. Ada empat algoritma dasar dari OpenStack Neat untuk dapat mendukung *dynamic resource allocation* yaitu *host underload detection, host overload detection, VM selection dan VM placement*. Hasil dari eksperimen dan perkiraan konsumsi energi menunjukan bahwa OpenStack Neat mampu mengurangi konsumsi energi antara 25-33%.

Penelitian yang dilakukan oleh Leelipushpam dan P Getzi Jeba berjudul Live *“Live VM Migration Techniques in Cloud Environment – A Survey”* menyatakan bahwa *live VM* *migration* adalah teknik migrasi atau memindahkan seluruh sistem operasi dan aplikasi yang sedang berjalan dari mesin fisik satu ke mesin fisik lain untuk menciptakan efisiensi energi, ketersediaan tinggi dari *load balancing* dari server fisik di tengah *Cloud Data. Live VM Migration* berjalan tanpa mengganggu aplikasi yang berjalan di atasnya.

Tabel 2.1 Daftar Penelitian

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Peneliti** | **Judul** | **Tool** | **Hasil** |
| 1. | A Beloglazov,  R Buyya  (2010) | Energy Efficient Allocation of Virtual Machines in Cloud Data Centers | CloudSim Toolkit | Realokasi VMS dinamis sesuai dengan penggunaan CPU dapat menghemat energi yang lebih tinggi dibandingkan dengan alokasi statis. |
| 2. | Nstallation, I  Erformance, P  Tudy, S  (2011) | Virtual Machines and Networks Advantages and Virtualization Options | VMware Workstation, Netperf Software | Virtualisasi dapat menciptakan lingkungan bisnis dunia nyata menjadi sedekat mungkin dengan lingkungan akademik, sehingga siswa dapat berinteraksi dengan teknologi seperti mereka berada di lingkungan kerja. |
| 3. | Jhu-jyun Jheng,  Fan-hsun Tseng,  Han-chieh Chao,  Li-der Chou  (2014) | A Novel VM Workload Prediction using Grey Forecasting Model in Cloud Data Center | Gray  Forecasting | Pertama memprediksi beban kerja di setiap mesin fisik dengan menggunakan Gray Forecasting. Hasil simulasi menunjukan bahwa hasil simulasi tersebut mampu menonaktifkan lebih banyak host fisik dan mengurangi konsumsi energi. |
| 4. | Koomey, Jonathan G  (2008) | Worldwide electricity used in data centers | IDC dan UEPA | Diperkirakan pada tahun 2008 konsumsi energi untuk *cloud data center* di seluruh dunia dari tahun 2005 sampai 2010 meningkat sebesar 56% dan akan tercat antara 1,1% hingga 1,5% dari penggunaan energi listrik global pada tahun 2010 dan jumlahnya akan terus meningkat setiap tahun |
| 5. | A Beloglazov,  R Buyya  (2010) | OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds | OpenStack Neat | Hasil dari eksperimen dan perkiraan konsumsi energi menunjukan bahwa OpenStack Neat mampu mengurangi konsumsi energi antara 25-33%. |
| 6. | Leelipushpam,  P Getzi Jeba  (2013) | Live VM Migration Techniques in Cloud Environment – A Survey | Teknik pengendalian migrasi | Studi komparatif dibuat antara teknik efisiensi energi yang merupakan teknik penghematan terbaik. Studi komparatif diadakan mengenai teknik parameter load balancing migration. Banyak teknik fault tolerant migration yang mencegah gangguan dan menjangkau ketersediaan server fisik. Keamanan adalah perhatian utama saat proses migrasi. |

## Tinjauan Pustaka

### Jaringan Komputer

Jaringan komputer merupakan sekumpulan komputer dan perangkat keras lain yang saling terhubung dengan menggunakan media tertentu untuk saling berbagi data, informasi dan penggunaan perangkat keras bersama seperti printer dan kamera.

Jaringan komputer yang menghubungkan beberapa *hardware* dan *software* untuk berkomunikasi dalam satu area yang terbatas disebut dengan *Local Area Network (LAN)*. Sedangkan untuk jaringan yang lebih luas lagi disebut dengan *Wide Area Network (WAN)*. Jaringan komputer juga menghubungkan seluruh komputer di dunia, yang disebut dengan jaringan internet.

### Virtual Machine

Menurut Ali (2011)[3] konsep mesin *virtual* pertama kali dikembangkan oleh IBM pada tahun 1960 untuk menyediakan akses interaktif ke komputer *mainframe* secara bersama. Setiap mesin *virtual* merupakan replika mesin fisik dan pengguna diberikan sebuah ilusi pada mesin fisik secara langsung. Mesin virtual juga memberikan manfaat seperti isolasi, berbagi sumber daya, dan kemampuan untuk menjalankan beberapa *flavors* dan konfigurasi dari sistem operasi dengan *setting* yang berbeda. *Virtual machine (VM*) mengemulasikan keseluruhan sistem perangkat keras, mulai dari prosessor sampai *network card*, memungkinkan sistem operasi yang berbeda untuk berjalan secara simultan.

Setiap mesin fisik mampu menjalankan satu atau lebih *virtual machine* dengan memberikan sebagian resource yang dimiliknya untuk dijadikan resource *virtual machine*. Keuntungan utama dari *virtual machine* adalah beberapa sistem operasi dapat berjalan bersamaan dalam satu komputer, mampu menyediakan *Instruction Set Architecture (ISA)* meskipun berbeda dengan mesin sebenarnya, application provisioning, maintenance, high availability and disaster recovery[13].

### Cloud Computing

*Cloud Computing* berperan dalam layanan komunikasi antar komputer yang dikombinasikan dengan internet berbasis server *virtual*. Definisi lain menurut buku *A View of Cloud Computing*[14]*,* *cloud computing* mengacu pada sebuah aplikasi yang disampaikan sebagai layanan melalui internet berupa perangkat keras dan perangkat lunak sistem di *data center* yang menyediakan layanan tersebut. *Cloud computing* mempermudah pengguna untuk saling berbagi sumber daya perangkat keras dan perangkat lunak.

