

**PREDIKSI JUMLAH PENDUDUK DI SUMATRA SELATAN
MENGUNAKAN ALGORITMA *FUZZY TIME SERIES* DAN
*PARTICLE SWARM OPTIMIZATION***

Diajukan Sebagai Syarat Untuk Menyelesaikan
Pendidikan Program Strata-1 Pada
Jurusan Teknik Informatika

Oleh :

Virgiana Febrianti
09021381722085

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER UNIVERSITAS SRIWIJAYA
2022**

LEMBAR PENGESAHAN SKRIPSI

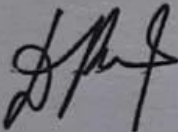
**PREDIKSI JUMLAH PENDUDUK DI SUMATRA
SELATAN MENGGUNAKAN ALGORITMA FUZZY
TIME SERIES DAN PARTICLE SWARM OPTIMIZATION**

Oleh:

Virgiana Febrianti
09021381722085

Palembang, Juli 2022

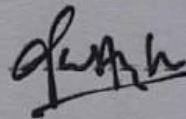
Pembimbing I



Dian Palupi Rini, S.Si., M.Kom., Ph.D.

NIP. 197802232006042002

Pembimbing II

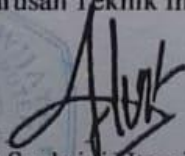


Desty Rodiah, M.T.

NIP. 198912212020122011

Mengetahui,

Ketua Jurusan Teknik Informatika



Alvi Syahrini Utami,

M.Kom. NIP.

197812222006042003

HALAMAN PERNYATAAN BEBAS PLAGIAT

Yang bertanda tangan di bawah ini :

Nama : Virgiana Febrianti

NIM : 09021381722085

Program Studi : Teknik Informatika Bilingual

Judul Skripsi : Prediksi jumlah penduduk di Sumatra Selatan menggunakan algoritma *fuzzy time series* dan *Particle swarm Optimization*

Hasil Pengecekan Software iThenticate/Turnitin : 17 %

Menyatakan bahwa Laporan Projek saya merupakan hasil karya sendiri dan bukan hasil penjiplakan/plagiat. Apabila ditemukan unsur penjiplakan/plagiat dalam laporan projek ini, maka saya bersedia menerima sanksi akademik dari Universitas Sriwijaya sesuai dengan ketentuan yang berlaku.

Demikian, pernyataan ini saya buat dengan sebenarnya dan tidak ada paksaan oleh siapapun.

Palembang, 03 Agustus 2022



Virgiana febrianti
NIM. 09021381722085

**Prediction Of Population In South Sumatra Using *Fuzzy Time Series*
Algorithm And *Particle Swarm Optimization***

**by :
Virgiana Febrianti
09021381722085**

ABSTRACT

Population growth occurs due to the increase in the number of births. The impact of population growth affects human welfare, both in the economic, health, social, political, and cultural fields. Therefore, it is necessary to conduct population forecasting so that it can control population growth and population welfare on a regular basis, based on the results of the forecasting carried out. One of the forecasting methods used is *Fuzzy Time Series (FTS)*. However, this method still has drawbacks, namely in determining the value of the interval that is not precise. So an optimization algorithm is needed to find the optimal value of the interval. This study aims to forecast the population in the city of Palembang using the *FTS* method which is optimized using the *Particle Swarm Optimization (PSO)*. The amount of data used is 60 data for 5 years and the historical population for the period 2010-2020, the method used to calculate error obtained using RMSE and AFER so that the average accuracy result is obtained. From the overall test of $RMSE = 22182,58487$ and $AFER = 9.8175400\%$.

Keywords: *Fuzzy Time Series*, population, *Particle Swarm Optimization*, Facing, South Sumatra

**Prediksi Jumlah Penduduk Di Sumatra Selatan Menggunakan Algoritma
*FuzzyTime Series Dan Particle Swarm Optimization***

oleh :
Virgiana Febrianti
09021381722085

ABSTRAK

Pertumbuhan penduduk terjadi dikarenakan meningkatnya jumlah kelahiran. Dampak pertumbuhan penduduk ini berpengaruh terhadap kesejahteraan manusia, baik dalam bidang ekonomi, kesehatan, sosial, politik dan budaya. Oleh karena itu perlunya untuk melakukan, peramalan jumlah penduduk sehingga dapat mengontrol pertumbuhan penduduk dan kesejahteraan penduduk secara berkala, berdasarkan hasil peramalan yang dilakukan. Salah satu metode peramalan yang digunakan yaitu *Fuzzy Time Series (FTS)*. Namun, metode ini masih memiliki kekurangan yaitu pada penentuan nilai interval yang kurang tepat. Maka diperlukan algoritma optimasi untuk mencari nilai yang optimal dari interval tersebut. Penelitian ini bertujuan untuk melakukan peramalan jumlah penduduk pada kota Palembang dengan menggunakan metode *FTS* yang di optimasi menggunakan algoritma *Particle Swarm Optimization (PSO)*. Optimasi yang dilakukan untuk mendapatkan nilai interval optimal pada *FTS* dan nilai parameter optimal pada *PSO*. Jumlah data yang digunakan sebanyak 60 data selama 5 tahun data histori jumlah penduduk periode tahun 2016-2020. Metode yang digunakan untuk menghitung error yang didapatkan menggunakan RMSE dan AFER sehingga didapatkan hasil rata-rata akurasi dari keseluruhan pengujian sebesar $RMSE = 22182,58487$ dan $A FER = 0,8175440\%$.

Kata Kunci : *Fuzzy Time Series* , *jumlah penduduk*, *Particle Swarm Optimization*,
Peramalan,
Sumatra Selatan

DAFTAR ISI

	Halaman
LEMBAR PENGESAHAN LAPORAN SKRIPSI.....	ii
ABSTRAK	iii
DAFTAR ISI	v
DAFTAR GAMBAR	ix
DAFTAR TABEL.....	x
BAB I	I-1
PENDAHULUAN	I-1
1.1 Pendahuluan	I-1
1.2 Latar Belakang.....	I-1
1.3 Rumusan Masalah.....	I-5
1.4 Tujuan Penelitian	I-5
1.5 Manfaat Penelitian	I-6
1.6 Batasan Masalah	I-6
1.7 Sistematika Penulisan	I-7
1.8 Kesimpulan.....	I-7
BAB II	II-1
KAJIAN TEORITIS.....	II-1
2.1 Pendahuluan	II-1
2.2 Peramalan.....	II-1
2.3 Data Runtun Waktu (<i>Time Series</i>).....	II-2
2.4 <i>Fuzzy Time Series</i>	II-2
2.5 Penentuan Interval Berbasis Rata-rata pada <i>Fuzzy Time Series</i>	II-5
2.6 <i>Particle Swarm Optimization</i>	II-6
2.7 Pengukuran Peramalan.....	II-12
2.8 Rational Unified Process (RUP)	II-13
2.9 Penelitian Lain Yang Relevan	II-16
2.10 Kesimpulan.....	II-19
BAB III.....	III-1
METODOLOGI PENELITIAN.....	III-1
3.1 Pendahuluan	III-1
3.2 Unit Penelitian	III-1

3.3	Data.....	I-1
3.3.1	Jenis Sumber Data.....	III-1
3.3.2	Metode Pengumpulan Data.....	III-2
3.4	Tahapan Penelitian.....	III-2
3.4.1	Menetapkan Kerangka Kerja/ <i>Framework</i>	III-2
1	Menganalisis metode <i>Fuzzy Time Series</i> dan algoritma <i>PSO</i>	III-3
2	Mengumpulkan data	III-4
2.1	Data yang digunakan.....	III-4
2.2	Melakukan proses penguji	III-4
2.3	melakukan analisis terhadap hasil pengujian	III-4
2.4	membuat kesimpulan	III-5
3.4.2	Menetapkan Kriteria Pengujian	III-5
3.4.3	Menetapkan Format Data Pengujian	III-5
3.4.4	Menetapkan Pengujian Penelitian	III-6
3.4.5	Melakukan Analisis Hasil Pengujian Dan Membuat Kesimpulan	III-8
3.5	Metode Pengembangan Perangkat Lunak	III-9
3.6	Manajemen Proyek Penelitian	III-13
3.7	Kesimpulan.....	III-17
BAB IV		IV-1
PENGEMBANGAN PERANGKAT LUNAK		IV -1
4.1	Pendahuluan	IV -1
4.2	Fase Insepsi	IV -1
4.2.1	Pemodelan Bisnis	IV-1
4.2.2	Kebutuhan Sistem	IV-2
4.2.3	Analisis dan Desain.....	IV-3
4.2.3.1	Analisisa Kebutuhan Prangkat Lunak.....	IV-4
4.2.3.2	Analisis Data.....	IV-4
4.2.3.3	Analisis <i>Fuzzy Time Series</i> dan <i>PSO</i>	IV-5
4.2.3.4	Desain Perangkat Lunak	IV-7
1	<i>use case</i>	IV-7
a)	diagram <i>use case</i>	IV-7
b)	Tabel Definisi Aktor	IV-8

