



TUGAS AKHIR – KM4801

**PENGEMBANGAN PROTOTIPE BIG DATA
PADA EKOSISTEM HADOOP
MENGUNAKAN HDFS DAN MAPREDUCE
SEBAGAI MODEL KOMPUTASI PARALEL
(STUDI KASUS : SAMPEL SISTEM
TRANSPORTASI KOTA SURABAYA)**

**M. FIAN FACHRY A.
NRP 0611144000068**

**Dosen Pembimbing
Dr. Budi Setiyono, S.Si, MT**

**DEPARTEMEN MATEMATIKA
Fakultas Matematika, Komputasi, dan Sains Data
Institut Teknologi Sepuluh Nopember
Surabaya 2019**



FINAL PROJECT – KM4801

**BIG DATA PROTOTYPE DEVELOPMENT
ON HADOOP ECOSYSTEM
USING HDFS AND MAPREDUCE
AS THE PARALLEL COMPUTATION MODEL
(CASE STUDY : SAMPLE OF INTELLIGENT
TRANSPORTATION SYSTEM OF SURABAYA
CITY)**

**M. FIAN FACHRY A.
NRP 0611144000068**

**Supervisor
Dr. Budi Setiyono, S.Si, MT**

**DEPARTMENT OF MATHEMATICS
Faculty of Mathematics, Computation, and Data Science
Institut Teknologi Sepuluh Nopember
Surabaya 2019**

1


LEMBAR PENGESAHAN

PENGEMBANGAN PROTOTIPE BIG DATA PADA EKOSISTEM
HADOOP MENGGUNAKAN HDFS DAN MAPREDUCE
SEBAGAI MODEL KOMPUTASI PARALEL
(STUDI KASUS :
SAMPel SISTEM TRANSPORTASI CERDAS KOTA SURABAYA)


Diajukan untuk memenuhi salah satu syarat
untuk memperoleh gelar Sarjana Sains pada
Bidang studi Ilmu Komputer
Program Studi S-1 Departemen Matematika
Fakultas Matematika Komputasi dan Sains Data
Institut Teknologi Sepuluh Nopember Surabaya

Oleh :
M. FIAN FACHRY A.
NRP. 06111440000368

Menyetujui
Dosen Pembimbing,


Dr. Budi Setiyono, S.Si, MT
NIP. 19720207 199702 1 001

Mengetahui,
Kepala Departemen Matematika


Dr. Inam Mukhlash, S.Si, M.T
NIP. 19700831 199403 1 003
Surabaya, Januari 2019

**PENGEMBANGAN PROTOTIPE BIG DATA PADA
EKOSISTEM HADOOP MENGGUNAKAN HDFS DAN
MAPREDUCE SEBAGAI MODEL KOMPUTASI PARALEL
(STUDI KASUS : SAMPEL SISTEM TRANSPORTASI KOTA
SURABAYA)**

Nama Mahasiswa : M. Fian Fachry A.
NRP : 0611144000068
Departemen : Matematika
Dosen Pembimbing : Dr. Budi Setiyono, S.Si, MT

Abstrak

Dengan perkembangan sosial yang begitu cepat, industri transportasi juga menghadapi tantangan yang belum pernah dihadapi sebelumnya. Salah satu daerah yang mengalami dampak tersebut adalah Kota Surabaya, yaitu tantangan pertumbuhan jumlah kendaraan pribadi yang begitu pesat. Maka dari itu dibutuhkan suatu sistem transportasi umum yang memadai, yaitu dengan menyediakan rute yang sesuai dengan kebutuhan masyarakat Kota Surabaya. Dengan mengamati jalur perjalanan kendaraan yang berlalu – lalang di Kota Surabaya, akan diketahui preferensi jalur perjalanan yang dilalui oleh masyarakat, sehingga dapat dijadikan acuan untuk pemerintah Kota Surabaya, khususnya Dinas Perhubungan, sebagai dasar membuka rute transportasi umum dalam Kota Surabaya. Dengan memanfaatkan big data, maka data jalur perjalanan kendaraan dapat diolah sehingga mendapatkan hasil jalur perjalanan kendaraan yang paling sering dilalui oleh masyarakat Kota Surabaya. Big data tersebut diterapkan pada ekosistem Hadoop dengan menggunakan HDFS dan MapReduce sebagai model komputasi paralel. Model komputasi paralel digunakan karena data yang diolah berjumlah besar, sehingga akan lebih efisien dari model komputasi sekuensial. Dikarenakan area Kota Surabaya yang begitu luas dan prasarana untuk mengambil data yang dibutuhkan kurang memadai, sehingga penelitian ini dilakukan pada sampel 5 titik CCTV, yaitu Jl. Kayoon, Jl. Panglima Sudirman, Jl. Urip Sumoharjo, Keputran, dan Jl. Raya Ngagel. Berdasarkan hasil pengolahan data pada 5 titik

CCTV tersebut, didapatkan bahwa jalur perjalanan kendaraan yang paling sering dilalui adalah jalur perjalanan dari Jl. Panglima Sudirman menuju Jl. Urip Sumoharjo.

Kata kunci : big data, sistem transportasi, implementasi, teknologi, transportasi umum, hadoop, HDFS, MapReduce

**BIG DATA PROTOTYPE DEVELOPMENT ON HADOOP
ECOSYSTEM USING HDFS AND MAPREDUCE AS THE
PARALLEL COMPUTATION MODEL
(CASE STUDY : SAMPLE OF INTELLIGENT
TRANSPORTATION SYSTEM OF SURABAYA CITY)**

Name of Student : M. Fian Fachry A.
NRP : 0611144000068
Department : Mathematics
Supervisor : Dr. Budi Setiyono, S.Si, MT

Abstract

Because of the rapid social development, the transportation industry also faces challenges that have never been faced before. One of the region that experienced this impact is the Surabaya City, that is the challenge of the rapid growth of the number of private vehicles. Therefore, an adequate public transportation system is needed, that is by providing a route that fits the needs of the society of Surabaya City. By observing the travel path of passing vehicles in the Surabaya City, the route path preferences traveled by the society will be known, so that it can be used as a reference for the Surabaya City government, especially the Department of Transportation, as the basis for opening public transportation routes in Surabaya City. By utilizing big data, the vehicle path data can be processed so that the results of the vehicle paths that are most frequently traveled by the society of Surabaya City. Big data is applied to the Hadoop ecosystem using HDFS and MapReduce as the parallel computation model. The parallel computation model is used because the processed data is large, so it will be more efficient than the sequential computing model. Due to the vast area of the Surabaya City and the infrastructure to acquire the data needed is not sufficient, so this research was conducted on samples of 5 CCTVs points, those are Kayoon street, Panglima Sudirman street, Urip Sumoharjo street, Keputran, and Raya Ngagel street. Based on the results of data processed at the 5 CCTVs points, it was found that the

vehicle's most frequently traveled route was the travel route from Panglima Sudirman street headed for Urip Sumoharjo street.

Key words : big data, transportation system, implementation, technology, public transportation, hadoop, HDFS, MapReduce

KATA PENGANTAR

Segala puji syukur penulis panjatkan ke hadirat Allah SWT, karena dengan ridho-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul

“PENGEMBANGAN PROTOTIPE BIG DATA PADA EKOSISTEM HADOOP MENGGUNAKAN HDFS DAN MAPREDUCE SEBAGAI MODEL KOMPUTASI PARALEL (STUDI KASUS : SAMPEL SISTEM TRANSPORTASI CERDAS KOTA SURABAYA)”

Merupakan salah satu persyaratan akademis dalam menyelesaikan Program Sarjana Departemen Matematika, Fakultas Matematika Komputasi dan Sains Data, Institut Teknologi Sepuluh Nopember Surabaya.

Tugas Akhir ini dapat diselesaikan dengan baik berkat kerja sama, bantuan, dan dukungan dari banyak pihak. Sehubungan dengan hal tersebut, penulis ingin mengucapkan terimakasih kepada :

1. Dr. Imam Mukhlas, S.Si, MT selaku Kepala Departemen Matematika ITS.
2. Dr. Dra. Mardlijah, MT selaku Dosen Wali yang telah memberikan arahan akademik selama penulis menempuh pendidikan di Departemen Matematika ITS.
3. Dr. Budi Setiyono, S.Si, MT selaku Dosen Pembimbing yang telah memberikan bimbingan dan motivasi kepada penulis dalam mengerjakan Tugas Akhir ini sehingga dapat terselesaikan dengan baik.
4. Dr. Didik Khusnul Arif, S.Si, M.Si selaku ketua program studi S1 Departemen Matematika ITS.
5. Drs. Iis Herisman, M.Si selaku Sekretaris Program Studi S1 Departemen Matematika ITS.
6. Seluruh jajaran dosen dan staf Departemen Matematika ITS.
7. Orang tua dan kedua adik penulis yang senantiasa memberikan dukungan dan do'a yang tak terhingga.

8. Semua pihak yang tak bisa penulis sebutkan satu-persatu, terima kasih telah membantu sampai terselesaikannya Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih jauh dari kesempurnaan. Oleh karena itu, penulis mengharapkan kritik dan saran dari pembaca. Akhir kata, semoga Tugas Akhir ini dapat bermanfaat bagi semua pihak yang berkepentingan.

Surabaya, Januari 2019

Penulis

DAFTAR ISI

KATA PENGANTAR.....	xi
DAFTAR ISI	xiii
DAFTAR GAMBAR.....	xv
DAFTAR TABEL	xvi
BAB I. PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan	3
1.5 Manfaat	3
1.6 Sistematikan Penulisan Tugas Akhir	3
BAB II. TINJAUAN PUSTAKA	5
2.1. Big Data	5
2.2. Hadoop.....	7
2.2.1. MapReduce	9
2.2.2. HDFS	11
2.3. Implementasi Big Data pada <i>ITS</i>	12
2.4. Kondisi Sistem Lalu Lintas Cerdas Kota Surabaya	12
BAB III. METODE PENELITIAN	15
3.1 Tahapan Penelitian	15
3.2 Objek dan Aspek Penelitian	16
3.3 Blok Diagram.....	16
BAB IV. PERANCANGAN DAN IMPLEMENTASI SISTEM	19
4.1 Perancangan Sistem	19
4.1.1. Perancangan Data	19
4.1.2. Perancangan Blok Diagram Sistem	19
4.1.2.1. Akuisisi Data (Pra-Proses).....	20
4.1.2.2. Arsitektur Sistem	27
4.1.2.3. Perancangan Model Paralel.....	28
4.1.2.4. Perancangan Struktur Data dan Penyimpanan Data pada HDFS.....	30
4.1.2.5. Perancangan Fungsi Mapper	31
4.1.2.6. Perancangan Fungsi Reducer	36
4.1.3. Perancangan <i>Class Diagram</i>	39

4.2 Implementasi Sistem	40
4.2.1. Implementasi Struktur Data dan Penyimpanan Data pada HDFS	40
4.2.2. Implementasi Fungsi Mapper	41
4.2.3. Implementasi Fungsi Reducer	45
4.2.4. Running Program.....	45
BAB V. HASIL UJI COBA DAN PEMBAHASAN	49
5.1 Hasil Akuisisi Data	49
5.2 Hasil Uji Coba Sistem.....	49
5.3 Analisa Hasil Pengolahan Data.....	50
BAB VI. PENUTUP.....	53
6.1 Kesimpulan	53
6.2 Saran	53
DAFTAR PUSTAKA.....	55
LAMPIRAN	57

DAFTAR GAMBAR

Gambar 2.1.	Pertumbuhan dan digitalisasi kapasitas penyimpanan informasi global.....	6
Gambar 2.2.	Sebuah cluster multi-node Hadoop.....	8
Gambar 2.3.	Contoh MapReduce kanonik menghitung kemunculan setiap kata dalam sekumpulan dokumen	10
Gambar 2.4.	Arsitektur HDFS	11
Gambar 3.1.	Blok diagram penelitian.....	17
Gambar 4.1.	Blok Diagram Sistem.....	20
Gambar 4.2.	Peta Titik CCTV yang Diamati	21
Gambar 4.3.	Tampilan Masing - Masing CCTV yang Diamati..	21
Gambar 4.4.	Sketsa Titik CCTV yang Diamati	22
Gambar 4.5.	Kendaraan k1 Terbaca pada CCTV 1	23
Gambar 4.6.	Kendaraan k1 Terbaca pada CCTV 2	24
Gambar 4.7.	Kendaraan k1 Terbaca pada CCTV 3	24
Gambar 4.8.	Kendaraan k2 Terbaca pada CCTV 2	25
Gambar 4.9.	Kendaraan k2 Terbaca pada CCTV 3	25
Gambar 4.10.	Kendaraan k2 Terbaca pada CCTV 4	25
Gambar 4.11.	Kendaraan k2 Terbaca pada CCTV 5	25
Gambar 4.12.	Kendaraan k3 Terbaca pada CCTV 2	26
Gambar 4.13.	Kendaraan k4 Terbaca pada CCTV 2	26
Gambar 4.14.	Kendaraan k4 Terbaca pada CCTV 5	27
Gambar 4.15.	Arsitektur Sistem.....	27
Gambar 4.16.	Grafik Model Paralel	28
Gambar 4.17.	Grafik Model Paralel	29
Gambar 4.18.	Grafik Model Paralel	29
Gambar 4.19.	Struktur Data Input untuk HDFS	30
Gambar 4.20.	<i>Flowchart</i> fungsi <i>mapper</i>	35
Gambar 4.21.	<i>Flowchart</i> fungsi <i>reducer</i>	37
Gambar 4.22.	Ilustrasi tahap 4.1.2.5. – 4.1.2.6.....	38
Gambar 4.23.	<i>Class</i> diagram	39
Gambar 4.24.	Pembuatan Direktori pada HDFS	40
Gambar 4.25.	Direktori pada HDFS	40
Gambar 4.26.	Input Data ke Dalam HDFS	41
Gambar 4.27.	Hasil Input Data pada HDFS	41

Gambar 4.28.	Running Program MapReduce.....	45
Gambar 5.1.	Menampilkan Output dari Program	49
Gambar 5.2.	Output Program.....	49
Gambar 5.3.	Analisa Hasil Pengamatan	52

DAFTAR TABEL

Tabel 4.1. Rata – Rata Waktu Perpindahan antar CCTV	23
Tabel 4.2. Analisa Hasil Pengamatan	51

BAB I PENDAHULUAN

Pada bab ini akan dibahas mengenai latar belakang yang mendasari penulisan Tugas Akhir. Di dalamnya mencakup identifikasi permasalahan pada topik Tugas Akhir kemudian dirumuskan menjadi permasalahan yang selanjutnya diberikan batasan-batasan dalam pembahasan pada Tugas Akhir ini.

1.1. Latar Belakang

Dengan perkembangan sosial yang begitu cepat, industri transportasi juga menghadapi tantangan yang belum pernah dihadapi sebelumnya. Salah satunya adalah pertumbuhan jumlah kendaraan pribadi yang begitu pesat.[1] Ditambah lagi dengan tidak tersedianya sistem transportasi umum yang cukup memadai kebutuhan sosial masyarakatnya, maka tantangan tersebut semakin menjadi besar. Salah satu faktor penyebab sistem transportasi umum yang kurang memadai kebutuhan sosial masyarakat adalah penentuan rute atau trayek transportasi umum yang kurang sesuai dengan kebutuhan mobilitas masyarakat.

Maka dari itu, dibutuhkan suatu teknologi yang mampu merumuskan suatu sistem yang dapat mengatasi permasalahan penentuan rute transportasi umum. Dengan memanfaatkan *Closed Circuit Television* (CCTV) yang telah tersebar di beberapa ruas persimpangan jalan di Kota Surabaya, penulis telah melakukan pemantauan jalur perjalanan pada setiap kendaraan bermotor yang melewati beberapa titik CCTV di Kota Surabaya. Dikarenakan teknologi yang digunakan oleh CCTV tersebut masih belum memadai untuk membaca jalur perjalanan suatu kendaraan maupun hanya untuk membaca plat nomor suatu kendaraan secara otomatis, maka pengenalan kendaraan beserta jalur perjalanannya dilakukan secara manual.

Pada Tugas Akhir ini, penulis telah mengimplementasikan big data pada data jalur perjalanan kendaraan yang lalu-lalang di ruas jalan beberapa CCTV yang telah dipantau sebelumnya. Menurut Badan Pusat Statistik, pada tahun 2015, terdapat 2.126.168

kendaraan bermotor yang berada di Kota Surabaya,[2] sehingga dibutuhkan suatu implementasi big data yang dapat mengolah jumlah volume data yang begitu besar. Implementasi big data ini diterapkan pada ekosistem Hadoop, dimana HDFS sebagai sistem file untuk penyimpanan big data dan MapReduce sebagai model komputasi paralel. Karena data yang diolah cukup besar, maka penulis menggunakan model komputasi paralel untuk mengolah data tersebut, agar pemrosesan data dapat dilakukan secara jauh lebih efisien dibandingkan model komputasi biasa (sekuensial).

Diharapkan dengan diterapkannya big data pada sistem transportasi ini, dapat digunakan sebagai dasar bagi pemerintah Kota Surabaya dalam pengambilan keputusan untuk pembukaan rute transportasi umum.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang sudah diuraikan, pokok permasalahan yang dikaji dalam tugas akhir ini adalah :

1. Bagaimana implementasi data transportasi ke dalam ekosistem Hadoop?
2. Bagaimana mengolah data jalur perjalanan kendaraan bermotor pada ekosistem Hadoop?
3. Bagaimana menentukan urutan jalur perjalanan terbanyak yang dilewati oleh kendaraan bermotor?

1.3. Batasan Masalah

Penulisan Tugas Akhir ini difokuskan pada pembahasan dengan beberapa batasan masalah sebagai berikut :

1. Sistem transportasi yang diteliti adalah sistem transportasi di lima titik CCTV di Surabaya Pusat.
2. Kendaraan roda dua, angkutan kota, dan bus kota tidak diperhitungkan di dalam penelitian ini.
3. Framework atau platform yang digunakan pada penelitian ini adalah Hadoop.
4. Bahasa pemrograman yang digunakan adalah Java.

1.4. Tujuan

Berdasarkan rumusan masalah di atas, tujuan dari penulisan Tugas Akhir ini sebagai berikut :

1. Mengimplementasi data transportasi ke dalam ekosistem Hadoop.
2. Mengolah data jalur perjalanan kendaraan bermotor pada ekosistem Hadoop.
3. Menentukan urutan jalur perjalanan terbanyak yang dilewati oleh kendaraan bermotor.

1.5. Manfaat

Berdasarkan rumusan masalah di atas, tujuan dari penulisan Tugas Akhir ini sebagai berikut :

1. Memberikan pengetahuan bagi pembaca mengenai implementasi big data pada ekosistem Hadoop.
2. Memberikan masukan terhadap pemerintah Kota Surabaya, khususnya Dinas Perhubungan mengenai efektifitas rute transportasi umum Kota Surabaya.
3. Memberikan informasi untuk digunakan dalam riset selanjutnya.