Menurut *NIST (National Institute of Standards and Technology)* ada tiga karakteristik *cloud computing* yaitu :

1. *On-demand self-service*.

Pengguna cloud dapat mengkonfigurasi *cloud* yang dimilikinya tanpa harus berinteraksi dengan penyedia layanan *cloud*. Hal ini disebut dengan layan mandiri.

1. *Broad network access*

Layanan *cloud* harus bisa diakses dari berbagai perangkat yang didukung dengan internet seperti smart phone, tablet PC, komputer dan sebagainya.

1. *Resource pooling*

*Resource* dikumpulkan dalam satu sistem untuk disediakan sebagai layanan yang kemudian dikonfigurasi secara mandiri oleh pengguna dalam bentuk *virtual machine*. *Resource* yang disediakan termasuk *processor*, *memory*, *storage* dan *bandwidth.*

1. Rapid elasticity

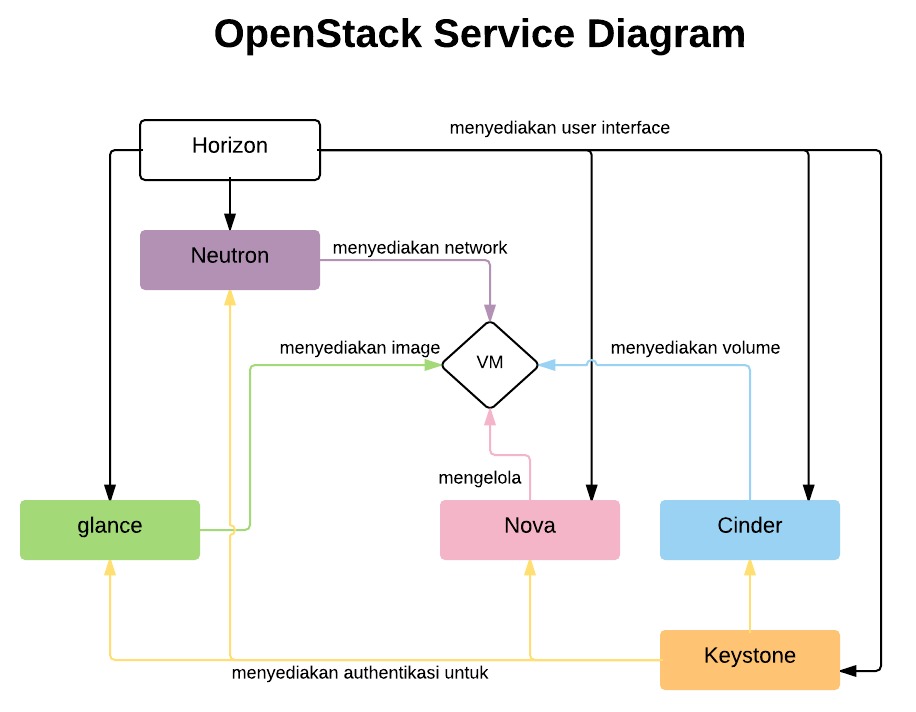
*Resource* yand disediakan dalam *cloud* dapat digunakan sesuai dengan kebutuhan. Pengguna bisa menggunakan sesuai dengan kebutuhan dan keinginannya seperti menggunakan seluruh *resource* dalam satu *virtual machine* atau membagi *resource* dalam beberapa *virtual machine*.

1. Measure Service

Menyediakan layanan untuk mengoptimalkan dan memonitor resource layanan yang telah digunakan. Layanan ini dapat melihat *resource* komputasi yang telah digunakan dan yang masih tersedia.

### OpenStack

*OpenStack* adalah sebuah *software open source* dalam *cloud* *computing* yang berorientasi dibidang *Infrastruktur as Service (Iass*) (Iftakhul Anwar: 2011). *Openstack* bertindak sebagai *middleware* yang mengkomunikasikan beberapa layanan untuk mengelola berbagai macam *resource* dan membentuk sebuah sistem komputasi besar. Administrasi dan monitor *resource* bisa dilakukan dari *dashboard* yang bisa diakses melalui antarmuka web. Beberapa layanan pada OpenStack adalah :



1. Horizon

*Horizon* merupakan layanan *Dashboard* pada OpenStack yang digunakan untuk mengelola layanan *cloud computing* seperti mengunggah *image*, memebuat *instance*, mengatur hak akses dan sebagainya. Layanan ini berbasis web dan bisa diakses melalui browser.

1. Nova

*Nova* atau *compute service* merupakan layanan yang digunakan untuk mengelola komputasi. Pada dasarnya layanan ini mengontrol *virtual machine* seperti *spawning, starting, scheduling, shutdown* dan *terminate*.

1. Neutron

*Neutron* menyediakan konesi jaringan sebagai layanan untuk layanan OpenStack lainnya seperti *compute service*. *Modular layer plug-in* memungkinkan konektifitas antar *virtual machine* dan juga *virtual router*.

1. Swift

*Swift* atau *object storage* merupakan media penyimpanan data berbentuk objek tidak terstruktur dengan API berbasis HTTP. Penyimpanan ini sangat toleran terhadap kesalahan karena mendukung replikasi data dan arsitektur yang besar. Implementasinya tidak seperti file server.

1. Cinder

*Cinder* menyediakan *block storage* sebagai penyimpanan dan tempat berjalannya *virtual machine*. Pada dasarnya setiap *volume* yang dibuat oleh *cinder* merupakan sebuah *logical volume* yang dibuat oleh *Logical Volume Manager*.

1. Keystone

Menyediakan autentikasi dan autorisasi bagi setiap layanan OpenStack. Mengelola *user* dan *tenant* untuk diakses oleh pengguna yang berhak.

1. Glance

Menyimpan dan mengelola *disk image* yang digunakan oleh virtual machine sebagai sumber sistem operasinya.

