BAB 2

LANDASAN TEORI

2.1 Definisi Penjadwalan

Jadwal adalah daftar *output* (tabel kegiatan)/rencana kegiatan yang harus dihasilkan dalam jangka waktu tertentu, biasanya disusun menurut urutan prioritas dengan pembagian waktu pelaksanaan yang terperinci, sedangkan penjadwalan adalah proses pembuatan/cara menjadwalkan suatu data menjadi jadwal (Harding, 1984). Penjadwalan merupakan proses untuk menyusun suatu jadwal atau urutan proses yang diperlukan dalam sebuah permasalahan.

Fogarty (1991) mengatakan bahwa penjadwalan mencakup dua hal, yaitu *scheduling* dan *sequencing* yang masing-masing didefinisisikan sebagai berikut: ”*Scheduling is the assigning of starting and completion times orders (job) and frequently includes the times when orders are to arrive and leave each department*”. *Scheduling* (penjadwalan) merupakan proses penugasan kapan pekerjaan harus dimulai dan diselesaikan, sedangkan *sequencing* (pengurutan) merupakan proses pengaturan urutan atas pekerjaan-pekerjaan yang harus diselesaikan tersebut. Karena eratnya hubungan diantara kedua istilah ini, maka biasanya dalam penggunaan kata *scheduling* (penjadwalan), pengertian *sequencing* sudah tercakup didalamnya.

Bedworth and Bailey (1987) mengidentifikasikan tujuan dari aktivitas penjadwalan, yaitu meningkatkan penggunaan sumber daya atau mengurangi waktu tunggunya, sehingga total waktu proses dapat berkurang dan produktivitas dapat meningkat.

2.2 Optimasi

2.2.1 Pengertian Optimasi

Optimasi adalah suatu proses untuk mencapai hasil yang ideal atau optimal (nilai efektif yang dapat dicapai). Dalam disiplin matematika optimasi merujuk pada studi permasalahan yang mencoba untuk mencari nilai minimal atau maksimal dari suatu fungsi nyata. Untuk dapat mencari nilai optimal secara sistematis dilakukan pemilihan nilai variabel integer atau nyata yang akan memberikan solusi optimal (Wardy, 2007).

2.2.2 Macam-Macam Permasalahan Optimasi

Permasalahan yang berkaitan dengan optimasi sangat komplek dalam kehidupan masing-masing. Nilai yang didapat dalam optimasi dapat berupa besaran panjang, jarak, waktu dan lain-lain (Wardy, 2007).

Masalah optimasi dapat dikategorikan ke dalam dua kelas besar, yaitu optimasi tanpa pembatas (*unconstrained optimization*) dan optimasi dengan pembatas (*constrained optimization*). Dari namanya kita bisa mengetahui bahwa optimasi tanpa pembatas hanya melibatkan fungsi tujuan, tidak ada pembatas (*constraint*), sedangkan optimasi dengan pembatas, selain fungsi tujuan kita juga mempunyai tambahan pembatas yang membuat permasalahan lebih rumit. Berikut ini adalah termasuk beberapa permasalahan optimasi:

1. Mengatur penjadwalan penyewaan sumber daya pada *cloud computing.*
2. Menentukan pendistribusian beban kerja pada *cloud computing.*

Selain beberapa contoh diatas, masih banyak persoalan lainnya yang terdapat dalam berbagai bidang.

2.2.3 Penyeleesaian Masalah Optimasi

Prosedur pemecahan masalah optimasi adalah memodelkan persoalannya ke dalam sebuah program matematis dan kemudian memecahkannya dengan menggunakan teknik-teknik atau metode optimasi seperti program integer, program linier, program nonlinier, program tujuan ganda, dan metode-metode lainnya yang sudah berkembang saat ini.

2.3 Program integer

Program integer merupakan suatu program linear dengan variabel keputusannya merupakan bilangan bulat, sehingga pada bentuk umum program linear terdapat tambahan syarat bahwa variabel keputusannya harus bilangan bulat. Pada masalah program linear bilangan bulat untuk pola memaksimumkan, nilai tujuan dari program linear bilangan bulat tidak akan pernah melebihi nilai tujuan dari program linear (Wahyujati A., 2009).