1.6. Sistematika Penulisan Tugas Akhir

Sistematika dari penulisan Tugas Akhir ini adalah sebagai berikut :

1. BAB I PENDAHULUAN

Bab ini menjelaskan tentang gambaran umum dari penulisan Tugas Akhir ini yang meliputi latar belakang masalah, perumusan masalah, batasan masalah, tujuan, manfaat penelitian, dan sistematika penulisan.

2. BAB II TINJAUAN PUSTAKA

Bab ini berisi tentang materi-materi yang mendukung Tugas Akhir ini, antara lain big data, hadoop, MapReduce, HDFS, dan Kondisi Sistem Lalu Lintas Cerdas Kota Surabaya.

3. BAB III METODE PENELITIAN

Pada bab ini dibahas tentang langkah – langkah dan metode yang digunakan untuk menyelesaikan Tugas Akhir.

4. **BAB IV PERANCANGAN DAN IMPLEMENTASI SISTEM**

Pada bab ini akan menguraikan bagaimana perancangan dan tahapan tahapan dalam implementasi sistem, yaitu.

5. **BAB V HASIL UJI COBA DAN PEMBAHASAN**

Bab ini menjelaskan mengenai hasil dari sistem yang telah dibuat disertai dengan penjelasan pembahasan.

6. **BAB VI PENUTUP**

Bab ini berisi kesimpulan yang diperoleh dari pembahasan masalah sebelumnya serta saran yang diberikan untuk pengembangan selanjutnya.

BAB II TINJAUAN PUSTAKA

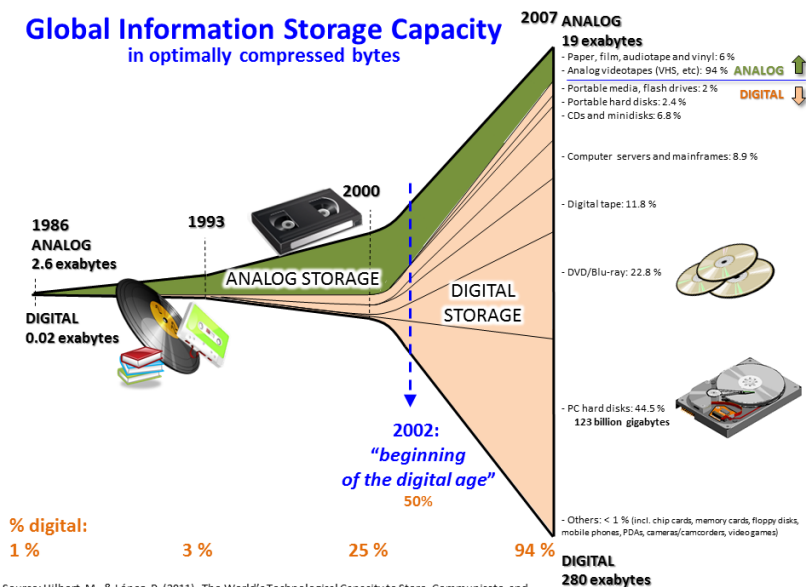
Pada bab ini dibahas mengenai dasar teori yang digunakan dalam penyusunan Tugas Akhir ini. Dasar teori yang dijelaskan adalah big data, Hadoop, MapReduce, HDFS, dan kondisi sistem lalu lintas cerdas Kota Surabaya.

2.1. Big Data

Big data adalah istilah yang digunakan untuk merujuk pada studi dan aplikasi dari himpunan data yang besar dan kompleks dimana aplikasi perangkat lunak pemrosesan data tradisional tidak memadai untuk berurusan dengan himpunan data tersebut. Tantangan big data meliputi penangkapan data, penyimpanan data, analisis data, pencarian, *sharing*, transfer, visualisasi, query, *updating*, privasi informasi dan sumber data. Terdapat beberapa konsep yang berkaitan dengan big data, semula terdapat tiga konsep, yaitu *volume*, *variety* (keragaman data), dan *velocity* (kecepatan pertumbuhan data).[3] konsep lain yang kemudian dikaitkan dengan big data adalah *veracity* (kebenaran atau ketelitian) dan *value* (nilai dari data itu sendiri).[4]

Belakangan ini, istilah big data cenderung merujuk pada penggunaan analisis prediktif, analisis tingkah laku user, atau metode-metode analisis data lanjutan tertentu yang mengekstrak nilai dari data, dan kadang juga diterapkan kepada ukuran himpunan data tertentu. Ada sedikit keraguan bahwa jumlah data yang tersedia sekarang memang besar, tetapi itu bukan karakteristik yang paling relevan dari ekosistem data baru ini. Analisis himpunan data dapat menemukan korelasi baru untuk melihat tren bisnis, mencegah penyakit, memerangi kejahatan dan sebagainya. Para ilmuwan, eksekutif bisnis, praktisi kedokteran, iklan, dan pemerintahan biasanya menemukan kesulitan dengan himpunan data yang besar di berbagai bidang termasuk pencarian Internet, *fintech*, informatika perkotaan, dan informatika bisnis. Para ilmuwan menemukan keterbatasan dalam pekerjaan *e-Science*, termasuk meteorologi,

konektromiks, simulasi fisika kompleks, biologi, dan penelitian lingkungan.[5]



Gambar 2.1. Pertumbuhan dan digitalisasi kapasitas penyimpanan informasi global

Sumber : <http://www.martinhilbert.net/WorldInfoCapacity.html>

Himpunan - himpunan data tumbuh dengan sangat cepat, sebagian karena himpunan - himpunan data tersebut semakin dikumpulkan oleh penginderaan informasi internet yang murah dan beragam dari berbagai gawai, seperti perangkat seluler, aerial, log perangkat lunak, kamera, mikrofon, pembaca Identifikasi Frekuensi Radio (*Radio-Frequency Identification –RFID*), dan jaringan - jaringan sensor nirkabel.[6] Kapasitas per kapita teknologi dunia untuk menyimpan informasi telah meingkat dua kali lipat setiap 40 bulan sejak tahun 1980-an, pada 2012, setiap hari 2,5 exabytes ($2,5 \times 10^{18}$ byte) data dihasilkan.[7] Berdasarkan prediksi laporan IDC, volume data global akan tumbuh secara eksponensial dari 4,4 zettabyte menjadi 44 zettabytes antara 2013 dan 2020. Pada 2025,

IDC memprediksi akan ada 163 zettabytes data. Satu pertanyaan untuk perusahaan besar adalah menentukan siapa yang harus memiliki inisiatif big-data yang mempengaruhi seluruh organisasi.

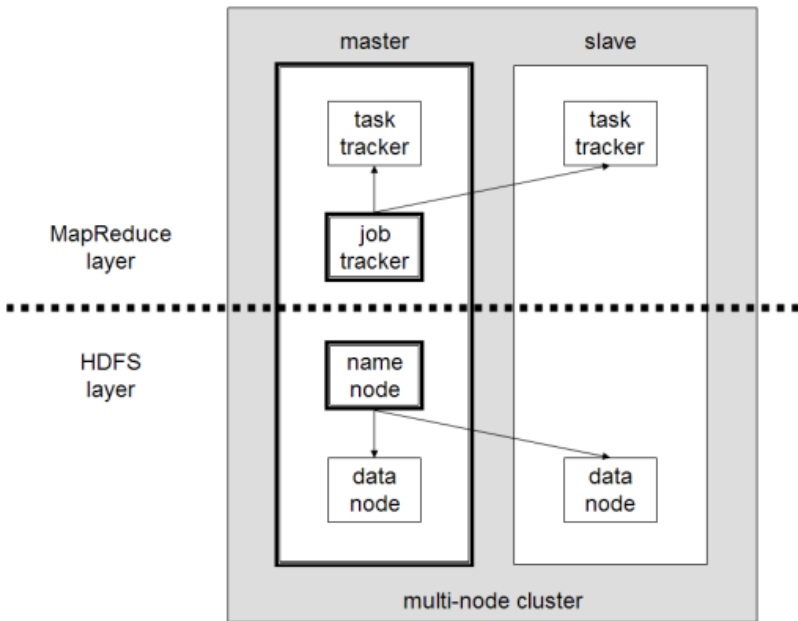
Sistem manajemen basis data relasional, statistik desktop, dan paket perangkat lunak untuk memvisualisasikan data sering mengalami kesulitan menangani big data. Pekerjaan tersebut mungkin memerlukan perangkat lunak paralel besar-besaran yang berjalan pada puluhan, ratusan, atau bahkan ribuan server. Apa yang dianggap sebagai "big data" bervariasi tergantung pada kemampuan *user* dan alat mereka, dan kemampuan yang semakin meluas membuat data besar menjadi target yang bergerak. Untuk beberapa organisasi, menghadapi ratusan gigabyte data untuk pertama kalinya dapat memicu kebutuhan untuk mempertimbangkan kembali opsi manajemen data. Bagi yang lain, mungkin diperlukan puluhan atau ratusan terabyte sebelum ukuran data menjadi pertimbangan yang signifikan.

2.2. Hadoop

Apache Hadoop adalah kumpulan utilitas perangkat lunak *open source* yang memfasilitasi penggunaan jaringan multi komputer untuk menyelesaikan masalah yang sejumlah besar data (atau biasa kita sebut dengan big data) dan komputasi.[8] Utilitas tersebut menyediakan kerangka kerja perangkat lunak untuk penyimpanan terdistribusi dan pengolahan big data menggunakan model pemrograman MapReduce. Awalnya dirancang untuk kelompok komputer yang dibangun dari perangkat keras yang biasa-biasa saja, meskipun tetap digunakan secara umum, namun Apache Hadoop juga telah digunakan pada kelompok perangkat keras kelas atas. Semua modul di Hadoop dirancang dengan asumsi mendasar bahwa kegagalan perangkat keras adalah kejadian umum dan harus ditangani secara otomatis oleh kerangka kerja.

Inti dari Apache Hadoop terdiri dari bagian penyimpanan, yang dikenal sebagai *Hadoop Distributed File System* (HDFS) atau jika diterjemahkan secara harfiah berarti Sistem File Terdistribusi Hadoop, dan bagian pemrosesan yang merupakan model pemrograman MapReduce. Hadoop membagi file menjadi blok-blok

besar dan mendistribusikannya di seluruh node dalam sebuah kelompok, kemudian mentransfer kode yang dikemas ke dalam node untuk memproses data secara paralel. Pendekatan ini mengambil keuntungan dari lokasi data, dimana node-node tersebut memanipulasi data yang mereka akses. Hal ini memungkinkan dataset untuk diproses lebih cepat dan lebih efisien daripada yang ada dalam arsitektur superkomputer yang lebih konvensional yang bergantung pada sistem file paralel di mana komputasi dan data didistribusikan melalui jaringan berkecepatan tinggi.



Gambar 2.2. Sebuah cluster multi-node Hadoop

Kerangka dasar Apache Hadoop terdiri dari modul-modul berikut :

1. *Hadoop Common* - berisi pustaka dan utilitas yang dibutuhkan oleh modul Hadoop lainnya,

2. *Hadoop Distributed File System* (HDFS) - sistem file terdistribusi yang menyimpan data pada komputer biasa, menyediakan *bandwidth* yang sangat tinggi di seluruh kelompok,
3. *Hadoop YARN* - diperkenalkan pada tahun 2012, adalah platform yang bertanggung jawab untuk mengelola sumber daya komputasi dalam setiap kelompok dan menggunakannya untuk menjadwalkan aplikasi para *user*,

Hadoop MapReduce - implementasi dari model pemrograman MapReduce untuk pemrosesan data yang berskala besar.

Istilah Hadoop tidak hanya merujuk pada modul dasar dan sub-modul yang disebutkan di atas, tetapi juga untuk ekosistem atau kumpulan paket perangkat lunak tambahan yang dapat dipasang 'pada' atau 'bersama' Hadoop, seperti Apache Pig, Apache Hive, Apache HBase, Apache Phoenix, Apache Spark, Apache ZooKeeper, Cloudera Impala, Apache Flume, Apache Sqoop, Apache Oozie, dan Apache Storm. Komponen MapReduce dan HDFS dari Apache Hadoop terinspirasi oleh paper Google di MapReduce dan Sistem File Google.

Kerangka Hadoop sendiri sebagian besar ditulis dalam bahasa pemrograman Java, dengan beberapa kode asli dalam C dan utilitas command line ditulis sebagai shell script. Meskipun kode MapReduce Java itu umum, namun bahasa pemrograman apa pun dapat digunakan dengan "Hadoop Streaming" untuk menerapkan bagian "map" dan "reduce" dari program user. Proyek lain dalam ekosistem Hadoop mengekspos User Interface yang lebih beragam.

2.2.1. MapReduce

MapReduce adalah sebuah model pemrograman dan implementasi yang berhubungan untuk memproses dan menghasilkan himpunan data yang besar dengan algoritma yang terdistribusi dan paralel pada sebuah kelompok data. Program MapReduce terdiri dari method 'map', yang melakukan penyaringan dan penyortiran (seperti menyortir siswa dengan nama depan ke dalam antrian, satu antrian untuk setiap nama), method 'reduce', yang melakukan operasi ringkasan (seperti menghitung jumlah siswa di setiap antrian, menghasilkan frekuensi nama). Kerangka kerja

MapReduce mengatur pemrosesan dengan menyusun server yang terdistribusi, menjalankan berbagai tugas secara paralel, mengelola semua komunikasi dan transfer data antara berbagai bagian sistem, dan menyediakan redundansi dan toleransi kesalahan.

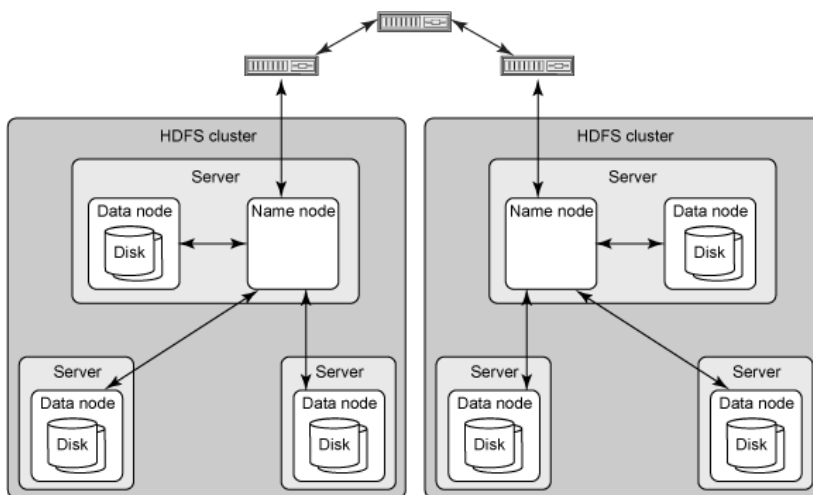
```
function map(String name, String document):  
  // name: document name  
  // document: document contents  
  for each word w in document:  
    emit (w, 1)  
  
function reduce(String word, Iterator partialCounts):  
  // word: a word  
  // partialCounts: a list of aggregated partial counts  
  sum = 0  
  for each pc in partialCounts:  
    sum += pc  
  emit (word, sum)
```

Gambar 2.3. Contoh MapReduce kanonik menghitung kemunculan setiap kata dalam sekumpulan dokumen

Model ini adalah spesialisasi dari strategi split-apply-combine untuk analisis data.[9] Hal ini terinspirasi oleh fungsi map dan reduce yang biasa digunakan dalam pemrograman fungsional, meskipun tujuannya dalam kerangka MapReduce tidak sama dengan bentuk aslinya.[10] Kontribusi utama dari kerangka MapReduce bukanlah fungsi map dan reduce yang aktual, tetapi skalabilitas dan toleransi kesalahan yang dicapai untuk berbagai aplikasi dengan mengoptimalkan mesin eksekusi. Dengan demikian, implementasi MapReduce single threaded biasanya tidak akan lebih cepat daripada implementasi tradisional (non MapReduce), setiap keunggulan dari MapReduce biasanya hanya terlihat pada implementasi multi threaded. Penggunaan model ini bermanfaat hanya ketika operasi shuffle terdistribusi dioptimalkan) dan fitur toleransi kesalahan dari kerangka MapReduce ikut bekerja. Mengoptimalkan biaya komunikasi sangat penting untuk algoritma MapReduce yang baik.[11]

Library dari MapReduce telah ditulis dalam banyak bahasa pemrograman, dengan tingkat pengoptimalan yang berbeda-beda. Implementasi open source yang populer yang mendukung shuffle terdistribusi adalah bagian dari Apache Hadoop. Istilah MapReduce awalnya hanya mengacu pada teknologi milik Google, tetapi sejak itu telah digeneralisasikan. Pada tahun 2014, Google tidak lagi menggunakan MapReduce sebagai model pemrosesan big data utama mereka, dan pengembangan pada Apache Mahout telah beralih ke mekanisme yang lebih kapabel dan mekanisme yang lebih tidak berorientasi pada disk yang menggabungkan kapabilitas map dan reduce secara utuh.

2.2.2. HDFS



Gambar 2.4. Arsitektur HDFS

HDFS adalah proyek Apache Software Foundation dan sub proyek dari proyek Apache Hadoop. Hadoop sangat ideal untuk menyimpan data dalam jumlah besar hingga ber-terabyte dan petabyte, dan menggunakan HDFS sebagai sistem penyimpanannya. HDFS memungkinkan Anda menghubungkan node (PC biasa) yang terdapat dalam kelompok di mana file data didistribusikan. Kemudian kita dapat mengakses dan menyimpan file data sebagai

satu sistem file yang mulus. Akses ke file data ditangani secara streaming, artinya aplikasi atau perintah dijalankan secara langsung menggunakan model pemrosesan MapReduce.

2.3. Implementasi Big Data pada ITS

Dewasa ini big data big data menjadi fokus penelitian dalam Sistem Transportasi Cerdas atau *Intelligent Transportation System (ITS)* yang dapat dilihat di banyak proyek di seluruh dunia.[12] *ITS* akan menghasilkan sejumlah besar data. Big data yang dihasilkan akan memiliki dampak besar pada desain dan penerapan *ITS* sehingga menjadi lebih aman, lebih efisien, dan menguntungkan.

Salah satu studi kasus aplikasi analisis big data dalam ITS adalah pembacaan jalur perjalanan kendaraan bermotor. Dari data perjalanan kendaraan bermotor tersebut, dapat dilakukan pengolahan data lebih lanjut, yaitu penentuan rute transportasi umum yang mengadopsi rute perjalanan kendaraan pribadi yang paling banyak terjadi.

2.4. Kondisi Sistem Lalu Lintas Cerdas Kota Surabaya

Surabaya, Sidoarjo, Mojokerto, Gresik, Bangkalan, Pasuruan, Lamongan dalam satu sistem tata ruang yang terintegrasi dalam struktur jaringan infrastruktur jalan, transportasi, komunikasi dan seterusnya. Integrasi sistem tata ruang dalam struktur jaringan infrastruktur jalan, transportasi, teknologi komunikasi dan informasi dan seterusnya menuju Greater Surabaya.

