

SKRIPSI

**STUDI DAN IMPLEMENTASI POLA DESAIN MAPREDUCE
II**



Irvan Wijaya Ludianto

NPM: 2015730008

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2019**

UNDERGRADUATE THESIS

**STUDY AND IMPLEMENTATION MAPREDUCE DESIGN
PATTERN II**



Irvan Wijaya Ludianto

NPM: 2015730008

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2019**

LEMBAR PENGESAHAN

STUDI DAN IMPLEMENTASI POLA DESAIN MAPREDUCE II

Irvan Wijaya Ludianto

NPM: 2015730008

Bandung, 24 Juli 2019

Menyetujui,

Pembimbing

Dr. Veronica Sri Moertini

Ketua Tim Penguji

Anggota Tim Penguji

Raymond Chandra Putra, S.T., M.T.

Dr.rer.nat. Cecilia Esti Nugraheni

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

STUDI DAN IMPLEMENTASI POLA DESAIN MAPREDUCE II

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 24 Juli 2019

Meterai Rp. 6000

Irvan Wijaya Ludianto
NPM: 2015730008

ABSTRAK

Istilah *Big Data* sudah bukanlah lagi istilah yang asing. Istilah ini muncul karena jumlah data yang terus bertambah dan semakin besar setiap tahunnya. Data ini tentu saja perlu diolah untuk menghasilkan suatu informasi dan menarik suatu kesimpulan, namun hal ini sulit dilakukan dan membutuhkan waktu yang lama mengingat ukuran data yang begitu besar. Oleh karena itu sebuah *framework* bernama *Hadoop* diciptakan agar dapat mengolah *Big Data* secara terdistribusi dengan menggunakan model pemrograman *MapReduce*.

MapReduce terus digunakan dan menjadi semakin populer hingga akhirnya terlihatlah banyak permasalahan umum yang membutuhkan solusi yang serupa dalam mengolah data dengan *MapReduce*. Masalah-masalah umum tersebut dapat diselesaikan dengan menggunakan *design pattern* sehingga lebih mudah untuk dipecahkan dan digunakan kembali. Jenis-jenis *design pattern* antara lain adalah *summarization*, *filtering*, *data organization*, *join*, *meta*, dan *input and output*. Tentu saja pola desain tersebut harus dimodifikasi, dipelajari, dan diuji performanya sebelum digunakan.

Penelitian ini memuat 3 macam pola desain yang dikostumisasi sehingga dapat digunakan untuk beberapa kasus nyata tertentu yang menggunakan data berbentuk *JSON*. Ketiga pola desain yang dibuat tersebut adalah *Filter Pattern*, *Join Pattern*, dan *Input and Output Pattern*. Tentu ketiga pola desain ini memiliki kebutuhan dan fungsi yang berbeda-beda. Seluruh perangkat lunak demo untuk pola-pola desain tersebut telah berhasil disusun dan dapat digunakan sesuai dengan kasus-kasus nyata tertentu dimana data yang digunakan memiliki format *JSON*. Seluruh pola desain yang dibuat kemudian diuji dan dilihat waktu eksekusinya dengan menggunakan beberapa data dengan ukuran yang berbeda.

Hasil pengujian membuktikan bahwa performa yang dimiliki oleh setiap pola desain berbeda-beda tergantung dari komponen yang digunakan dan implementasi dari pola-pola desain tersebut. Selain itu performa itu juga dipengaruhi oleh berbagai faktor lainnya seperti ukuran dari data yang diproses, jumlah *Block* data, performa dari komputer yang melakukan proses sendiri, dan hal-hal lainnya.

Kata-kata kunci: *Big Data*, *Hadoop*, *MapReduce*, Pola Desain, Pola *Filter*, Pola *Join*, Pola *Input and Output*

ABSTRACT

The term of Big Data is no longer a foreign term. This term arises because the amount of data is keep increasing every year. The increasing data of course needs to be processed to produce an information and draw a conclusion, but this is hard to do and requires a long processing time considering the size of the data is so large. Therefore a framework called Hadoop was created so that it can process Big Data in a distributed manner using the MapReduce programming model.