Terdapat tiga macam permasalahan dalam program linear bilangan bulat, yaitu sebagai berikut :

1. Program bulat murni (*Pure Integer Programming*), yaitu program linear bilangan bulat yang menghendaki semua variabel keputusan harus merupakan bilangan bulat non-negatif.
2. Program bulat campuran (*Mixed Integer Programming*), yaitu program linear bilangan bulat yang menghendaki beberapa, tetapi tidak semua variabel keputusan harus merupakan bilangan bulat non-negatif.
3. Program bulat biner (*Zero One Integer Programming*), yaitu program linear bilangan bulat yang menghendaki semua variabel keputusan harus bernilai 0 atau 1 (*binary*).

Bentuk umum model program integer adalah:

$$Max(min)\ Z = \sum c_{ij}\, x_j \tag{2.1}$$

dengan kendala,

$$\sum a_{ij}\, x_j\ (\leq, =, \geq) b_i\ , (i = 1,2,3, \dots, m), \tag{2.2}$$

$x_j \geq 0, (j = 1,2,3, \dots, n)$

x_j bernilai integer untuk beberapa atau semua j

Bentuk umum model program integer 0-1 adalah:

$$Max(min)Z = \sum c_{ij}\, x_j \tag{2.3}$$

dengan kendala,

$$\sum a_{ij}\, x_j\ (\leq, =, \geq) b_i\ , (i = 1,2,3, \dots, m), \tag{2.4}$$

$x_j = 0\ atau\ x_j = 1, (j = 1,2,3, \dots, n)$

Keterangan:

x_j = variabel keputusan

c_j = koefisien fungsi tujuan

a_{ij} = koefisien fungsi kendala

b_i = koefisien pembatas

2.4 Definisi *Cloud Computing*

Cloud computing merupakan teknologi dimana sebagian besar proses dan komputasi terletak di jaringan Internet sehingga memungkinkan pengguna dapat mengakses layanan yang diperlukan dari manapun (Hewitt, 2008).

Cloud computing merupakan suatu model yang mempermudah ketersediaan dan konfigurasi layanan baik berupa perangkat lunak, jaringan, server, media penyimpanan maupun aplikasi (Grance, 2009).

Cloud computing adalah sebuah mekanisme yang memungkinkan kita "menyewa" sumber daya teknologi informasi (*software*, *processing power*, *storage*, dan lainnya) melalui internet dan memanfaatkan sesuai kebutuhan kita dan membayar sesuai dengan yang digunakan oleh kita saja. Berbagai definisi mengenai *cloud computing* banyak diungkapkan oleh para ahli dan peneliti seperti, Peter Mell dan Tim Grance dari National Institute of Standards and Technology (NIST), Information Technology Laboratory mendefinisikan *cloud computing* sebagai suatu model yang mempermudah ketersediaan dan konfigurasi layanan baik berupa perangkat lunak, jaringan, *server*, media penyimpanan maupun aplikasi.

Cloud computing adalah penggunaaan oleh *user* pada sebuah computer dan menjalankan sebuah aplikasi dimana file-file tersebut tidak terdapat di komputer yang digunakannya namun berada di komputer lain yang dihubungkan oleh jaringan. Dalam *Cloud computing* terdapat istilah *front-end* (Desktop-PC) dan *back-end* (*Server*). Keduanya harus saling terhubung oleh sebuah jaringan yang dapat berupa internet atau untuk skala yang lebih kecil. *Front-end* (Desktop-PC) yang mengambil data dan menjalankan aplikasi, sedangkan *back-end* merupakan *resource* yang diistilahkan dengan awan (*cloud*).

Sebagaimana telah dijelaskan pada defenisi di atas bahwa *cloud computing* adalah layanan teknologi informasi yang di manfaatkan melalui jaringan internet, namun tidak semua layanan yang ada di internet dapat dikategorikan sebagai

layanan *cloud computing*. Ada pun beberapa syarat yang harus dipenuhi agar layanan yang ada di internet dikatakan sebagai layanan *cloud computing* yaitu,

1. Layanan bersifat "*On Demand*", pengguna dapat berlangganan hanya yang dia butuhkan saja, dan membayar hanya untuk yang mereka gunakan saja. Misalkan sebuah sebuah internet *service provider* menyediakan 5 macam pilihan atau paket-paket internet dan *user* hanya mengambil 1 paket internet maka *user* hanya membayar paket yang diambil saja.