Juni 2012, Surabaya dinobatkan sebagai salah satu kota terbaik pada Peningkatan e-Government Indonesia (PeGi) 2012. Dinas Perhubungan kota Surabaya terus berupaya berkontribusi pada pemeringkatan PeGI dengan inisiatif memanfaatkan Teknologi Informasi dan Komunikasi dalam program-program pembangunan sistem cerdas (intelligent) untuk manajemen dan rekayasa Lalu-Lintas. Salah satu hasil program tersebut adalah penerapan awal Intelligent Transportation System (ITS). Pada tahap awal, sistem ini berupa sebuah Adaptive Traffic Control System sebagai upaya modernisasi ATCS konvensional yang telah dimiliki oleh Surabaya.

Upaya ini didukung oleh jaringan CCTV Surabaya sebagai bagian dari Traffic Management System. Intelligent Transport Sistem yaitu Sistem cerdas untuk mendukung manajemen transportasi dengan memanfaatkan teknologi (informasi, komunikasi, sensor, kontrol dan komputerisasi) untuk membangun sistem informasi dan manajemen transportasi secara otomatis. Adaptive Traffic Control System merupakan sistem yang mampu melakukan pengaturan waktu nyala lampu lalu-lintas (signal timing) secara real-time berdasarkan kondisi traffic saat itu, termasuk akibat keperluan (demand) khusus dan optimasi kapasitas arus lalu-lintas secara total.

Tujuan dari penerapan SITS ini antara lain adalah [13] :

1. Meningkatkan Keselamatan Lalu Lintas, diantaranya dengan cara mencegah/mengurangi kecelakaan lalu lintas dan mengurangi kerusakan akibat kecelakaan.
2. Meningkatkan Kelancaran Lalu Lintas, diantaranya dengan cara mengoptimalkan siklus lampu lalu lintas baik secara otomatis maupun secara manual.
3. Menjaga kelestarian lingkungan dengan cara mengurangi polusi kendaraan akibat antrian kendaraan di ruas dan persimpangan. Dengan meningkatnya waktu tempuh dan berkurangnya waktu antrian dipersimpangan, diharapkan polusi kendaraan juga makin berkurang.

BAB III METODE PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang digunakan dalam penyusunan Tugas Akhir. Disamping itu, dijelaskan pula prosedur dan proses pelaksanaan tiap-tiap langkah yang dilakukan dalam menyelesaikan Tugas Akhir.

3.1. Tahapan Penelitian

Pada penelitian tugas akhir ini dilakukan beberapa langkah sebagai berikut :

1. Studi Literatur

Pada tahap ini dilakukan identifikasi permasalahan dengan mencari referensi yang menunjang penelitian yang berupa Tugas Akhir, Jurnal Internasional, buku, maupun artikel yang berhubungan dengan topik Tugas Akhir ini.

2. Akuisisi Data

Akuisisi data ini dilakukan untuk mengumpulkan himpunan data kendaraan yang melalui CCTV yang terletak di kawasan Surabaya Pusat selama 30 menit. Lalu data tersebut dikonversi sehingga dapat diproses lebih lanjut.

3. Pembacaan Jalur Perjalanan Kendaraan pada MapReduce

Pada tahap ini dilakukan pembacaan jalur perjalanan kendaraan dengan cara membuat fungsi yang dapat membaca jalur perjalanan kendaraan dengan memanfaatkan data yang sudah didapat.

4. Pengolahan Data Jalur Perjalanan Kendaraan pada HDFS

Data jalur perjalanan kendaraan yang telah dibaca pada tahap sebelumnya diolah pada database yang berbasis HDFS.

5. Penyusunan Laporan

Pada tahap ini dilakukan penyusunan laporan berdasarkan pada hasil penelitian yang telah dilakukan.

3.2. Objek dan Aspek Penelitian

Objek yang digunakan pada penelitian ini adalah data kendaraan yang melalui CCTV di Kawasan Surabaya Pusat selama 30 menit kendaraan. Rekaman yang diamati adalah rekaman pada hari Senin, tanggal 25 November 2018 pukul 11.00 – 11.30. Sedangkan CCTV yang diamati adalah CCTV yang terletak pada :

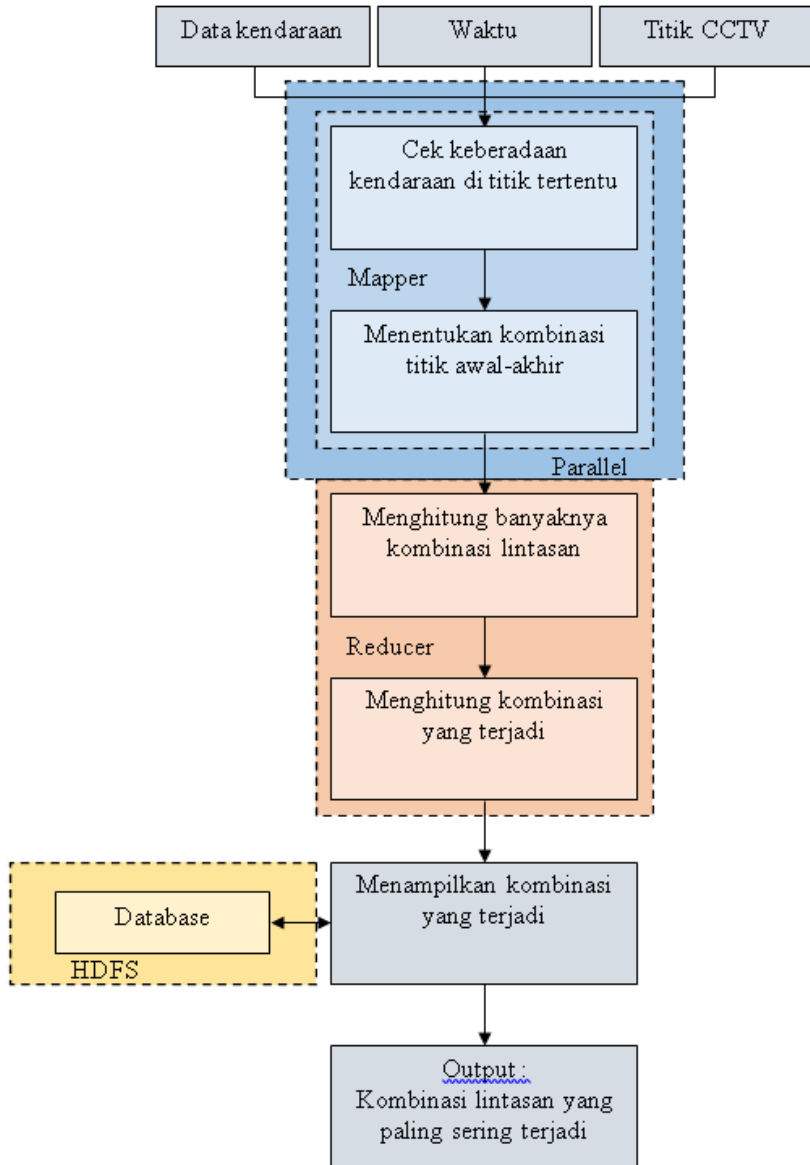
1. Jl. Kayoon – Sonokembang
2. Jl. Panglima Sudirman
3. Jl. Urip Sumoharjo – Pandegiling
4. Keputran – Dinoyo
5. Jl. Raya Ngagel – Sulawesi

Data penelitian diambil dengan cara mengamati setiap kendaraan yang terekam oleh 5 CCTV tersebut.

3.3. Blok Diagram

Data yang telah diperoleh dikonversi sehingga didapatkan himpunan data berupa data kendaraan dan lokasi dimana kendaraan tersebut terdeteksi (titik CCTV). Setelah itu, setiap kendaraan dicek keberadaannya di setiap titik, lalu ditentukan di titik mana kendaraan tersebut pertama dan terakhir kali terdeteksi. Langkah ini dilakukan oleh mapper dan dikerjakan secara parallel. Kemudian setelah diketahui pasangan titik awal-akhir dari setiap kendaraan, dihitunglah berapa kali pasangan tersebut terjadi. Langkah tersebut dilakukan oleh reducer. Setelah menghitung jumlah kejadian pasangan, data – data tersebut diolah dan disimpan di database yang berbasis HDFS, sekaligus ditampilkan hasilnya. Output yang ditampilkan berupa pasangan lintasan beserta banyaknya pasangan tersebut terjadi.

Untuk lebih ringkasnya, berikut adalah blok diagram yang digunakan untuk mengolah data yang telah diperoleh :



Gambar 3.1. Blok diagram penelitian

BAB IV

PERANCANGAN DAN IMPLEMENTASI SISTEM

Bab ini menjelaskan tentang perancangan yang diperlukan dan implementasi sistem.

4.1. Perancangan Sistem

Pada bagian ini, akan dijelaskan tentang perancangan sistem yang meliputi perancangan data, perancangan blok diagram sistem, serta perancangan *class* diagram.

4.1.1. Perancangan Data

Data-data yang digunakan dalam program ini dapat dibedakan menjadi dua jenis, yaitu data input dan data output.

1. Data Input

Data input adalah data yang digunakan sebagai masukan (input) dari program. Input ini yang kemudian diolah oleh program melalui tahap tahap tertentu sehingga menghasilkan keluaran (output) yang diinginkan. Data input yang digunakan adalah data kendaraan bermotor yang melewati setiap CCTV selama 30 menit.

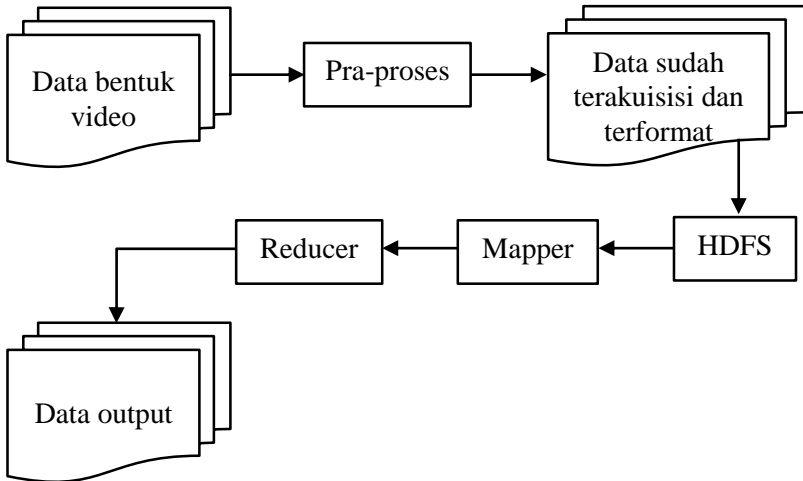
2. Data Output

Data output merupakan data yang dihasilkan oleh program setelah proses-proses tertentu selesai dilakukan. Data keluaran pada program ini berupa data pasangan titik awal-akhir perjalanan kendaraan bermotor beserta banyaknya pasangan tersebut dilalui. Kemudian dari data output tersebut akan dianalisa bagaimana preferensi jalur perjalanan kendaraan bermotor di lima titik CCTV tersebut.

4.1.2. Perancangan Blok Diagram Sistem

Pada Gambar 4.1 di bawah dijelaskan tentang tahapan-tahapan yang dilakukan untuk mencapai tujuan Tugas Akhir. Dalam tugas akhir ini *tools* untuk komputasi menggunakan bahasa pemrograman java pada ekosistem Hadoop dengan menggunakan NetBeans IDE

8.2. Program yang dibuat dalam NetBeans IDE 8.2 disimpan dengan nama *VehicleRoute*. Terdapat dua *class* yang keduanya akan disimpan dalam *package* default, yaitu *Source Packages*.



Gambar 4.1. Blok Diagram Sistem

4.1.2.1. Akuisisi Data (Pra-Proses)

Pada tahap pra-proses, dilakukan pengkonversian data secara manual, yaitu pengenalan kendaraan yang terekam oleh kamera CCTV, dicatat dengan memperhatikan waktu, letak CCTV, dan data kendaraan. Letak CCTV sendiri akan dinotasikan sebagai berikut :

1. Jl. Kayoon – Sonokembang sebagai CCTV 1
2. Jl. Panglima Sudirman sebagai CCTV 2
3. Jl. Urip Sumoharjo – Pandegiling sebagai CCTV 3
4. Keputran – Dinoyo sebagai CCTV 4
5. Jl. Raya Ngagel – Sulawesi sebagai CCTV 5

Gambar 4.2 adalah peta jalur perjalanan kendaraan bermotor dan titik – titik CCTV yang diamati. Sedangkan 8 adalah tampilan setiap titik CCTV yang diamati.

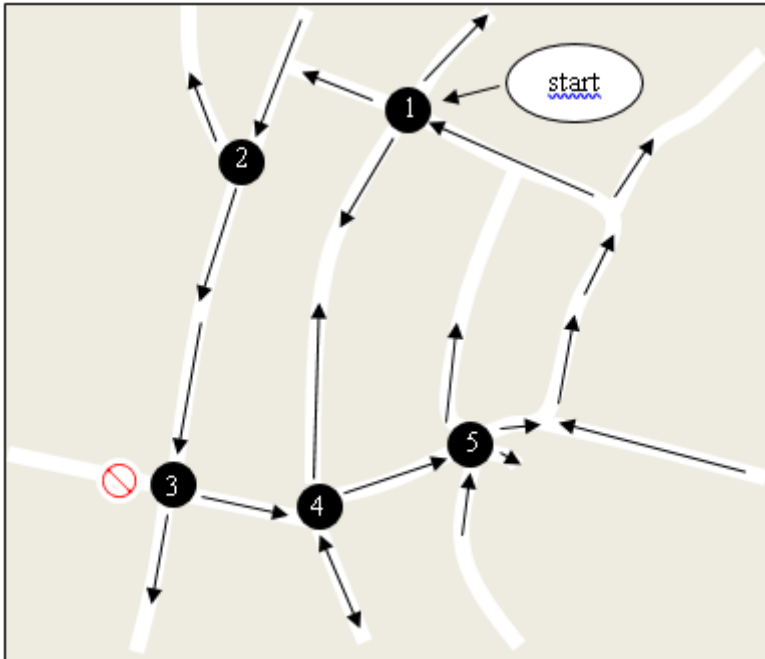


Gambar 4.2. Peta Titik CCTV yang Diamati



Gambar 4.3. Tampilan Masing - Masing CCTV yang Diamati

Gambar 4.3 merupakan tampilan visual dari setiap titik CCTV yang diamati dalam penelitian ini. Sedangkan berikut ini adalah sketsa dari titik – titik CCTV tersebut beserta arah arus lalu lintas di berbagai ruas simpangannya.



Gambar 4.4. Sketsa Titik CCTV yang Diamati

Sketsa arah arus lalu lintas pada Gambar 4.4 telah disesuaikan dengan sudut pandang kamera CCTV yang terpasang di setiap titik.

Proses akuisisi data akan menghasilkan data yang dapat digunakan sebagai data input yang dapat disimpan pada HDFS dan nantinya dapat diolah oleh program mapreduce.

Berikut akan dilakukan simulasi proses pengakuisisian data sehingga didapat data input yang dapat disimpan pada HDFS dan kemudian dapat diolah oleh program MapReduce.



Gambar 4.5. Kendaraan k1
Terbaca pada CCTV 1

Pertama, setiap kendaraan yang terekam oleh setiap CCTV diamati dengan cara diberi ID dan dicatat waktu terekamnya dan titik CCTV yang merekam kendaraan yang diamati tersebut. Sebagai contoh, kendaraan pada gambar disamping diberi ID = k1, dengan waktu = 00.15 (waktu yang dicatat hanya bagian menit dan detikanya saja), dan CCTV yang merekam adalah CCTV 1.

Berikut ini adalah tabel yang menunjukkan jarak antar CCTV dan berdasarkan data pada Lampiran 1, didapatkan pula rata – rata waktu yang dibutuhkan untuk berpindah antar CCTV. Dengan demikian, kita dapat menghitung rata – rata kecepatan kendaraan yang melewati setiap segmen antar CCTV.

Tabel 4.1. Rata - Rata Waktu Perpindahan Kendaraan antar CCTV

Titik CCTV	Jarak	Waktu	Kecepatan
1 – 2	350 m	183 s	6,84 km/h
2 – 3	500 m	43 s	41,76 km/h
3 – 4	280 m	39 s	25,92 km/h
4 - 5	300 m	131 s	8,28 km/h

Berdasarkan tabel di atas, dapat kita simpulkan bahwa rata – rata waktu yang dibutuhkan setiap kendaraan untuk berpindah antar CCTV kurang dari 5 menit, sehingga jika terdapat kendaraan yang waktu perpindahan antar CCTV lebih dari 5 menit, maka kendaraan tersebut akan dianggap sebagai kendaraan baru dan akan diberi ID yang berbeda dengan ID sebelumnya.

Dalam proses akuisisi data, akan dijumpai banyak kemungkinan kasus kendaraan yang terekam oleh sistem CCTV, di antaranya :

1. Kasus pertama adalah kasus dimana kendaraan melewati beberapa CCTV secara berurutan dengan rentang waktu yang dibutuhkan untuk berpindah antar CCTV kurang dari 5 menit. Berikut adalah contoh kendaraan yang mengalami kasus pertama.



Gambar 4.6. Kendaraan k1 terbaca pada CCTV 2



Gambar 4.7 Kendaraan k1 Terbaca pada CCTV 3

Pada Gambar 4.6 dan 4.7, kendaraan dengan ID k1 melewati CCTV 2 menuju CCTV 3 dengan rentang waktu kurang dari 5 menit. Sehingga k1 dikatakan sebagai kendaraan yang mengalami kasus pertama.

2. Kasus kedua adalah kasus dimana kendaraan melewati beberapa CCTV secara berurutan selayaknya pada kasus pertama, namun pada kasus kedua ini, kendaraan berpindah antar CCTV membutuhkan waktu lebih dari 5 menit. Berikut adalah contoh kendaraan yang mengalami kasus kedua.



Gambar 4.8. Kendaraan k2 Terbaca pada CCTV 2



Gambar 4.9. Kendaraan k2 Terbaca pada CCTV 3



Gambar 4.10. Kendaraan k2 Terbaca pada CCTV 4



Gambar 4.11. Kendaraan k2 Terbaca pada CCTV 5

Pada Gambar 4.8 sampai 4.11, kendaraan dengan ID k2 melewati CCTV 1 menuju CCTV 2 dengan rentang waktu kurang dari 5 menit, namun dari CCTV 2 menuju CCTV 3, rentang waktu yang dibutuhkan lebih dari 5 menit. Sehingga k2 dikatakan sebagai kendaraan yang mengalami kasus kedua.