MapReduce continues to be used and is becoming increasingly popular until finally there are many common problems that require similar solutions in processing data with MapReduce. These general problems can be solved using a design pattern so that it is easier to solve and reuse. The types of pattern design include summarization, filtering, data organization, join, meta, and input and output. Of course these design patterns must be modified, studied, and tested for performance before use.

This study contains 3 kinds of customized design patterns so that it can be used for certain specific cases that use JSON formatted data. The three design patterns created are Filters Pattern , Join Patterns, and Input and Output Patterns. Of course these three design patterns have different needs and functions. All demo software for these design patterns have been successfully build and can be used in accordance with certain real cases where the input data used has the JSON format. All design patterns created are then tested and seen when they are executed by using several data of different sizes.

The test results prove that the performance of each design pattern varies depending on the components used and the implementation of the design patterns. In addition, performance is also influenced by various other factors such as the size of the data being processed, the number of blocks of data, the performance of the computer that performs the process itself, and several other things.

Keywords: *Big Data, Hadoop, MapReduce, Design Pattern, Filter Pattern, Join Pattern, Input and Output Pattern*

*Skripsi ini dipersembahkan kepada Ayah, Ibu, saudara-saudara,
teman-teman, dan seluruh orang yang membutuhkan hasil
penelitian dari penelitian ini.*

KATA PENGANTAR

Puji dan syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya sehingga penulis bisa terus diberikan kekuatan sehingga dapat menyelesaikan skripsi ini sesuai dengan waktu yang diharapkan. Penulis menyadari bahwa di dalam skripsi ini masih terdapat banyak kekurangan yang disebabkan oleh keterbatasan dari segi kemampuan dan pengetahuan yang dimiliki oleh penulis. Oleh karena itu penulis dengan senang hati untuk menerima saran dan kritik yang membangun. Pada kesempatan kali ini, penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Orang tua penulis yaitu Lo Tjauw Fong dan Tan Kim Hwei yang selalu memberikan dukungan sehingga penulis dapat terus memiliki kekuatan untuk terus mengerjakan dalam keadaan sesulit apapun.
2. Ibu Veronica Sri Moertini sebagai dosen pembimbing yang selalu memberikan arahan yang baik berupa saran yang kritis sehingga skripsi ini dapat diselesaikan dengan hasil yang baik.
3. Bapak Raymond Chandra Putra dan Ibu Cecilia Esti Nugraheni sebagai dosen penguji yang telah memberikan masukan yang kritis sehingga hasil dari skripsi ini semakin layak dan baik.
4. Para admin lab yang bekerja dengan sangat baik dan giat sehingga setiap kali penulis mengalami masalah komputer saat melakukan pengujian, masalah tersebut dapat diselesaikan dengan cepat.
5. Anggota Founder Nibble Softworks yang dapat mengerti keadaan penulis dan memberikan waktu bagi penulis untuk menyusun skripsi sehingga skripsi dapat diselesaikan tepat pada waktunya.
6. Teman-teman terdekat penulis yang diantaranya adalah anggota *party DOTA* (Matthew, Edrick, Cornel, Himawan, dll), anggota *B&B* (Bernard, Raymond, Nico, dll), dan anggota grup *Terong* (Melinda dan Joshua) yang saling memberikan dukungan dan saling tolong menolong dalam berbagai hal.
7. Seluruh rekan-rekan lainnya yang berada satu jurusan maupun diluar jurusan yang tidak dapat disebutkan satu persatu.

Akhirnya penulis berharap semoga amal baik dari semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini mendapatkan balasan pahala dari Tuhan Yang Maha Esa. Semoga apa yang telah ditulis dalam skripsi ini dapat bermanfaat bagi penulis dan seluruh pihak yang membutuhkan hasil penelitian ini.