2. Layanan bersifat *elastis* atau *scalable*, di mana pengguna bisa menambah atau mengurangi jenis dan kapasitas layanan yang dia inginkan kapan saja dan sistem selalu bisa mengakomodasi perubahan tersebut. Misalkan *user* berlangganan internet pada yang bandwidthnya 512 Kb/s lalu ingin menambahkan kecepatannya menjadi 1Mb/s kemudian *user* menelpon *costumer service* meminta untuk penambahan *bandwitch* lalu *customer service* merespon dengan mengubah *bandwidth* menjadi 1Mb/s.

3. Layanan sepenuhnya dikelola oleh *provider*, yang dibutuhkan oleh pengguna hanyalah komputer ditambah koneksi internet.

4. Sumber Daya Terkelompok (*Resource pooling*)

   Penyedia layanan cloud computing memberikan layanan melalui sumber daya yang dikelompokkan di satu atau berbagai lokasi data *center* yang terdiri dari sejumlah *server* dengan mekanisme *multi-tenant.* Mekanisme *multi-tenant* ini memungkinkan sejumlah sumber daya komputasi digunakan secara bersama-sama oleh sejumlah *user,* dimana sumber daya tersebut baik yang berbetuk fisik atau virtual, dapat dialokasikan secara dinamis untuk kebutuhan pengguna atau pelanggan sesuai permintaan. Dengan demikian pelanggan tidak perlu tahu bagaimana dan darimana permintaan akan sumber daya komputasinya terpenuhi oleh penyedia layanan yang ada di *cloud computing*. Yang penting setiap permintaan dapat dipenuhi. Sumber daya komputasi ini meliputi media penyimpanan, *memory*, *processor*, pita jaringan dan mesin virtual.

5. Akses Pita Lebar

Layanan yang terhubung melalui jaringan pita lebar, terutama dapat diakses secara memadai memalui jaringan internet. Baik menggunakan *thin client, thick client,* ataupun media lain seperti *smartphone.*

6. Layanan yang terukur. (*Measured Service*)

Sumber daya *cloud* yang tersedia harus dapat diatur dan dioptimasi penggunaannya, dengan suatu sistem pengukuran yang dapat mengukur penggunaan dari setiap sumber daya komputasi yang digunakan (penyimpanan, *memory, processor*, lebar pita, aktivitas *user*, dan lainnya). Dengan demikian, jumlah sumber daya yang digunakan dapat secara transparan diukur yang akan menjadi dasar bagi *user* untuk membayar biaya penggunaan layanan.

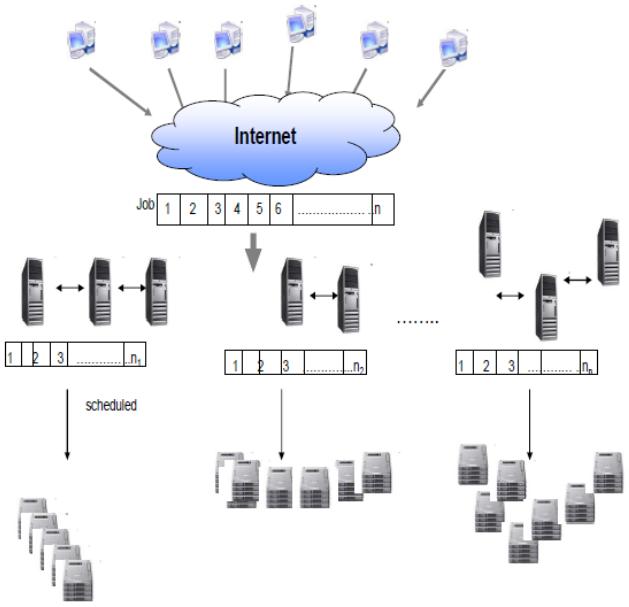
Selain itu karakterisik dari *cloud computing* adalah sangat cepat di *deploy*, *instant* untuk implementasi. Dalam hal ini :

a. Biaya *start up* teknologi ini (*cloud computing*) mungkin akan sangat murah ataupun tidak ada, dan juga tidak ada investasi kapital.
b. Biaya dari *service* dan pemakaian akan berdasarkan komitmen yang tidak fix.
c. Pelayanan ini (*cloud computing*) dapat dengan mudah di *upgrade* atau *downgrade* dengan cepat tanpa adanya *penalty.*
d. Pelayanan akan menggunakan metode *multi-tenant* (banyak *customer* dalam 1 *platform*)
e. Kemampuan untuk meng-*customize* pelayanan akan menjadi terbatas.