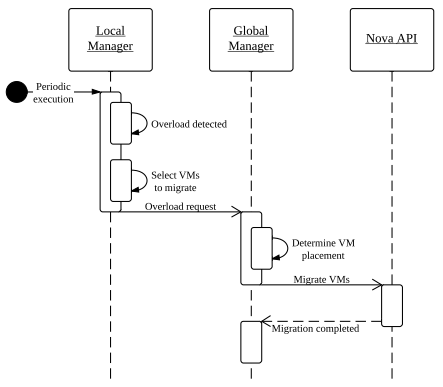
### OpenStack Neat

*OpenStack Neat* didesain dan diimplementasikan untuk mendukung *dynamic VM consolidation* pada *OpenStack*. Tujuannya untuk meningkatkan penggunaan *resource* dan mengurangi konsumsi energi listrik. OpenStack Neat terdiri dari beberapa komponen yaitu :

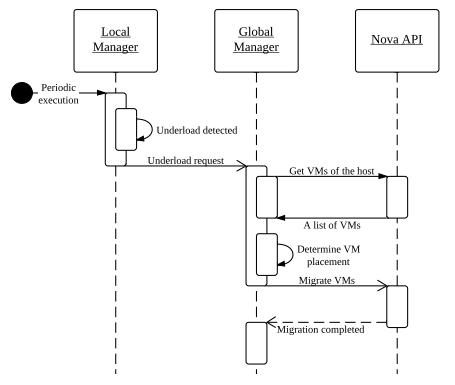
1. Global Manager

Global manager dipasang di controller host untuk memberikan respon berdasarkan *request* yang dikirimkan oleh local manager. Global manager memberikan keputusan VM placement dan menginisiasi live VM migration ke nova API. Global manager memroses dua jenis *request* dari local manager yaitu overload dan underload *request.*

Overload request diterima beserta id VM yang akan dipindahkan, kemudian global manager akan membuat keputusan menentukan compute host tujuan dengan menggunakan algoritma VM placement. Jika status host tujuan telah suspend maka akan dihidupkan melalui NIC. Tahap terahir adalah inisiasi live VM migration dikirimkan ke nova API. Diagram alur global manager dalam merespon overload request digambarkan dalam gambar berikut.



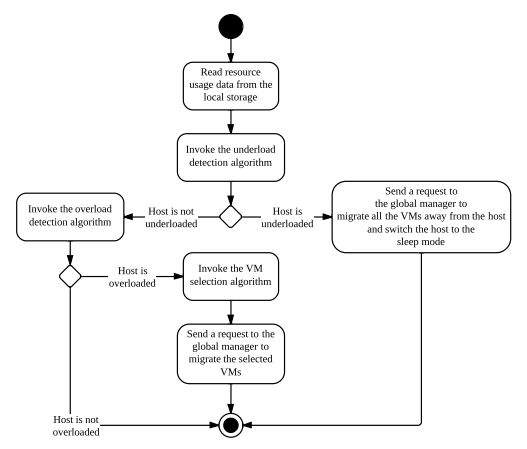
Underload request diterima beserta *hostname* compute *host* yang mengalami underload. Global manager akan menentukan *host* tujuan setiap VM yang ada di dalam *host* yang mengalami underload tersebut kemudian menginisiasi live VM migration. Setelah semua VM dipindahkan maka status compute *host* menjadi *idle* dan statusnya diubah ke suspend. Alur global manager dalam merespon underload request digambarkan pada diagram berikut.



1. Local Manager

Local manager dipasang di setiap compute host untuk membuat keputusan lokal masing-masing compute host. Local manager secara periodik menjalankan fungsi membaca data yang telah disimpan oleh data collector. Kemudian dengan menggunakan algoritma host underload detection ditentukan keptusan apakah host berada dalam kondisi underload atau tidak. Ketika sebuah host dinyatakan dalam kondisi underload, maka local manager akan mengirimkan underload request ke global manager.

Jika host dinyatakan tidak dalam kondisi underload maka ditentukan keputusan apakah host berada dalam kondisi overload. Ketika sebuah host dinyatakan dalam kondisi overload, maka local manager akan menentukan VM yang akan dipindahkan. VM dipilih berdasarkan algoritma VM selection yang digunakan pada konfigurasi local manager. Tahap akhir adalah mengirimkan overload request ke global manager untuk memindahkan VM yang telah dipilih. Aktifitas local manager digambarkan dengan activity diagram berikut.



Activity diagram local manager

1. Data Collector

Data collector dipasang di setiap compute host. Data collector secara periodik menjalankan fungsi mengumpulkan data utilisasi CPU dari compute host dan VM yang berjalan di atasnya. Data yang dikumpulkan disimpan di dalam file lokal dalam compute host dan juga disimpan di database server sehingga bisa diakses secara global.

*OpenStack Neat* menyediakan *tools* untuk mengukur hasil kerja dan evaluasi dengan menghitung *Agregated Overload Time Fraction (AOTF)* dan *Agregated UnderLoad Time Fraction (AITF).* *AOTF* merupakan gabungan waktu *overload* dari semua *host* aktif yang memiliki beban kerja atau *busy*. *AOTF* dirumuskan sebagai berikut :

Dimana merupakan waktu *overload* yang dikalkulasi dari host dihitung yang melewati ambang batas *overload* dan merupakan total semua host *bussy* (tidak dalam keadaan *idle*).

AITF merupakan gabungan waktu idle dari semua host yang aktif. AITF dirumuskan sebagai berikut :

Dimana merupakan waktu *underload* yang dikalkulasi dari host dan merupakan total semua host aktif.

### Algoritma *VM Consolidation*

Dalam sebuah journal ilmiah yang berjudul “Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers” membagi masalah dynamic *VM consolidation* menjadi empat bagian yaitu :

1. *Host overload detection*

Sebuah *host* fisik ditentukan berada dalam kondisi *overload* atau tidak dengan menggunakan algoritma *host overload detection*. Ada beberapa algoritma yang bisa digunakan yaitu :

1. *Static CPU Utilization Threshold (THR)*

Pendeteksian overload yang paling mudah adalah dengan menentukan batasan penggunaan CPU antara kondisi overload dan tidak overload. Jika Utilisasi CPU melewati batasan maka ditetapkan keputusan status overload..