Bandung, Juli 2019

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xxi
DAFTAR TABEL	xxiii
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	2
1.6 Sistematika Pembahasan	3
2 LANDASAN TEORI	5
2.1 Hadoop [1]	5
2.1.1 Hadoop Distributed File System	6
2.1.2 MapReduce	7
2.1.3 Yet Another Resource Negotiator	10
2.2 Pola Desain MapReduce [2]	12
2.2.1 Pola Desain Filtering	12
2.2.2 Pola Desain Join	16
2.2.3 Pola Desain Input dan Output	21
2.3 Redis	24
3 STUDI EKSPLORASI HADOOP	25
3.1 Studi Eksplorasi Data Input	25
3.2 Studi Eksplorasi Word Count	26
3.3 Studi Eksplorasi MapReduce Design Pattern	29
3.3.1 Experimen Filtering Pattern	29
3.3.2 Experimen Bloom Filter Pattern	30
3.3.3 Experimen Top Ten Pattern	32
3.3.4 Experimen Distinct Pattern	34
3.3.5 Experimen Reduce Side Join Pattern	35
3.3.6 Experimen Replicated Join Pattern	38
3.3.7 Experimen Composite Join Pattern	40
3.3.8 Experimen Cartesian Product Pattern	41
3.3.9 Experimen Generating Data Pattern	45
3.3.10 Experimen External Source Output Pattern	47
3.3.11 Experimen External Source Input Pattern	49
3.3.12 Experimen Partition Pruning Output Pattern	51

3.3.13	Experimen Partitioning Input Pattern	53
4	ANALISIS DAN PERANCANGAN	57
4.1	Analisis Kebutuhan	57
4.2	Analisis Perancangan Perangkat Lunak Pola Desain MapReduce	57
4.2.1	Analisis Perancangan Penghubung GUI, PHP, dan Jar	57
4.2.2	Analisis Perancangan Pengiriman Data String Ke Mapper dan Reducer	58
4.2.3	Analisis Perancangan Pengiriman File Ke Mapper dan Reducer	58
4.2.4	Analisis Input Output dan Generik Filter Pattern	59
4.2.5	Analisis Input Output dan Generik Bloom Filter Pattern	59
4.2.6	Analisis Input Output dan Generik Simple Random Sampling Pattern	60
4.2.7	Analisis Input Output dan Generik Distinct Pattern	60
4.2.8	Analisis Input Output dan Generik TopN Pattern	60
4.2.9	Analisis Input Output dan Generik Reduce Side Join Pattern	61
4.2.10	Analisis Input Output dan Generik Replicated Join Pattern	61
4.2.11	Analisis Input Output dan Generik Composite Join Pattern	61
4.2.12	Analisis Input Output dan Generik Cartesian Product Pattern	62
4.2.13	Analisis Input Output dan Generik Generating Pattern	63
4.2.14	Analisis Input Output dan Generik External Input Pattern	63
4.2.15	Analisis Input Output dan Generik External Output Pattern	63
4.2.16	Analisis Input Output dan Generik Partitioning Input Pattern	64
4.2.17	Analisis Input Output dan Generik Partitioning Output Pattern	64
4.3	Perancangan Kelas Program Pola Desain MapReduce	64
4.3.1	Perancangan Kelas dan Method Filter	66
4.3.2	Perancangan Kelas dan Method Bloom Filter	67
4.3.3	Perancangan Kelas dan Method Simple Random Sampling	69
4.3.4	Perancangan Kelas dan Method Distinct	70
4.3.5	Perancangan Kelas dan Method Top N	71
4.3.6	Perancangan Kelas dan Method Reduce Side Join	73
4.3.7	Perancangan Kelas dan Method Replicated Join	75
4.3.8	Perancangan Kelas dan Method Composite Join	77
4.3.9	Perancangan Kelas dan Method Cartesian Product	79
4.3.10	Perancangan Kelas dan Method Generating	82
4.3.11	Perancangan Kelas dan Method External Input	84
4.3.12	Perancangan Kelas dan Method External Output	87
4.3.13	Perancangan Kelas dan Method Partitioning Input	89
4.3.14	Perancangan Kelas dan Method Partitioning Output	92
4.4	Perancangan Antarmuka	94
4.5	Format File Input	98
5	IMPLEMENTASI DAN PENGUJIAN	105
5.1	Deskripsi Perangkat Lunak dan Keras yang Digunakan	105
5.2	Implementasi Perangkat Lunak Design Pattern MapReduce dan Antarmuka	106
5.3	Pengujian Kebenaran Perangkat Lunak	117
5.4	Pengujian Performa Setiap Design Pattern	119
5.4.1	Pengujian Filter Pattern	119
5.4.2	Pengujian Bloom Filter Pattern	120
5.4.3	Pengujian Simple Random Sampling Pattern	120
5.4.4	Pengujian Distinct Pattern	121
5.4.5	Pengujian TopN Pattern	121
5.4.6	Pengujian Reduce Side Join Pattern	122
5.4.7	Pengujian Replicated Join Pattern	122

