Pengembangan *Database Genbank* UAI-*Bioinformatics* Menggunakan Sistem Terdistribusi

Ade Jamal, Denny Hermawan, Muhammad Nugraha

Program Studi Teknik Informatika, Fakultas Sains dan Teknologi
Universitas Al Azhar Indonesia, Jl. Sisingamangaraja, Jakarta 12110

Penulis untuk korespondensi/E-mail: adja@uai.ac.id

Abstrak – **Telah dilakukan penelitian tentang pengolahan terdistribusi data genbank menggunakan *Hadoop Distributed Filesystem* (HDFS) dengan tujuan mengetahui efektifitas pengolahan data genbank khususnya pada pencarian sequens dengan data masukan yang berukuran besar. Penelitian dilakukan di Laboratorium Jaringan Universitas Al Azhar Indonesia dengan menggunakan 6 komputer dan satu *server* dimana dalam *Hadoop* menjadi 7 *node* dengan rincian 1 *namenode*, 7 *datanode*, 1 secondary *namenode*. Dengan eksperimen HDFS menggunakan 1 *node*, 2 *node*, 4 *node*, 6 *node*, dan 7 *node* dibandingkan dengan *Local Filesystem*. Hasil menunjukan proses pencarian sequens data genbank menggunakan 1 – 7 *node* pada skenario eksperimen pertama dengan *output* yang menampilkan hasil 3 *field (Locus, Definition,* dan *Authors*), skenario eksperimen kedua dengan *output* yang menampilkan hasil 3 *field (Locus, Authors,* dan *Origin)*, dan skenario eksperimen ketiga menggunakan HDFS dan LFS dengan *output* yang menampilkan seluruh *field* yang terdapat dalam data genbank (*Locus, Definition, Accesion, Version, Keywords, Source, Organism, Reference, Authors, Title, Journal, Pubmed, Comment, Features,* dan *Origin*). Evaluasi menunjukan bahwa proses pencarian sequens data genbank menggunakan HDFS dengan 7 *node* adalah 4 kali lebih cepat dibandingkan dengan menggunakan 1 *node*. Sedangkan perbedaan waktu pada penggunaan HDFS dengan 1 *node* adalah 1.02 kali lebih cepat dibandingkan dengan *Local Filesystem* dengan 4 *core processor*.**

Abstract - **A research on distributed processing of GenBank data using Hadoop Distributed File System GenBank (HDFS) in order to know the effectiveness of data processing, especially in the search sequences with large input data. Research conducted at the Network Laboratory of the University of Al Azhar Indonesia using 6 computers and a server where the Hadoop to 7 nodes with details 1 namenode, 7 datanode, 1 secondary namenode. With HDFS experiments using 1 node, node 2, node 4, node 6, and 7 nodes compared with the Local Filesystem. The results show the search process of data GenBank sequences using 1-7 nodes in the first experiment scenario with an output that displays the results of 3 fields (Locus, Definition, and Authors), a second experiment scenario with an output that displays the results of 3 fields (Locus, Authors, and Origin) , and the third experiment scenarios using HDFS and LFS with output that displays all the data fields contained in GenBank (Locus, Definition, Accesion, Version, Keywords, Source, Organism, Reference, Authors, Title, Journal, Pubmed, Comment, Features, and Origin). Evaluation shows that the search process of data GenBank sequences using HDFS with 7 nodes is 4 times faster than using one node. While the time difference in the use of HDFS with one node is 1:02 times faster than the Local File System with 4 core processor.**

Keywords – *genbank, sequens, distributed computing, Hadoop, HDFS*

PENDAHULUAN

National Center for Biotechnology Information (NCBI) merupakan institutsi pemerintah yang menyediakan sumber informasi mengenai biologi molekuler. NCBI memiliki salah satu *database* yang dapat diakses oleh publik berhubungan dengan *database* sequens DNA lebih dari 100.000 organisme yang berbeda, yaitu Genbank. Untuk melakukan pencarian sequens data Genbank diperlukan *web* NCBI dan aplikasi pendukung lain yang harus digunakan. Aplikasi tersebut merupakan aplikasi tersendiri dan tidak disediakan oleh NCBI. Namun untuk melakukan ini merupakan hal yang merepotkan karena membutuhkan koneksi internet yang cepat dan stabil, serta tidak mudah bagi yang belum terbiasa menggunakannya. Solusinya adalah memindahkan *database* Genbank ke dalam *Local Filesystem* agar tidak memerlukan koneksi internet lagi [1]. Dibutuhkan spesifikasi komputer yang tinggi untuk dapat melakukan pencarian sequens data Genbank yang sedemikian besar. Hal ini tentu merugikan dari segi biaya karena komputer dengan spesifikasi tinggi dan terbaik mempunyai harga yang mahal.