3. Kasus ketiga adalah kasus dimana kendaraan hanya melewati satu titik CCTV. Berikut adalah contoh kendaraan yang mengalami kasus ketiga.



Gambar 4.12. Kendaraan k3 Terbaca pada CCTV 2

Pada gambar di atas, kendaraan dengan ID k3 hanya melewati CCTV 2 saja dan tidak pernah terbaca lagi di CCTV berikutnya. Sehingga k3 dikatakan sebagai kendaraan yang mengalami kasus ketiga.

4. Kasus keempat adalah kasus terakhir, yaitu kasus diaman suatu kendaraan melewati beberapa CCTV tidak secara berurutan. Berikut adalah contoh kendaraan yang mengalami kasus ketiga.



Gambar 4.13. Kendaraan k4 Terbaca pada CCTV 2

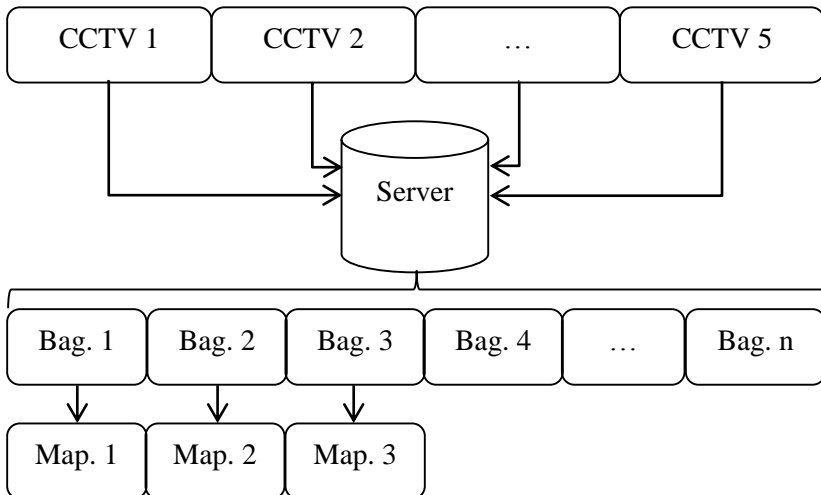


Gambar 4.14. Kendaraan k4 Terbaca pada CCTV 5

Pada gambar di atas, kendaraan dengan ID k4 melewati CCTV 2, mengikuti alur lalu-lintas pada sistem, seharusnya kendaraan tersebut akan terbaca di CCTV 3 lalu kemudian di CCTV 4, namun, kendaraan tersebut malah terbaca pada CCTV 5. Sehingga k4 dikatakan sebagai kendaraan yang mengalami kasus keempat.

4.1.2.2. Arsitektur Sistem Secara Umum

Berikut adalah gambaran tentang sistem yang akan dibuat :



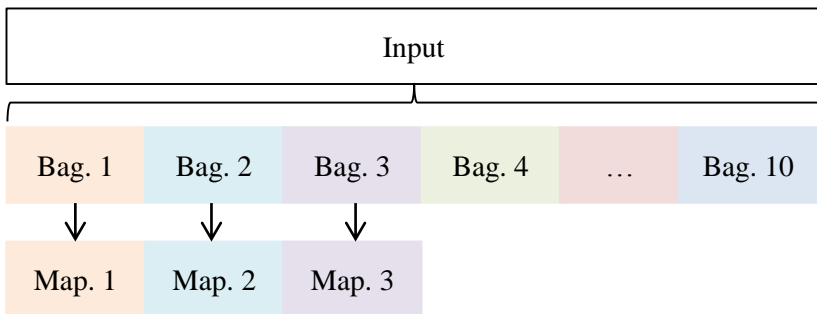
Gambar 4.15. Arsitektur Sistem

Arsitektur sistem secara umum idealnya mengambil nilai input dari setiap CCTV yang diamati. Setiap CCTV mengirim data kendaraan yang terbaca pada CCTV tersebut setiap satuan waktu tertentu kepada server. Selanjutnya server mengatur data yang diterima menjadi bagian – bagian yang akan diproses oleh fungsi *mapper* secara paralel. Karena prasarana yang belum memadai untuk menerapkan sistem yang ideal, maka tugas dari server yang mengatur data input menjadi bagian – bagian yang akan diproses oleh fungsi *mapper* dilakukan secara manual.

4.1.2.3. Perancangan Model Paralel

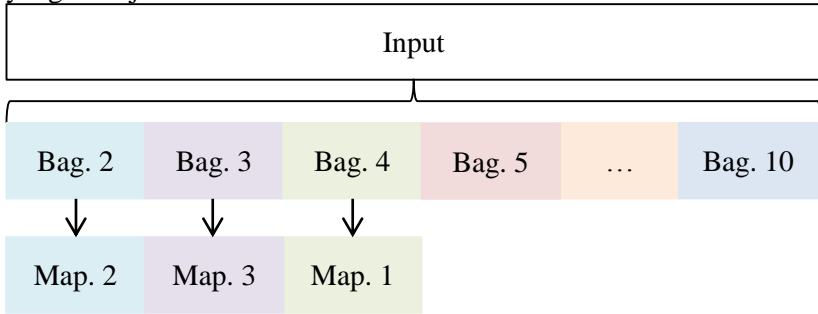
Pada tahap ini, akan dirancang model yang akan mengolah data input secara paralel. Diasumsikan setiap CCTV mengirim data input kepada server selama 10 satuan waktu, maka server akan mengatur data tersebut menjadi 10 bagian, dimana setiap bagian akan diolah oleh fungsi *mapper*. Terdapat tiga fungsi *mapper* yang akan mengolah setiap bagian data input tersebut yang dilakukan secara paralel. Sehingga jika salah satu fungsi *mapper* telah selesai mengolah suatu bagian data input, maka fungsi tersebut akan langsung mengambil bagian data input selanjutnya untuk diproses.

Gambar 4.16 menunjukkan bagaimana proses paralel dilakukan oleh fungsi *mapper*, dimana bagian input pertama hingga bagian input ketiga diproses terlebih dahulu oleh ketiga fungsi *mapper*.



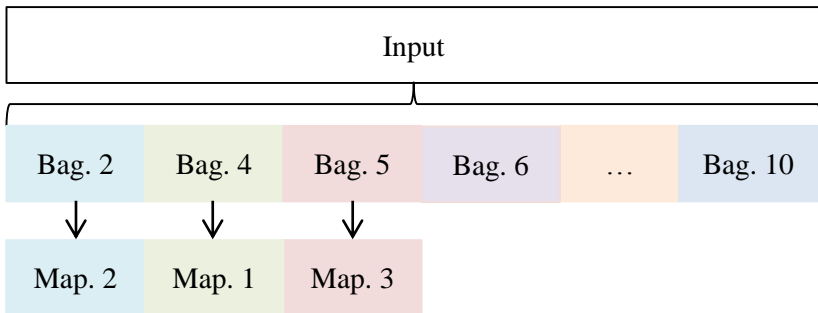
Gambar 4.16. Grafik Model Paralel

Dimisalkan *mapper* pertama telah terlebih dahulu selesai mengolah bagian input pertama, maka selanjutnya fungsi mapper pertama akan mengambil bagian input keempat untuk diolah seperti yang ditunjukkan oleh Gambar 4.17



Gambar 4.17. Grafik Model Paralel

Lalu dimisalkan lagi mapper ketiga selesai mengolah bagian input ketiga, maka fungsi mapper ketiga akan mengambil bagian selanjutnya, yakni bagian kelima untuk diolah seperti yang ditunjukkan pada Gambar 4.18, dan proses seperti ini dilakukan hingga semua bagian input diolah.



Gambar 4.18. Grafik Model Paralel

Jika terdapat input baru yang akan diolah, maka input baru tersebut akan ditempatkan setelah bagian input 10, dengan kata lain setiap input baru akan ditempatkan pada akhir antrean.

Untuk menetapkan banyaknya fungsi mapper yang akan bekerja, maka pada mapred-site yang terdapat pada folder etc/hadoop

harus ditambahkan sebuah property yang menyatakan bahwa banyaknya fungsi reduce yang bekerja adalah sebanyak tiga.

Berikut adalah property untuk menentukan banyaknya fungsi mapper yang akan bekerja.

```
<property>
  <name>mapreduce.map</name>
  <value>3</value>
</property>
```

4.1.2.4. Perancangan Struktur Data dan Penyimpanan Data pada HDFS

HDFS adalah suatu sistem file terdistribusi, yaitu sistem file yang menyimpan data tidak hanya dalam satu media penyimpanan. Disinilah nantinya data yang telah diakuisisi sebelumnya akan diinputkan, lalu data tersebut akan diolah oleh program mapreduce.

Karena data yang akan diinputkan ke dalam HDFS haruslah dapat diolah oleh fungsi *mapper*, sehingga data hasil akuisisi akan disimpan dengan struktur data seperti pada Gambar 4.19. Data tersebut disimpan dalam *array of String* satu dimensi, agar proses *parsing* oleh fungsi *mapper* dapat dilakukan lebih mudah.

```
0102 2 k1 0123 3 k1 0012 2 k2 0102 3 k2 0647 4 k2
0740 5 k2 0824 2 k3 0534 2 k4 1019 5 k4
```

Gambar 4.19. Struktur Data Input untuk HDFS

Pada Gambar 4.19, index ke- i pada array tersebut menunjukkan waktu suatu kendaraan terbaca pada CCTV, index ke- $i+1$ menunjukkan titik CCTV yang membaca kendaraan tersebut, dan index ke- $i+2$ adalah ID kendaraan yang terbaca.

Output dari data yang diolah secara otomatis akan tersimpan pada HDFS dengan direktori yang telah dibuat pada saat menjalankan program.

4.1.2.5. Perancangan Fungsi Mapper

Pada tahap ini, fungsi *mapper* akan memetakan pasangan nilai input menjadi suatu himpunan pasangan – pasangan nilai intermediate. Pada kasus ini, nilai input yang dimaksud berupa pasangan waktu suatu kendaraan terbaca oleh CCTV, lokasi CCTV yang membaca kendaraan tersebut, dan ID kendaraan yang terbaca. Sedangkan output dari fungsi *mapper* ini, yaitu pasangan – pasangan nilai intermediate, adalah titik dimana suatu kendaraan pertama kali terbaca oleh CCTV dan titik terakhir kendaraan tersebut terbaca.

Fungsi *mapper* akan mengolah bagian inputnya sehingga menghasilkan output yang dapat diolah oleh fungsi reducer, yaitu pasangan CCTV – CCTV dimana suatu kendaraan pertama kali terbaca dan terakhir kali terbaca oleh CCTV. Misalkan kendaraan A terbaca pada CCTV X pada T_1 lalu terbaca lagi pada CCTV Y pada waktu T_2 , maka jalur perjalanan kendaraan A tersebut akan disimpan dalam bentuk String sebagai “X – Y”.

Berdasarkan contoh kejadian yang terlihat di pada Gambar 4.19, fungsi *mapper* akan membaca jalur perjalanan dari setiap kendaraan yang terekam, sehingga didapat kendaraan k1 bepergian dari CCTV 2 ke CCTV 3, kendaraan k2 terbaca sebagai dua kendaraan berbeda yaitu kendaraan yang bepergian dari CCTV 2 ke CCTV 3, dan kendaraan yang bepergian dari CCTV 4 ke CCTV 5, hal ini dikarenakan rentang waktu pada perjalanan dari CCTV 2 menuju CCTV 3 lebih dari 5 menit, sehingga pada CCTV 3, k2 terbaca sebagai kendaraan baru lagi. Hasil *mapping* untuk kendaraan k3 tidak ditampilkan karena kendaraan k3 hanya terbaca pada satu CCTV saja. Lalu kendaraan k4 terbaca dari CCTV 2 ke CCTV 5.

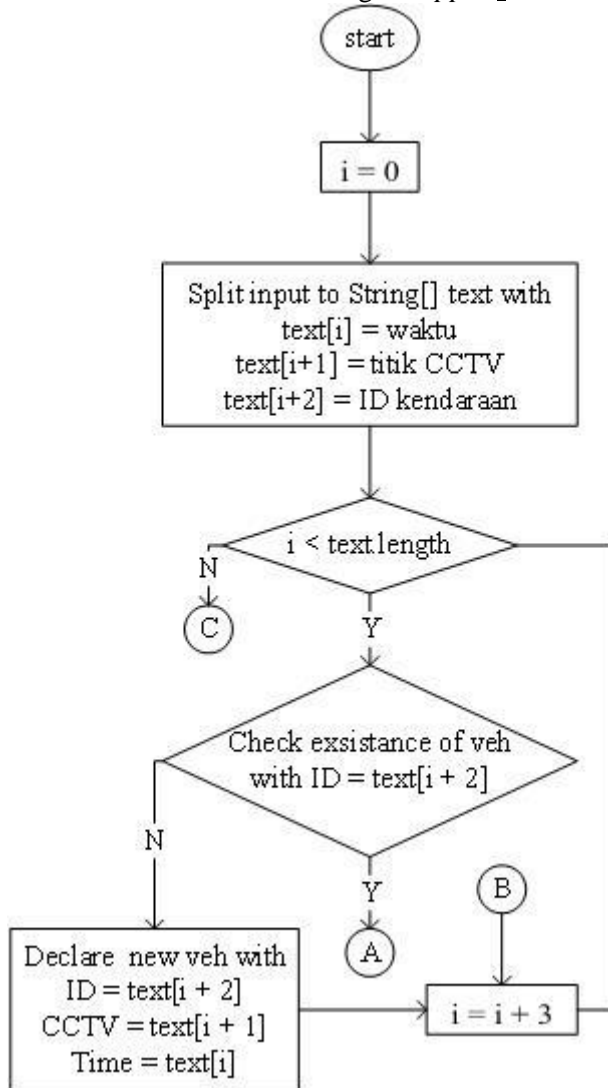
Untuk dapat diolah, data input terlebih dahulu dijadikan suatu array, dimana array tersebut memuat waktu kendaraan terbaca oleh CCTV pada index ke- i (yang selanjutnya disebut “waktu”), CCTV yang membaca kendaraan tersebut pada index ke- $i+2$ (yang selanjutnya disebut “CCTV”), dan ID kendaraan yang terbaca pada index ke- $i+3$ (yang selanjutnya disebut “ID”), dimana i adalah bilangan asli dari 0 sampai 99.

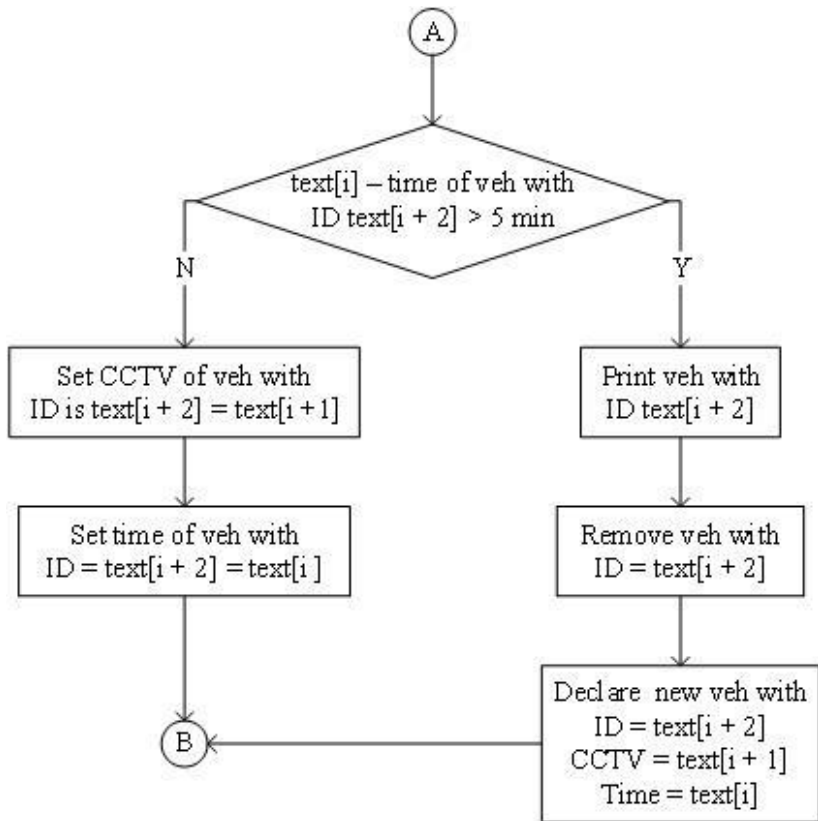
Setelah data input disimpan pada array, selanjutnya array tersebut diproses dengan cara dicek apakah kendaraan dengan ID =

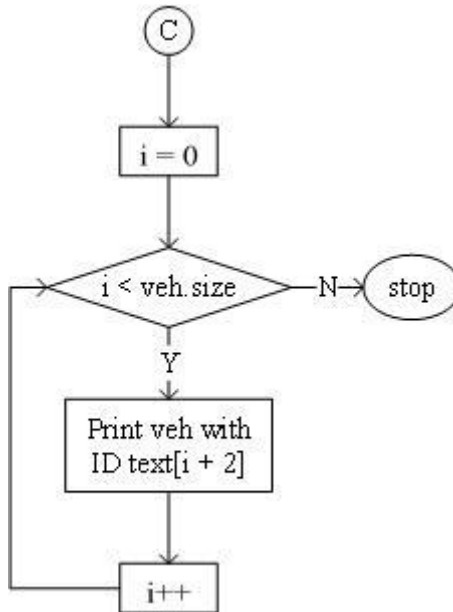
elemen array pada index ke- $i+2$ terdapat pada himpunan kendaraan yang sudah dicek (telah terbaca oleh suatu CCTV) atau belum. Jika sudah terdapat pada himpunan tersebut, dicek kembali apakah kendaraan tersebut terbaca pada CCTV sebelumnya lebih dari 5 menit yang lalu atau tidak. Jika kendaraan tersebut terbaca pada CCTV sebelumnya lebih dari 5 menit, maka kendaraan tersebut dinyatakan sebagai kendaraan yang baru terbaca, sehingga kendaraan tersebut dideklarasikan pada himpunan tersebut dengan atribut waktu = elemen pada array dengan index ke- i , CCTV = elemen pada array dengan index ke- $i+1$, dan ID = elemen pada array dengan index ke- $i+2$. Jika ternyata kendaraan tersebut terbaca oleh CCTV sebelumnya kurang dari 5 menit yang lalu, maka kendaraan tersebut hanya mengupdate atribut waktu-nya menjadi elemen array pada index ke- i dan menambahkan atribut CCTV dengan elemen array pada index ke- $i+1$, sehingga nantinya setiap kendaraan yang dicek akan memiliki himpunan CCTV yang pernah dilaluinya. Sama seperti kasus pertama, jika kendaraan dengan ID = elem array ke- $i+2$ belum sama sekali pernah terbaca pada CCTV manapun, maka kendaraan tersebut akan dideklarasikan pada himpunan kendaraan yang sudah terbaca sebagai suatu anggota baru dengan atribut waktu = elemen pada array dengan index ke- i , CCTV = elemen pada array dengan index ke- $i+1$, dan ID = elemen pada array dengan index ke- $i+2$.