5.4.8	Pengujian Composite Join Pattern	123
5.4.9	Pengujian Cartesian Product Pattern	124
5.4.10	Pengujian Generating Pattern	124
5.4.11	Pengujian External Input Pattern	125
5.4.12	Pengujian External Output Pattern	125
5.4.13	Pengujian Partitioning Input Pattern	125
5.4.14	Pengujian Partitioning Output Pattern	125
5.5	Kesimpulan Hasil Eksperimen	126
6	KESIMPULAN DAN SARAN	127
6.1	Kesimpulan	127
6.2	Saran	127
	DAFTAR REFERENSI	129
A	KODE PROGRAM DESIGN PATTERN MAPREDUCE	131
A.1	<i>Package Filtering</i>	131
A.1.1	Kelas FilterPatternDriver.java	131
A.1.2	Kelas BloomFilterTrainer.java	132
A.1.3	Kelas BloomFilterDriver.java	134
A.1.4	Kelas SimpleRandomSamplingDriver.java	135
A.1.5	Kelas DistinctPatternDriver.java	136
A.1.6	Kelas TopNPatternDriver.java	137
A.2	<i>Package Join</i>	140
A.2.1	Kelas ReduceSideJoinPatternDriver.java	140
A.2.2	Kelas ReplicatedJoinPatternDriver.java	143
A.2.3	Kelas CompositeFormatter.java	144
A.2.4	Kelas CompositeJoinPatternDriver.java	145
A.2.5	Kelas CartesianProduct.java	146
A.3	<i>Package InputAndOutput</i>	149
A.3.1	Kelas RandomDataGeneratorPatternDriver.java	150
A.3.2	Kelas InputPatternDriver.java	153
A.3.3	Kelas OutputPatternDriver.java	155
A.3.4	Kelas PartitionPruningInputPatternDriver.java	158
A.3.5	Kelas PartitionPruningOutputDriver.java	161
B	KODE PROGRAM GRAPHICAL USER INTERFACE	165
B.1	<i>Package controllers</i>	165
B.1.1	Kelas Config_Controller	165
B.1.2	Kelas Home_Controller	166
B.2	<i>Package controllers/filter</i>	166
B.2.1	Kelas Filter_Controller	166
B.2.2	Kelas Bloom_Controller	167
B.2.3	Kelas SRS_Controller	167
B.2.4	Kelas Distinct_Controller	168
B.2.5	Kelas Top_N_Controller	168
B.3	<i>Package controllers/join</i>	169
B.3.1	Kelas Reduce_Side_Controller	169
B.3.2	Kelas Replicated_Controller	169
B.3.3	Kelas Composite_Controller	170
B.3.4	Kelas Cartesian_Product_Controller	171
B.4	<i>Package controllers/inputoutput</i>	171