Dibutuhkan suatu teknologi dalam mengakomodir kebutuhan ini dengan biaya yang minimal namun dengan hasil optimal. Maka solusi terbaik adalah menggunakan teknologi *distributed computing* dengan sistem *cluster*. *Distributed computing* merupakan suatu teknologi yang dapat memecahkan problem besar ke dalam proses-proses kecil ke banyak komputer untuk kemudian proses kecil itu dipecahkan secara simultan dan apabila sudah didapatkan solusi-solusi kecil maka disatukan kembali dalam satu solusi yang besar dan terintegrasi.

Hadoop merupakan *framework software* berbasis *Java* dan *open source* yang berfungsi untuk mengolah data yang sangat besar secara terdistribusi dan berjalan di atas *cluster* yang terdiri dari beberapa komputer yang saling terhubung. *Hadoop* dapat mengolah data dalam jumlah yang sangat besar hingga *petabyte* dan dijalankan di atas ribuan komputer. Dengan menggunakan *Hadoop* melalui HDFS, maka pencarian sequens data Genbank dapat dilakukan tanpa menggunakan koneksi internet yang cepat dan stabil serta dapat menggunakan beberapa komputer personal yang terdapat pada Laboratorium Jaringan Universitas Al Azhar Indonesia.

KERANGKA TEORI

Hadoop

Hadoop adalah *framework software* berbasis Java yang berfungsi untuk mengolah data berukuran besar secara terdistribusi dan berjalan di atas *cluster* komputer yang saling terhubung serta jumlah komputer dapat ditambahkan tanpa batasan [2]. *Hadoop* memiliki 2 inti, yaitu HDFS (*Hadoop Distributed Filesystem*) dan *MapReduce.*

HDFS (*Hadoop Distributed Filesystem*)

HDFS merupakan *filesystem* yang menyimpan data sangat besar, berjalan di atas *cluster* dengan spesifikasi komputer pada umumnya dan disimpan secara terdistribusi di banyak komputer dalam satu *cluster* yang saling berhubungan. Terdapat tiga keunggulan dalam HDFS, yaitu 1) File yang sangat besar. HDFS dapat digunakan untuk data sebesar ratusan megabyte, gigabyte, terabyte, bahkan saat ini sudah ada *cluster Hadoop* yang menggunakan data sampai *petabyte*, 2) *Streaming data access*. HDFS dibangun dengan pola streaming data access, yaitu data cukup ditulis satu kali ke dalam HDFS, namun dapat dibaca atau diproses lebih dari sekali, 3) Penggunaan *hardware* pada umumnya tidak membutuhkan *hardware* dengan spesifikasi khusus dalam menggunakan dan mengimplementasikan HDFS.

HDFS juga memiliki *block size* sebesar 64 MB. *Block size* pada HDFS sebesar itu karena untuk meminimalisir waktu membaca suatu data. Dengan *block size* yang cukup besar, waktu yang dibutuhkan untuk transfer data dari *disk drive* akan lebih besar secara signifikan dibandingkan dengan *single disk drive*. Ada keuntungan apabila menggunakan HDFS yang memiliki *block size* besar, yaitu file yang disimpan dalam HDFS dapat berukuran lebih besar dibandingkan dengan satu *hard disk drive*. Misalnya, file berukuran 10TB tidak akan bisa disimpan kedalam *hard disk drive* berukuran 3TB yang merupakan *hard disk drive* dengan kapasitas terbesar saat ini. Namun dengan HDFS, penyimpanan file sebesar itu dapat dilakukan [4].

HDFS *cluster* memiliki dua tipe *node* yang beroperasi, yaitu *namenode* (yang bekerja sebagai *master*) dan *datanode* (yang berkerja sebagai *slave*) [3]. *Namenode* bertugas untuk *mengatur filesystem namespace*. *Filesystem namespace* memelihara *filesystem tree* dan *metadata* untuk semua file dan direktori dalam *tree* yang ada dalam *cluster*. Informasi ini disimpan dalam *disk*

dengan format dua file, yaitu *namespace image* dan *edit log* [4].