Selanjutnya setiap kendaraan yang telah dicek tersebut akan diambil dua lokasi atau titik CCTV pada himpunan CCTV yang pernah dilaluinya, yaitu elemen pertama pada himpunan CCTV tersebut dan dinyatakan sebagai titik awal kendaraan tersebut terbaca atau memasuki sistem transportasi yang diteliti, dan elemen terakhir himpunan pada himpunan CCTV tersebut yang akan dinyatakan sebagai titik akhir kendaraan tersebut terbaca atau titik dimana kendaraan tersebut keluar dari sistem transportasi yang diteliti. Yang pada akhirnya, fungsi *mapper* ini menghasilkan output berupa pasangan – pasangan titik awal dan titik akhir dari setiap kendaraan yang terbaca oleh sistem.

Berikut adalah flowchart dari fungsi *mapper* :







Gambar 4.20. Flowchart fungsi *mapper*

Flowchart di atas menerangkan bagaimana suatu bagian input yang berisi 100 kendaraan diolah pada fungsi *mapper*.

Langkah – langkah di atas juga dapat ditulis dengan *pseudocode* sebagai berikut :

```
1. String[] text = split input;
2. ArrayList<veh> vehicle;

3. For i = 0 as i < text.length with i = i+3 {
4.   If checkExistance of vehicle with ID = text[i+2] {
5.     If text[i] - time of vehicle with ID = text[i+2] <
       5 min {
6.       If first and last CCTV of vehicle with ID =
           text[i+2] different {
7.         Print (the first CCTV and the last CCTV);
8.       }
9.       Remove vehicle with ID = text[i+2];
10.      Add new vehicle with ID = text[i+2], CCTV =
           text[i+1], time = [i];
11.     }
12.     Else {
13.       add CCTV of the vehicle = text[i+1];
14.       add time of the vehicle = text[i];
15.     }
16.     Else add new vehicle with ID = text[i+2], CCTV =
           text[i+1], time = [i];
17. }

18. For i = 0 as i < vehicle.size with i++ {
19.   If first and last CCTV of vehicle with ID = text[i+2]
       different {
20.     Print (the first CCTV and the last CCTV);
21.   }
22. }
```

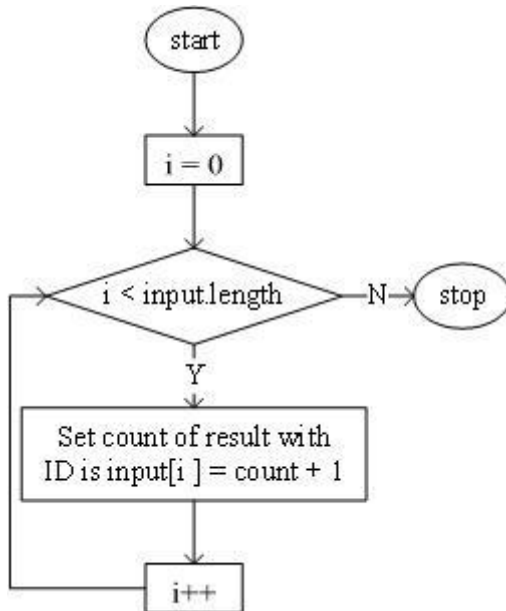
4.1.2.6. Perancangan Fungsi Reducer

Setelah pasangan nilai input diproses menjadi pasangan nilai intermediate oleh fungsi *mapper*, selanjutnya pasangan nilai intermediate tersebut diolah oleh fungsi *reducer* menjadi nilai output final. Pada kasus ini, pasangan titik awal dan titik akhir suatu kendaraan akan diolah oleh fungsi *reducer*, yaitu dengan cara dihitung frekuensi kemunculan dari setiap pasangan titik awal dan titik akhir. Output dari fungsi *reducer* ini adalah pasangan titik awal dan titik akhir perjalanan setiap kendaraan beserta frekuensi kemunculan dari setiap pasangan tersebut.

Misal terdapat n perjalanan kendaraan yang telah terbaca pada fungsi mapper, maka dari n perjalanan tersebut akan dihitung berapa banyak perjalanan yang melalui jalur " $X_k - Y_k$ ". Sehingga nantinya akan dihitung jumlah kejadian setiap pasangan jalur perjalanan yang telah terbaca pada fungsi mapper.

Berdasarkan contoh kejadian pada Gambar 4.19, fungsi reducer akan menghitung setiap pasangan jalur perjalanan kendaraan yang telah dibaca oleh fungsi mapper, yaitu jalur perjalanan kendaraan dari CCTV 2 menuju CCTV 3 sebanyak dua kali, dan masing – masing satu kali untuk jalur perjalanan dari CCTV 4 menuju CCTV 5 dan dari CCTV 2 menuju CCTV 5.

Berikut adalah flowchart dari fungsi *reducer* :



Gambar 4.21. Flowchart fungsi *reducer*

Flowchart di atas menjelaskan bagaimana fungsi *reducer* mengolah data pasangan data intermediate yang dihasilkan oleh fungsi *mapper*. Output dari fungsi mapper yang berupa himpunan pasangan titik awal dan titik akhir dari suatu kendaraan yang

melewati sistem adalah suatu input bagi fungsi *reducer* ini. Setiap pasangan titik awal dan titik akhir dihitung jumlah kejadiannya, sehingga fungsi ini menghasilkan output berupa himpunan pasangan titik awal dan titik akhir beserta dengan jumlah kejadian pasangan tersebut.

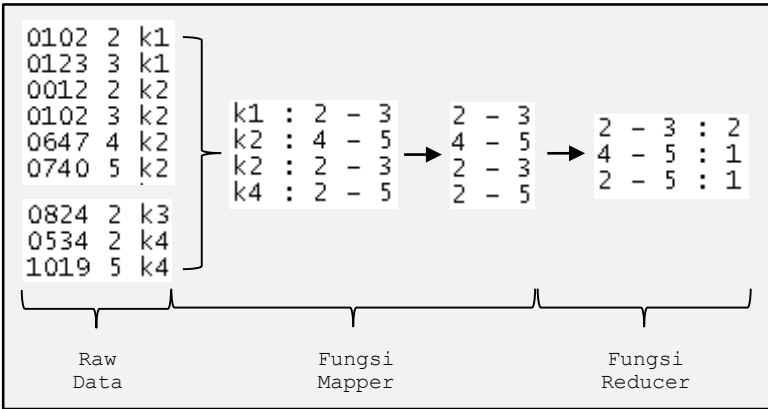
Langkah – langkah di atas juga dapat ditulis dengan *pseudocode* sebagai berikut :

```

1. IntWritable result;
2. For i = 0 as i < input.length with i++ {
3.     Set count of result with ID = input[i] = count+1;
4. }

```

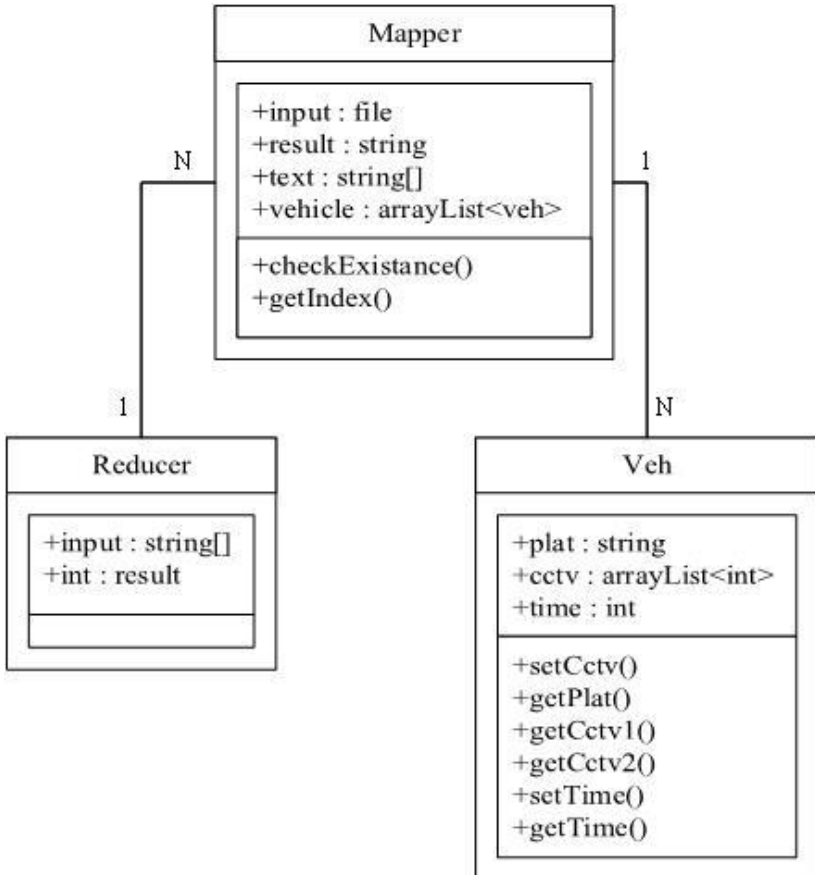
Gambar 4.22 menunjukkan ilustrasi bagaimana fungsi *mapper* mengolah *raw* data sehingga menjadi suatu output yang selanjutnya dapat diolah oleh fungsi *reducer* menjadi *final* output. Pada ilustrasi tersebut, maksud dari hasil pembacaan dari fungsi mapper “k1 : 2 – 3” kendaraan k1 bepergian dari CCTV 2 menuju CCTV 3, dan maksud dari “2 – 3” adalah terdapat sebuah kendaraan yang bepergian dari CCTV 2 menuju CCTV 3. Lalu hasil penghitungan oleh fungsi reducer yang bertuliskan “2 – 3 : 2” berarti bahwa jumlah kendaraan yang melalui CCTV 2 menuju CCTV 3 sebanyak dua buah kendaraan.



Gambar 4.22. Ilustrasi tahap 4.1.2.5. - 4.1.2.6.

4.1.3. Perancangan *Class Diagram*

Pada class diagram di bawah ini akan dijelaskan tentang UML yang akan dijadikan dasar sebagai pembuatan program MapReduce. Program yang akan dibuat nantinya memiliki tiga kelas, yakni *mapper*, *reducer*, dan *veh*, yang memiliki atribut dan method seperti yang terlihat pada diagram di berikut.



Gambar 4.23. *Class diagram*

4.2. Implementasi Sistem

Pada bagian ini dijelaskan tentang implementasi struktur data dan penyimpanan data pada HDFS, implementasi model paralel pada fungsi *mapper*, dan implementasi fungsi *reducer*.

4.2.1. Implementasi Struktur Data dan Penyimpanan Data pada HDFS

Hasil dari akuisisi data dicatat dengan struktur data seperti pada Subbab 4.1.2.1.. Sebelum menginput data hasil akuisisi tersebut ke dalam HDFS, terlebih dahulu dibuat direktori pada HDFS dimana data tersebut akan diinputkan. Berikut adalah sintaks untuk membuat direktori pada HDFS :

```
C:\Users\John>hdfs dfs -mkdir -p /user/input
```

Gambar 4.24. Pembuatan Direktori pada HDFS

Gambar di atas menunjukkan bahwa direktori “input” akan dibuat pada super direktori “user”, dimana super direktori ini dibuat pada PC “John”. Berikut adalah tampilan direktori sebelum data diinputkan :



Gambar 4.25. Direktori pada HDFS

Setelah direktori tersedia, data input telah dapat diinputkan ke dalam HDFS dengan cara sebagai berikut :

```
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input1.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input2.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input3.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input4.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input5.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input6.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input7.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input8.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input9.txt /user/input
C:\Users\John>hdfs dfs -put C:\Users\John\Desktop\input10.txt /user/input
```

Gambar 4.26. Input Data ke Dalam HDFS

Gambar di atas menunjukkan bahwa file input1.txt – input10.txt yang berada pada desktop pada PC “John” telah berhasil diinputkan ke dalam HDFS dengan direktori “input” yang berada pada super direktori “user”. Berikut adalah tampilan direktori /user/input setelah data berhasil diinput :

Browse Directory

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	John	supergroup	2.33 kB	1/7/2019, 10:37:26 PM	1	128 MB	input1.txt
-rw-r--r--	John	supergroup	2.1 kB	1/7/2019, 10:38:59 PM	1	128 MB	input10.txt
-rw-r--r--	John	supergroup	2.19 kB	1/7/2019, 10:37:35 PM	1	128 MB	input2.txt
-rw-r--r--	John	supergroup	2.44 kB	1/7/2019, 10:37:42 PM	1	128 MB	input3.txt
-rw-r--r--	John	supergroup	2.32 kB	1/7/2019, 10:38:09 PM	1	128 MB	input4.txt
-rw-r--r--	John	supergroup	2.68 kB	1/7/2019, 10:38:18 PM	1	128 MB	input5.txt
-rw-r--r--	John	supergroup	2.04 kB	1/7/2019, 10:38:25 PM	1	128 MB	input6.txt
-rw-r--r--	John	supergroup	2.73 kB	1/7/2019, 10:38:32 PM	1	128 MB	input7.txt
-rw-r--r--	John	supergroup	2.18 kB	1/7/2019, 10:38:43 PM	1	128 MB	input8.txt
-rw-r--r--	John	supergroup	2.53 kB	1/7/2019, 10:38:50 PM	1	128 MB	input9.txt

Hadoop, 2018.

Gambar 4.27. Hasil Input Data pada HDFS

4.2.2. Implementasi Fungsi Mapper

Data yang sebelumnya telah diinputkan ke HDFS kemudian diolah oleh fungsi *mapper* untuk dijadikan kumpulan String, String

tersebut adalah pasangan titik awal dan titik akhir jalur perjalanan suatu kendaraan bermotor.

Pada tahap ini, dibuat suatu subclass dari class Mapper, yang diberi nama “MyMapper”. Class MyMapper hanya memiliki satu atribut, yaitu “word”, yang merepresentasikan pasangan titik awal-akhir yang akan dihitung frekuensi kejadiannya. Class MyMapper memiliki method “map” yang digunakan untuk mengolah data input menjadi pasangan-pasangan titik awal dan titik akhir jalur perjalanan suatu kendaraan dalam bentuk String.

Berikut adalah implementasi dari fungsi *mapper* :

```
public static class MyMapper extends Mapper<LongWritable,
Text, Text, IntWritable> {
    private Text word = new Text();
    public void map(LongWritable key, Text value, Context
context) throws IOException, InterruptedException {
        String[] stringArr = value.toString().split("\\s+");
        ArrayList<veh> sortedVeh = new ArrayList<veh>();
        for (int i = 0; i < stringArr.length; i = i + 3) {
            if (checkExistence(sortedVeh, stringArr[i + 2])) {
                if ((Integer.parseInt(stringArr[i]) -
sortedVeh.get(getIndex(sortedVeh, stringArr[i +
2])).getTime()) > 500) {
                    if (sortedVeh.get(i).getCctv1() !=
sortedVeh.get(i).getCctv2()) {
                        word.set(sortedVeh.get(getIndex(sortedVeh
, stringArr[i + 2])).getCctv1() + " - " +
sortedVeh.get(getIndex(sortedVeh,
stringArr[i + 2])).getCctv2());
                        context.write(word, new IntWritable(1));
                    }
                }
                sortedVeh.remove(getIndex(sortedVeh,
stringArr[i + 2]));
                sortedVeh.add(new veh(stringArr[i + 2],
Integer.parseInt(stringArr[i + 1]),
Integer.parseInt(stringArr[i])));
            } else {
                sortedVeh.get(getIndex(sortedVeh, stringArr[i +
2])).setCctv(Integer.parseInt(stringArr[i +
1]));
                sortedVeh.get(getIndex(sortedVeh, stringArr[i +
2])).setTime(Integer.parseInt(stringArr[i]));
            }
            } else {
                sortedVeh.add(new veh(stringArr[i + 2],
Integer.parseInt(stringArr[i + 1]),
Integer.parseInt(stringArr[i])));
            }
        }
        for (int i = 0; i < sortedVeh.size(); i++) {
            if (sortedVeh.get(i).getCctv1() !=
sortedVeh.get(i).getCctv2()) {
                word.set(sortedVeh.get(i).getCctv1() + " - " +
sortedVeh.get(i).getCctv2());
                context.write(word, new IntWritable(1));
            }
        }
    }
}
```

Pada class “MyMapper”, terdapat dua method yang dipanggil, yaitu :

1. `checkExistence(ArrayList<veh> x, String y)`

Method ini digunakan untuk mengecek apakah suatu String y (merepresentasikan plat nomor) dimiliki oleh salah satu objek di ArrayList x (merepresentasikan kumpulan data kendaraan) atau tidak. Berikut adalah implementasi dari method `checkExistence(ArrayList<veh> x, String y)`.

```
public static boolean checkExistence(ArrayList<veh> x,
String y) {
    boolean flag = false;
    for (int i = 0; i < x.size(); i++) {
        if (x.get(i).getPlat().equals(y)) {
            flag = true;
            break;
        }
    }
    return flag;
}
```

2. `getIndex(ArrayList<veh> x, String y)`

Method ini digunakan untuk mencari index dari suatu objek di ArrayList x (kumpulan kendaraan) yang memiliki atribut String y (plat nomor). Berikut adalah implementasi dari method `getIndex(ArrayList<veh> x, String y)`

```
public static int getIndex(ArrayList<veh> x, String y) {
    int flag = 0;
    for (int i = 0; i < x.size(); i++) {
        if (x.get(i).getPlat().equals(y)) {
            flag = i;
            break;
        }
    }
    return flag;
}
```

4.2.3. Implementasi Fungsi Reducer

Setelah mendapatkan pasangan titik awal dan titik akhir dalam bentuk String, dilakukan penghitungan frekuensi kemunculan pasangan – pasangan tersebut oleh fungsi *reducer*. Berikut adalah implementasi dari fungsi *reducer* :

```
public static class MyReducer extends Reducer<Text,
IntWritable, Text, IntWritable> {
    private IntWritable result = new IntWritable();
    public void reduce(Text key, Iterable<IntWritable>
values, Context context)
throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        result.set(sum);
        context.write(key, result);
    }
}
```

“MyReducer” merupakan subclass dari class Reducer, yang memiliki atribut “result”, yaitu atribut yang merepresentasikan banyaknya kejadian untuk setiap pasangan yang ada. Selain itu, class ini juga memiliki method “reduce” yang berfungsi untuk menghitung banyaknya kejadian dari suatu pasangan titik awal-akhir.