1. *Local regression* (LR)

Algoritma ini berbasis penghitungan estimasi parameter LOESS dan menggunakannya untuk memprediksi utilitas CPU pada waktu berikutnya. Dalam algoritma LR juga didefinisikan batasan antara overload dan tidak overload. Sehingga Jika utilisasi CPU pada waktu berikutnya lebih besar dari batasan maka akan ditetapkan status overload.

1. *Markov overload detection* (MHOD)

Algoritma MHOD menggunakan markov chain model sebagai metode untuk memprediksi utilisasi CPU dalam beberapa waktu yang akan datang berdasarkan history utilisasi CPU. Algoritma ini menerima parameter masukan ambang batas overload time fraction (OTF). Apabila diprediksi OTF akan melewati batas maka ditetapkan status overload.

1. *Host underload detection*

Sebuah *host* fisik ditentukan berada dalam kondisi *underload* atau tidak dengan menggunakan algoritma *host underload detection*. Saat ini hanya ada satu algoritma yang telah diimplementasikan yaitu *Static CPU utilization threshold* (THR).

1. *VM selection*

Ketika sebuah *host* diputuskan mengalami kondisi *overload* maka harus ada *VM* yang dipindahkan dari host tersebut. Dibutuhkan sebuah algoritma yang menentukan *VM* yang akan dipindahkan. Salah satu algoritma yang digunakan adalah *minimum migration time (MMT)*. Algoritma ini memilih *VM* dengan penggunaan *RAM* paling kecil untuk meminimalkan *live migration time*.

1. *VM placement*

*VM placement* merupakan algoritma untuk menentukan *host* yang akan menerima *VM* yang dipindahkan dari *host* yang mengalamai *overload* atau *underload.* Permasalahan pada algoritma ini merupakan masalah *bin packing* dimana sebuah objek dalam kasus ini adalah *VM* dari beberapa *host* yang berbeda harus ditempatkan dalam *host* dengan jumlah seminimal mungkin. Salah satu algoritma yang digunakan adalah *best fit decreasing (BFD).* Algoritma ini menempatkan *VM* ke dalam *host* yang memiliki *available memory* paling kecil yang mencukupi untuk ditempati oleh *VM* yang akan dipindahkan.

### Live Migration

Menurut Strunk, Anja Dargie Waltenegus (2013) [15], *live migration* merupakan migrasi *virtual machine (VM)* dari satu *physical host (PM)* ke *host* fisik yang lain dalam keadaan *VM* masih berjalan. Teknologi virtualisasi saat ini (berdasarkan *hypervisors*) tidak menggunakan *local disk* untuk menyimpan *image VM*. Karena itu virtualisasi membutuhkan penyimpanan yang bisa bida diakses dalam sebuah jaringan atau disebut juga *network attached strorage (NAS).* Storage dalam *NAS* diakses oleh *host* fisik dan digunakan sebagai *storage VM*. Dengan menggunakan *NAS*, proses *live migration* hanya memindahkan *memmory* dan *registry* VM dari PM sumber ke tujuan.

Menurut Jose Sinti (et.al) (2014) [16] ada beberapa tahapan *Live Migration* yaitu:

* + - 1. *The Set up Stage :*

Merupakan proses menentukan *VM* yang akan dimigrasikan dari *PM* sumber dan menentukan *PM* tujuan dimana *VM* akan diletakan serta menentukan koneksi *TCP* antara *PM* sumber dan tujuan untuk mentransfer data konfigurasi *VM*. Pada *PM* tujuan, memori dialokasikan dan kerangka *VM* diatur.

* + - 1. *The Memory Transfer Stage :*

Pada tahap ini memori *VM* disalin dari *PM* sumber ke *PM* tujuan. Sementara *VM* masih berjalan pada *PM* sumber.

* + - 1. *The Storage Transfer Stage :*

Menghubungkan *PM* tujuan ke *NAS* supaya *VM* di *PM* tujuan dapat mengakses media penyimpanan seperti *virtual hard drive* *(VHD)*.

* + - 1. *The Network Clean-Up Stage :*

Agar transparan, semua koneksi jaringan yang terbuka harus tetap terbuka hingga migrasi selesai. Karena setiap *VM* akan memiliki kartu *Virtual Network Interface (vnic*) yang diidentifikasi oleh *MAC address*.

### TCP-DS

TCP-DS merupakan *dessicion support benchmark* saat ini dikembangkan oleh *Transaction Processing Performance Council* (TPC). merupakan model dari sistem pendukung keputusan yang difungsikan untuk ritail dan *supplier* produk[17]. Mendukung skema yang berisi informasi bisnis vital seperti pelanggan, pesanan dan data produk.

TPC-DS menjalankan benchmark yang terdiri dari dalam 3 tahap yaitu :

1. *Load test*, mendemonstrasikan performa beban dari system. Data yang disajikan ke sistem dalam bentuk flat file yang menyerupai skema bintang dari data warehouse.
2. *Query run*, mengukur kemampuan sistem untuk menggunakan semua sumber daya untuk memenuhi penggunaan bersama dalam waktu sesingkat mungkin. Dijalankan satu set skenario dimana setiap skenario terdiri dari satu atau beberapa *query*.
3. *Data maintenance*, berjalan setelah *query* pertama selesai dieksekusi dan sebelum *query* kedua dieksekusi. Pengukuran waktu *maintencance* dimulai saat kumpulan data baru diakses dan selesai ketika semua pembaruan yang diperlukan pada databasenya lengkap.

Karakteristik load pada TPC-DS akan sama persis dengan karakteristik pada mesin pendukung keputusan yang sebenarnya sehingga bisa digunakan sebagai simulaisi membangkitkan beban kerja mesin pendukung keputusan.

### Evaluasi

Evaluasi yang akan dilakukan pada penelitian ini adalah a*gregated overload time fraction (AOTF)* yaitu persentase total waktu *overload* semua *compute host* dari masing-masing algoritma *overload detection*.