Komponen dari *cloud computing* terdiri dari:

1. *Infrastructure*: infrastruktur yang menunjukkan konsep "*Hadware as a service*" yaitu perangkat keras yang digunakan secara bersama sebagai layanan.
2. *Storage*: konsep yang menunjukkan pemisahan data dari pengolahan dan penyimpanan yang terletak pada tempat yang berjauhan. Layanan ini juga disebut layanan *database*.

3. *Platform*: layanan untuk men-*deploy* aplikasi dan mengelola perangkat keras dan perangkat lunak yang diperlukan.
4. *Application*: konsep perangkat lunak yang menawarkan asitektur tanpa proses menginstal, menjalankan, dan memelihara aplikasi pada pengguna *desktop*.
5. *Service*: bagian yang independen dari perangkat lunak yang dapat digunakan bersama-sama untuk mendukung interaksi antar aplikasi melalui jaringan.
6. *Client*: perangkat keras dan lunak yang memanfaatkan layanan *cloud computing* melalui jaringan. Perangkat *client* dapat berupa web *browser*, leptop, PC, dan lain-lain.



Gambar 2.1. Cloud computing
(Sumber: Paul et al., 2011)

Dilihat dari jenis layanan tersendiri, *cloud computing* terbagi dalam 3 jenis layanan secara umum, yaitu sebagai berikut:

1. SaaS *(Software as a Services)*.

   SaaS adalah istilah terhadap *software* atau aplikasi tertentu berbasis internet yang ditawarkan oleh *provider* kepada pengguna.

2. PaaS *(Platform as a Services).*

   PaaS adalah layanan yang menyediakan modul-modul siap pakai yang dapat digunakan untuk mengembangkan sebuah aplikasi, yang tentu saja hanya bisa berjalan diatas *platform* tersebut.

3. IaaS *(Infrastructure as a Services).*

   IaaS adalah sebuah layanan yang menyewakan sumberdaya teknologi informasi dasar, yang meliputi media penyimpanan, *processing power*, *memory*, sistem operasi, kapasitas jaringan dan lain-lain, yang dapat digunakan oleh pelanggannya untuk menjalankan aplikasi yang dimilikinya. IaaS umumnya memiliki fitur:

   a. Memiliki pilihan virtual *machine* yang beragam, baik yang sama sekali kosong, memiliki OS *preinstalled*, bahka telah memiliki beberapa *office productivity tools* terinstal.
   b. Kemampuan untuk meningkatkan atau menurunkan kemampuan *computing* baik secara manual atau otomatis.
   c. Terdapat *tools* untuk memproses banyak data ataupun memproses aplikasi dengan perhitungan yang rumit
   d. Dapat menyimpan data pada beberapa lokasi geografis fisik (memudahkan *download*).

Virtualisasi bisa diartikan sebagai pembuatan suatu bentuk atau versi virtual dari sesuatu yang bersifat fisik, misalnya sistem operasi, perangkat *storage*/penyimpanan data atau sumber daya jaringan (Harry Sufehmi, Pengenalan Virtualisasi, 2009). Virtualisasi bisa diimplementasikan kedalam berbagai bentuk, antara lain:

1. *Network Virtualization* : VLAN, *Virtual* IP (*untclustering*), *Multilink*

2. *Memory Virtualization* : *pooling memory* dari *node-node* di *cluster*

3. *Grid Computing* : banyak komputer = satu

4. *Application Virtualization* : *Dosemu, Wine*

5. *Storage Virtualization* : RAID, LVM

6. *Platform Virtualization : virtual computer*

Sementara dari sifat jangkauan layanan, *cloud computing* terbagi menjadi 4 jenis layanan yaitu *Public Cloud, Private Cloud dan Hybrid Cloud*.

a. *Public Cloud*.