Filesystem image adalah file yang berisi tentang informasi seluruh direktori dan file yang terdapat dalam HDFS. Informasi yang ada meliputi level replikasi, modifikasi dan *access time*, *access permission*, *block size*, dan apa yang terdapat di dalam *block* tersebut. *Edit log* adalah hasil sinkronisasi setelah setiap *write* sebelum kode sukses dikembalikan ke *datanode*. Saat *namenode* melakukan *write* dengan multi direktori, maka *write* harus disinkronisasi sebelum kode sukses dikembalikan ke *datanode*. Ini untuk menjaga dan meyakinkan bahwa tidak ada operasi yang hilang karena komputer *failure* [5].

Datanode adalah pekerja atau *slave* dalam HDFS. *Datanode* menyimpan dan menerima *block* ketika *datanode* diberitahu oleh *datanode* lain atau *namenode* dan *datanode* melaporkan kembali ke *namenode* secara periodik dilengkapi dengan daftar *block* apa saja yang disimpan dalam *datanode* tersebut [3].

Tanpa *namenode*, maka HDFS tidak bisa digunakan sama sekali. Apabila *Hadoop Cluster* telah dijalankan namun *namenode* terhapus, semua data yang tersimpan dalam HDFS akan hilang karena tidak ada cara lain untuk mengetahui keberadaan data dan merekonstruksi file yang terbagi di *block – block datanode* dalam HDFS selain oleh *namenode*. Itu merupakan alasan betapa pentingnya *namenode* sampai tidak terjadi kegagalan dan *Hadoop* mempunyai solusi untuk masalah ini. Pertama adalah melakukan *backup file* yang terdapat dalam HDFS secara periodik. *Hadoop* bisa dikonfigurasi untuk melakukan *backup* ke dalam *Local Filesystem* dari HDFS. Kedua adalah dengan menggunakan secondary *namenode*. Aturannya adalah melakukan *merge namespace* secara periodik berdasarkan *edit log* yang tersimpan dalam *namenode*. Biasanya *secondary namenode* disimpan ditempat yang berbeda dengan *namenode* namun memiliki spesifikasi komputer yang sama untuk dapat melakukan *merge* yang sempurna. *Secondary namenode* menyalin *namespace image*, agar bisa digunakan seketika ketika *namenode* gagal melakukan tugasnya. Namun secondary *namenode* tidak bisa menggantikan posisi *primary namenode* karena *secondary namenode* hanya menyimpan *metadata* terbaru agar bisa digunakan oleh *namenode* lain untuk melanjutkan *job* baru berdasarkan *edit log* terakhir menggantikan *namenode* yang *failed* [4].

MapReduce

MapReduce adalah model pemrograman yang diperkenalkan oleh *Google* dan digunakan untuk mendukung *distributed computing* yang dijalankan di atas *data set* yang sangat besar dan dijalankan secara simultan dibanyak computer [6]. Model pemrograman ini terinspirasi oleh konsep fungsi *map* dan *reduce* yang biasa digunakan di *functional programming* [2].

1. Proses "*Map*" :

Masternode menerima *input*, kemudian *input* tersebut dipecah menjadi beberapa sub problem yang kemudian didistribusikan ke *worker node*s. Worker *node*s ini akan memproses sub problem yang diterimanya untuk kemudian apabila problem tersebut sudah diselesaikan, maka akan dikembalikan ke *masternode*.

2. Proses "*Reduce*" :

Masternode menerima jawaban dari semua sub problem dari banyak data *node*s, menggabungkan jawaban-jawaban tersebut menjadi satu jawaban besar untuk mendapatkan penyelesaian dari permasalahan utama.

Keuntungan dari *MapReduce* ini adalah proses *map* and *reduce* dijalankan secara terdistribusi. Dalam setiap proses *mapping* bersifat independen sehingga proses dapat dijalankan secara simultan dan paralel. Demikian pula dengan proses *reducer* dapat dilakukan secara paralel diwaktu yang sama, selama *output* dari operasi mapping mengirimkan *key value* yang sesuai dengan proses *reducer*-nya. Proses *MapReduce* ini dapat diaplikasikan di *cluster server* yang jumlahnya sangat banyak sehingga dapat mengolah data dalam jumlah *petabyte* hanya dalam waktu beberapa jam[4].

Didalam *Hadoop*, *MapReduce engine* ini terdiri dari satu *jobtracker* dan satu/banyak *tasktracker*. *Jobtracker* adalah *server* penerima *job* dari *client*, sekaligus akan mendistribusikan *job* tersebut ke *tasktracker* yang akan mengerjakan *sub job* sesuai yang diperintahkan *jobtracker*. Strategi ini akan mendekatkan pengolahan data dengan datanya sendiri, sehingga ini akan sangat signifikan mempercepat proses pengolahan data.