## Kerangka Pemikiran

Berdasarkan dukungan landasan teoritik yang diperoleh dari eksplorasi teori yang dijadikan rujukan konsepsional variabel penelitian, maka dapat disusun Kerangka Pemikiran pada Gambar 2.3 berikut :

Masalah:

1. Bagaimana membangun infrastruktur *cloud computing* yang mampu menerapkan *dynamic VM consolidation*?
2. Bagaimana performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *static CPU utilization threshold* (THR)?
3. Bagaimana performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *local regression* (LR)?
4. Bagaimana performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *Markov host overload detection* (MHOD)?
5. Manakah metode *overload detection* yang menghasilkan performa terbaik?

Tujuan:

1. Membangun infrastruktur *cloud computing* yang mampu menerapkan *dynamic VM consolidation*.
2. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *static CPU utilization threshold* (THR).
3. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *local regression* (LR).
4. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *Markov host overload detection* (MHOD).
5. Menentukan metode *overload detection* yang menghasilkan performa terbaik.

Hasil:

1. Membangun infrastruktur *cloud computing* yang mampu menerapkan *dynamic VM consolidation*.
2. Menentukan performa AOTF *dynamic VM consolidation* dengan menggunakan metode *overload detection* *static CPU utilization threshold* (THR), *local regression* (LR), *Markov host overload detection* (MHOD).
3. Menentukan metode *overload detection* yang menghasilkan performa terbaik.

Literatur:

1. *Energy Efficient Allocation of Virtual Machines in Cloud Data Centers*
2. *Managing Overloaded Hosts for dynamic VM consolidation* *s in Cloud Data Centers Under Quality of Service Constraints*
3. *OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds*
4. *VM Migration Techniques in Cloud Environment – A Survey*

“Analisis Performa *Host Overload Detection* pada *dynamic VM consolidation* dengan Beban Kerja Pendukung Keputusan”.

Metode Eksperimen

Gambar 2.3 Kerangka Pemikiran

# BAB III METODE PENELITIAN

Pada bab ini dijelaskan mengenai metode penelitian yang digunakan dalam penulisan Tugas Akhir ini.

## Instrumen Penelitian

Meliputi bahan dan peralatan dalam melakukan penelitian, dalam penelitian ini diperlukan beberapa perangkat agar penelitian dapat berjalan dengan lancar dan sesuai dengan tema penelitian. Perangkat tersebut dibagi menjadi dua yaitu perangkat keras dan perangkat lunak.

### Perangkat Keras (Hardware)

Merupakan perangkat keras yang digunakan dalam penelitian ini. Berikut adalah perangkat keras yang digunakan untuk membangun *server* *cloud computing* :

1. Personal computer, merupakan komputer berskala kecil yang biasa digunakan oleh perorangan untuk mengerjakan tugas yang diinstruksikan kepadanya dan bukan memberikan layanan kepada komputer lainnya. Pada penelitian ini personal computer digunakan sebagai server, hal ini dilakukan karena keterbatasan perangkat keras.
2. Virtual machine, merupakan komputer dalam bentuk virtual yang menggunakan resource komputer fisik untuk membuat host tersendiri dengan pengelolaan yang terpisah dengan komputer fisik. Virtual machine berjalan di atas komputer server dengan *CPU Intel xeon CPU E5-2650 0 @ 2.0 GHz.*
3. Perangkat jaringan yang digunakan berkecapatan satu gigabit untuk menjamin kualitas komunikasi dalam sistem.

### Perangkat Lunak (Software)

Untuk dapat membangun *cloud computing* yang mendukung *dynamic VM consolidation* dibutuhkan layanan perangkat lunak. Penulis mengelompokan perangkat lunak yang digunakan. Pertama adalah perangkat lunak yang diinstall di server sebagai berikut :

1. Sistem operasi menggunakan CentOS 6.7
2. OpenStack service, merupakan sepaket perangkat lunak yang menyediakan layanan cloud computing.
3. Database service sebagai layanan pengelolaan data.
4. NTP service digunakan untuk menyingkronkan waktu antar host.
5. Message broker, merupakan layanan distribusi pesan antar service untuk berbagi informasi.
6. Logical volume manager sebagai backend layanan block storage.
7. KVM sebagai backend layanan virtualisasi.
8. OpenStack Neat, sebagai framework dynamic VM consolidation.

Kemudian perangkat lunak yang diinstall di VM dengan *workload* TPC-DS yaitu :

1. Sistem operasi menggunakan CentOS 7.
2. Oracle Database Server.
3. SQL Developer

Kelompok terahir adalah perangkat lunak yang diinstall di VM dengan *workload* CPU load generator yaitu :

1. Sistem operasi menggunakan Ubuntu 14.04.
2. CPU load generator.

## Pengumpulan Data

Di dalam penelitian ini, peneliti menggunakan data ritail dan suplier produk yang di-*generate* dari TPC-DS data generator. Dengan tools ini dihasilkan data dalam bentuk *flat file* sebesar 10 GB. Data yang dihasilkan kemudian diimport ke dalam database server.

## Model Penelitian

Model penelitian yang dilakukan oleh peneliti pada penelitian ini digambarkan pada gambar 3.1 di bawah ini.

Identifikasi Masalah

Studi Literatur

Pengumpulan Data

Analisa Kebutuhan dan Desain Arsitektur *Cloud Computing*

Membanguninfrastruktur *Cloud Computing*

*framework Dynamic VM Consolidation*

Pengujian dan Analisa

Gambar 3.1 Model Penelitian

### Identifikasi Masalah

Identifikasi masalah merupakan tahap mengenali sebuah masalah penelitian. Masalah penelitian bisa ditemukan melalui studi literatur dan pengamatan secara langsung seperti observasi atau survei. Pada penelitian ini identifikasi masalah telah diuraikan pada bagian latar belakang masalah.

### Studi Literatur

Merupakan tahap mempelajari literatur yang berhubungan dengan masalah penelitian yang telah dirumuskan sebelumnya. Literatur yang digunakan bisa bersumber dari publikasi penelitian, buku dan situs website di internet.

### Pengumpulan Data

Pada tahap ini penulis mengumpulkan data-data penelitian yang dibutuhkan. Data-data yang dikumpulkan bersumber dari journal, buku, buku-elektronik, dokumentasi projek, dokumentasi penelitian dan data generator.