4.2.4. Running Program

Setelah data terinput ke dalam HDFS dan fungsi *mapper* dan *reducer* telah terimplementasi, maka program MapReduce siap untuk di-running. Berikut adalah cara running program MapReduce pada ekosistem Hadoop :



```
C:\Users\John>hadoop jar C:\Users\John\Desktop\VehicleRoute.jar VehicleRoute /user/input /user/output
```

Gambar 4.28. Running Program MapReduce

Gambar di atas menunjukkan bahwa program MapReduce yang pada kasus ini disimpan dengan nama “VehicleRoute” yang

tersimpan pada Desktop pada PC “John” yang memiliki *main class* mengeksekusi data input yang berada pada direktori /user/input dan nantinya hasil dari eksekusi oleh program ini disimpan pada direktori /user/output. Berikut merupakan proses *running* dari program MapReduce :


```

19/01/10 13:41:38 INFO client.RMProxy: Connecting to ResourceManager at localhos
t/127.0.0.1:8032
19/01/10 13:41:39 WARN mapreduce.JobResourceUploader: Hadoop command-line option
 parsing not performed. Implement the Tool interface and execute your applicatio
n with ToolRunner to remedy this.
19/01/10 13:41:40 INFO input.FileInputFormat: Total input paths to process : 10
19/01/10 13:41:40 INFO mapreduce.JobSubmitter: number of splits:10
19/01/10 13:41:40 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_15
47102344447_0001
19/01/10 13:41:41 INFO impl.YarnClientImpl: Submitted application application_15
47102344447_0001
19/01/10 13:41:41 INFO mapreduce.Job: The url to track the job: http://10.212.1.
56:8088/proxy/application_1547102344447_0001/
19/01/10 13:41:41 INFO mapreduce.Job: Running job: job_1547102344447_0001
19/01/10 13:41:51 INFO mapreduce.Job: Job job_1547102344447_0001 running in uber
 mode : false
19/01/10 13:41:51 INFO mapreduce.Job: map 0% reduce 0%
19/01/10 13:42:00 INFO mapreduce.Job: map 30% reduce 0%
19/01/10 13:42:07 INFO mapreduce.Job: map 60% reduce 0%
19/01/10 13:42:15 INFO mapreduce.Job: map 90% reduce 0%
19/01/10 13:42:20 INFO mapreduce.Job: map 100% reduce 0%
19/01/10 13:42:21 INFO mapreduce.Job: map 100% reduce 100%
19/01/10 13:42:22 INFO mapreduce.Job: Job job_1547102344447_0001 completed succe
ssfully
19/01/10 13:42:22 INFO mapreduce.Job: Counters: 49
  File System Counters
    FILE: Number of bytes read=10038
    FILE: Number of bytes written=1377024
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=25177
    HDFS: Number of bytes written=47
    HDFS: Number of read operations=33
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
  Job Counters
    Launched map tasks=10
    Launched reduce tasks=1
    Data-local map tasks=10
    Total time spent by all maps in occupied slots (ms)=64194
    Total time spent by all reduces in occupied slots (ms)=5759
    Total time spent by all map tasks (ms)=64194
    Total time spent by all reduce tasks (ms)=5759
    Total vcore-milliseconds taken by all map tasks=64194
    Total vcore-milliseconds taken by all reduce tasks=5759
    Total megabyte-milliseconds taken by all map tasks=65734656
    Total megabyte-milliseconds taken by all reduce tasks=5897216
  Map-Reduce Framework
    Map input records=10
    Map output records=836
    Map output bytes=8360
    Map output materialized bytes=10092
    Input split bytes=1001
    Combine input records=0
    Combine output records=0
    Reduce input groups=5
    Reduce shuffle bytes=10092
    Reduce input records=836
    Reduce output records=5
    Spilled Records=1672
    Shuffled Maps =10
    Failed Shuffles=0
    Merged Map outputs=10
    GC time elapsed (ms)=1673
    CPU time spent (ms)=5280
    Physical memory (bytes) snapshot=2916249600
    Virtual memory (bytes) snapshot=4412289024
    Total committed heap usage (bytes)=2240806912
  Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
  File Input Format Counters
    Bytes Read=24096
  File Output Format Counters
    Bytes Written=47

```


BAB V HASIL UJI COBA DAN PEMBAHASAN

Pada bab ini dijelaskan tentang hasil dan pembahasan dari program yang telah dibuat.

5.1. Hasil Akuisisi Data

Pada penelitian ini, data yang digunakan adalah data rekaman CCTV di lima lokasi yang telah disebutkan sebelumnya. Data tersebut didapatkan dari Dinas Perhubungan Kota Surabaya pada tanggal 26 November 2018. Hasil akuisisi data dari video menghasilkan input sebanyak 1000 kendaraan, yang dapat dilihat pada Lampiran 1.

5.2. Hasil Uji Coba Sistem

Program yang telah dijalankan sebelumnya menghasilkan data pasangan titik awal dan titik akhir dari jalur perjalanan kendaraan-kendaraan yang melewati setiap CCTV yang diamatai beserta frekuensi kejadian pasangan – pasangan tersebut. Hasil running program akan menghasilkan output yang disimpan pada direktori yang telah disebutkan pada saat running, yaitu /user/output dengan nama “part-r-00000”. Berikut adalah cara menampilkan data output yang terdapat pada HDFS :

```
C:\Users\John>hdfs dfs -cat /user/output/part-r-00000
```

Gambar 5. Menampilkan Output dari Program

Sedangkan output dari running program tersebut adalah sebagai berikut :

```
1 - 2 - 3          32
2 - 3          635
2 - 3 - 4          16
2 - 3 - 4 - 5     20
4 - 5          132
```

Gambar 6. Output Program

Gambar 5.2 adalah hasil dari pengolahan data input pada HDFS oleh program MapReduce.

5.3. Analisa Hasil Pengolahan Data

Berdasarkan hasil yang telah didapatkan sebelumnya, maka dapat dianalisa bahwa sebanyak 635 kendaraan memiliki jalur perjalanan yang berawal di CCTV 2 dan berakhir di CCTV 3, yang berarti 635 kendaraan tersebut memasuki area pengamatan melalui Jl. Panglima Sudirman dan keluar dari pengamatan melalui Jl. Urip Sumoharjo. Pasangan jalur perjalanan kendaraan ini adalah pasangan titik awal dan titik akhir yang paling banyak dilalui oleh kendaraan. Sementara pasangan titik awal dan titik akhir yang paling sedikit dilalui adalah pasangan yang berawal dari Jl. Panglima Sudirman menuju Keputran. Selain itu, dapat dilihat bahwa kendaraan hanya dapat memasuki sistem transportasi melalui tiga titik, yaitu Jl. Kayoon, Jl. Panglima Sudirman, dan Keputran, yang berarti setiap kendaraan yang melewati Jl. Urip Sumoharjo pasti juga melewati Jl. Panglima Sudirman, dan setiap kendaraan yang melewati Jl. Raya Ngagel juga pasti melewati Keputran. Hal ini karena tidak ada kendaraan yang pertama kali terbaca pada Jl. Urip Sumoharjo ataupun Jl. Raya Ngagel. Hal ini juga sesuai dengan sketsa sistem transportasi yang telah dibuat pada BAB IV.

Total jumlah kendaraan pada data input adalah 1000 kendaraan, namun pada akhirnya hanya ada 835, sehingga terdapat sebanyak 165 kendaraan yang tadinya terdapat pada data input, namun tidak ditampilkan pada output. Hal ini diakibatkan karena beberapa kendaraan mengalami proses akuisisi seperti kasus 2 dan kasus 3. Dimana kasus 2 adalah kendaraan yang melewati beberapa CCTV dengan rentang waktu lebih dari 5 menit, sehingga satu kendaraan yang sama akan terbaca oleh program sebanyak dua kali. Namun masalah utamanya bukan pada kendaraan yang mengalami proses akuisisi seperti kasus 2, melainkan kendaraan – kendaraan yang mengalami proses akuisisi seperti kasus 3, yaitu setiap kendaraan hanya melewati satu CCTV, sehingga kendaraan tersebut tidak diproses. Selisih antara kendaraan yang mengalami proses akuisisi seperti kasus 3 dengan kendaraan yang mengalami proses

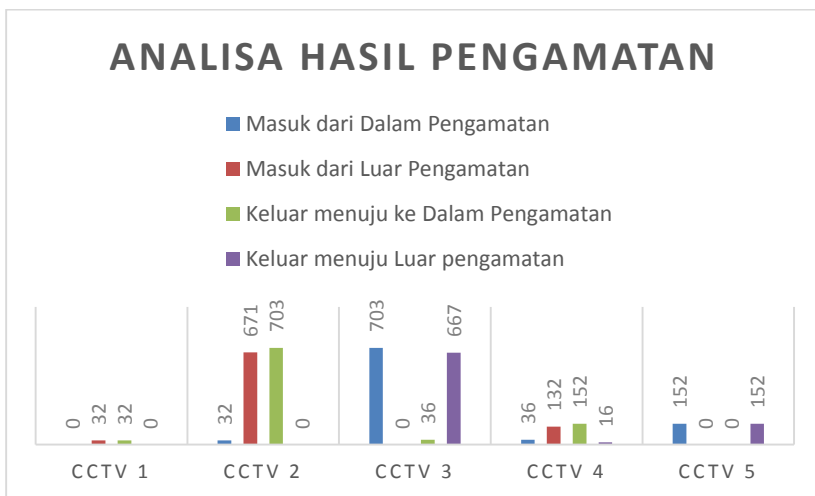
akuisisi seperti kasus 2 inilah data input yang tidak ditampilkan pada output, yaitu sebanyak 164 kendaraan.

Dari total 835 kendaraan tersebut, dapat dianalisa bahwa sebanyak 32 kendaraan masuk ke pengamatan melalui Jl. Kayoon, 671 kendaraan masuk ke pengamatan melalui Jl. Panglima sudirman, dan 132 lainnya masuk melalui keputran. Dari 32 kendaraan yang masuk ke Jl. Kayoon, seluruhnya akan menuju Jl. Panglima sudirman, sehingga jumlah kendaraan yang keluar dari Jl. Panglima sudirman berjumlah 703 kendaraan, dimana dari 703 kendaraan tersebut seluruhnya akan menuju Jl. Urip sumoharjo. Jl. Urip sumoharjo tidak menerima kendaraan dari luar pengamatan, yang berarti, sebanyak 703 kendaraan yang melewati Jl. Urip sumoharjo dipastikan adalah kendaraan yang melewati Jl. Panglima sudirman terlebih dahulu. Dari 703 kendaraan yang keluar dari Jl. Urip sumoharjo, 36 kendaraan diantaranya menuju keputran dan 667 kendaraan lainnya keluar dari pengamatan. Dikarenakan 36 kendaraan yang keluar dari Jl. Urip sumoharjo menuju keputran, maka jumlah kendaraan yang menuju keputran sebanyak 169 kendaraan. Dari 169 kendaraan tersebut, sebanyak 152 kendaraan menuju Jl. Raya ngagel dan sisanya sebanyak 16 kendaraan keluar dari pengamatan. Seluruh kendaraan yang menuju cctv adalah kendaraan yang sebelumnya telah melewati keputran. Akhirnya, seluruh kendaraan yang keluar dari Jl. Raya ngagel akan keluar dari pengamatan. Analisa di atas dapat dirangkum dalam bentuk tabel seperti yang tertera berikut :

Tabel 5.1. Analisa Hasil Pengamatan

CCTV	Masuk dari		Keluar ke	
	Dalam Pengamatan	Luar Pengamatan	Dalam Pengamatan	Luar Pengamatan
1	-	32	32	-
2	32	671	703	-
3	703	-	36	667
4	36	132	152	16
5	152	-	-	152

Selain dapat disajikan dalam bentuk tabel, hasil analisa tersebut juga dapat disajikan dalam bentuk diagram sebagai berikut :



Gambar 7. Analisa Hasil Pengamatan

BAB VI PENUTUP

Pada bab ini berisi tentang beberapa kesimpulan yang dihasilkan berdasarkan penelitian yang telah dilaksanakan dan saran yang dapat digunakan jika penelitian ini dikembangkan.

6.1. Kesimpulan

Berdasarkan analisis terhadap hasil pengujian program, maka dapat diambil kesimpulan sebagai berikut :

1. Data input yang berhasil diakuisisi dari data video yang diperoleh dari Dinas Perhubungan Kota Surabaya pada hari Senin tanggal 26 November 2018 selama 30 menit pada pukul 11.00 – 11.30 sebanyak 1000 kendaraan. Data tersebut lalu dikonversi menjadi *array of String* 1D sehingga dapat diproses oleh program MapReduce.
2. Data input yang telah terkonversi diimplementasikan ke dalam ekosistem Hadoop dengan cara diinputkan ke HDFS dan diproses oleh program MapReduce.
3. Jalur kendaraan yang paling banyak dialui oleh kendaraan adalah jalur perjalanan yang berawal dari Jl. Panglima sudirman dan berakhir pada Jl. Urip sumoharjo. Sehingga jika pemerintah kota surabaya akan membuka rute atau trayek transportasi umum dalam kota, maka sebaiknya pemerintah kota surabaya membuka rute yang berawal dari Jl. Panglima sudirman dan berakhir di Jl. Urip sumoharjo.

6.2. Saran

Ada beberapa hal yang penulis sarankan untuk pengembangan penelitian selanjutnya :

1. Kualitas gambar CCTV ditingkatkan agar setiap kendaraan yang melewati CCTV tersebut dapat terlihat plat nomornya.
2. Terapkan teknologi pembacaan plat nomor secara otomatis sehingga proses akuisisi data menjadi jauh lebih cepat.

3. Jika kedua hal di atas telah diterapkan, maka dapat dilakukan penelitian tentang sistem transportasi Kota Surabaya secara keseluruhan, tidak menggunakan *sampling*, sehingga hasil dari penelitian tersebut dapat langsung diimplementasikan sesuai kebutuhan sistem transportasi Kota Surabaya.

DAFTAR PUSTAKA

- [1] Zeng, Gang (2015). "Application of Big Data in Intelligent Traffic System". IOSR Journal of Computer Engineering (IOSR-JCE). 17 (1).
- [2] <https://surabayakota.bps.go.id/statictable> (dikunjungi pada 7 Desember 2018).
- [3] Laney, Doug (2001). "3D data management: Controlling data volume, velocity and variety". META Group Research Note. 6 (70).
- [4] Goes, Paulo B. (2014). "Design science research in top information systems journals". MIS Quarterly: Management Information Systems. 38 (1)
- [5] Reichman, O.J.; Jones, M.B.; Schildhauer, M.P. (2011). "Challenges and Opportunities of Open Data in Ecology". Science. 331 (6018): 703–5.
- [6] Segaran, Toby; Hammerbacher, Jeff (2009). Beautiful Data: The Stories Behind Elegant Data Solutions. O'Reilly Media. p. 257.
- [7] Hilbert, Martin; López, Priscila (2011). "The World's Technological Capacity to Store, Communicate, and Compute Information". Science. 332 (6025): 60–65.
- [8] <http://hadoop.apache.org/releases.html> (dikunjungi pada 20 September 2018)
- [9] Wickham, Hadley (2011). "The split-apply-combine strategy for data analysis". Journal of Statistical Software. 40: 1–29.
- [10] Lämmel, R. (2008). "Google's Map Reduce programming model — Revisited". Science of Computer Programming. 70: 1–30.
- [11] Ullman, J. D. (2012). "Designing good MapReduce algorithms". XRDS: Crossroads, The ACM Magazine for Students. Association for Computing Machinery. 19: 30.
- [12] Zhu, Li; Yu, Fei R.; Wang, Yige; Ning, Bin; Tang, Tao (2018). "Big Data Analytics in Intelligent Transportation Systems: A Survey".
- [13] <http://sits.dishub.surabaya.go.id/ver2/tentang-sits> (dikunjungi pada 21 Agustus 2018).

LAMPIRAN

1. Hasil Akuisisi Data

0016 1 k1 0103 2 k1 0125 3 k1 0234 1 k2 0539 2 k2 0621 3 k2
0253 1 k3 0555 2 k3 0636 3 k3 0346 1 k4 0347 1 k5 0354 1 k6
0652 2 k6 0655 2 k4 0721 3 k6 0823 3 k4 0443 1 k7 0447 1 k8
0452 1 k9 0743 2 k7 0752 2 k8 0844 3 k7 0904 3 k8 0001 2 k10
0007 2 k11 0007 2 k12 0007 2 k13 0012 2 k14 0030 3 k10 0045 3
k12 0047 3 k13 0056 3 k11 0057 3 k14 0014 2 k15 0017 2 k16
0019 2 k17 0019 2 k18 0020 2 k19 0023 2 k20 0028 2 k21 0028 2
k22 0058 3 k17 0100 3 k15 0100 3 k18 0104 3 k16 0106 3 k20
0106 3 k21 0107 3 k19 0113 3 k22 0547 4 k16 0640 5 k16 0031 2
k23 0032 2 k24 0033 2 k25 0037 2 k26 0041 2 k27 0107 3 k25
0108 3 k23 0110 3 k24 0110 3 k26 0113 3 k27 0042 2 k28 0043 2
k29 0043 2 k30 0046 2 k31 0048 2 k32 0117 3 k28 0120 3 k29
0123 3 k30 0126 3 k31 0128 3 k32 0055 2 k33 0056 2 k34 0056 2
k35 0101 2 k36 0104 2 k37 0105 2 k38 0139 3 k33 0142 3 k34
0150 3 k35 0158 3 k36 0204 3 k37 0206 3 k38 0106 2 k39 0106 2
k40 0108 2 k41 0109 2 k42 0116 2 k43 0116 2 k44 0140 3 k39
0142 3 k40 0146 3 k41 0154 3 k42 0155 3 k43 0155 3 k44 0117 2
k45 0119 2 k46 0121 2 k47 0125 2 k48 0127 2 k49 0150 3 k45
0156 3 k46 0158 3 k47 0204 3 k48 0206 3 k49 0128 2 k50 0157 3
k50 0141 2 k51 0142 2 k52 0143 2 k53 0143 2 k54 0146 2 k55
0147 2 k56 0148 2 k57 0148 2 k58 0150 2 k59 0151 2 k60 0151 2
k61 0212 3 k51 0214 3 k52 0216 3 k53 0223 3 k54 0226 3 k55
0227 3 k56 0228 3 k57 0228 3 k58 0229 3 k59 0230 3 k60 0235 3
k61 0153 2 k62 0153 2 k63 0154 2 k64 0159 2 k65 0201 2 k66
0202 2 k67 0203 2 k68 0230 3 k62 0231 3 k63 0233 3 k64 0237 3
k65 0238 3 k66 0239 3 k67 0244 3 k68 0204 2 k69 0205 2 k70
0209 2 k71 0210 2 k72 0210 2 k73 0214 2 k74 0240 3 k70 0240 3
k71 0245 3 k72 0253 3 k73 0301 3 k74 0315 3 k69 0215 2 k75
0216 2 k76 0217 2 k77 0219 2 k78 0220 2 k79 0223 2 k80 0224 2
k81 0224 2 k82 0224 2 k83 0225 2 k84 0304 3 k75 0304 3 k76
0316 3 k77 0317 3 k78 0318 3 k79 0320 3 k80 0322 3 k81 0424 3
k82 0425 3 k83 0438 3 k84 0234 2 k85 0243 2 k86 0356 3 k85
0427 3 k86 0249 2 k87 0253 2 k88 0255 2 k89 0255 2 k90 0257 2
k91 0258 2 k92 0424 3 k87 0426 3 k88 0432 3 k89 0436 3 k90