Pencarian Sequens Data Genbank Menggunakan *Hadoop*

Dengan menggunakan *Hadoop* maka proses pencarian sequens data Genbank dapat dilakukan dengan cepat dan efisien dibandingkan dengan

proses pencarian sequens data Genbank menggunakan *single* processing. *Hadoop* memiliki 2 inti, yaitu *MapReduce* dan HDFS. Data Genbank yang telah diunduh menggunakan *Wget* diunggah kedalam HDFS (*Hadoop Distributed Filesystem*) yang merupakan *Filesystem Hadoop*. Kemudian dilakukan proses pencarian sequens data berdasarkan *field – field* dalam data Genbank sesuai dengan yang diinginkan dan diproses oleh *MapReduce*. *MapReduce* merupakan *framework software* yag digunakan untuk mendukung *distributed computing* yaitu *Hadoop* dan dijalankan dalam data dengan jumlah besar yang tersimpan dalam HDFS.

Hadoop mengintegrasikan *MapReduce* dan HDFS untuk mencapai tujuan yaitu melakukan proses pencarian sequens data Genbank yang telah diunggah kedalam HDFS dan diolah dengan menggunakan *MapReduce* untuk menampilkan hasil pencarian berdasarkan *record* dan *field – field* yang telah ditentukan sebelumnya.

Tempat dan Waktu Penelitian

Berikut adalah penjelasan tentang tempat dan waktu penelitian yang dilakukan dalam penerapan *Hadoop Cluster* untuk proses pencarian sequens data Genbank:

Tempat Penelitian : Universitas Al Azhar Indonesia
Waktu Penelitian : Februari 2013 – Mei 2013
Alamat : Komplek Masjid Agung Al Azhar, Jakarta

Bahan Penelitian

Bahan yang diperlukan untuk melakukan penelitian tentang *Hadoop* ini adalah data Genbank yang terdapat di FTP NCBI, yaitu ftp://ftp.ncbi.nih.gov/genbank. Data Genbank yang diambil adalah Gbbct1.seq.gz sampai dengan Gbbct100.seq.gz sebesar 6 GB [7].

Perangkat Penelitian

Perangkat dalam penelitian ini meliputi *software* dan *hardware* yang digunakan untuk pembuatan infrastrukstur *Hadoop Cluster* [8][9].

1. *Hardware*
   a. *Master*
      - *Dell Power Edge R200*
      - *Intel Xeon Quad Core X3210 @ 2.13 GHz*
      - *VGA Controller ATI ES1000 32MB Memory*
      - *RAM 4GB 667MHz*
      - *Hard disk drive Seagate 250GB 7200 rpm 6Gbps*
      - *Network Card Broadcom NetXtreme BCM5721 Gigabit Ethernet Adapter*
      - *DVD-ROM OEM*
   b. *Slave*

      Slave menggunakan 6 PC dengan spesifikasi yang sama, yaitu:
      - *Intel Pentium Dual Core E2160 @ 1.8 GHz*
      - *VGA Controller SiS771/671*
      - *RAM 1GB*
      - *Hard disk drive Maxtor 80GB 7200 rpm 6Gbps*
      - *Network Card SiS191 Gigabit Ethernet Adapter*
      - *DVD-ROM Samsung*
   c. *Network*
      - *Switch 16 port 10/100 Mbps D-Link DES-1016D*
      - *UTP Cable Belden Cat5e*
2. *Software*
   a. *Master*
      - *Ubuntu 9.04 LTS Karmic Koala*
      - *Kernel 2.6.31-20-generic*
      - *Sun Java 6 JDK*
      - *Hadoop 1.0.4*
      - *Firewall Shorewall*
      - *OpenSSH-Server*
   b. *Slave*
      - *Ubuntu 10.04 LTS Lucid Lynx*
      - *Kernel 2.6.32.38-generic*
      - *Openjdk 6 JDK*
      - *Hadoop 1.0.4*
      - *OpenSSH-Client*

Spesifikasi minimal untuk *Hadoop datanode* sebenarnya lebih tinggi dari yang dipakai untuk penelitian ini, namun karena ini untuk kepentingan pendidikan dan memanfaatkan komputer yang ada pada Lab Jaringan di Universitas Al Azhar Indonesia serta bukan untuk kepentingan komersial, maka spesifikasi yang ada tidak sesuai dengan yang tertera pada *Minimum Requirement* untuk instalasi *Hadoop*, yaitu [10]:

1. *Processor, 2 quad-core 2 – 2.5 GHz CPU*
2. *Memory, 16-24 GB ECC RAM*
3. *Storage, 4x 1TB SATA Hard disk drive*
4. *Network, Gigabit Ethernet*

EVALUASI KINERJA HDFS

Pengujian ini meliputi pengukuran waktu menggunakan aplikasi pencarian sequens genbank untuk mendukung sistem *Hadoop Cluster* yang

telah dibuat dalam penelitian ini dan dibandingkan dengan aplikasi pencarian sequens data genbank menggunakan *Local Filesystem*.