B.4.1	Kelas Generator_Controller	171
B.4.2	Kelas External_Input_Controller	172
B.4.3	Kelas External_Output_Controller	173
B.4.4	Kelas Partition_Input_Controller	173
B.4.5	Kelas Partition_Output_Controller	174
B.5	<i>Package views</i>	175
B.5.1	View config_view	175
B.5.2	View home_view	177
B.6	<i>Package views/filter</i>	177
B.6.1	View filter_view	178
B.6.2	View bloom_trainer_view	179
B.6.3	View bloom_filter_view	180
B.6.4	View srs_view	181
B.6.5	View distinct_view	182
B.6.6	View top_n_view	183
B.7	<i>Package views/join</i>	185
B.7.1	View reduce_side_view	185
B.7.2	View replicated_view	186
B.7.3	View composite_formatter_view	188
B.7.4	View composite_view	189
B.7.5	View cartesian_product_view	190
B.8	<i>Package views/inputoutput</i>	191
B.8.1	View generator_formatter_view	192
B.8.2	View generator_view	193
B.8.3	View external_input_view	195
B.8.4	View external_output_view	196
B.8.5	View partition_input_formatter_view	197
B.8.6	View partition_input_view	198
B.8.7	View partition_output_formatter_view	199
B.8.8	View partition_output_view	201

DAFTAR GAMBAR

2.1	Arsitektur <i>Hadoop Distributed File System</i> .	7
2.2	<i>MapReduce Word Count</i> .	8
2.3	<i>Mapreduce data flow</i> .	9
2.4	Struktur YARN.	11
2.5	Struktur <i>Filtering</i> .	12
2.6	Struktur <i>Bloom Filter</i> .	14
2.7	Struktur <i>Top Ten</i> .	16
2.8	Struktur <i>Reduce Side Join</i> .	17
2.9	Struktur <i>Replicated Join Pattern</i> .	18
2.10	<i>Foreign key data sets</i> dibagi-bagi sesuai dengan pasangannya dan isinya telah terurut.	19
2.11	Struktur <i>Composite Join Pattern</i> .	20
2.12	Struktur <i>Cartesian Product Pattern</i> .	21
2.13	Struktur <i>Generating Data Pattern</i> .	22
2.14	Struktur <i>External Source Output Pattern</i> .	22
2.15	Struktur <i>External Source Input Pattern</i> .	23
2.16	Struktur <i>Partition Pruning Pattern</i> .	24
4.1	<i>Package</i> perangkat lunak.	65
4.2	Diagram kelas <i>Filter Pattern</i> .	66
4.3	Diagram kelas <i>Bloom Filter Pattern</i> .	67
4.4	Diagram kelas <i>Simple Random Sampling Pattern</i> .	69
4.5	Diagram kelas <i>Distinct Pattern</i> .	70
4.6	Diagram kelas <i>TopN Pattern</i> .	71
4.7	Diagram kelas <i>Reduce Side Join Pattern</i> .	73
4.8	Diagram kelas <i>Replicated Join Pattern</i> .	75
4.9	Diagram kelas <i>Composite Join Pattern</i> .	77
4.10	Diagram kelas <i>Cartesian Product Pattern</i> .	79
4.11	Diagram kelas <i>Generating Pattern</i> .	82
4.12	Diagram kelas <i>External Input Pattern</i> .	84
4.13	Diagram kelas <i>External Output Pattern</i> .	87
4.14	Diagram kelas <i>Partitioning Input Pattern</i> .	89
4.15	Diagram kelas <i>Partitioning Output Pattern</i> .	92
4.16	Halaman <i>Home</i> .	95
4.17	Halaman konfigurasi.	95
4.18	Halaman-halaman pola desain MapReduce.	96
4.19	Tampilan menu samping ditekan.	96
4.20	Halaman pola desain yang perlu data format.	97
4.21	Halaman pembuat data format.	97
4.22	Tampilan saat sedang memproses data.	98
5.1	Tampilan <i>Home</i> .	106
5.2	Tampilan <i>Config</i> .	106
5.3	Tampilan <i>Web HDFS</i> .	107