0437 3 k91 0444 3 k92 0303 2 k93 0304 2 k94 0307 2 k95 0308 2
k96 0309 2 k97 0356 3 k93 0421 3 k94 0425 3 k95 0429 3 k96
0437 3 k97 0812 4 k96 0822 4 k97 0817 5 k96 0827 5 k97 0316 2
k98 0316 2 k99 0317 2 k100 0318 2 k101 0320 2 k102 0323 2
k103 0325 2 k104 0325 2 k105 0325 2 k106 0326 2 k107 0428 3
k98 0431 3 k99 0434 3 k100 0437 3 k101 0438 3 k102 0439 3
k103 0441 3 k104 0443 3 k105 0445 3 k106 0449 3 k107 0329 2
k108 0330 2 k109 0331 2 k110 0333 2 k111 0334 2 k112 0335 2
k113 0338 2 k114 0450 3 k108 0450 3 k109 0454 3 k110 0457 3
k111 0500 3 k112 0504 3 k113 0514 3 k114 0343 2 k115 0345 2
k116 0352 2 k117 0442 3 k115 0443 3 k116 0444 3 k117 0358 2
k118 0358 2 k119 0400 2 k120 0401 2 k121 0403 2 k122 0403 2
k123 0404 2 k124 0405 2 k125 0406 2 k126 0408 2 k127 0443 3
k118 0450 3 k119 0450 3 k120 0452 3 k121 0454 3 k122 0502 3
k123 0500 3 k124 0502 3 k125 0504 3 k126 0506 3 k127 0409 2
k128 0410 2 k129 0411 2 k130 0413 2 k131 0413 2 k132 0416 2
k133 0417 2 k134 0418 2 k135 0450 3 k128 0455 3 k129 0458 3
k130 0500 3 k131 0503 3 k132 0503 3 k133 0504 3 k134 0510 3
k135 0420 2 k136 0420 2 k137 0421 2 k138 0423 2 k139 0423 2
k140 0425 2 k141 0425 2 k142 0427 2 k143 0428 2 k144 0429 2
k145 0504 3 k136 0505 3 k137 0506 3 k138 0508 3 k139 0508 3
k140 0509 3 k141 0510 3 k142 0516 3 k143 0517 3 k144 0524 3
k145 1012 4 k144 1203 4 k145 0431 2 k146 0432 2 k147 0432 2
k148 0433 2 k149 0434 2 k150 0435 2 k151 0436 2 k152 0438 2
k153 0438 2 k154 0441 2 k155 0519 3 k146 0520 3 k147 0526 3
k148 0527 3 k149 0528 3 k150 0528 3 k151 0530 3 k152 0531 3
k153 0531 3 k154 0533 3 k155 0447 2 k156 0448 2 k157 0448 2
k158 0451 2 k159 0455 2 k160 0456 2 k161 0536 3 k156 0536 3
k157 0537 3 k158 0540 3 k159 0546 3 k160 0545 3 k161 0458 2
k162 0459 2 k163 0541 3 k162 0551 3 k163 0325 4 k172 0328 4
k173 0331 4 k174 0333 4 k175 0334 4 k176 0337 4 k177 0340 5
k172 0343 5 k173 0345 5 k174 0354 5 k175 0356 5 k176 0318 4
k178 0318 4 k179 0326 4 k180 0328 4 k181 0423 5 k178 0426 5
k179 0426 5 k180 0343 4 k182 0345 4 k183 0350 4 k184 0353 4
k185 0355 4 k186 0356 4 k187 0438 5 k182 0439 5 k183 0442 5
k184 0447 5 k185 0449 5 k186 0542 5 k187 0421 4 k188 0423 4
k189 0424 4 k190 0428 4 k191 0430 4 k192 0432 4 k193 0435 4

k194 0438 4 k195 0545 5 k188 0550 5 k189 0552 5 k190 0555 5
k191 0601 5 k192 0602 5 k193 0633 1 k788 0639 1 k789 0643 1
k196 0655 1 k197 0948 2 k788 0953 2 k195 0957 2 k196 1001 2
k197 1044 3 k788 1049 3 k789 1054 3 k196 1115 3 k197 0905 1
k198 0910 1 k199 0936 1 k200 0945 1 k201 1201 2 k198 1227 2
k199 1242 2 k200 1245 3 k198 1323 3 k199 1339 3 k200 0501 2
k202 0505 2 k207 0509 2 k203 0510 2 k204 0511 2 k205 0511 2
k206 0540 3 k202 0543 3 k203 0546 3 k204 0547 3 k205 0549 3
k206 0550 3 k207 0513 2 k208 0515 2 k209 0518 2 k210 0518 2
k211 0519 2 k212 0521 2 k213 0555 3 k208 0557 3 k209 0559 3
k210 0601 3 k211 0602 3 k212 0604 3 k213 0525 2 k214 0527 2
k215 0529 2 k216 0529 2 k217 0531 2 k218 0533 2 k219 0534 2
k220 0534 2 k221 0603 3 k214 0605 3 k215 0606 3 k216 0609 3
k217 0610 3 k218 0611 3 k219 0611 3 k220 0611 3 k221 0706 4
k221 0800 5 k221 0537 2 k222 0538 2 k223 0538 2 k224 0541 2
k225 0542 2 k226 0546 2 k227 0612 3 k222 0619 3 k223 0621 3
k224 0622 3 k225 0622 3 k226 0623 3 k227 0548 2 k228 0550 2
k229 0551 2 k230 0553 2 k231 0555 2 k232 0555 2 k233 0556 2
k234 0619 3 k228 0621 3 k229 0629 3 k230 0631 3 k231 0634 3
k232 0634 3 k233 0637 3 k234 0603 2 k235 0606 2 k236 0612 2
k237 0612 2 k238 0637 3 k235 0638 3 k236 0641 3 k237 0656 3
k238 0614 2 k239 0615 2 k240 0616 2 k241 0617 2 k242 0619 2
k243 0622 2 k244 0623 2 k245 0644 3 k239 0645 3 k240 0653 3
k241 0658 3 k242 0700 3 k243 0702 3 k244 0658 3 k245 0630 2
k246 0634 2 k247 0636 2 k248 0638 2 k249 0639 2 k250 0705 3
k246 0707 3 k247 0712 3 k248 0712 3 k249 0717 3 k250 0641 2
k251 0642 2 k252 0642 2 k253 0647 2 k254 0648 2 k255 0713 3
k251 0721 3 k252 0740 3 k253 0817 3 k254 0825 3 k255 0905 4
k255 0652 2 k256 0652 2 k257 0652 2 k258 0655 2 k259 0655 2
k260 0656 2 k261 0658 2 k262 0700 2 k263 0821 3 k256 0823 3
k257 0823 3 k258 0823 3 k259 0823 3 k260 0826 3 k261 0832 3
k262 0845 3 k263 0929 4 k262 0941 4 k263 1019 5 k263 0703 2
k264 0706 2 k265 0708 2 k266 0826 3 k264 0827 3 k265 0830 3
k266 0711 2 k267 0711 2 k268 0712 2 k269 0715 2 k270 0716 2
k271 0717 2 k272 0720 2 k273 0721 2 k274 0831 3 k267 0831 3
k268 0832 3 k269 0833 3 k270 0834 3 k271 0835 3 k272 0836 3
k273 0839 3 k274 0934 4 k274 0723 2 k275 0724 2 k276 0724 2

k277 0726 2 k278 0726 2 k279 0727 2 k280 0730 2 k281 0731 2
k282 0732 2 k283 0733 2 k284 0734 2 k285 0734 2 k285 0735 2
k286 0737 2 k287 0738 2 k288 0840 3 k275 0840 3 k276 0840 3
k377 0841 3 k378 0841 3 k379 0841 3 k380 0842 3 k381 0842 3
k382 0842 3 k382 0843 3 k384 0843 3 k385 0844 3 k386 0844 3
k387 0845 3 k288 0740 2 k289 0740 2 k290 0743 2 k291 0744 2
k292 0746 2 k293 0748 2 k294 0748 2 k295 0750 2 k296 0752 2
k297 0757 2 k298 0758 2 k299 0759 2 k300 0759 2 k301 0800 2
k302 0846 3 k289 0847 3 k290 0848 3 k291 0849 3 k292 0850 3
k293 0851 3 k294 0852 3 k295 0853 3 k296 0854 3 k297 0855 3
k298 0856 3 k299 0857 3 k300 0900 3 k301 0903 3 k302 1032 4
k302 0801 2 k303 0802 2 k304 0907 3 k303 0911 3 k304 1044 4
k304 0803 2 k305 0804 2 k306 0805 2 k307 0806 2 k308 0807 2
k309 0808 2 k310 0809 2 k311 0810 2 k312 0811 2 k313 0812 2
k314 0813 2 k315 0814 2 k316 0815 2 k317 0816 2 k318 0817 2
k319 0818 2 k320 0819 2 k321 0820 2 k322 0821 2 k323 0822 2
k324 0823 2 k325 0824 2 k326 0825 2 k327 0826 2 k328 0827 2
k329 0828 2 k330 0829 2 k331 0830 2 k332 0831 2 k333 0832 2
k334 0833 2 k335 0834 2 k336 0835 2 k337 0836 2 k338 0837 2
k339 0838 2 k340 0839 2 k341 0840 2 k342 0841 2 k343 0842 2
k344 0843 2 k345 0844 2 k346 0845 2 k347 0846 2 k348 0847 2
k349 0848 2 k350 0850 2 k351 0852 2 k352 0854 2 k353 0856 2
k354 0858 2 k355 0900 2 k356 0902 2 k357 0904 2 k358 0907 2
k359 0910 2 k360 0913 2 k361 0916 2 k362 0919 2 k363 0922 2
k364 0925 2 k365 0928 2 k366 0931 2 k367 0934 2 k368 0937 2
k369 0947 2 k370 0957 2 k371 1007 2 k372 1037 2 k373 0912 3
k305 0913 3 k306 0914 3 k307 0915 3 k308 0916 3 k309 0917 3
k310 0918 3 k311 0919 3 k312 0920 3 k313 0921 3 k314 0922 3
k315 0923 3 k316 0924 3 k317 0925 3 k318 0926 3 k319 0927 3
k320 0928 3 k321 0929 3 k322 0930 3 k323 0931 3 k324 0933 3
k325 0935 3 k326 0937 3 k327 0939 3 k328 0941 3 k329 0943 3
k330 0945 3 k331 0947 3 k332 0949 3 k333 0951 3 k334 0953 3
k335 0955 3 k336 0957 3 k337 0959 3 k338 1001 3 k339 1004 3
k340 1007 3 k341 1010 3 k342 1013 3 k343 1016 3 k344 1019 3
k345 1022 3 k346 1025 3 k347 1028 3 k348 1031 3 k349 1035 3
k350 1040 3 k351 1045 3 k352 1050 3 k353 1055 3 k354 1100 3
k355 1101 3 k356 1102 3 k357 1103 3 k358 1104 3 k359 1105 3

k360 1106 3 k361 1107 3 k362 1108 3 k363 1109 3 k364 1110 3
k365 1112 3 k366 1113 3 k367 1114 3 k368 1115 3 k369 1120 3
k370 1125 3 k371 1126 3 k372 1131 3 k373 1246 4 k373 1313 5
k373 0555 4 k380 0559 4 k381 0600 4 k382 0604 4 k383 0605 4
k384 0606 4 k385 0608 4 k386 0608 4 k387 0610 4 k388 0613 4
k389 0614 4 k390 0616 4 k391 0617 4 k392 0618 4 k393 0621 4
k394 0621 4 k395 0622 4 k396 0624 4 k397 0624 4 k398 0626 4
k399 0626 4 k400 0629 4 k401 0632 4 k402 0634 4 k403 0636 4
k404 0643 4 k405 0700 4 k406 0703 4 k407 0717 4 k408 0700 5
k380 0702 5 k381 0704 5 k382 0706 5 k383 0708 5 k384 0710 5
k385 0712 5 k386 0714 5 k387 0716 5 k388 0718 5 k389 0720 5
k390 0722 5 k391 0724 5 k392 0725 5 k393 0726 5 k394 0727 5
k395 0728 5 k396 0729 5 k397 0730 5 k398 0731 5 k399 0732 5
k400 0733 5 k401 0734 5 k402 0736 5 k403 0738 5 k404 0740 5
k405 0742 5 k406 0745 5 k407 0748 5 k408 0750 4 k409 0753 4
k410 0756 4 k411 0759 4 k412 0802 4 k413 0805 4 k414 0808 4
k415 0811 4 k416 0817 4 k417 0823 4 k418 0829 4 k419 0835 4
k420 0841 4 k421 0847 4 k422 0853 4 k423 0859 4 k424 0911 4
k425 0919 5 k409 0922 5 k410 0925 5 k411 0928 5 k412 0931 5
k413 0934 5 k414 0937 5 k415 0941 5 k416 0944 5 k417 0951 5
k418 0958 5 k419 1005 5 k420 1012 5 k421 1019 5 k422 1026 5
k423 1033 5 k424 1035 5 k425 1050 1 k426 1358 2 k426 1443 3
k426 1216 1 k427 1218 1 k428 1236 1 k429 1237 1 k430 1516 2
k428 1527 2 k427 1542 2 k429 1543 2 k430 1629 3 k428 1704 3
k430 1705 3 k429 1715 3 k427 1352 1 k431 1724 2 k431 1042 2
k432 1043 2 k433 1044 2 k434 1045 2 k435 1046 2 k436 1047 2
k437 1048 2 k438 1049 2 k439 1050 2 k440 1051 2 k441 1052 2
k442 1054 2 k443 1056 2 k444 1058 2 k445 1100 2 k446 1102 2
k447 1104 2 k448 1106 2 k450 1108 2 k451 1110 2 k452 1112 2
k453 1114 2 k454 1116 2 k455 1118 2 k456 1120 2 k457 1122 2
k458 1124 2 k459 1126 2 k460 1128 2 k461 1130 2 k462 1132 2
k463 1134 2 k464 1132 3 k431 1133 3 k432 1134 3 k433 1135 3
k434 1136 3 k435 1137 3 k436 1138 3 k437 1139 3 k438 1140 3
k439 1141 3 k440 1142 3 k441 1143 3 k442 1144 3 k443 1145 3
k444 1146 3 k445 1147 3 k446 1148 3 k447 1149 3 k448 1150 3
k450 1151 3 k451 1152 3 k452 1153 3 k453 1154 3 k454 1155 3
k455 1156 3 k456 1157 3 k457 1158 3 k458 1159 3 k459 1216 3

k460 1227 3 k461 1229 3 k462 1231 3 k463 1244 3 k464 1309 4
k462 1312 4 k461 1314 4 k463 1318 4 k464 1425 4 k460 1426 5
k462 1433 5 k461 1449 5 k463 1138 2 k465 1140 2 k466 1142 2
k467 1144 2 k468 1146 2 k469 1148 2 k470 1149 2 k471 1150 2
k472 1152 2 k473 1154 2 k474 1156 2 k475 1158 2 k476 1213 2
k477 1224 2 k478 1245 3 k465 1246 3 k466 1247 3 k467 1248 3
k468 1249 3 k469 1254 3 k470 1256 3 k471 1258 3 k472 1300 3
k473 1303 3 k474 1305 3 k475 1307 3 k476 1309 3 k477 1315 3
k478 1423 4 k478 1619 5 k478 1224 2 k479 1225 2 k480 1226 2
k481 1227 2 k482 1228 2 k483 1229 2 k484 1230 2 k485 1231 2
k486 1232 2 k487 1233 2 k488 1235 2 k489 1237 2 k490 1239 2
k491 1241 2 k492 1244 2 k493 1247 2 k494 1250 2 k495 1300 2
k496 1309 2 k497 1316 3 k479 1317 3 k480 1318 3 k481 1319 3
k482 1320 3 k483 1321 3 k484 1322 3 k485 1323 3 k486 1324 3
k487 1325 3 k488 1330 3 k489 1336 3 k490 1342 3 k491 1346 3
k492 1348 3 k493 1349 3 k494 1350 3 k495 1350 3 k496 1352 3
k497 1425 4 k496 1427 4 k497 1604 5 k497 1622 5 k496 1315 2
k497 1316 2 k498 1317 2 k499 1318 2 k500 1319 2 k501 1321 2
k502 1323 2 k503 1325 2 k504 1337 2 k505 1339 2 k506 1340 2
k507 1341 2 k508 1342 2 k509 1343 2 k510 1343 2 k511 1344 2
k512 1345 2 k513 1345 2 k514 1346 2 k515 1347 2 k516 1348 2
k517 1353 3 k497 1354 3 k498 1355 3 k499 1356 3 k500 1357 3
k501 1359 3 k502 1401 3 k503 1403 3 k504 1405 3 k505 1407 3
k506 1409 3 k507 1412 3 k508 1415 3 k509 1418 3 k510 1421 3
k511 1424 3 k512 1427 3 k513 1430 3 k514 1433 3 k515 1436 3
k516 1439 3 k517 1700 4 k517 1356 2 k518 1357 2 k519 1358 2
k520 1359 2 k521 1400 2 k522 1402 2 k523 1404 2 k524 1406 2
k525 1408 2 k526 1410 2 k527 1413 2 k528 1416 2 k529 1419 2
k530 1422 2 k531 1425 2 k532 1426 2 k533 1427 2 k534 1429 2
k535 1440 3 k518 1441 3 k519 1442 3 k520 1443 3 k521 1444 3
k522 1446 3 k523 1448 3 k524 1450 3 k525 1452 3 k526 1454 3
k527 1457 3 k528 1500 3 k529 1503 3 k530 1506 3 k531 1509 3
k532 1519 3 k533 1529 3 k534 1535 3 k535 1713 4 k535 1755 5
k535 1432 2 k536 1433 2 k537 1434 2 k538 1435 2 k539 1436 2
k540 1438 2 k541 1440 2 k542 1442 2 k543 1445 2 k544 1536 3
k536 1537 3 k537 1538 3 k538 1539 3 k539 1540 3 k540 1545 3
k541 1550 3 k542 1555 3 k543 1558 3 k544 1716 4 k544 1448 2