   Jenis cloud ini diperuntukkan untuk umum oleh penyedia layanannya.

b. *Private Cloud.*

   Merupakan infrastruktur layanan *cloud*, yang dioperasikan hanya untuk sebuah organisasi tertentu. Infrastruktur *cloud* itu bisa saja dikelola oleh sebuah organisasi itu atau oleh pihak ketiga. Lokasinya pun bisa *on-site* ataupun *off-site*. Biasanya organisasi dengan skala besar saja yang mampu memiliki/mengelola *private cloud* ini.

c. *Community cloud.*

   Dalam model ini, sebuah infrastruktur *cloud* digunakan bersama-sama oleh beberapa organisasi yang memiliki kesamaan kepentingan, misalnya dari sisi misinya, atau tingkat keamanan yang dibutuhkan, dan lainnya.

d. *Hybrid Cloud.*

   Untuk jenis ini, infrastruktur *cloud* yang tersedia merupakan komposisi dari dua atau lebih infrastruktur *cloud* (*private*, *community*, atau *public*). meskipun secara entitas mereka tetap berdiri sendiri, tapi dihubungkan oleh suatu teknologi / mekanisme yang memungkinkan portabilitas data dan aplikasi antar *cloud* itu. Misalnya, mekanisme *load balancing* yang antar *cloud*, sehingga alokasi sumberdaya bisa dipertahankan pada level yang optimal.

Pada *cloud computing*, ada sebuah layanan pembatas dalam melakukan transaksi data, layanan ini disebut *Service Level Agreement* (SLA). *Service Level Agreement* (SLA) merupakan sebuah kontrak di mana dua pihak telah bersepakat tentang *terms* dan *conditions* yang terkait dengan penyediaan layanan. Masing-masing pihak harus mengerti peran dan tanggungjawab yang terkait dengan layanan tersebut. Keberadaan SLA sangat penting dalam menjamin kepuasan pengguna layanan. SLA dapat digambarkan sebagai jaminan secara hukum tentang layanan minimal yang harus disediakan oleh *provider* (Schaaf, 2009).

Sebuah SLA yang baik setidaknya memuat komponen komponent berikut ini:

a. *Purpose*: menggambarkan alasan.
b. SLA *parties*: menerangkan pihak-pihak yang terlibat dalam SLA beserta peranan masing masing (*provider* dan *consumer*).
c. *Validity period*: menentukan jangka waktu sampai berapa lama masa berlakunya SLA. Hal ini ditunjukkan dengan tanggal mulai dan berakhirnya layanan.
d. *Scope:* menentukan batasan jasa-jasa apa saja yang ditanggung dalam SLA tersebut.
e. *Restrictions:* menentukan langkah-langkah yang diperlukan agar layanan yang dikehendaki dapat disediakan.
f. *Service level objectives*: merupakan tingkatan sejauh mana layanan yang disepakati oleh masing-masing pihak (*user* dan *service provider*), biasanya termasuk juga indikator-indikator seperti *availability, performance dan reliability*.
g. *Penalties:* menjelaskan denda apa jika terjadi kasus jika layanan yang disediakan di bawah performa atau bahkan tidak dapat dipenuhi sesuai kesepakatan dalam SLA. Karenanya skema pemutusan secara sepihak juga harus dipersiapkan.
h. *Optional services:* menyediakan jasa-jasa yang biasanya secara normal tidak dibutuhkan oleh pengguna, tapi mungkin dibutuhkan sebagai sebuah pengecualian.
i. *Exclusions:* menetapkan hal hal yang tidak diatur dalam SLA.
j. *Administration:* menjelaskan proses pembuatan SLA dalam mengukur dan menetapkan jenis-jenis layanan beserta tanggungjawab masing masing pihak selama proses pembuatan SLA (Sahai et al, 2002).