5.4	Tampilan <i>Web Yarn Resource Manager</i> .	107
5.5	Tampilan <i>Filter Design Pattern</i> .	108
5.6	Tampilan <i>Bloom Filter Design Pattern</i> .	108
5.7	Tampilan <i>Bloom Filter Trainer</i> .	109
5.8	Tampilan <i>Simple Random Sampling Design Pattern</i> .	109
5.9	Tampilan <i>Distinct Design Pattern</i> .	110
5.10	Tampilan <i>TopN Design Pattern</i> .	110
5.11	Tampilan <i>Reduce Side Join Design Pattern</i> .	111
5.12	Tampilan <i>Replicated Join Design Pattern</i> .	111
5.13	Tampilan <i>Composite Join Design Pattern</i> .	112
5.14	Tampilan <i>Composite Formatter</i> .	112
5.15	Tampilan <i>Cartesian Product Design Pattern</i> .	113
5.16	Tampilan <i>Generating Design Pattern</i> .	113
5.17	Tampilan <i>Generating Formatter</i> .	114
5.18	Tampilan <i>External Input Design Patter</i> .	114
5.19	Tampilan <i>External Output Design Patter</i> .	115
5.20	Tampilan <i>Partitioning Input Design Patter</i> .	115
5.21	Tampilan <i>Partitioning Input Formatter</i> .	116
5.22	Tampilan <i>Partitioning Output Design Patter</i> .	116
5.23	Tampilan <i>Partitioning Output Formatter</i> .	117

DAFTAR TABEL

5.1	Tabel hasil pengujian kebenaran dari perangkat lunak pola desain <i>MapReduce</i> . . .	117
5.2	Hasil Eksperimen <i>Filter Pattern</i>	119
5.3	Hasil Eksperimen <i>Bloom Filter Pattern</i>	120
5.4	Hasil Eksperimen <i>Simple Random Sampling Pattern</i>	120
5.5	Hasil Eksperimen <i>Distinct Pattern</i>	121
5.6	Hasil Eksperimen <i>TopN Pattern</i>	121
5.7	Hasil Eksperimen <i>Reduce Side Join Pattern</i>	122
5.8	Hasil Eksperimen <i>Replicated Join Pattern</i>	123
5.9	Hasil Eksperimen <i>Composite Join Pattern</i>	123
5.10	Hasil Eksperimen <i>Cartesian Product Pattern</i>	124
5.11	Hasil Eksperimen <i>Generating Pattern</i>	124
5.12	Hasil Eksperimen <i>External Output Pattern</i>	125
5.13	Hasil Eksperimen <i>Partitioning Output Pattern</i>	125

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Data adalah catatan atas fakta yang telah terjadi. Dengan adanya kumpulan data kita dapat menyimpulkan sesuatu bila kumpulan data tersebut memiliki suatu hubungan satu sama lain serta memiliki ukuran yang cukup besar. Hal tersebut membuat orang-orang berlomba-lomba untuk mendapatkan data agar mereka dapat melihat suatu informasi yang berharga. Ukuran data terus bertambah setiap tahunnya dan pada akhirnya berukuran sangat besar yang sering kita dengar dengan sebutan *Big Data*. Ukuran dari *Big Data* cukup relatif, ada orang yang mengatakan data berukuran ratusan *Gigabyte* adalah *Big Data* namun ada juga orang yang mengatakan bahwa data berukuran *Terabyte* baru termasuk *Big Data*. Oleh karena itu secara umum *Big Data* adalah kumpulan data yang berukuran sangat besar sehingga perlu waktu yang lama untuk diproses lebih lanjut sehingga dapat menjadi informasi yang berharga.