k545 1449 2 k546 1450 2 k547 1451 2 k548 1452 2 k549 1453 2
k550 1454 2 k551 1455 2 k552 1456 2 k553 1457 2 k554 1459 2
k555 1501 2 k556 1503 2 k557 1505 2 k558 1506 2 k559 1507 2
k560 1559 3 k545 1600 3 k546 1601 3 k547 1602 3 k548 1603 3
k549 1604 3 k550 1605 3 k551 1606 3 k552 1607 3 k553 1608 3
k554 1609 3 k555 1610 3 k556 1611 3 k557 1612 3 k558 1612 3
k559 1614 3 k560 1722 4 k560 1803 5 k560 0951 4 k561 0952 4
k562 0953 4 k563 0954 4 k564 0955 4 k565 0957 4 k566 0959 4
k567 1001 4 k569 1003 4 k570 1005 4 k571 1015 4 k572 1030 4
k573 1049 4 k574 1149 5 k561 1153 5 k568 1157 5 k574 1143 4
k575 1144 4 k576 1145 4 k577 1146 4 k578 1147 4 k579 1149 4
k580 1151 4 k581 1153 4 k582 1155 4 k583 1157 4 k584 1200 4
k585 1203 4 k586 1206 4 k587 1209 4 k588 1212 4 k589 1216 4
k590 1220 4 k591 1224 4 k592 1228 4 k593 1232 4 k594 1252 4
k595 1307 4 k596 1318 4 k597 1411 5 k575 1412 5 k576 1413 5
k577 1414 5 k578 1415 5 k579 1417 5 k580 1419 5 k581 1421 5
k582 1423 5 k583 1425 5 k584 1428 5 k585 1431 5 k586 1434 5
k587 1437 5 k588 1440 5 k589 1444 5 k590 1448 5 k591 1449 5
k592 1450 5 k593 1451 5 k594 1452 5 k595 1453 5 k596 1454 5
k597 1343 4 k598 1345 4 k599 1346 4 k600 1347 4 k601 1348 4
k602 1349 4 k603 1351 4 k604 1353 4 k605 1355 4 k606 1357 4
k607 1359 4 k608 1402 4 k609 1405 4 k610 1408 4 k611 1411 4
k612 1414 4 k613 1418 4 k614 1422 4 k615 1426 4 k616 1430 4
k617 1434 4 k618 1436 4 k619 1438 4 k620 1439 4 k621 1549 5
k610 1555 5 k611 1601 5 k612 1607 5 k613 1613 5 k614 1624 5
k615 1635 5 k616 1646 5 k617 1657 5 k618 1708 5 k619 1728 5
k620 1743 5 k621 1509 1 k622 1807 2 k622 1840 3 k622 1611 1
k623 1912 2 k623 2050 3 k623 1709 1 k624 1712 1 k625 1716 1
k626 2016 2 k625 2022 2 k626 2106 3 k625 2113 3 k626 1819 1
k627 2125 2 k627 2213 3 k627 1929 1 k628 1934 1 k629 1937 1
k630 1941 1 k631 2243 2 k629 2246 2 k628 2246 2 k630 2422 3
k630 2426 3 k628 2426 3 k629 1508 2 k630 1509 2 k631 1510 2
k632 1511 2 k633 1512 2 k634 1514 2 k635 1516 2 k636 1518 2
k637 1520 2 k638 1522 2 k639 1525 2 k640 1528 2 k641 1531 2
k642 1534 2 k643 1537 2 k644 1539 2 k645 1541 2 k646 1543 2
k647 1545 2 k648 1547 2 k649 1548 2 k650 1549 2 k651 1550 2
k652 1551 2 k653 1552 2 k654 1553 2 k655 1554 2 k656 1555 2

k657 1556 2 k658 1557 2 k659 1558 2 k660 1600 2 k661 1602 2
k662 1604 2 k663 1605 2 k664 1615 3 k630 1616 3 k631 1617 3
k632 1618 3 k633 1619 3 k634 1621 3 k635 1623 3 k636 1625 3
k637 1627 3 k638 1629 3 k639 1632 3 k640 1635 3 k641 1638 3
k642 1641 3 k643 1644 3 k644 1645 3 k645 1646 3 k646 1647 3
k647 1648 3 k648 1649 3 k649 1650 3 k650 1651 3 k651 1652 3
k652 1653 3 k653 1654 3 k654 1655 3 k655 1656 3 k656 1657 3
k657 1658 3 k658 1659 3 k659 1700 3 k660 1701 3 k661 1702 3
k662 1703 3 k663 1704 3 k664 1727 4 k664 1808 5 k664 1607 2
k665 1608 2 k666 1609 2 k667 1610 2 k668 1611 2 k669 1613 2
k670 1615 2 k671 1617 2 k672 1619 2 k673 1621 2 k674 1624 2
k675 1627 2 k676 1630 2 k677 1633 2 k678 1636 2 k679 1638 2
k680 1640 2 k681 1642 2 k682 1644 2 k683 1646 2 k684 1648 2
k685 1650 2 k686 1652 2 k687 1654 2 k688 1656 2 k689 1658 2
k690 1700 2 k691 1706 2 k692 1712 2 k693 1705 3 k665 1706 3
k666 1707 3 k667 1708 3 k668 1709 3 k669 1711 3 k670 1713 3
k671 1715 3 k672 1717 3 k673 1719 3 k674 1722 3 k675 1725 3
k676 1728 3 k677 1731 3 k678 1734 3 k679 1738 3 k680 1742 3
k681 1746 3 k682 1750 3 k683 1754 3 k684 1759 3 k685 1804 3
k686 1809 3 k687 1814 3 k688 1819 3 k689 1820 3 k690 1821 3
k691 1822 3 k692 1823 3 k693 1918 4 k693 1714 2 k694 1715 2
k695 1716 2 k696 1717 2 k697 1718 2 k698 1720 2 k699 1722 2
k700 1724 2 k701 1726 2 k702 1728 2 k703 1731 2 k704 1734 2
k705 1737 2 k706 1740 2 k707 1743 2 k708 1745 2 k709 1747 2
k710 1749 2 k711 1751 2 k712 1753 2 k713 1754 2 k714 1755 2
k716 1756 2 k717 1757 2 k718 1758 2 k719 1759 2 k720 1800 2
k721 1801 2 k722 1802 2 k723 1803 2 k724 1804 2 k725 1805 2
k726 1807 2 k727 1808 2 k728 1810 2 k729 1811 2 k730 1824 3
k694 1825 3 k695 1826 3 k696 1827 3 k697 1828 3 k698 1829 3
k699 1830 3 k700 1831 3 k701 1832 3 k702 1833 3 k703 1834 3
k704 1835 3 k705 1836 3 k706 1837 3 k707 1838 3 k708 1839 3
k709 1839 3 k710 1839 3 k711 1840 3 k712 1840 3 k713 1841 3
k714 1841 3 k716 1842 3 k717 1842 3 k718 1843 3 k719 1843 3
k720 1844 3 k721 1844 3 k722 1845 3 k723 1845 3 k724 1846 3
k725 1846 3 k726 1847 3 k727 1847 3 k728 1848 3 k729 1849 3
k730 1920 4 k730 1812 2 k731 1814 2 k732 1816 2 k733 1818 2
k734 1822 2 k735 1826 2 k736 1850 3 k731 1852 3 k732 1854 3

k733 1856 3 k734 1901 3 k735 1905 3 k736 1929 4 k736 1957 5
k736 1828 2 k737 1829 2 k738 1830 2 k739 1831 2 k740 1832 2
k741 1834 2 k742 1836 2 k743 1838 2 k744 1840 2 k745 1842 2
k746 1845 2 k747 1848 2 k748 1851 2 k749 1854 2 k750 1857 2
k751 1901 2 k752 1902 2 k753 1903 2 k754 1904 2 k755 1904 2
k756 1906 3 k737 1907 3 k738 1908 3 k739 1909 3 k740 1910 3
k741 1912 3 k742 1914 3 k743 1916 3 k744 1918 3 k745 1920 3
k746 1925 3 k747 1930 3 k748 1935 3 k749 1940 3 k750 1945 3
k751 1950 3 k752 1955 3 k753 2000 3 k754 2009 3 k755 2019 3
k756 2148 4 k755 2151 4 k756 1905 2 k757 1906 2 k758 1907 2
k759 1908 2 k760 1909 2 k761 1911 2 k762 1913 2 k763 1915 2
k764 1917 2 k765 1919 2 k766 1922 2 k767 1925 2 k768 1928 2
k769 1931 2 k770 1934 2 k771 1938 2 k772 1942 2 k773 1946 2
k774 1950 2 k775 1954 2 k776 1959 2 k777 2004 2 k778 2009 2
k779 2010 2 k780 2011 2 k781 2012 2 k782 2013 2 k783 2014 2
k784 2015 2 k785 2015 2 k786 2016 2 k787 2020 3 k757 2021 3
k758 2022 3 k759 2023 3 k760 2024 3 k761 2026 3 k762 2028 3
k763 2030 3 k764 2032 3 k765 2034 3 k766 2037 3 k767 2040 3
k768 2043 3 k769 2046 3 k770 2049 3 k771 2054 3 k772 2059 3
k773 2104 3 k774 2109 3 k775 2114 3 k776 2124 3 k777 2134 3
k778 2144 3 k779 2154 3 k780 2204 3 k781 2114 3 k782 2115 3
k783 2116 3 k784 2117 3 k785 2118 3 k786 2220 3 k787 2237 4
k787 2312 5 k787 1607 4 k790 1608 4 k791 1609 4 k792 1610 4
k793 1611 4 k794 1613 4 k795 1615 4 k796 1617 4 k797 1619 4
k798 1621 4 k799 1631 4 k800 1641 4 k801 1651 4 k802 1701 4
k803 1711 4 k804 1731 4 k805 1741 4 k806 1742 4 k807 1743 4
k808 1744 4 k809 1745 4 k810 1745 4 k811 1746 4 k812 1746 4
k813 1747 4 k814 1633 5 k790 1726 5 k791 1727 5 k792 1728 5
k793 1729 5 k794 1731 5 k795 1733 5 k796 1735 5 k797 1737 5
k798 1739 5 k799 1744 5 k800 1749 5 k801 1754 5 k802 1759 5
k803 1804 5 k804 1809 5 k805 1810 5 k806 1811 5 k807 1812 5
k808 1814 5 k809 1906 5 k810 1907 5 k811 1836 4 k815 1837 4
k816 1838 4 k817 1839 4 k818 1850 4 k819 1852 4 k820 1854 4
k821 1856 4 k822 1858 4 k823 1900 4 k824 1905 4 k825 1910 4
k826 1915 4 k827 1920 4 k828 1925 4 k829 1929 4 k830 1914 5
k815 1915 5 k816 1916 5 k817 1917 5 k818 1918 5 k819 1920 5
k820 1922 5 k821 1924 5 k822 1926 5 k823 1928 5 k824 1933 5

k825 1938 5 k826 1943 5 k827 1948 5 k828 1953 5 k829 1957 5
k830 1958 4 k831 2000 4 k832 2013 4 k833 2015 4 k834 2019 4
k835 2025 4 k836 2031 4 k837 2033 4 k838 2039 4 k839 2043 4
k840 2053 5 k831 2054 5 k832 2055 5 k833 2056 5 k834 2057 5
k835 2102 5 k836 2107 5 k837 2112 5 k838 2117 5 k839 2118 5
k840 2103 1 k841 2151 1 k842 2201 1 k843 2239 1 k844 2544 2
k844 2551 3 k844 2422 1 k845 2441 1 k846 2716 2 k845 2744 2
k846 2837 3 k845 2904 3 k846 2017 2 k847 2018 2 k848 2019 2
k849 2020 2 k850 2022 2 k851 2024 2 k852 2026 2 k853 2028 2
k854 2030 2 k855 2033 2 k856 2036 2 k857 2039 2 k858 2104 2
k859 2221 3 k847 2222 3 k848 2223 3 k849 2224 3 k850 2225 3
k851 2227 3 k852 2229 3 k853 2231 3 k854 2233 3 k855 2235 3
k856 2238 3 k857 2241 3 k858 2245 3 k859 2357 4 k859 2536 5
k859 2106 2 k860 2107 2 k861 2108 2 k862 2109 2 k863 2110 2
k864 2112 2 k865 2114 2 k866 2116 2 k867 2118 2 k868 2120 2
k869 2123 2 k870 2126 2 k871 2129 2 k872 2132 2 k873 2135 2
k874 2139 2 k875 2143 2 k876 2147 2 k877 2151 2 k878 2155 2
k879 2156 2 k880 2157 2 k881 2158 2 k882 2159 2 k883 2200 2
k884 2201 2 k885 2201 2 k886 2202 2 k887 2202 2 k888 2203 2
k889 2203 2 k890 2204 2 k891 2204 2 k892 2205 2 k893 2205 2
k894 2246 3 k860 2247 3 k861 2248 3 k862 2249 3 k863 2250 3
k864 2252 3 k865 2254 3 k866 2256 3 k867 2258 3 k868 2300 3
k869 2303 3 k870 2306 3 k871 2309 3 k872 2312 3 k873 2315 3
k874 2319 3 k875 2323 3 k876 2327 3 k877 2331 3 k878 2335 3
k879 2340 3 k880 2345 3 k881 2350 3 k882 2355 3 k883 2400 3
k884 2401 3 k885 2402 3 k886 2403 3 k887 2404 3 k888 2405 3
k889 2406 3 k890 2406 3 k891 2407 3 k892 2407 3 k893 2407 3
k894 2633 4 k894 2207 2 k895 2208 2 k896 2209 2 k897 2210 2
k898 2212 2 k899 2214 2 k900 2216 2 k901 2218 2 k902 2220 2
k903 2223 2 k904 2226 2 k905 2229 2 k906 2232 2 k907 2235 2
k908 2239 2 k909 2243 2 k910 2247 2 k911 2251 2 k912 2255 2
k913 2300 2 k914 2305 2 k915 2310 2 k916 2311 2 k917 2312 2
k918 2313 2 k919 2314 2 k920 2315 2 k921 2316 2 k922 2316 2
k923 2317 2 k924 2317 2 k925 2318 2 k926 2318 2 k927 2318 2
k928 2318 2 k929 2318 2 k930 2408 3 k895 2409 3 k896 2410 3
k897 2412 3 k898 2414 3 k899 2416 3 k900 2418 3 k901 2420 3
k902 2423 3 k903 2426 3 k904 2429 3 k905 2432 3 k906 2435 3

k907 2439 3 k908 2443 3 k909 2447 3 k910 2448 3 k911 2449 3
k912 2450 3 k913 2451 3 k914 2452 3 k915 2453 3 k916 2454 3
k917 2455 3 k918 2456 3 k919 2456 3 k920 2457 3 k921 2457 3
k922 2458 3 k923 2458 3 k924 2458 3 k925 2458 3 k926 2458 3
k927 2458 3 k928 2458 3 k929 2458 3 k930 2632 4 k930 2736 5
k930 2323 2 k931 2324 2 k932 2325 2 k933 2327 2 k934 2329 2
k935 2331 2 k936 2333 2 k937 2335 2 k938 2338 2 k939 2341 2
k940 2344 2 k941 2347 2 k942 2350 2 k943 2354 2 k944 2358 2
k945 2402 2 k946 2406 2 k947 2410 2 k948 2415 2 k949 2420 2
k950 2425 2 k951 2426 2 k952 2427 2 k953 2428 2 k954 2429 2
k955 2430 2 k956 2431 2 k957 2432 2 k958 2433 2 k959 2434 2
k960 2435 2 k961 2436 2 k962 2437 2 k963 2438 2 k964 2438 2
k965 2439 2 k964 2459 3 k931 2500 3 k932 2501 3 k933 2502 3
k934 2503 3 k935 2504 3 k936 2505 3 k937 2506 3 k938 2507 3
k939 2508 3 k940 2509 3 k941 2510 3 k942 2511 3 k943 2512 3
k944 2513 3 k945 2514 3 k946 2515 3 k947 2516 3 k948 2517 3
k949 2518 3 k950 2519 3 k951 2520 3 k952 2521 3 k953 2522 3
k954 2523 3 k955 2524 3 k956 2525 3 k957 2525 3 k958 2525 3
k959 2526 3 k960 2526 3 k961 2526 3 k962 2527 3 k963 2527 3
k964 2527 3 k965 2527 3 k964 2634 4 k964 2845 5 k964 2441 2
k965 2442 2 k966 2443 2 k967 2444 2 k968 2445 2 k969 2447 2
k970 2449 2 k971 2451 2 k972 2453 2 k973 2455 2 k974 2458 2
k975 2501 2 k976 2504 2 k977 2507 2 k978 2510 2 k979 2512 2
k980 2514 2 k981 2528 3 k965 2529 3 k966 2530 3 k967 2332 3
k968 2334 3 k969 2336 3 k970 2338 3 k971 2340 3 k972 2343 3
k973 2346 3 k974 2349 3 k975 2352 3 k976 2354 3 k977 2355 3
k978 2356 3 k979 2356 3 k980 2557 3 k981 2636 4 k981 2516 2
k982 2517 2 k983 2518 2 k984 2519 2 k985 2520 2 k986 2522 2
k987 2524 2 k988 2526 2 k989 2528 2 k990 2530 2 k991 2533 2
k992 2536 2 k993 2539 2 k994 2542 2 k995 2545 2 k996 2550 2
k997 2551 2 k998 2551 2 k999 2552 2 k1000 2558 3 k982 2559 3
k983 2600 3 k984 2602 3 k985 2604 3 k986 2606 3 k987 2608 3
k988 2610 3 k989 2613 3 k990 2616 3 k991 2619 3 k992 2622 3
k993 2625 3 k994 2639 3 k995 2640 3 k996 2641 3 k997 2642 3
k998 2643 3 k999 2634 3 k1000

BIODATA PENULIS



Penulis dengan sapaan Fachry dan nama lengkap Mohammad Fian Fachry Alfahmi ini bertempat tinggal di Kabupaten Situbondo. Anak pertama dari tiga bersaudara ini lahir pada tanggal 12 Mei 1996. Selama menjalani masa perkuliahan di Departemen Matematika ITS, penulis aktif sebagai anggota UKM Technopreneurship Development Center (TDC ITS) pada tahun pertama. Pada tahun kedua, penulis melanjutkan kiprahnya di UKM TDC ITS sebagai asisten manajer Entrepreneurship Development. Di tahun ketiganya, penulis sempat “vakum” untuk berkuliah di lingkungan kampus ITS untuk mengikuti program *Study Exchange* selama dua semester (sepuluh bulan) di Catalunya, Spanyol. Kampus tempat penulis mengikuti program *study exchange* adalah *Universitat Rovira I Virgili* dan mengambil jurusan *Finance and Accounting*. Pada tahun keempat, penulis kembali melanjutkan perkuliahan di ITS untuk fokus mengejar ketertinggalan yang disebabkan oleh *study exchange* tersebut.

Tugas Akhir ini tak lepas dari kritik dan saran. Jika ada yang ingin didiskusikan ataupun ditanyakan, jangan ragu untuk menghubungi penulis via email : fian.fachry.fifa@gmail.com . Terimakasih dan semoga Tugas Akhir ini bermanfaat.