Skenario Eksperimen

Skenario eksperimen pada penelitian ini adalah pencarian sequens data Genbank sebesar 6 GB dengan menggunakan HDFS pada *Hadoop Cluster* dan sebagai perbandingan menggunakan *Local Filesystem* pada satu komputer.

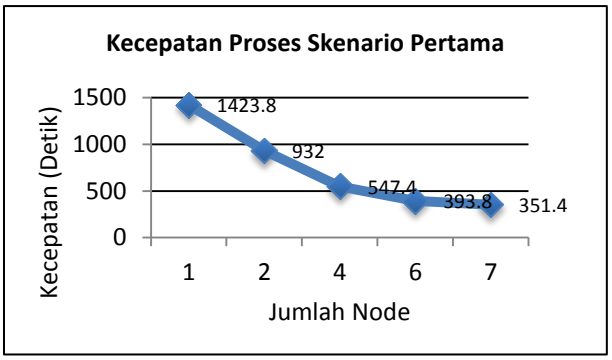
Eksperimen ini melakukan pencarian sequens data genbank yang telah ditentukan dengan menggunakan *Hadoop Cluster* yang telah dikonfigurasi sebelumnya, yaitu:

1. Mencari *Authors* bernama *Harano* dengan *output* yang diinginkan berupa *field Locus, Definition*, dan *Authors*
2. Mencari *Authors* bernama *Harano* dengan *output* yang diinginkan berupa *field Locus, Authors,* dan *Origin.*
3. Mencari *Authors* bernama *Harano* dengan *output* yang diinginkan berupa semua *field* maupun sub-*field* pada data Genbank, yaitu: *Locus, Definition, Accesion, Version, Keywords, Source, Organism, Reference, Authors, Title, Journal, Pubmed, Comment, Features,* dan *Origin.*

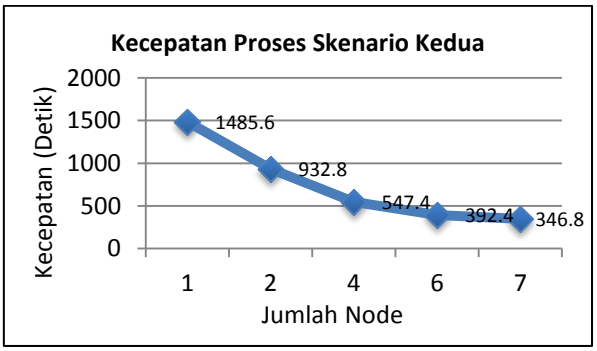
Pada eksperimen ini menggunakan *Local Filesystem* pada satu komputer. *Local Filesystem* sudah terdapat data Genbank yang telah diunduh sebelumnya menggunakan program *Wget* dan ini merupakan data yang sama dengan yang digunakan oleh *Hadoop Cluster*. Pada data Genbank dilakukan pencarian berupa *Authors* bernama *Harano* dengan *output* yang diinginkan berupa semua *field* maupun *sub-field* pada data Genbank, yaitu: *Locus, Definition, Accesion, Version, Keywords, Source, Organism, Reference, Authors, Title, Journal, Pubmed, Comment, Features,* dan *Origin.* Eksperimen ini dilakukan sebagai perbandingan *Local Filesystem* dalam satu komputer dengan HDFS pada *Hadoop Cluster*.

Evaluasi Kinerja HDFS dan LFS

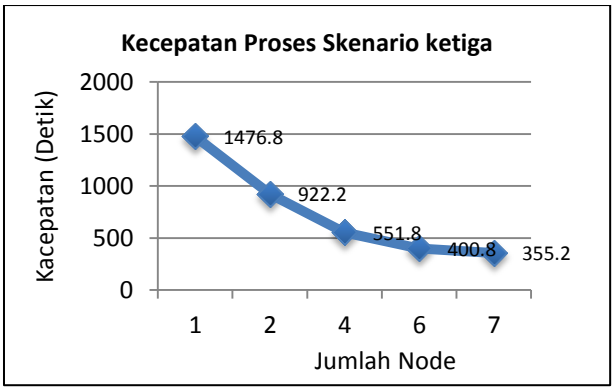
Untuk mengetahui rata-rata waktu yang dibutuhkan pada skenario eksperimen pertama, kedua, dan ketiga terdapat pada gambar 1, 2 dan 3.