	c) Tabel Definisi <i>use case</i>	V-9
	d) Sekenario <i>use case</i>	IV-9
	2 Kelas Analisis.....	IV-12
	a) Kelas Analisis Memuat Data	IV-12
	b) Kelas Analisis meramalkan jumlah penduduk dengan FTS dan algoritma PSO.....	12
	c) Kelas Analisis Akurasi Peramalan	IV-13
4.3	Fase Elaborasi.....	IV -14
4.3.1	Pemodelan Bisnis	IV-15
4.3.1.1	Perancangan Data	IV-15
4.3.1.2	Perancangan Antar Muka.....	IV-16
4.3.2	Kebutuhan Sistem	IV-16
4.3.3	Diagram <i>Sequence</i>	IV-17
4.3.3.1	<i>Sequence Diagram</i> Memuat Data	IV-17
4.3.3.2	<i>Sequence diagram</i> peramalan jumlah penduduk dengan FTS dengan Algoritma PSO.....	IV-18
4.3.3.3	<i>Sequence Diagram</i> akurasi	IV-19
4.4	Fase Konstruksi	IV -19
4.4.1	Kebutuhan Sistem	IV-19
4.4.2	Diagram Kelas	IV-20
4.4.3	Implementasi.....	IV-21
4.4.3.1	Implementasi Kelas	IV-21
4.4.3.1	Implementasi Antarmuka.....	IV-22
4.5	Fase Transisi.....	IV- 22
4.5.1	Pemodelan Bisnis	IV-23
4.5.2	Rencana Pengujian	IV-23
4.5.2.1	Rencana Pengujian <i>Use Case</i> Memuat Data.....	IV-23
4.5.2.2	Rencana Pengujian <i>Use Case</i> meramalkan jumlah penduduk dengan FTS	IV-23
4.5.2.3	Rencana Pengujian <i>Use Case</i> Optimasi akurasi peramalan dengan PSO.....	IV-24
4.5.2.4	Rencana Pengujian <i>Use Case</i> akurasi peramalan.	IV-24
4.5.3	Implementasi.....	IV-25
4.5.3.1	Rencana Pengujian <i>Use Case</i> Memuat Data.....	IV-26

4.5.3.2 Pengujian <i>Use Case</i> Meramalkan jumlah penduduk dengan <i>Fuzzy Time Series</i> dan <i>Particle Swarm Optimize</i>	IV-27
4.5.3.1 Pengujian <i>Use Case</i> akurasi peramalan	V-28
4.6 Kesimpulan	IV-29
BAB V	V-1
ANALISIS PENELITIAN	V-1
5.1 Pendahuluan	V-1
5.2 Hasil Percobaan Penelitian	V-1
5.2.1 Data set Pengujian	V-2
5.2.2 Hasil Pengujian <i>Fuzzy Time Series</i> dengan <i>Particle Swarm Optimization</i>	V-7
5.2.3 Analisis Hasil Pengujian Perbandingan Akurasi Peramalan	V-
95.3 Kesimpulan	V-10
BAB VI	VI-1
KESIMPULAN DAN SARAN	VI-1
6.1 Pendahuluan	VI-1
6.2 Kesimpulan	VI-1
6.3 Saran	1-2
DAFTAR PUSTAKA	xii

DAFTAR GAMBAR

Gambar II-1 Arsitektur RUP (Sumber: Kruchten,2004).....	I-14
Gambar III-1 diagram kerangka kerja.....	III-3
Gambar III-2 Tahapan Pengujian Penelitian	III-6
Gambar IV-1 Diagram Alur FTS-PSO	IV-7
Gambar IV-2 Diagram <i>use case</i>	IV-8
Gambar IV-3 Kelas Analisis Memuat Data	IV-12
Gambar IV-4 Kelas Analisis Optimasi dengan <i>Fuzzy time series</i> dengan algoritma <i>particle swarm optimize</i>	IV-13
Gambar IV-5 Kelas peramalan akurasi.....	IV-14
Gambar IV-6 Rancangan antarmuka perangkat lunak.....	IV-16
Gambar IV-7 <i>Sequence diagram</i> Memuat Data	IV-18
Gambar IV-8 <i>sequence diagram</i> peramalan jumlah penduduk dengan <i>FTS</i> dengan Algoritma <i>PSO</i>	IV-18
Gambar IV-9 <i>sequence diagram</i> Akurasi	IV-19
Gambar IV-10 Diagram Kelas Perangkat Lunak	IV-20
Gambar IV-11 Antarmuka Halaman Utama Perangkat Lunak	IV-22
Gambar V-1 Grafik Hasil Pengujian Jumlah Particle.....	V-8

DAFTAR TABEL

Tabel II- 1 Tabel basis interval	I-6
Tabel III- 1 Data Masukkan	III-5
Tabel III- 2 Rancangan Hasil Pengujian	III-6
Tabel III- 3 Rancangan perbandingan hasil akurasi peramalan	III-9
Tabel III- 4 jadwal penelitian	III-14
Tabel IV-1 Tabel Kebutuhan Fungsional.....	IV-3
Tabel IV-2 Tabel kebutuhan non fungsional.....	IV-3
Tabel IV-3 Definisi Aktor <i>Use case</i>	IV-8
Tabel IV-4 Definisi <i>Use case</i>	IV-9
Tabel IV-5 Sekenarion Memuat Data.....	IV-10
Tabel IV-6 Sekenario Meramalkan jumlah penduduk dengan <i>Fuzzy Time Series</i> dengan <i>Particle swarm optimization</i>	IV-10
Tabel IV-7 Sekenarion Akurasi Peramalan.....	IV-11
Tabel IV-8 Implementasi Kelas.....	IV-21
Tabel IV-9 rencana pengujian <i>use case</i> Memuat Data	IV-23
Tabel IV-10 rencana pengujian <i>use case</i> meramalkan jumlah penduduk menggunakan dengan <i>FTS</i> dan <i>PSO</i>	IV-24
Tabel IV-11 Rencana Pengujian <i>Use Case</i> Akurasi Peramalan.....	IV-24
Tabel IV-12 Pengujian <i>Use Case</i> Memuat Data	IV-26

Tabel IV-13 Pengujian <i>Use Case</i> Meramalkan Jumlah Penduduk Menggunakan Dengan <i>FTS</i> dan <i>PSO</i>	V-27
Tabel IV-14 Pengujian <i>Use Case</i> Akurasi Peramalan	IV-28

BAB I

PENDAHULUAN

1.1 Pendahuluan

Pada bab ini membahas latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, dan batasan masalah. Bab ini akan memberikan penjelasan umum mengenai keseluruhan penelitian.

Pendahuluan dimulai dengan penjelasan mengenai latar belakang penelitian peramalan dimana metode yang digunakan yaitu *Fuzzy Time Series* dan *algoritma Particle Swarm Optimization* untuk menghasilkan peramalan jumlah penduduk di Sumatra Selatan dengan tepat serta penjelasan penelitian yang berkaitan dengan kombinasi teknik optimasi ke dalam proses peramalan.

1.2 Latar Belakang

Jumlah penduduk kota Palembang setiap tahunnya mengalami peningkatan yang cukup signifikan, berdasarkan sensus penduduk 2000 oleh Badan Pusat Statistik berjumlah 6.899.675 jiwa. Jumlah itu meningkat sekitar 546.726 orang dalam waktu 10 tahun terakhir, yakni tahun 2010 sebanyak 7.446.401 jiwa atau dengan pertumbuhan sebesar 9,7 persen¹. Karena adanya peningkatan jumlah penduduk di kota Palembang maka muncul permasalahan yang terjadi diantaranya

¹ Website” <https://palembangkota.bps.go.id.”>, 05 Februari 2022

pengangguran, tindakan kriminal, dan bangunan sebagai tempat tinggal. Dari peristiwa tersebut, maka pemerintah kota Palembang perlu untuk melakukan peramalan jumlah penduduk, agar pemerintah dapat menanggulangi permasalahan jumlah penduduk yang terjadi di kota Palembang.

Sistem Peramalan adalah metode untuk memperkirakan suatu nilai dimasa depan dengan menggunakan data masa lalu (Wardah,2016). Dalam permalan terdapat beberapa metode seperti *metode naif*, *metode moving Average*, *metode kausal*, *metode regresi*, *metode korelasi* dan *Fuzzy time series*. Metode *Fuzzy Time Series* dapat menangkap pola dari data masa lalu untuk memproyeksikan data yang akan datang dengan berdasarkan konsep *fuzzy*. Kelebihan logika *Fuzzy* yang mendorong perkembangan mesin masa kini adalah penggunaan *soft computing* untuk meniru kemampuan otak manusia yang secara efektif melakukan pendekatan daripada pertimbangan kepastian perhitungan matematika, dan memberikan toleransi ketidaktepatan, ketidakpastian, dan kebenaran parsial (robandi, 2016)

Fuzzy Time series memiliki kesulitan dalam menemukan jumlah kelas dan panjang interval yang efektif. Perhitungan prediksi dengan menggunakan *fuzzy time series* standar, panjang interval telah ditentukan secara manual di awal proses perhitungan. Sedangkan penentuan panjang interval sangat berpengaruh dalam pembentukan *fuzzy relationship* yang tentunya akan memberikan dampak perbedaan

hasil perhitungan peramalan. Dalam penelitian ini akan menggunakan algoritma *Particle Swarm Optimization* untuk mengoptimasi pada interval nilai linguistik. Dengan melakukan optimasi, perhitungan peramalan dapat menghasilkan perhitungan peramalan yang lebih baik dan dapat membantu meningkatkan akurasi peramalan. *PSO* memiliki beberapa kesamaan dengan teknik komputasi evolusioner seperti *algoritma genetika*. Dibandingkan dengan algoritma genetika, *PSO* memiliki beberapa kelebihan, antara lain mudah di implementasikan dan memiliki lebih sedikit fungsi operasi dan parameter yang harus ditentukan. (Haupt, 2004).