### Analisa Kebutuhan dan Desain Arsitektur *Cloud Computing*

Beradasarkan studi literatur dan data yang telah dikumpulkan, penulis menganalisa kebutuhan untuk membangun sebuah *server cloud computing* yang mampu menerapkan *dynamic VM consolidation*.

### Membangun Infrastruktur *Cloud Computing*

Tahap ini adalah tahap membangun infrastruktur *cloud computing*. *Server* dibuat dengan spesifikasi yang sesuai dengan yang telah dijelaskan pada sub bab 3.1 instrumen penelitian.

### *Framework Dynamic VM Consolidation*

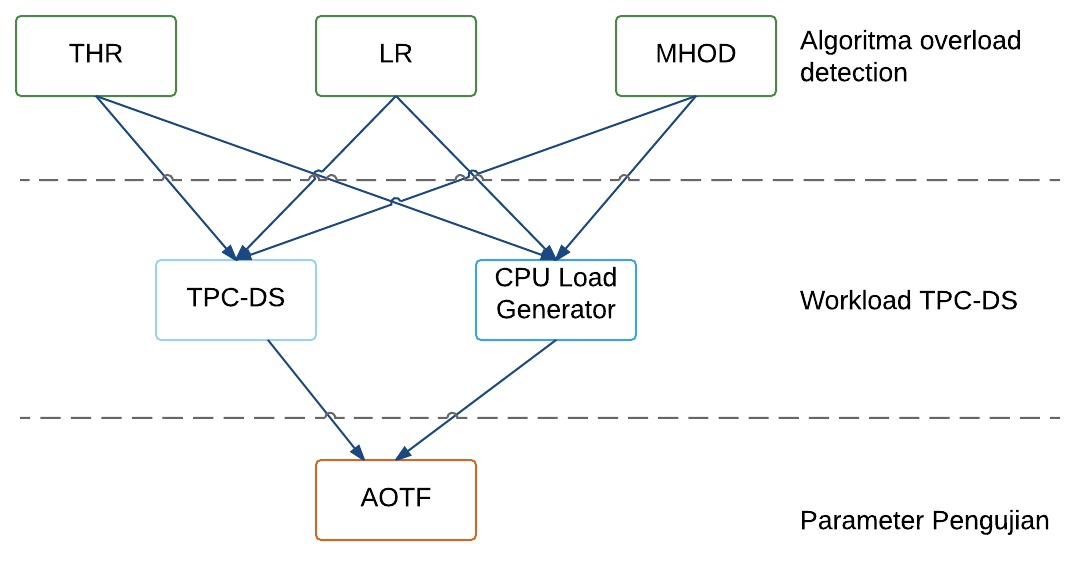
Tahap ini dilakukan setelah *server cloud computing* berjalan dengan baik. *Framework* dimasukan ke dalam *server* untuk dapat menerapkan *dynamic VM consolidation*.

### Pengujian dan Analisis

Merupakan tahap menguji algoritma *host overload detection* di dalam *Dynamic VM Consolidation*. Data hasil pengujian dikumpulkan dan dianalisa untuk dapat menentukan algoritma yang terbaik.

## Desain Pengujian dan Analisis

Pengujian algoritma overload detection di dalam *dynamic VM consolidation* digambarkan dengan skenario berikut.



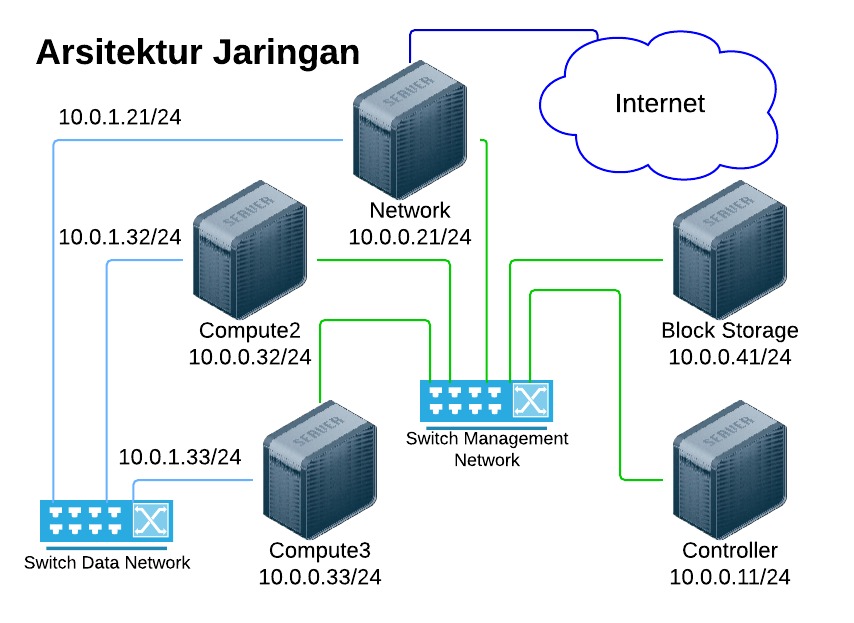
Masing-masing algotima overload detection diuji dengan diterapkan di dalam *dynamic VM consolidation* pada infrastruktur *cloud computing* yang menjalankan VM dengan *workload* TPC-DS. Parameter pengujian yang digunakan adalah Agregated Overload Time Fraction *(AOTF)*, diperoleh dengan menghitung wkatu *overload* semua *host* dibagi dengan waktu kerja semua *host*.

# BAB IV PERCOBAAN

## Desain Cloud Data center

### Arsitektur Jaringan

Ada tiga jaringan lokal dalam cloud data center yaitu management network untuk komunikasi antar service OpenStack, data network sebagai jembatan untuk virtual network, serta external network sebagai jembatan antara jaringan luar dengan virtual network. Arsitektur jaringan cloud data center digambarkan dalam gambar berikut.



Gambar 4.1 Arsitektur Jaringan

### Kebutuhan Hardware

Rincian spesifikasi perangkat keras yang digunakan untuk membangun *server* *cloud computing*.