Gambar 1 Kecepatan Proses Skenario Pertama

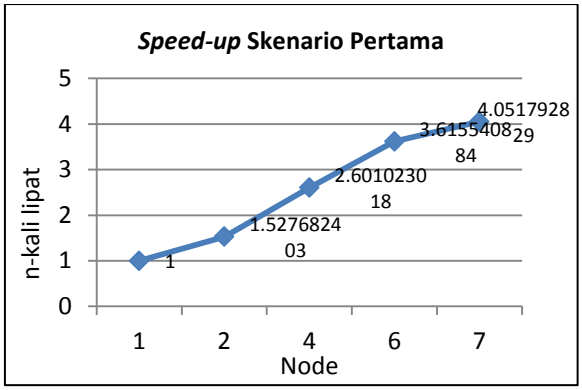


Gambar 2 Kecepatan Proses Skenario kedua

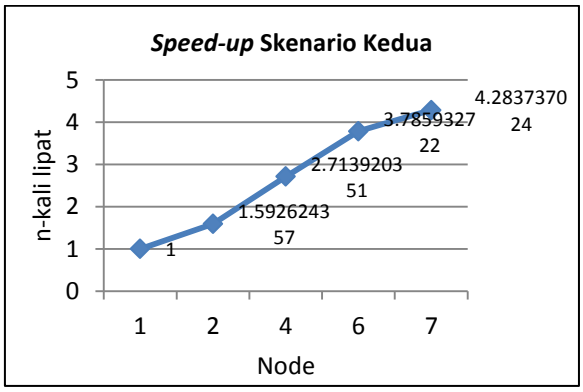


Gambar 3 Kecepatan Proses Skenario Ketiga

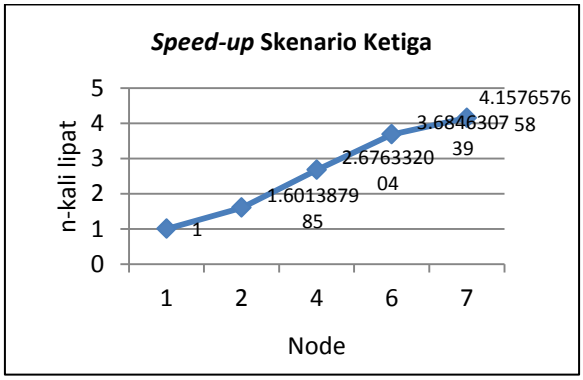
Pada gambar 1, 2 dan 3 terlihat bahwa terjadi penurunan waktu yang dibutuhkan untuk melakukan pencarian sequens data genbank menggunakan 1 *node*, 2 *node*, 4 *node*, 6 *node*, dan 7 *node*. Sedangkan untuk perbandingan dari *speed-up* dari skenario eksperimen pertama dapat dilihat pada gambar 4, 5 dan 6.



Gambar 4 Speed-up Skenario Pertama



Gambar 5 Speed-up Skenario Kedua



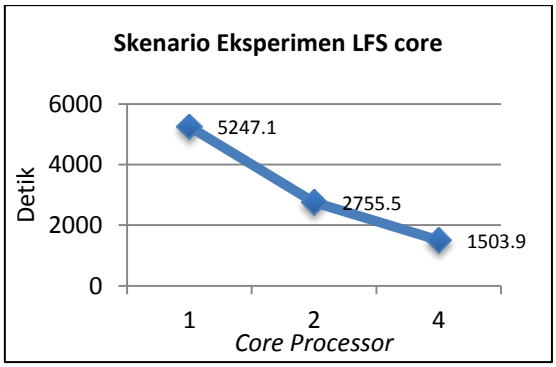
Gambar 6 Speed-up Skenario Ketiga

Speed-up merupakan perbedaan waktu yang dibutuhkan dari satu node dengan node yang lain. Penggunaan 7 *node* 4,05x lebih cepat dibandingkan dengan penggunaan 1 *node*. Walaupun secara teoritis seharusnya penggunaan 7 *node* dapat lebih cepat 7x lipat dibandingkan dengan penggunaan 1 *node*. Hal ini kemungkinan terjadi diakibatkan beberapa faktor, diantaranya:

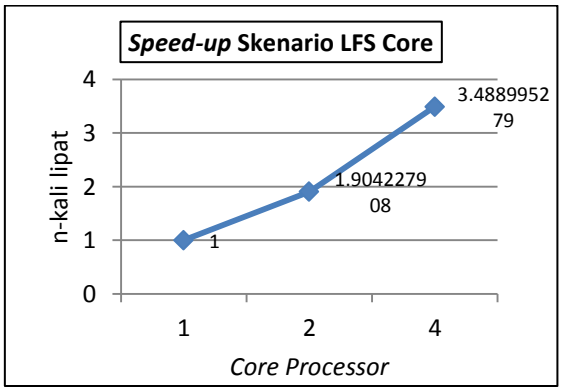
a. *Overhead* yang terjadi pada proses *input/output* (I/O) antara prosesor dengan memori dan dengan HDFS
b. *Overhead* yang terjadi pada proses I/O antara *master* dengan *node* saat *namenode* melalui *jobtracker* memberikan delegasi pekerjaan kepada *datanode* melalui *tasktracker*
c. Penggunaan *Switch* yang berspesifikasi 10/100 Mbps, bukan *Gigabit Switch*
d. Menggunakan spesifikasi komputer yang dibawah *minimum requirement* dari *Hadoop Cluster*
e. Menggunakan kabel *UTP cat5e*, bukan *UTP cat6* untuk spesifikasi *Gigabit Network*

Sebagai perbandingan HDFS, evaluasi juga diperlukan pada penggunaan *Local Filesystem*. Pada skenario eksperimen menggunakan konfigurasi penggunaan sebanyak 1, 2, dan 4 *core processor* Untuk mengetahui rata-rata waktu yang dibutuhkan saat menggunakan *Local Filesystem*, terlihat pada gambar 7.

Pada gambar 7 terlihat bahwa terjadi penurunan waktu yang dibutuhkan untuk melakukan pencarian sequens data genbank menggunakan 1 *core*, 2 *core*, dan 4 *core processor*. Sedangkan untuk perbandingan dari *speed-up* dari skenario eksperimen ketiga dapat dilihat pada gambar 8.



Gambar 7 Skenario Eksperimen LFS Core



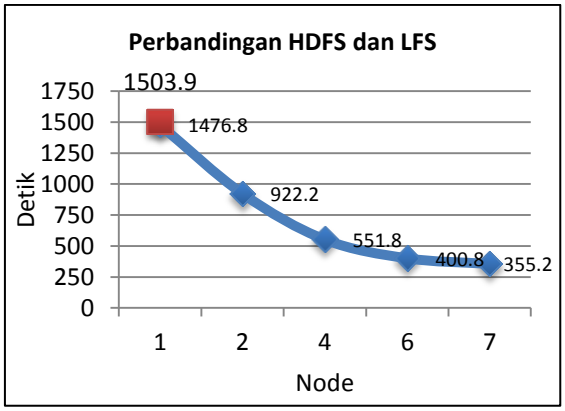
Gambar 8 Speed-up Skenario LFS Core

Speed-up pada skenario LFS *Core* merupakan perbedaan waktu yang dibutuhkan dari satu core dengan core yang lain. Penggunaan 4 *core* 3.4x lebih cepat dibandingkan dengan penggunaan 1 *core processor*. Walaupun secara teoritis seharusnya penggunaan 4 *core* dapat lebih cepat 4x lipat dibandingkan dengan penggunaan 1 *core*. Hal ini kemungkinan terjadi diakibatkan beberapa faktor, diantaranya:

a. *Overhead* yang terjadi pada proses I/O antara prosesor dengan memori
b. *Overhead* yang terjadi pada proses I/O antara memori dengan *Local Filesystem*
c. Kemampuan baca/tulis dari *Local Filesystem* yang terbatas

Evaluasi perbandingan kinerja HDFS dan LFS

Setelah semua skenario telah dilakukan dan di evaluasi, maka akan terlihat bahwa penggunaan HDFS dengan 7 *node* 4.2x lebih cepat dibandingkan dengan penggunaan *Local Filesystem* dengan 4 *core processor*. Hal ini dapat dilihat pada gambar 9.



Gambar 9 Perbandingan HDFS dan LFS

Pada penggunaan HDFS dengan 1 *node* 1,8% atau 1.02x lebih cepat dibandingkan dengan LFS yang menggunakan 4 *core processor*. Ini kemungkinan diakibatkan karena HDFS dengan 1 *node* juga menggunakan 4 *core processor*.