Penelitian tentang peramalan penduduk sebelumnya sudah dilakukan dengan *Optimasi Fuzzy Time Series Menggunakan Algoritma Particle Swarm Optimization Untuk Peramalan Jumlah Penduduk Di Kabupaten Probolinggo* penelitian ini dilakukan oleh (Marji 2018). Hasil penelitian menunjukkan metode *PSO* berhasil di implementasikan pada backpropagation untuk mengoptimalkan bobot. *PSO Metode Fuzzy Time Series (FTS)* dapat digunakan untuk melakukan peramalan. peramalan yang dilakukan menggunakan metode *FTS* memeberikan nilai RMSE, yaitu: 52,05198333. Metode *Particle Swarm Optimization (PSO)* dapat digunakan untuk melakukan optimasi metode *FTS*. Hasil dari pengujian yang dilakukan dengan menggunakan data jumlah penduduk Kabupaten Probolinggo pada setiap kecamatan mulai tahun 2013 sampai 2016, didapatkan nilai paramater terbaik yaitu: $w = 0,6$,

$c_1 = 1,8$ dan $c_2 = 2,4$. Sehingga diperoleh nilai fitness terbaik dari peramalan tersebut, yaitu: 0,445334.

Dalam pengembangan sistem peramalan jumlah penduduk maupun lainnya biasanya juga dilakukan dengan memproses informasi secara statistik dan matematis dengan *knowledge base* yang disesuaikan. Penelitian sebelumnya mengenai peramalan penjualan penelitian ini dilakukan oleh (Diponegoro, U. 2013) dengan menerapkan peramalan jumlah penjualan dengan metode *Fuzzy Time Series*. Untuk dapat memperkirakan jumlah suatu penduduk, salah satunya adalah dengan menggunakan metode *Fuzzy Time Series*.

Penelitian terkait dengan *Fuzzy Time Series* dan *Particle Swarm Optimization* penelitian dilakukan oleh (Qiu et al., 2015) yang melakukan penelitian pada data shanghai stock exchange composite index dan data pendaftaran University of Alabama. Penelitian ini menerapkan metode *Generalized Fuzzy Time Series Forecasting Model Enhanced* dengan *Particle Swarm Optimization*. Dalam percobaan menunjukkan bahwa hasil *Root Mean Squared Error (RMSE)* sebesar 2.59, *Mean Absolute Error (MAE)* sebesar 0,64 serta *Mean Absolute Percentage Error (MAPE)* sebesar 0.0004.

Selanjutnya penelitian pada metode *Fuzzy Time Series* dan *Particle Swarm Optimization* penelitian yang dilakukan oleh (Setiawan, 2018) dengan penelitiannya Peramalan Permintaan Darah. Berdasarkan hasil dari serangkaian pengujian didapatkan solusi optimum bernilai

cost (MSE) sebesar 60435.685 dengan jumlah partikel sebanyak 40, jumlah dimensi sebanyak 30, kombinasi nilai c_1 dan c_2 masing-masing 1.5 dan 1.5, bobot inersia sebesar 0.3, dan jumlah iterasi maksimum sebesar 950. Tingkat kesalahan dari sistem ini (*MAPE*) sebesar 7.50330% dari 12 data uji yang digunakan.

Berdasarkan permasalahan dan penjelasan yang telah diuraikan, penulis mengusulkan penelitian yang berjudul “Peramalan jumlah penduduk di kota Palembang” dengan Kombinasi Metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization* “. Dengan menggabungkan karakteristik yang dimiliki Metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization* diharapkan dapat peramalan jumlah penduduk dengan optimal.

1.3 Rumusan Masalah

Untuk menjawab rumusan masalah tersebut, diuraikan beberapa pertanyaan penelitian sebagai berikut:

1. Bagaimana cara mengembangkan perangkat lunak untuk meramalkan jumlah penduduk dengan menggunakan *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization* ?
2. Bagaimana tingkat kesalahan dalam meramalkan jumlah penduduk dengan menggunakan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization* ?

1.4 Tujuan Penelitian

Tujuan dilakukannya penelitian adalah sebagai berikut:

1. Mengembangkan perangkat lunak untuk meramalkan jumlah penduduk di Sumatra Selatan dengan menggabungkan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*.
2. Mengetahui tingkat kesalahan dalam meramalkan jumlah penduduk dengan menggunakan metode. *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*.

1.5 Manfaat Penelitian

Manfaat yang dapat diperoleh dari penelitian ini adalah sebagai berikut :

1. Menghasilkan perangkat lunak yang mampu membantu meramalkan jumlah penduduk di Sumatra Selatan.
2. Membantu pemerintah dalam memperoleh informasi atas peramalan jumlah penduduk untuk menentukan atau merencanakan pembangunan di kemudian hari.
3. Menjadi masukan dalam memprediksi jumlah penduduk secara akurat untuk menentukan perencanaan dan anggaran daerah ke depan.
4. Dapat digunakan sebagai pembanding dengan metode lain pada studi kasus peramalan jumlah penduduk.

1.6 Batasan Masalah

Batasan masalah pada penelitian ini adalah sebagai berikut :

1. Penelitian yang dilakukan berdasarkan data riil atau data aktual pada tahun 2016 – 2020 dengan tipe data per bulan yang didapatkan dari Dinas Kependudukan dan Catatan Sipil Kota Palembang.
2. Data yang digunakan terdiri dari 1 jenis data yaitu data seluruh penduduk di kota Palembang berdasarkan 60 bulan (2016 – 2020)

1.7 Sistematika Penulisan

Penyusunan skripsi ini disusun dengan sistematika penulisan sebagaiberikut :

BAB I. PENDAHULUAN

Pada bab ini diuraikan mengenai latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan.

BAB II. KAJIAN LITERATUR

Pada bab ini membahas dasar-dasar teori yang digunakan dalam penelitian serta beberapa kajian literatur mengenai penelitian lain yang relevan pada penelitian ini.

BAB III. METODOLOGI PENELITIAN

Pada bab ini membahas mengenai tahapan yang akan dilaksanakan pada penelitian ini. Masing-masing rencana tahapan penelitian dideskripsikan dengan rinci mengacu pada suatu kerangka kerja. Di akhir bab, berisi perancangan manajemen proyek pelaksanaan

penelitian.

BAB IV. PENGEMBANGAN PERANGKAT LUNAK

Pada bab ini membahas mengenai analisis dan perancangan perangkat lunak yang akan digunakan sebagai alat penelitian. Dimulai dari pengumpulan dan analisa kebutuhan, rancangan dan konstruksi perangkat lunak serta pengujian untuk memastikan semua kebutuhan pengembangan perangkat lunak sesuai dengan kebutuhan. Penyusunan pada bab ini memiliki kerangka penulisan dengan fase-fase dan elemen-elemen pengembangan perangkat lunak bersifat berorientasi objek.

BAB V. HASIL DAN ANALISIS PENELITIAN

Pada bab ini diuraikan hasil pengujian berdasarkan langkah-langkah yang telah direncanakan. Analisa hasil pengujian disajikan sebagai basis dari kesimpulan yang akan diambil dalam penelitian ini.

BAB VI. KESIMPULAN DAN SARAN

Pada bab ini berisi kesimpulan dari semua uraian-uraian pada bab-bab sebelumnya dan juga saran-saran yang diharapkan berguna dalam penerapan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization* untuk peramalan jumlah penduduk

I.8 Kesimpulan

Bab ini telah mengenai penelitian yang akan dilakukan yaitu peramalan jumlah penduduk di Sumatra Selatan dengan menggunakan Kombinasi metode *Fuzzy Time Series* dan algoritma *Particle Swarm*

Optimization. Selanjutnya teori- teori yang berkaitan dengan penelitian akan dibahas pada bab II.

II BAB II

KAJIAN TEORITIS

2.1 Pendahuluan

Pada bab sebelumnya dijelaskan rumusan masalah pada penelitian ini adalah Bagaimana pengaruh algoritma *Particle Swarm Optimization* dalam meningkatkan akurasi metode *Fuzzy Time Series* untuk melakukan peramalan jumlah penduduk. Untuk memahami fundamental objek penelitian, maka dilakukan *literature review* terhadap jurnal, buku, dan artikel yang terkait dengan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*.