1. *Compute host* 2 dan *Compute host* 4 menggunakan dua personal komputerdengan spesifikasi masing-masing :
2. *RAM 8 GB, DDR3, 1066MHz.*
3. *CPU intel core i7 @ 2.80 GHz.*
4. *2 Network Interface Card Gigabit Ethernet.*
5. *HDD Seagate 1 TB, SCSI*.
6. *Network Host* menggunakan satu personal komputer dengan spesifikasi :
7. *CPU intel dual core E2160 @ 1.80 GHz.*
8. *RAM 1 GB, DDR2.*
9. *2 Network Interface Card Gigabit Ethernet..*
10. *1 Network Interface Card 100 Mb.*
11. *HDD Samsung 160 GB, IDE.*
12. *Controller Host* menggunakan satu *virtual* komputer dengan spesifikasi virtual :
13. *RAM 2 GB, DDR3, 1066MHz.*
14. Intel xeon CPU E5-2650 0 @ 2.0 GHz dengan alokasi 2 core.
15. *Network Interface Card Gigabit Ethernet.*
16. *Virtual Disk Image 150 GB, SCSI*.
17. *Block Storage* menggunakan personal komputerdengan spesifikasi :
18. *RAM 8 GB, DDR3, 1066MHz.*
19. *CPU intel core i7 @ 2.80 GHz.*
20. *Ethernet intel Gigabit Ethernet.*
21. *HDD Seagate 1 TB, SCSI*.
22. *HDD Western Digital 160 GB, SCSI*
23. 2 switch berkecepatan gigabit8 *port*.

### Arsitektur Layanan OpenStack

Dalam membangun *cloud computing* paket-paket aplikasi berbeda diinstall di setiap *host* sesuai dengan fungsinya masing-masing.

1. *Controller host*, menggunakan sistem operasi *Centos 6.7*, dengan paket aplikasi yang diinstal antara lain :
2. *MySQL*
3. *RabbitMQ*
4. *Keystone*
5. *Glance*
6. *Nova Management*
7. *Neutron Server*
8. *ML2 Plug-in*
9. *Horizon*
10. *Cinder Management*
11. *Swift Proxy*
12. *NTP*
13. *Compute host*, menggunakan sistem operasi *CentOS 6.7*, dengan paket aplikasi yang diinstal antara lain :
14. *Nova Hypervisor*
15. *KVM*
16. *ML2 Plug-in*
17. *Layer 2 Agent (Open VSwitch)*
18. *Nova Management*
19. *NTP*
20. *Network host*, menggunakan sistem operasi *CentOS 6.7*, dengan paket aplikasi yang diinstal antara lain :
21. *ML2 Plug-in*
22. *Layer 2 Agent (Open VSwitch)*
23. *Layer 3 Agent*
24. *DHCP Agent*
25. *NTP*
26. *NFS*
27. *Block Storage*, menggunakan sistem operasi *Centos 6.7*, dengan paket aplikasi yang diinstal antara lain :
28. *Lvm2*
29. *Cinder Volume*
30. *NTP*

## Live Virtual Machine Migration

### Kebutuhan Live Virtual Machine Migration

Live virtual machine migration merupakan salah satu fitur yang didukung oleh OpenStack. Namun untuk bisa menerapkan live VM migration, sebuah cloud data center harus memenuhi sarat sebgai berikut :

1. Menggunakan shared storage untuk mensingkronkan file instance virtual machine.
2. Menggunakan libvirt sebagai backend hypervisor serta mengaktifkan listen protocol tcp pada libvirt.
3. Menggunakan shared storage yang dihubungkan ke virtual machine sebagai virtual hard drive.
4. Mengatur firewall supaya mengizinkan libvirt untuk berkomunikasi antar compute host.
5. Mengkonfigurasi nova untuk mengaktifkan live VM migration.

### Inisiasi Live Virtual Machine Migration

Live VM migration digunakan untuk memindahkan VM dari satu compute host ke compute host yang lain. Sehingga untuk menginisiasi Live VM migration kita harus melengkapi dua buah parameter yang dikirimkan bersamaan dengan inisiasi live VM migration yaitu ID dari VM dan hostname dari compute host tujuan. Untuk itu tahapan dalam menginisiasi live vm migration adalah sebagai berikut :

* + 1. Melihat daftar instance VM yang sedang berjalan untuk memilih dan mendapatkan ID dari VM.
    2. Melihat informasi tentang VM yang telah dipilih untuk mengetahui alokasi CPU, memori dan media penyimpanan.
    3. Melihat hostname compute host dalam cloud data center untuk memilih compute host tujuan.
    4. Memastikan bahwa compute host tujuan siap menerima VM. Host tujuan siap menerima VM jika sisa sumber daya compute host yang dimiliki lebih besar dari sumberdaya yang digunakan oleh VM.
    5. Inisiasi live VM migration dengan menyertakan ID VM dan hostname compute host yang telah dipilih.

## Dynamic VM Consolidation

### Kebutuhan Dynamic VM Consolidation

Untuk dapat melakukan *Dynamic VM Consolidation* arsitektur *cloud computing* harus memenuhi kriteria berikut :

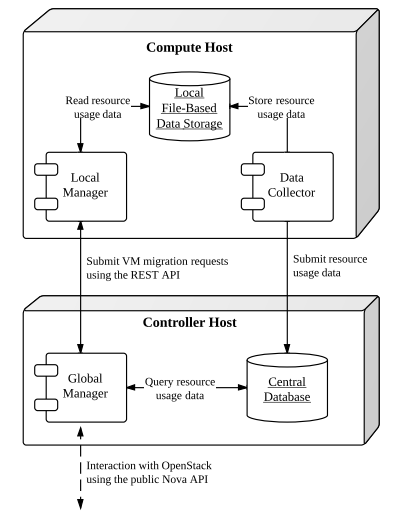
1. Menggunakan shared storage yang dihubungkan ke virtual machine sebagai virtual hard drive.
2. OpenStack Neat menggunakan *live VM migration* untuk memindahkan VM sehingga live VM migration harus diaktifkkan.
3. *Remote login* sebagai *user root* pada compute host diijinkan untuk dapat mematikan PM ke status suspend.
4. Network interface card (NIC) pada compute host mendukung layanan WakeOnLAN sehingga compute host dapat dihidupkan melalui controller host.
5. Paket *dependecy* yang dibutuhkan *Openstack Neat* adalah distribute, pyqcy, mocktest, PyContracts, SQLAlchemy, bottle, requests, libvirt, python-novaclient, NumPy, SciPy.