KESIMPULAN DAN SARAN

Berdasarkan implementasi eksperimen dan evaluasi yang telah dilakukan terhadap pencarian sequens data Genbank menggunakan *Hadoop Distributed* Filesystem, maka dapat diperoleh kesimpulan sebagai berikut:

1. *Hadoop Cluster* berhasil diimplementasikan dalam Laboratorium Jaringan Universitas Al Azhar Indonesia menggunakan 1 komputer sebagai *master namenode* dan 7 komputer sebagai *slave datanode* (termasuk *master namenode* didalamnya merangkap sebagai *datanode*) dengan topologi jaringan yang digunakan adalah topologi *star*.
2. Data Genbank dapat diunduh melalui FTP menggunakan program *Wget* dan berhasil diunggah ke dalam HDFS.
3. Besar jumlah data berbanding lurus dengan waktu yang dibutuhkan untuk menyelesaikan proses pencarian sequens data Genbank. Semakin besar data yang diproses, maka akan semakin lama waktu yang dibutuhkan untuk mendapatkan hasil dari proses pencarian dengan menggunakan konfigurasi dan spesifikasi yang sama.
4. *Hadoop Cluster* dapat digunakan dengan *single node* ataupun *multi node*, disesuaikan dengan eksperimen yang telah ditentukan sebelumnya.
5. Waktu yang dibutuhkan untuk melakukan proses pencarian sequens data Genbank menggunakan 7 *node* rata-rata sebesar 351 detik dan rata-rata sebesar 1462 detik menggunakan 1 *node*.
6. Proses pencarian data menggunakan 7 *node* adalah rata-rata 4x lebih cepat dibandingkan dengan menggunakan 1 *node* tergantung dari *output* yang diinginkan oleh *user*.
7. Proses pencarian menggunakan HDFS dengan 1 *node* memiliki perbedaan sebesar 1.8% dibandingkan dengan menggunakan LFS dengan 4 *core processor*.

Adapun saran untuk pengembangan selanjutnya dari tugas akhir ini adalah:

1. Penggunaan spesifikasi komputer dan perangkat jaringan yang sesuai atau melebihi dari kebutuhan minimum untuk melakukan implementasi *Hadoop Cluster* sehingga dalam melakukan pemrosesan suatu data dapat berjalan lebih cepat.
2. Pengguna dapat melakukan proses pencarian dimanapun dan kapanpun diperlukan dengan menggunakan *VPN (Virtual Private Network)*, *port forwarding*, atau *firewall masquerading*.
3. Melakukan pengunggahan data ke dalam HDFS tidak dalam *format file compressed* untuk mengurangi waktu yang dibutuhkan akibat proses *uncompressed*.

DAFTAR PUSTAKA

[1] I. Chabib, Pengembangan *Database* Genbank UAI-Bioinformatics [skripsi], Jurusan Teknik Informatika Universitas Al Azhar Indonesia, 2013.

[2] A. Hisyam, Analisis Dan Pengembangan Uai-Bioinformatics Dengan Proses Pengolahan Data Terdistribusi Menggunakan *Framework Hadoop* [skripsi], Jurusan Teknik Informatika Universitas Al Azhar Indonesia, 2013

[3] J. Venner, Pro Hadoop, New York: Apress, 2009.

[4] T. White, Hadoop: The Definitive Guide, Third Edition, United State of America: O'Reilly, 2012.

[5] E. Sammer, Hadoop Operations, United State of America: O'Reilly, 2012.

[6] Komput@si. "Komputasi Terdistribusi", 2010. [online].Available:http://www.komputasi.lipi.go.id/utama.cgi?cetakartikel&1271412582 [Accessed 16 Mei 2013 11.52 AM].

[7] D.A. Benson, I. K. Mizrachi, K. Clarck, D. J. Lipman, J. Ostell, E. W. Sayers, "Genbank", Nucleic Acids Research, Vol. 40, *Database* issue, 2012, D48-D53, 2012.

[8] M. G. Noll, "*Running Hadoop* On Ubuntu Linux (*Single-Node Cluster*)", 2013. [online]. Available:http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/ [Accessed 16 Mei 2013 11.53 AM].

[9] M. G. Noll, "*Running Hadoop* On Ubuntu Linux (*Multi-Node Cluster*)", 2013. [online]. Available: http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/ [Accessed 16 Mei 2013 11.53 AM].

[10] Laboratorium Sistem Komputer Lanjut, Jaringan Komputer Dasar, Jakarta: Universitas Gunadarma, 2012.