2.2 Peramalan

Peramalan merupakan suatu proses pendugaan terhadap kejadian yang akan terjadi pada masa depan. Peramalan juga dapat diartikan sebagai proses yang dilakukan ketika ada kesenjangan waktu (lag) dari data aktual pada waktu tertentu dengan data yang ingin diketahui pada waktu yang akan datang. Peramalan diperlukan untuk mengetahui kapan atau bagaimana suatu peristiwa akan terjadi sehingga tindakan yang tepat dapat dilakukan. Peramalan identik dengan analisis data time series (deret waktu). Data *time series* merupakan serangkaian data yang berupa nilai pengamatan yang diukur selama kurun waktu tertentu berdasarkan interval waktu yang tetap (Susetyoko.R, 2016). Metode peramalan data

yang menggunakan prinsip-prinsip *fuzzy* sebagai dasarnya peramalan dengan menggunakan *fuzzy time series* menangkap pola dari data yang telah lalu kemudian digunakan untuk memproyeksikan data yang akan datang. Oleh karena itu perlunya untuk melakukan, peramalan jumlah penduduk sehingga dapat mengontrol pertumbuhan penduduk dan kesejahteraan penduduk secara berkala, berdasarkan hasil peramalan yang dilakukan.

Peramalan sering dilakukan untuk memprediksi dan membuat perencanaan dengan menggunakan data masa lalu dan data masa sekarang, agar dapat membuat prediksi di masa yang akan datang. Berdasarkan jangka waktunya, peramalan dapat dibagi menjadi tiga yaitu:

- (1) Peramalan Jangka Panjang (*Long-Term Forecasting*), yang memprediksikan keadaan dalam jangka waktu beberapa tahun ke depan (tahunan).
- (2) Peramalan Jangka Menengah (*Mid-Term Forecasting*), yang memprediksikan keadaan dalam jangka waktu bulanan atau mingguan.
- (3) Peramalan Jangka Pendek (*Short-Term Forecasting*), yang memprediksikan keadaan dalam jangka waktu harian hingga tiap jam.

2.1 Data Runtun Waktu (*Time Series*)

Data time series adalah data dari waktu ke waktu yang memiliki nilai tertentu untuk setiap waktunya. Data time series berisi data tentang objek

tertentu (Rizky P, 2018). Dapat dilihat dari contoh data *time series* pada data harga saham, data ekspor, data nilai tukar (*kurs*), data produksi, dan lain-lain sebagainya. Jika diamati masing-masing data tersebut terkait dengan waktu (*time*) dan terjadi berurutan. Misalnya data produksi minyak sawit dari tahun 2000 hingga 2009, data kurs Rupiah terhadap dollar Amerika Serikat dari tahun 2000 - 2006, dan lain-lain. Dengan demikian maka akan sangat mudah untuk mengenali jenis data ini. Data *time series* juga sangat berguna bagi pengambil keputusan untuk memperkirakan kejadian di masa yang akan datang. Karena diyakini pola perubahan data *time series* beberapa periode masa lampau akan kembali terulang pada masa kini.

2.3 *Fuzzy Time Series*

Fuzzy Time Series adalah metode prediksi data yang menggunakan prinsip-prinsip *fuzzy* sebagai dasarnya. Sistem prediksi dengan *FTS* menangkap pola dari data yang telah lalu kemudian digunakan untuk memproyeksikan data yang akan datang (Anwary, 2011). Pertama kali dikembangkan oleh Song and Chissom (1993) kemudian disempurnakan oleh Chen dan memiliki keuntungan mengurangi waktu perhitungan dan menyederhanakan proses perhitungan (Chen, 2004). *Fuzzy Time Series* adalah sebuah konsep baru yang dapat digunakan untuk menangani masalah prediksi di mana data historis adalah nilai-nilai linguistik (Chen, 1996). Metode ini sering digunakan oleh para peneliti untuk

menyelesaikan masalah prediksi. Dalam FTS, himpunan semesta didefinisikan sesuai Persamaan 1 (Chenn, 1996)

$$U=(Dmin-D_1Dmax+D_2) \quad (II-1)$$

Keterangan:

$Dmin$: data historis minimum

$Dmax$: data historis maksimum

$D1$ dan $D2$: bilangan positif sembarang yang ditentukan oleh peneliti untuk menentukan himpunan semesta dari himpunan data historis

Adapun metode *Fuzzy Time Series* dalam penyelesaian masalah prediksi adalah sebagai berikut (Chen, 2004):

1. Mendefinisikan himpunan *fuzzy* pada himpunan semesta. Tahap ini mengubah himpunan semesta yang telah terbagi dan masih berupa himpunan bilangan crisp menjadi himpunan *fuzzy* berdasarkan interval.

$$A_1 = a_{11}/ U_1 + a_{12}/ U_2 + \dots + + a_{1n}/ U_n$$

$$A_2 = a_{21}/ U_1 + a_{22}/ U_2 + \dots + + a_{2n}/ \quad (II-2)$$

$$UA_m = a_{m1}/ U + a_{m2}/ U_2 + \dots + + a_{mn}/ U_n$$

Melakukan fuzzifikasi pada data historis. Tahap ini menentukan nilai keanggotaan pada masing-masing himpunan *fuzzy* dari data historis, dengan nilai keanggotaan 0 sampai 1. Nilai keanggotaan ini diperoleh dari fungsi keanggotaan yg telah dibuat sebelumnya.

2. Membentuk *fuzzy relationship*, $A_i \rightarrow A_j$ berdasarkan nilai A_i yang

telah ditentukan pada langkah sebelumnya, dimana A_i adalah hari ke n dan A_j adalah hari ke $n + 1$. Kemudian semua *fuzzy* relasinya dikelompokkan. Contohnya sebagai berikut :

A_i mempunyai relasi relasi, yaitu $A_i \rightarrow A_j$; $A_i \rightarrow A_{j1}$; $A_i \rightarrow A_j$.

Dari 3 *fuzzy*

relationship dapat dikelompokkan menjadi $A_i \rightarrow A_j, A_{j1}$.

3. Melakukan defuzzifikasi output yang diramalkan. Tahap ini menentukan nilai hasil prediksi yang berupa nilai crisp, dengan aturan sebagai berikut:

Misalkan (t) adalah data yang akan diramalkan dimana $F(t - 1) =$

A_i , maka:

- a. Jika hanya terdapat satu *fuzzy relationship group* dari A_i yaitu $A_i \rightarrow A_s$, maka $(t) = A_s$ dimana defuzzifikasinya adalah nilai tengah dari interval dimana memiliki nilai keanggotaan maksimum pada A_s .
- b. Jika A_i tidak memiliki relasi maka defuzzifikasi $F(t)$ diperoleh dari nilai tengah interval yang memiliki nilai keanggotaan maksimum pada A_i
- c. jika hanya terdapat lebih dari satu *fuzzy relationship group* dari A_i yaitu $A_i \rightarrow A_j, A_{j1}, A_{j2}, \dots, A_{jn}$, maka $F(t)$ diperoleh dari rata rata nilai tengah dari masing masing interval yang memiliki nilai keanggotaan maksimum pada masing masing $A_j, A_{j1}, A_{j2}, \dots, A_{jn}$.

2.2 Penentuan Interval Berbasis Rata-rata pada *Fuzzy Time Series*

Dalam perhitungan peramalan dengan menggunakan *fuzzy time series* standar, panjang interval telah ditentukan di awal proses perhitungan. Sedangkan penentuan panjang interval sangat berpengaruh dalam pembentukan *fuzzy relationship* yang tentunya akan memberikan dampak perbedaan hasil perhitungan peramalan. Oleh karena itu, pembentukan *fuzzy relationship* haruslah tepat dan hal ini mengharuskan penentuan panjang interval yang sesuai. Kunci utama dalam penentuan panjang interval adalah tidak boleh terlalu besar dan tidak boleh terlalu kecil, karena jika interval itu terlalu besar maka tidak akan terjadi fluktuasi dalam proses perhitungan *fuzzy time series*, demikian juga jika interval tersebut terlalu kecil maka makna dari *fuzzy time series* sendiri akan hilang (karena himpunan yang terbentuk cenderung ke himpunan tegas/*crisp*).

Salah satu metode untuk penentuan panjang interval yang efektif adalah dengan metode berbasis rata-rata (*average-based*), yang memiliki algoritma sebagaimana berikut :

1. Hitung semua nilai absolute selisih antara A_{i+1} dan A_i ($i=1, \dots, n-1$) sehingga diperoleh rata-rata nilai absolute selisih;
2. Tentukan setengah dari rata-rata yang diperoleh dari langkah pertama untuk kemudian dijadikan sebagai panjang interval.
3. Berdasarkan panjang interval yang diperoleh dari langkah kedua,

ditentukan basis dari panjang interval sesuai dengan tabulasi basis berikut.

4. Panjang interval kemudian dibulatkan sesuai dengan tabel basis interval.

2.3 Particle Swarm Optimization

Particle Swarm Optimization disingkat sebagai *PSO*, merupakan algoritma berbasis populasi yang mengeksploitasi individu dalam pencarian. Dalam *PSO* populasi disebut swarm dan individu disebut partikel. Tiap partikel berpindah dengan kecepatan yang diadaptasi dari daerah pencarian dan menyimpannya sebagai posisi terbaik yang pernah dicapai. *PSO* didasarkan pada perilaku sosial sekawanan burung atau sekumpulan ikan. Algoritma *PSO* meniru perilaku sosial organisme ini. Perilaku sosial terdiri dari tindakan individu dan pengaruh dari individu-individu lain dalam suatu kelompok. Kata partikel menunjukkan,

misalnya, seekor burung dalam kawanan burung. Setiap individu atau partikel berperilaku secara terdistribusi dengan cara menggunakan kecerdasannya (*intelligence*) sendiri dan juga dipengaruhi perilaku kelompok kolektifnya. Dengan demikian, jika satu partikel atau seekor burung menemukan jalan yang tepat atau pendek menuju ke sumber makanan, sisa kelompok yang lain juga akan dapat segera mengikuti jalan meskipun lokasi mereka jauh di kelompok tersebut.