Untuk menganalisa data dengan ukuran yang sangat besar tidaklah mudah serta memakan banyak waktu dan biaya. Bila waktu yang dibutuhkan untuk mengubah suatu data menjadi informasi terlalu lama maka informasi yang didapatkan terkadang sudah tidak lagi berharga. Hal tersebut menghasilkan kebutuhan komputasi yang cepat. Komputasi paralel adalah sebuah solusi dimana proses dapat dilakukan secara bersamaan oleh banyak sistem sehingga pemrosesan *Big Data* dapat dilakukan dengan lebih cepat. Kecepatan adalah salah satu keuntungan komputasi paralel namun berbagai masalah yang harus ditangani oleh komputasi paralel juga harus diperhitungkan.

MapReduce adalah salah satu teknik yang pertama kali diperkenalkan oleh *Google* [1] untuk melakukan komputasi pada data yang berskala sangat besar secara terdistribusi dan paralel. Teknik ini secara umum akan memecah suatu masukan (*input*) kedalam sebuah daftar (*map*) yang berisikan pasangan kunci dan nilai. Daftar tersebut yang kemudian akan diproses kembali (*reduce*) untuk menghasilkan keluaran (*output*) yang diinginkan. *MapReduce* menjadi suatu solusi yang sangat baik untuk melakukan pemrosesan *Big Data* sehingga pada akhirnya muncullah sebuah *framework* yang bernama *Hadoop*. *Hadoop* adalah *framework* atau *platform open source* berbasis Java yang menggunakan teknologi *Google MapReduce* serta *Google File System (GFS)* sebagai fondasinya. Pada skripsi ini *framework Hadoop* inilah yang akan digunakan.

Setelah *MapReduce* menjadi populer dan sering digunakan akhirnya orang-orang tersadar bahwa banyak permasalahan umum yang selalu diselesaikan dengan solusi serupa yang dibuat berkali-kali. Masalah tersebut memacu munculnya berbagai pola desain. Ada 6 pola desain pada buku "Design Pattern Mapreduce" [2] yang diantaranya adalah *summarization*, *filtering*, *data organization*, *join*, *meta*, dan *input and output*. Pola-pola desain ini memiliki banyak keuntungan namun tentu saja suatu permasalahan tidak dapat diselesaikan hanya dengan menjiplak suatu pola secara mentah-mentah dan perlu modifikasi sesuai kebutuhan [2]. Hal tersebut memacu kebutuhan untuk mempelajari, mengkostumisasi, serta mengukur performa dari setiap pola desain. Skripsi ini meneliti tentang 3 tipe pola desain, yaitu *Filtering Pattern*, *Join Pattern*, dan *Input and Output Pattern* dari segi konsep yang digunakan serta kinerja yang dihasilkan dari ketiga pola tersebut. Untuk dapat meneliti hal-hal tersebut maka perlu dibuat perangkat lunak yang generik untuk setiap pola sehingga dapat diuji untuk berbagai kasus-kasus nyata tertentu.

1.2 Rumusan Masalah

Berdasarkan apa yang sudah dijelaskan pada latar belakang subbab 1.1 maka berikut adalah rumusan masalah yang akan dibahas pada skripsi ini:

1. Bagaimana menerapkan pola desain *Filtering*, *Join*, dan *Input and Output* pada kasus-kasus nyata tertentu?
2. Bagaimana performa dari pola desain *Filtering*, *Join*, dan *Input and Output*?
3. Bagaimana membangun perangkat lunak demo untuk pola desain *Filtering*, *Join*, dan *Input and Output*?

1.3 Tujuan

Berdasarkan rumusan masalah yang telah disusun pada subbab 1.2 maka tujuan dari penelitian ini adalah:

1. Membangun modul-modul program untuk pola desain *Filtering*, *Join*, dan *Input and Output*.
2. Melakukan eksperimen untuk mengukur kinerja dari pola desain *Filtering*, *Join*, dan *Input and Output*.
3. Membangun perangkat lunak demo yang dapat menjalankan pola desain *Filtering*, *Join*, dan *Input and Output* untuk kasus-kasus tertentu.