### Komponen sistem

OpenStack Neat terdiri dari tiga komponen utama yang masing-masing komponen memiliki tugasnya masing-masing yaitu :

1. Global manager, merupkan komponen yang bertugas untuk membuat keputusan dan instruksi global seperti penempatan VM, inisiasi live VM migration dan mematikan host ke status suspend. Global manager diinstall di controller host.
2. Local manager, merupakan komponen yang diinstall di setiap compute host bertugas untuk membuat keputusan lokal di compute host seperti keputusan kondisi underload,overload atau tidak keduanya.
3. Data collector, merupakan komponen yang diinstall di setiap compute host bertugas untuk mengumpulkan data utilisasi cpu pada PM serta VM yang berjalan di atasnya.

Ketiga komponen tersebut digambarkan dalam *deployment diagram* pada gambar berikut.



## Metode Host Overload Detection

### Static CPU Utilization Threshold

### Local Regression

### Markov Host Overload Detection

## Workload

## Skenario Percobaan

# BAB V HASIL DAN ANALISIS

# Jadwal Penyusunan Tugas Akhir

Tabel 3.1 Jadwal Penyusunan Tugas Akhir

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **N0** | **Tahap Penelitian** | **Bulan ke -** | | | | |
| **1** | **2** | **3** | **4** | **5** |
| **1** | Tahap Identifikasi Masalah |  |  |  |  |  |
| **2** | Tahap Studi Literatur |  |  |  |  |  |
| **3** | Tahap Pengumpulan Data |  |  |  |  |  |
| **4** | Tahap Analisa Kebutuhan dan Desain Arsitektur *Cloud Computing* |  |  |  |  |  |
| **5** | Tahap Membangun Infratruktur *Cloud Computing* |  |  |  |  |  |
| **6** | Tahap Implementasi dan Analisa *Dynamic VM Consolidation* |  |  |  |  |  |
| **7** | Tahap Penyimpulan Hasil Penelitian |  |  |  |  |  |

# DAFTAR PUSTAKA

[1] Z. Jia, L. Wang, J. Zhan, L. Zhang, and C. Luo, “Characterizing data analysis workloads in data centers,” *2013 IEEE Int. Symp. Workload Charact.*, pp. 66–76, Sep. 2013.

[2] A. Beloglazov and R. Buyya, “Energy Efficient Allocation of Virtual Machines in Cloud Data Centers,” *2010 10th IEEE/ACM Int. Conf. Clust. Cloud Grid Comput.*, pp. 577–578, 2010.

[3] I. Nstallation, P. Erformance, and S. Tudy, “VIRTUAL MACHINES AND NETWORKS – ADVANTAGES AND VIRTUALIZATION O PTIONS,” *Int. J. Netw. Secur. Its Appl. (IJNSA), Vol.3, No.1, January 2011*, vol. 3, no. 1, pp. 1–15, 2011.

[4] J. Jheng, F. Tseng, H. Chao, and L. Chou, “A Novel VM Workload Prediction using Grey Forecasting Model in Cloud Data Center,” pp. 40–45, 2014.

[5] L. I. Hongyou, W. Jiangyong, P. Jian, W. Junfeng, and L. I. U. Tang, “Energy-Aware Scheduling Scheme Using Workload-Aware Consolidation Technique in Cloud Data Centres,” no. December, pp. 114–124, 2013.

[6] A. Beloglazov, “Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing,” The University of Melbourne, 2013.

[7] J. G. Koomey, “Worldwide electricity used in data centers,” *Environ. Res. Lett.*, vol. 3, no. 3, p. 034008, Jul. 2008.

[8] A. Beloglazov and R. Buyya, “OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds,” *Softw. Pract. Exp.*, p. n/a–n/a, Jun. 2014.

[9] P. G. J. Leelipushpam, “LIVE VM MIGRATION TECHNIQUES IN CLOUD ENVIRONMENT – A SURVEY,” *Proc. 2013 IEEE Conf. Inf. Commun. Technol. (ICT 2013)*, no. Ict, pp. 408–413, 2013.

[10] C. Clark, K. Fraser, S. Hand, and J. Hansen, “Live migration of virtual machines,” *Proc. 2nd Conf. Symp. Networked Syst. Des. Implement.*, vol. 2, no. Vmm, pp. 273–286, 2005.

[11] A. A. Guruh Fajar Shidik, “Efficiency Energy Consumption in Cloud Computing based on Constant Position Selection Policy in Dynamic VM Consolidation,” *Adv. Sci. Lett.*, vol. 4, no. 2, pp. 400–407, 2011.

[12] A. Beloglazov, R. Buyya, and S. Member, “Managing Overloaded Hosts for Dynamic Consolidation of Virtual Machines in Cloud Data Centers Under Quality of Service Constraints,” vol. 24, no. 7, pp. 1366–1379, 2013.

[13] P. F. Silvia, R. Karthiha, R. Aarthy, and C. S. G. Das, “Virtual machine vs real machine: Security systems,” *Int. J. Eng. Technol.*, vol. 1, no. 1, pp. 9–13, 2009.

[14] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, and I. Stoica, “A View of Cloud Computing,” *Communications*, vol. 53, pp. 50–58, 2010.

[15] A. Strunk and W. Dargie, “Does Live Migration of Virtual Machines Cost Energy?,” *2013 IEEE 27th Int. Conf. Adv. Inf. Netw. Appl.*, pp. 514–521, Mar. 2013.

[16] J. Sinti, F. Jiffry, and M. Aiash, “Investigating the Impact of Live Migration on the Network Infrastructure in Enterprise Environments,” *2014 28th Int. Conf. Adv. Inf. Netw. Appl. Work.*, pp. 154–159, May 2014.

[17] M. Poess and B. Smith, “TPC-DS, taking decision support benchmarking to the next level,” *ACM SIGMOD*, pp. 582–587, 2002.