Metode optimasi yang didasarkan pada *swarm intelligence* ini disebut algoritma *behaviorally inspired* sebagai alternatif dari algoritma genetika, yang sering disebut *evolution-based procedures*. Algoritma *PSO* ini awalnya diusulkan oleh J. Kennedy dan R. C. Eberhart (1995). Dalam konteks optimasi multivariabel, kawanan diasumsikan mempunyai ukuran tertentu atau tetap dengan setiap partikel posisi awalnya terletak di suatu lokasi yang acak dalam ruang multidimensi. Setiap partikel bergerak dalam ruang / *space* tertentu dan mengingat posisi terbaik yang pernah dilalui atau ditemukan terhadap sumber makanan atau nilai fungsi objektif. Setiap partikel menyampaikan informasi atau posisi bagusnya kepada partikel yang lain dan menyesuaikan posisi dan kecepatan masing-masing berdasarkan informasi yang diterima mengenai posisi yang bagus tersebut.

PSO memiliki banyak kesamaan dengan teknik-teknik evolutionary computation yang lain, seperti *Genetic Algorithms (GA)*, *Evolutionary Strategies (EA)*, *Evolutionary Programming*, dan

sebagainya. *PSO* maupun *GA* dimulaidengan suatu populasi yang terdiri dari sejumlah individu (yang menyatakan solusi) yang dibangkitkan secara acak dan selanjutnya melakukan pencarian solusioptimum melalui perbaikan individu untuk sejumlah generasi tertentu. Tetapi berbeda dengan *GA*, *PSO* tidak menggunakan operator-operator evolusi seperti rekombinasi (*cross over*) dan mutasi. *PSO* memiliki memori untuk menyimpan solusi terbaik, sedangkan *GA* tidak punya. Setiap partikel pada *PSO* tidak pernah mati, sedangkan individu pada *GA* bisa mati dan digantikan dengan individu baru. Pada *PSO*, posisi dan kecepatan terbang partikel di-update pada setiap iterasi sehingga partikel tersebut bisa menghasilkan solusi baru yang lebih baik menurut (Ifrans, 2013).

Pada algoritma *PSO* ini, pencarian solusi dilakukan oleh suatu populasi yang terdiri dari beberapa partikel. Populasi dibangkitkan secara random dengan batasan nilai terkecil dan terbesar. Setiap partikel merepresentasikan posisi atau solusi dari permasalahan yang dihadapi. Setiap partikel melakukan pencarian solusi yang optimal dengan melintasi ruang pencarian (*search space*). Hal ini dilakukan dengan cara setiap partikel melakukan penyesuaian terhadap posisi terbaik dari partikel tersebut (*local best*) dan penyesuaian terhadap posisi partikel terbaik dari seluruh kawanan (*global best*) selama melintasi ruang pencarian. Jadi, penyebaran pengalaman atau informasi terjadi di dalam partikel itu sendiri dan antara suatu partikel dengan partikel terbaik dari seluruh kawanan selama proses pencarian solusi. Setelah

itu, dilakukan proses pencarian untuk mencari posisi terbaik setiap partikel dalam sejumlah iterasi tertentu sampai didapatkan posisi yang relatif steady atau mencapai batas iterasi yang telah ditetapkan. Pada setiap iterasi, setiap solusi yang direpresentasikan oleh posisi partikel, dievaluasi performansinya dengan cara memasukkan solusi tersebut kedalam *fitness function*.

Setiap partikel diperlakukan seperti sebuah titik pada suatu dimensi ruang tertentu. Kemudian terdapat dua faktor yang memberikan karakter terhadap status partikel pada ruang pencarian yaitu posisi partikel dan kecepatan partikel (Kennedy and Eberhart, 1995).

Algoritma PSO terdiri dari tiga tahap, yaitu pembangkitan posisi serta kecepatan partikel, update velocity (*update* kecepatan), update position (*update* posisi). Pertama posisi x_k^i dan kecepatan v_k^i dari kumpulan partikel dibangkitkan secara random menggunakan batas atas (x_{max}) dan batas bawah (x_{min}) dari designvariable, seperti yang ditunjukkan pada persamaan (II-3) dan (II-4).

$$x^1 = x_{min} + r_1(x_{max} - x_{min}) \quad (\text{II-3})$$

$$v^1 = x_{min} + r_2(x_{max} - x_{min}) \quad (\text{II-4})$$

Dimana,

x^1 = posisi awal

v^1 = kecepatan awal

x_{min} = batas bawah

x_{max} = batas atas

r_1, r_2 = nilai random antara nilai 0 dan 1

Posisi dan kecepatan direpresentasikan dalam bentuk vektor di mana n dimensi vektor merepresentasikan jumlah dari desain variabel partikel, dengan *superscript* dan *subscript* menotasikan partikel ke i pada waktu ke k dengan proses inisialisasi ini maka kumpulan partikel dapat terdistribusi secara *random* pada *design space*. Vektor seperti ditunjukkan di bawah ini:

$$x^i = (x^{i1}, x^{i2}, \dots, x^{in})^T \quad (\text{II-5})$$

$$v^i = (v^{i1}, v^{i2}, \dots, v^{in})^T \quad (\text{II-6})$$

Langkah kedua adalah *update velocity* (kecepatan) untuk semua partikel pada waktu $k + 1$ menggunakan fungsi objektif atau nilai *fitness* posisi partikel saat ini pada *design space* saat waktu ke k . Dari nilai *fitness* dapat ditentukan partikel mana yang memiliki nilai *global* terbaik (*global best*) pada *swarm* saat ini (p^g), dan juga dapat ditentukan posisi terbaik dari tiap partikel pada semua waktu yang sekarang dan sebelumnya (p^i). Perumusan *update velocity* menggunakan dua informasi tersebut untuk semua partikel pada kumpulan dengan pengaruh perpindahan yang sekarang (v^i), untuk memberikan arah pencarian (v^i) untuk $kk + 1$ generasi selanjutnya.

Perumusan *update velocity* mencakup beberapa parameter *random*

($rand/r_1, r_2$), untuk mendapatkan cakupan yang baik pada *design space*, tiga parameter yang mempengaruhi arah pencarian, yaitu *self confidence* (c_1), *swarm confidence* (c_2) akan digabungkan dalam satu penyajian, seperti yang ditunjukkan persamaan berikut:

$$v_{k+1}^i = v^i + c_1 r_1 (p^i - x^i) + c_2 r_2 (p^g - x^i) \quad (\text{II-7})$$

Di mana,

v^i = kecepatan sekarang

x^i = posisi sekarang

c_1, c_2 = *self confidence*, *swarm confidence*, merupakan *learning rates* untuk kemampuan individu (*cognitive*) dan pengaruh sosial (*group*). Parameters c_1 dan c_2 menunjukkan bobot dari memori (*position*) sebuah partikel terhadap memori (*position*) dari kelompok (*swarm*). Nilai dari c_1 dan c_2 biasanya adalah 2 sehingga perkalian $c_1 r_1$ dan $c_2 r_2$ memastikan bahwa partikel-partikel akan mendekati target sekitar setengah selisihnya

r_1, r_2 = bilangan *random* yang memiliki interval 0 dan 1

p^i = *local best*, posisi terbaik dari tiap partikel pada semua waktu yang sekarang

k p^g = nilai *global* terbaik (*global best*) pada *swarm* saat ini

Langkah terakhir dari setiap iterasi adalah *update* posisi tiap partikel dengan vektor *velocity*, seperti yang ditunjukkan pada persamaan berikut :

$$x_{k+1}^i = x_k^i + v_{k+1}^i \quad (\text{II-8})$$

Dimana,

x_{k+1}^i = posisi pencarian

v_{k+1}^i = arah pencarian

x_k^i = posisi sekarang

Tiga tahapan di atas akan diulang sampai kriteria kekonvergenan terpenuhi, kriteria kekonvergenan sangat penting dalam menghindari penambahan fungsi evaluasi setelah solusi optimum didapatkan, namun kriteria kekonvergenan tidak selalu mutlak diperlukan, penetapan jumlah iterasi maksimal juga dapat digunakan sebagai *stopping condition* dari algoritma

Banyak cara untuk membangun kondisi berhenti, diantaranya adalah : iterasi dihentikan ketika PSO telah mencapai iterasi maksimum, atau PSO telah menemukan nilai optimum tertentu atau kesalahan minimum yang diinginkan.

2.4 Pengukuran Peramalan

Teknik peramalan tidak selamanya selalu tepat karena teknik peramalan yang digunakan belum tentu sesuai dengan sifat datanya atau disebabkan oleh kondisi di luar bisnis yang mengharuskan bisnis perlu menyesuaikan diri. Oleh karena itu, perlu diadakan pengawasan peramalan sehingga dapat diketahui sesuai atau tidaknya teknik peramalan yang digunakan. Sehingga dapat dipilih dan ditentukan teknik peramalan yang lebih sesuai dengan cara menentukan batas toleransi

peramalan atas penyimpangan yang terjadi (Jumiriani,2009).