Li (2012) dalam jurnalnya menunjukkan bahwa optimasi penjadwalan *resource* memiliki hubungan dengan SLA (*servis level agreement*) yaitu sebuah kontrak yang digambarkan sebagai jaminan secara hukum tentang layanan pada *cloud computing*. Model untuk penjadwalan *resource* menggunakan *stochastic integer programming* yaitu:

$$Min \sum_{i=1}^{|\alpha|} \sum_{j=1}^{|\beta_i|} c_{ij}\ x_j \tag{2.5}$$

dengan kendala:

$$\sum_{i=1}^{|\alpha|} \sum_{j=1}^{|\beta_i|} c_{ij}\ x_j \leq\ C_{SLA} \tag{2.6}$$

$$\sum_{j=1}^{|\beta_i|} x_j\ = 1, i \in \{1, \dots, |\alpha|\} \tag{2.7}$$

$$Pro\left\{ min\{\xi_{ij}^{t} : i \in \{1, \dots, |\alpha|\}, j \in \{1, \dots, |\beta_i|\}, x_j\ \neq 0\} \geq T_{SLA}, \right.$$

$$\left. \sum_{i=1}^{|\alpha|} \sum_{j=1}^{|\beta_i|} \xi_{ij}^{l}\ x_j\ \leq L_{SLA} \right\} \geq \gamma \tag{2.8}$$

$$x_j\ \in \{0,1\}$$

di mana,

α : himpunan layanan *server*

$|\alpha|$: banyak elemen di himpunan α

β_i : himpunan *resource* yang tersedia untuk layanan *server* ke-i

$|\beta_i|$: banyak elemen di himpunan β_i

x_j : $\begin{cases} 1 : \text{pilih } resource \text{ ke} - \text{j untuk layanan } server \text{ ke} - \text{i} \\ 0 : \text{sebaliknya} \end{cases}$

c_{ij} : biaya dari *resource* ke-j untuk layanan *server* ke-i

C_{SLA} : biaya SLA

T_{SLA} : nilai taksiran SLA

L_{SLA} : interval waktu SLA

ξ_{ij}^{t} : variabel acak untuk nilai taksiran *resource*

ξ_{ij}^{l} : variabel acak untuk interval waktu *resource*

γ : peluang yang diberikan SLA untuk penyimpanan data di *cloud computing*

Persamaan (2.5) artinya keseluruhan biaya adalah solusi minimal oleh variabel x_j. Persamaan (2.6) artinya keseluruhan biaya adalah kurang dari atau sama dengan biaya SLA. Persamaan (2.7) artinya hanya satu *resource* yang terpilih untuk diimplementasikan pada layanan *server*. Persamaan (2.8) artinya peluang diizinkannya penyimpanan data di *cloud computing* berdasarkan dua variabel acak SLA adalah lebih besar atau sama dengan γ.

2.5 LINDO

Linier interaktive discrate optimizer (LINDO) adalah sebuah paket program *under* Windows yang bisa digunakan untuk mengolah kasus pemrograman linier, dilengkapi dengan berbagai perintah yang memungkinkan pemakai menikmati kemudahan-kemudahan didalam memperoleh informasi maupun mengolah data atau memanipulasi data.

Dengan menggunakan *software* ini memungkinkan perhitungan masalah pemrograman linier dengan n variabel. Prinsip kerja utama LINDO adalah memasukkan data, menyelesaikan serta menaksirkan kebenaran dan kelayakan data berdasarkan penyelesaiannya. Menurut Linus Scharge (1991), perhitungan yang digunakan pada LINDO pada dasarnya menggunakan metode simpleks. Sedangkan untuk menyelesaikan masalah pemrograman linier *integer* nol-satu *software* LINDO menggunakan metode Branch and Bound (Mark Wiley, 2010).

Untuk menentukan nilai optimal dengan menggunakan LINDO diperlukan beberapa tahapan yaitu:

1. Menentukan model matematika berdasarkan data real.
2. Menentukan formulasi program untuk LINDO.
3. Membaca hasil *report* yang dihasilkan oleh LINDO.

Kegunaan utama dari program LINDO adalah untuk mencari penyelesaian dari masalah linier dengan cepat dengan memasukan data yang berupa rumusan dalam bentuk linier. LINDO memberikan banyak manfaat dan kemudahan dalam memecahkan masalah optimasi dan minimasi. Model LINDO minimal memiliki tiga syarat yaitu:

1. Memerlukan fungsi objectif.
2. Variabel.
3. Batasan (fungsi kendala).