1.4 Batasan Masalah

Batasan masalah pada skripsi ini antara lain adalah:

1. Skripsi ini hanya akan memuat 3 jenis pola desain *MapReduce* yang ada, yaitu pola desain *Filtering*, *Join*, dan *Input and Output*.
2. Tidak menangani keamanan atau validasi pengguna pada perangkat lunak demo.

1.5 Metodologi

Metodologi yang digunakan dalam pembuatan skripsi ini adalah:

1. Melakukan studi literatur tentang *Hadoop* secara keseluruhan yang meliputi *library Hadoop* sendiri, *HDFS* (Hadoop Distributed File System), dan *MapReduce*.
2. Melakukan studi literatur tentang pola desain *Filtering*, *Join*, dan *Input and Output* pada *MapReduce*.
3. Mengimplementasikan program *word count* sebagai eksperimen agar lebih mengerti dan terbiasa dengan konsep *MapReduce* dan *Hadoop*.
4. Mengimplementasikan 12 kode program *design pattern MapReduce* yang ada pada buku "Design Pattern Mapreduce" [2] sebagai eksperimen agar lebih mengerti tentang konsep *design pattern MapReduce*.
5. Mengkonfigurasi dan mencoba menjalankan program *MapReduce* pada mode *Pseudo-distributed* dan mode *Fully-distributed*.

6. Mencari *data set* yang dapat digunakan untuk pola desain *Filtering*, *Join*, dan *Input and Output*.
7. Merancang modul-modul program untuk pola desain *Filtering*, *Join*, dan *Input and Output*.
8. Mengimplementasikan modul-modul program yang sudah dirancang.
9. Menguji tingkat kebenaran dan kinerja dari masing-masing pola desain.
10. Membuat dokumen skripsi.

1.6 Sistematika Pembahasan

1. Bab 1 Pendahuluan

Bab ini membahas mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian, serta sistematika pembahasan yang digunakan dalam penelitian serta penyusunan dokumen skripsi ini.

2. Bab 2 Landasan Teori

Bab ini membahas mengenai teori-teori yang digunakan dalam penyusunan skripsi ini. Teori-teori tersebut antara lain adalah teori mengenai *Big Data*, sistem terdistribusi *Hadoop*, *HDFS* (*Hadoop Distributed File System*), *MapReduce*, pola desain pada umumnya, pola desain *Filtering*, pola desain *Join*, dan pola desain *Input and Output*.

3. Bab 3 Studi Eksplorasi Hadoop

Bab ini membahas mengenai studi eksplorasi menggunakan *Hadoop* berupa eksperimen kecil yang dilakukan. Eksperimen tersebut memuat program *Word Count* serta 12 *design pattern* yang berasal dari buku "Design Pattern Mapreduce" [2].

4. Bab 4 Analisis dan Perancangan

Bab ini membahas mengenai analisis serta perancangan perangkat lunak yang dibangun pada skripsi ini. Analisis dan perancangan yang dimuat adalah kebutuhan dari perangkat lunak yang ingin disusun, analisis masukan dan keluaran yang diharapkan, perancangan *GUI* (*Graphical User Interface*), perancangan kelas pada perangkat lunak.

5. Bab 5 Implementasi dan Pengujian

Bab ini berisi mengenai spesifikasi perangkat lunak yang digunakan selama pengujian, hasil implementasi perangkat lunak, hasil implementasi antarmuka, pengujian kebenaran dari modul-modul program hasil implementasi, eksperimen pengukuran performa yang dilakukan untuk mengukur kinerja dari tiap pola desain yang telah diimplementasikan, dan kesimpulan hasil eksperimen.

6. Bab 6 Kesimpulan dan Saran

Bab ini membahas mengenai kesimpulan-kesimpulan dan saran-saran dari penulis yang berhasil didapatkan dari penelitian ini.