Pada prinsipnya, pengawasan peramalan dilakukan dengan membandingkan hasil peramalan dengan kenyataan yang terjadi. Penggunaan teknik peramalan yang menghasilkan penyimpangan terkecil adalah teknik peramalan yang paling sesuai untuk digunakan (Jumiriani,2009).

Jilani, Burney, dan Ardil (2007) menggunakan metode AFER(*Average Forecasting Error Rate*) dan MSE (*Mean Square Error*) untuk mengetahui besarnya penyimpangan yang terjadi pada data hasil peramalan terhadap data riil.

a. RMSE (*Root Mean Square Error*)

Root Mean Squared Error (RMSE) adalah besarnya tingkat kesalahan hasil prediksi, dimana semakin kecil (mendekati 0) maka hasil RMSE semakin akurat. Adapun rumus perhitungan RMSE dapat dilihat pada II-9.

$$\text{RMSE} = \left(\frac{\sum(y_i - \hat{y})^2}{n} \right)^{1/2} \quad (\text{II-9})$$

Keterangan:

y = nilai hasil observasi

\hat{y} = nilai hasil prediksi

n = jumlah data

i = urutan data pada database

b. AFER (*Average Forecasting Error Rate*)

AFER (*Average Forecasting Error Rate*) adalah metode lain untuk mengevaluasi peramalan berdasarkan rata-rata kesalahan peramalan.

Adapun

perhitungan AFER dapat dilihat pada 2.10.

$$\text{AFER} = \frac{\sum | \text{actual}(t) - \text{forecast}(t) |}{n} \times 100\% \quad (\text{II-10})$$

Keterangan :

A_t = nilai hasil observasi/ data actual

F_t = nilai hasil peramalan data ke-i

n = banyaknya data

2.5 Rational Unified Process (RUP)

Pada penelitian ini menggunakan metode pengembangan perangkat lunak *Rational Unified Process* (RUP). RUP merupakan suatu pendekatan yang mengatur tugas-tugas serta tanggung jawab pada organisasi pengembang perangkat lunak (Krutchen, 2004). RUP menggunakan konsep berorientasi objek, dengan aktifitas yang berfokus pada pengembangan model menggunakan *Unified Model Language* (UML).

1. Struktur Proses RUP

Arsitektur RUP memiliki dua dimensi horizontal dan vertikal yang dapat dilihat pada Gambar Gambar II-7 dibawah ini.

Dilihat pada Gambar II-1 arsitektur RUP menunjukkan proses RUP yang memiliki dua dimensi tahapan, yaitu:

a. Dimensi Horizontal

Dimensi ini menggambarkan waktu dan menjelaskan aspek dinamis dari proses pengembangan perangkat lunak meliputi; siklus, fase, iterasi, dan *milestones* (batas pencapaian yang menandakan akhir dan awal dari fase selanjutnya). Dimensi horizontal ini terdiri atas fase *Inception*, *Elaboration*, *Construction*, dan *Transition*.

b. Dimensi Vertikal

Dimensi ini menjelaskan aspek-aspek statis dari proses pengembangan perangkat lunak (RUP) meliputi; aktivitas dan alur pekerjaan. Alur kerja pada dimensi vertikal ini terdiri atas; *Business Modeling*, *Requirement, Analysis and Design*, *Implementation*, *Test*, *Deployment*, *Configuration*, *Project Management*, dan *nvironment*.

2. Fase-Fase Pengembangan RUP
 - a. Fase Insepsi (*Inception*) merupakan tahapan pertama dari 4 tahapan RUP. Fase ini berfokus pada permodelan bisnis. Dalam tahap ini memahami tentang semua proyek dan ruang lingkup tujuan. Permodelan bisnis tersebut mencakup antara lain; mendefinisikan masalah, membatasi ruang lingkup proyek, dan *initial bussiness case*.
 - b. Fase Elaborasi (*Elaboration*) merupakan tahapan kedua RUP. Tujuannya adalah untuk menganalisis permasalahan, menentukan arsitektur dasar dari sistem untuk menyediakan dasar yang stabil sebagian besar dari desain dan pelaksanaan upaya dalam tahap konstruksi.
 - c. Fase Konstruksi (*Construction*), merupakan tahapan ketiga RUP. fase ketiga yang merupakan tahap untuk mengimplementasikan hasil desain ke *coding* dan melakukan pengujian hasil implementasi. Di fase ini semua komponen dan fitur aplikasi dibangun dan disatukan. Fase ini berupa sebuah proses *manufacturing* yang berfokus pada pengontrolan operasi untuk mengoptimalkan biaya, jadwal dan kualitas.
 - d. Fase Transisi (*Transition*), merupakan tahap keempat sekaligus tahap terakhir tahapan pelaksanaan lifecycle RUP yaitu membuat apa yang sudah dimodelkan sebelumnya menjadi suatu produk jadi. Fase ini terfokus untuk memastikan bahwa produk layak untuk pengguna

akhir untuk itu penulis menentukan *tools* pengujian yang diperlukan yaitu perangkat keras yang sama saat digunakan untuk pengembangan perangkat lunak.

2.6 Penelitian Lain Yang Relevan

1. Penelitian sebelumnya juga mengenai Tentang penentuan *base transceiver system* menggunakan *Partical Swarm Optimization (PSO)*, Salah satu jaringannya adalah jaringan 4GLTE. Jaringan 4G ini merupakan suatu sistem berbasis IP dengan kecepatan 100Mbps/detik dan 1Gb/detik. Teknologi pada jaringan 4G menggunakan menara untuk memproses transmisi yang disebut eNode B (Sri, 2014) Frekuensi jaringan 4G yang ada di kota Malang saat ini 2300 MHz dan jangkauannya masih kurang memenuhi kebutuhan masyarakat (Sri, 2014). Maka penentuan lokasi BTS sangatlah penting. Sehingga diperlukan suatu metode dalam penentuan lokasi BTS yang akan dipasang. Satu Base Transceiver System (BTS) akan melayani sebuah sel, dimana satu Base Transceiver System akan mengatur sebuah BCCH (Broadcast Control Channel) dengan jumlah kanal pembawa maksimum 8 kanal. Setiap transceiver mentransmisikan dengan daya yang sama. Fungsi internal dari Base Transceiver System (BTS) adalah sebagai protokol dari jalur sinyal radio, jalur sinyal informasi antara BSC dan MS, maupun protocol interface BSC (Ningsih dkk, 2015).

2. Penelitian tentang peramalan penduduk sebelumnya sudah dilakukan dengan *Optimasi Fuzzy Time Series Menggunakan Algoritma Particle Swarm Optimization Untuk Peramalan Jumlah Penduduk Di Kabupaten Probolinggo* oleh (Cahyo Adi Prasajo, Budi Darma Setiawan, Marji 2018). Hasil penelitian menunjukkan metode *PSO* berhasil di implementasikan pada backpropagation untuk mengoptimalkan bobot. *PSO Metode Fuzzy Time Series (FTS)* dapat digunakan untuk melakukan peramalan. peramalan yang dilakukan menggunakan metode *FTS* memberikan nilai MSE, yaitu: 52,05198333. Metode *Particle Swarm Optimization (PSO)* dapat digunakan untuk melakukan optimasi metode *FTS*. Langkah pertama dengan melakukan optimasi interval *FTS* dengan *PSO*. Interval pada *FTS* tersebut akan digunakan sebagai partikel pada *PSO* yang nantinya memiliki fitness yang tinggi sebagai penentu partikel yang optimal, dimana semakin tinggi fitness semakin baik nilai akurasi. Nilai interval dilakukan dengan mengambil nilainya secara acak dengan batasan tertentu. Hasil dari pengujian yang dilakukan dengan menggunakan data jumlah penduduk Kabupaten Probolinggo pada setiap kecamatan mulai tahun 2013 sampai 2016, didapatkan nilai parameter terbaik yaitu: $w = 0,6$, $c1 = 1,8$ dan $c2 = 2,4$. Sehingga diperoleh nilai fitness terbaik dari peramalan tersebut, yaitu: 0,445334. Pada penelitian jumlah penduduk dengan menggunakan metode *Fuzzy Time Series* yang

diptimasi dengan metode *Particle Swarm Optimization* masih terdapat banyak kekurangan. Kekurangan tersebut dapat dikembangkan dengan melakukan penelitian yang lebih baik lagi.

3. Penelitian terkait dengan *Fuzzy Time Series* dan *Particle Swarm Optimization* dilakukan oleh Qiu et al., (2015) yang melakukan penelitian pada data shanghai stock exchange composite index dan data pendaftaran University of Alabama. Penelitian ini menerapkan metode *Generalized Fuzzy Time Series Forecasting Model Enhanced* dengan *Particle Swarm Optimization*. Dalam percobaan menunjukkan bahwa hasil *Root Mean Squared Error (RMSE)* sebesar 2.59, *Mean Absolute Error (MAE)* sebesar 0,64 serta *Mean Absolute Percentage Error (MAPE)* sebesar 0.0004
4. Selanjutnya penelitian pada metode *Fuzzy Time Series* dan *Particle Swarm Optimization* yang dilakukan oleh Dwi, Rifandi, & Setiawan (2018) dengan penelitiannya Peramalan Permintaan Darah. Berdasarkan hasil dari serangkaian pengujian didapatkan solusi optimum bernilai *cost (MSE)* sebesar 60435.685 dengan jumlah partikel sebanyak 40, jumlah dimensi sebanyak 30, kombinasi nilai c_1 dan c_2 masing-masing 1.5 dan 1.5, bobot inersia sebesar 0.3, dan jumlah iterasi maksimum sebesar 950. Tingkat kesalahan dari sistem ini (*MAPE*) sebesar 7.50330% dari 12 data uji yang digunakan.

2.7 Kesimpulan

Pada bab ini telah dibahas dasar – dasar teori terkait yang dibutuhkan dalam melakukan penelitian ini. Selanjutnya pada bab III akan dibahas mengenai metodologi penelitian yang akan digunakan pada penelitian kali ini.

III. BAB III

METODOLOGI PENELITIAN

3.1 Pendahuluan

Bab ini akan menjelaskan unit dan tahapan penelitian yang diimplementasikan, metodologi penelitian serta manajemen proyek penelitian. Tahapan penelitian dijadikan sebagai acuan pada setiap fase pengembangan dan memberikan sebuah solusi untuk rumusan masalah dan mencapai tujuan penelitian.

3.2 Unit Penelitian

Dalam penelitian ini yang digunakan sebagai unit penelitian adalah Dinas Kependudukan dan Catatan Sipil kota Palembang Jl. Demang Lebar Daun No.4225, Lorok Pakjo, Kec. Ilir Bar. I, Kota Palembang, Sumatera Selatan 30151. Lokasi penelitian berada di Fakultas Ilmu Komputer, jurusan Teknik Informatika Bilingual, Universitas Sriwijaya Palembang.

3.3 Data

3.3.1 Jenis dan Sumber Data

Jenis data yang digunakan pada penelitian tugas akhir ini adalah data sekunder, yaitu berupa 60 data format.doc laporan jumlah keseluruhan

penduduk di kota Palembang per bulan dari tahun 2016-2020 yang didapat dari Dinas Kependudukan dan Catatan Sipil kota Palembang Jl. Demang Lebar Daun No.4225, Lorok Pakjo, Kec. Ilir Bar. I, Kota Palembang, Sumatera Selatan.

3.3.2 Metode Pengumpulan Data

Sumber data yang digunakan pada penelitian ini adalah berupa data laporan jumlah keseluruhan penduduk di kota Palembang dari tahun 2016-2020 yang didapat dari Dinas Kependudukan dan Catatan Sipil kota Palembang, dengan 60 data.

3.4 Tahapan Penelitian

Penelitian akan dilakukan dengan tahapan-tahapan yang akan dijelaskan pada subbab 3.4.1 sampai dengan 3.4.5.

3.4.1 Menetapkan Kerangka Kerja / *Framework*

Berikut ini adalah gambar kerangka kerja / framework

1. Menganalisis metode *Fuzzy Time Series* dan algoritma *PSO*

Tahapan pertama yang dilakukan pada penelitian ini adalah menganalisa metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*. Metode *Fuzzy Time Series* adalah metode yang dapat menangkap pola dari data masa lalu untuk memproyeksikan data yang akan datang. Sedangkan algoritma *Particle Swarm Optimization* adalah algoritma optimasi yang mudah diimplementasikan dan memiliki lebih sedikit fungsi operator dan parameter yang harus ditentukan. Dengan metode dan algoritma yang digunakan diharapkan mendapatkan hasil yang akurat.

2. Mengumpulkan data

2.1 Data yang digunakan

pada penelitian ini yaitu berupa data laporan jumlah keseluruhan penduduk di kota Palembang dari tahun 2016-2020 yang didapat dari Dinas Kependudukan dan Catatan Sipil kota Palembang. Melakukan Pengembangan perangkat lunak. Pengembangan perangkat lunak menggunakan *Rational Unified Process*. Alasan menggunakan RUP adalah untuk memastikan bahwa perangkat lunak yang dihasilkan memenuhi kebutuhan pengguna (*user requirements*) dan dapat diselesaikan tepat waktu.

2.2 Melakukan proses pengujian

Proses pengujian perangkat lunak yaitu *black-box testing* dan metode pengujian perangkat lunak. *Black-box testing* adalah pengujian yang didasarkan pada detail aplikasi seperti tampilan aplikasi, fungsi-fungsi yang ada pada aplikasi, dan kesesuaian alur fungsi kebutuhan perangkat lunak.

2.3 Melakukan analisis terhadap hasil pengujian

Setelah dilakukan pengujian perangkat lunak maka tahap selanjutnya yaitu melakukan analisis hasil. Apakah perangkat lunak sesuai dengan tujuan pembuatan dan apakah perangkat lunak yang dihasilkan berfungsi dengan baik dan juga apakah perangkat lunak memberikan hasil yang akurat.

2.4 Membuat kesimpulan

Tahap terakhir yaitu membuat kesimpulan terhadap perangkat lunak yang dibangun.

3.4.2 Menetapkan Kriteria Pengujian

Format data pengujian akan dijelaskan pada sub bab 3.4.3 sampai dengan 3.4.5.

3.4.3 Menetapkan Format Data Pengujian

Pada penelitian ini, format data masukan yang digunakan memilikiformat .xlsx.

3.4.4 Menetapkan Pengujian Penelitian

berikut ini gambar tahapan pengujian penelitian

1. Data Histori Penduduk

Input berupa data histori jumlah penduduk yang terdiri dari 1 jenis yaitu jumlah keseluruhan penduduk.

2. *Fuzzy Time Series* dengan *Particle Swarm Optimization*

Pada tahap ini, dilakukan perhitungan *Fuzzy Time Series* menggunakan optimasi *PSO* untuk mendapatkan hasil peramalan jumlah penduduk.

3. Perhitungan MSE dan AFER

Pada tahap ini, dilakukan perhitungan MSE dan AFER dengan menggunakan data aktual dan data hasil peramalan untuk

mendapatkan hasil akurasi peramalan/kesalahan penyimpangan peramalan (*error*).

4. Analisa Hasil dan membandingkan data aktual dengan data hasil peramalan serta akurasi peramalan.

Pada tahap ini dilakukan analisa dengan membandingkan hasil pendeteksian oleh sistem dengan data aktual yang didapatkan dari pakar untuk melihat seberapa akurat hasil pendeteksian yang dihasilkan oleh sistem. Semakin kecil kesalahan pendeteksian maka semakin baik hasil yang didapat.

5. Kesimpulan Analisa Hasil dan hasil akurasi peramalan antara metode *Fuzzy Time* dengan *Particle Swarm Optimization*.

Pada tahap ini dilakukan menarik kesimpulan dari Analisa hasil dan hasil akurasi peramalan dari rancangan perbandingan hasil pengujian akurasi peramalan penjumlahan penduduk dapat dilihat pada tabel III-3.

Setelah mendapatkan hasil analisa pengujian penelitian, maka langkah selanjutnya adalah membuat kesimpulan penelitian yang akan dijelaskan pada babV.

3.5 Metode Pengembangan Perangkat Lunak

Pengembangan perangkat lunak dalam penelitian ini didasarkan pada 4 fase RUP. Langkah-langkah yang akan dilakukan dalam penelitian ini akan dijelaskan pada patagraf berikut ini

1. Pemodelan bisnis dalam insepisi Fase ini menentukan ruang lingkup perangkat lunak yaitu perangkat lunak untuk menyusun peramalan penjumlahan penduduk di sumatra selatan.
 - Menentukan aktoryang terlibat dalam perangkat lunak.
 - Mendefinisikan *usecase* perangkat lunak

- Membuat skenario sementara dari *usecase*.

Dalam elaborasi Membuat *use case* yang telah dideskripsikan menjadi rinci.

Dalam konstruksi Penyempurnaan *use case* dan skenario.

- Membuat diagram aktor dan *usecase* menggunakan AstahCommunity. membuat skenario interaksi aktor Dalam transisi Menguji kesesuaian *usecase* dan skenario dengan tujuan analisis.

- Mendokumentasikan *usecase* dan skenario dalam bentuk laporan

- Mendokumentasikan *use case* dan skenario dalam bentuk laporan

Dalam transisi Menguji kesesuaian *use case* dan skenario dengan tujuan analisis. terhadap perangkat lunak.

2. kebutuhan perangkat Lunak dalam ensepsi:

- Mengumpulkan bahan pembelajaran yang berkaitan dengan metode

Fuzzy Time Series dan algoritma *Particle Swarm Optimization*.

- Mengumpulkan semua data yang diperlukan untuk meramalkan jumlah penduduk.

- Pengembangan perangkat lunak menggunakan Netbeans IDE 8.2.0

Dokumentasi perangkat lunak menggunakan Microsoft Word 2010 atau

Microsoft Excel 2010. Prosesor Core™i5- 7200U

Processor (2.5 GHz, 3M Cache) up to 3.10 GHz Memori Standar 4GB

DDR4 Harddisk 1TB.

- Perangkat keras yang digunakan untuk pengembangan perangkat lunak

Dalam konstruksi:

- Meninjau kembali kebutuhan perangkat lunak dan batasan masalah.

Dalam transisi:

- Mendokumentasikan kebutuhan perangkat lunak dalam bentuk laporan

3. Analisis dan desain

Dalam ensepsi :

- Membuat perancangan (*prototype*) perangkat lunak.
- Membuat *draft* diagram Kelas analisis, *sequence diagram*, dan *class diagram* berdasarkan *use case* yang telah dibuat sebelumnya.

Dalam elaborasi:

- Membuat diagram kelas analisis, *sequence diagram*, dan *class diagram*.
- Menganalisis setiap tahapan pada diagram secara rinci.

Dalam konstruksi:

- Membuat seluruh diagram menggunakan *AstahCommunity*

Dalam transisi:

- Mendokumentasikan seluruh diagram dalam bentuk laporan.

Dalam transisi:

- Mendokumentasikan seluruh diagram dalam bentuk laporan.

4. Implementasi

Dalam insepisi:

- Bahasa pemrograman yang digunakan pada perangkat lunak ini adalah Java, serta *xlsx*. untuk proses pengambilan data

Dalam elaborasi:

- Melakukan revisi terhadap *prototype* antarmuka.

Dalam konstruksi:

- Mengimplementasikan kode program dalam bahasa pemrograman Java.

Dalam transisi:

- Membuat rencana pengujian terhadap perangkat lunak.

- Dokumen proses pembuatan diagram proses Peramalan jumlah penduduk

5. Pengujian

Dalam ensepsi:

- Mempersiapkan data input untuk perangkat lunak.
- Membuat rencana pengujian terhadap perangkat lunak.

Dalam elaborasi:

- Mengimplementasikan rencana pengujian perangkat lunak dengan memasukkan data pengujian

Dalam konstruksi:

- Pengujian hasil akhir perangkat lunak peramalan penjumlahan penduduk

6. Mengevaluasi hasil pengujian.

Dalam transisi:

- Dokumentasi hasil perangkat lunak peramalan penjumlahan penduduk
Membuat kesimpulan berdasarkan hasil peramalan penjumlahan Penduduk.

3.6 Manajemen Proyek Penelitian

Manajemen proyek merupakan perencanaan aktivitas penelitian dari tahap inialisasi masalah sampai dengan pada tahap kesimpulan dari

penelitian. Adapun kegiatan-kegiatan yang berlangsung selama penelitian dapat dilihat dalam penjadwalan tugas akhir yang tertera pada Tabel III-5 sebagai berikut:

3.7 Kesimpulan

Bab ini memberikan rincian dari metodologi penelitian yang terdiri dari kerangka penelitian, metodologi pengembangan sistem, data, serta perangkat keras dan perangkat lunak sebagai persyaratan dalam membangun perangkat lunak peramalan penjumlahan penduduk di sumatra selatan

IV. BAB IV

PENGEMBANGAN PERANGKAT LUNAK

4.1 Pendahuluan

Bab ini akan menjelaskan mengenai alat yang dibutuhkan pada pelaksanaan penelitian seperti yang telah disebutkan pada bab 3. Penulis mengembangkan alat berupa perangkat lunak dengan metode pemrograman berorientasi obyek berdasarkan panduan *Rational Unified Process* yang di dalamnya terdapat fase insepisi, elaborasi, konstruksi, dan transisi. Pada tiap fase akan dijelaskan mengenai permodelan bisnis, kebutuhan, analisis dan desain, implementasi, dan pengujian.

4.2 Fase Insepisi

Aktivitas-aktivitas yang dilakukan pada fase ini meliputi penentuan user requirement dan fungsionalitas perangkat lunak pada permodelan bisnis, mengumpulkan data penelitian pada kebutuhan, membuat *use case diagram* pada analisis dan desain, mendokumentasikan *user requirement*, fungsionalitas perangkat lunak, dan *use case diagram* pada implementasi, serta memastikan *user requirement* dan fungsionalitas perangkat lunak *valid* pada pengujian.

4.2.1 Pemodelan Bisnis

Penelitian ini menggunakan metode *fuzzy time series*, dan *Particle Swarm Optimization (PSO)* untuk membantu *user* menganalisis data jumlah penduduk di

kota Palembang yang bersifat tidak pasti sehingga dapat menghasilkan nilai prediksi jumlah penduduk. Perhitungan Root Mean Squared Error (RMSE) digunakan untuk membantu *user* mengetahui metode mana yang menghasilkan nilai prediksi yang lebih baik.

Perangkat lunak yang dibangun merupakan perangkat lunak berbasis desktop yang digunakan untuk memprediksi jumlah penduduk di kota Palembang. Masukan untuk perangkat lunak yang dikembangkan berupa nama kecamatan, jumlah penduduk tertinggi, jumlah penduduk terendah, dan jumlah penduduk setiap bulan berikutnya, yang dimasukan secara manual melalui *keyboard*. Keluaran yang dihasilkan berupa nilai prediksi jumlah penduduk dari kedua metode.

4.2.2 Kebutuhan Sistem

Spesifikasi kebutuhan perangkat lunak dibangun berdasarkan pada pemodelan bisnis. Fitur-fitur utama yang disediakan perangkat lunak antara lain fitur menghitung defuzzyfikasi, menghitung RMSE, dan mengolah data.

Untuk dapat merealisasikan fitur-fitur tersebut, perangkat lunak harus memenuhi kebutuhan fungsional dan non-fungsional. Kebutuhan fungsional menjelaskan kebutuhan atau fasilitas utama perangkat lunak yang dibangun pada Tabel IV-1. Sedangkan, kebutuhan non fungsional menjelaskan kebutuhan atau fasilitas yang tidak wajib dimiliki oleh perangkat lunak, dalam artian hanya merupakan pelengkap agar perangkat lunak lebih baik kinerjanya yang dapat dilihat pada Tabel IV-2.

4.2.3 Analisis Dan Desain

Tahapan analisis dan desain pada fase inepsi akan menjelaskan analisis kebutuhan perangkat lunak dan desain dari analisis tersebut dimodelkan ke dalam bentuk *use case* diagram.

4.2.3.1 Analisis Kebutuhan Perangkat Lunak

Dari pemodelan bisnis yang telah dijabarkan, perangkat lunak harus memiliki kemampuan sebagai berikut.

1. Melakukan perhitungan metode *fuzzy Time Series* dan *Partical Swarm Optimization (PSO)*.
2. Melakukan perhitungan R MSE terhadap nilai yang dihasilkan kedua metode.
3. Mengedit data, menyimpan data, mereset data, dan menghapus data.

Proses perangkat lunak dimulai dengan memasukkan riwayat data jumlah penduduk berupa angka, jumlah penduduk minimal, dan nilai jumlah penduduk maximal untuk membenuk domain fuzzy. Kemudian memasukkan nilai total jumlah penduduk per bulan selama lima tahun terakhir yang akan diteliti. Hasil prediksi kedua metode akan dibandingkan dengan nilai jumlah penduduk sebelumnya sebagai nilai perbandingan dalam proses perhitungan RMSE.

4.2.3.2 Analisis Data

Data yang dibutuhkan pada perangkat lunak yang dibangun adalah sebanyak 60 data perbulan selama lima (5) tahun dengan masing-masing jumlah memiliki 3 variabel yaitu, bulan, jumlah penduduk dan total.

Data yang diperoleh disimpan dalam format file bertipe *.xlsx*. Keseluruhan data tersebut terbagi dalam 5 tahun data histori jumlah penduduk periode tahun 2016-2020.

4.2.3.3 Analisis *Fuzzy Time Series* dan *Partical Swarm Optimization (PSO)*

Fuzzy Time Series digunakan untuk melakukan proses peramalan pada data histori jumlah penduduk, sedangkan algoritma *Particle Swarm Optimization* digunakan untuk mengoptimasi fungsi keanggotaan *fuzzy* agar dapat memberikan hasil peramalan yang lebih baik dan dapat meminimalisir kesalahan penyimpangan akurasi hasil peramalan. Berikut ini langkah-langkah proses perhitungan peramalan *Fuzzy Time Series* dengan algoritma *Particle Swarm Optimization* (Huang, et al., 2011):

1. Menentukan Semesta Pembicaraan (*Universe of Discourse*).
2. Inisialisasi (xmin) nilai minimum (xmax) untuk nilai maksimum, (D1) margin bawah (D2) margin atas dan di mana N merupakan jumlah partikel.
3. Membagi Semesta Pembicaraan yang telah dibuat menjadi beberapa interval berdasarkan posisi partikel.
4. Melakukan fuzzifikasi data historis.
5. Membuat *fuzzy logical relationship*.
6. Menyusun *fuzzy logical relationship group*.
7. Menghitung nilai peramalan.
8. Menghitung akurasi peramalan dengan nilai RMSE.
9. Selama solusi optimal belum ditemukan atau belum mencapai iterasi maksimum, ulangi langkah 2-8.

Dalam proses optimasi oleh *PSO* ini representasi partikel *PSO* yang digunakan adalah *real-coded* (pengkodean real) karena menyesuaikan dengan batasan interval *fuzzy time series* berupa bilangan real sebagai posisi partikel dalam dimensi pencarian. Posisi partikel berkaitan dengan batas interval pada *fuzzy time series* sebagai posisi partikel, hanya saja batas interval yang menjadi batas bawah maupun batas atas semesta pembicaraan tidak diikutsertakan dalam

representasi partikel karena bersifat tetap.

Diagram alur optimasi interval *fuzzy time series* menggunakan *PSO* ditunjukkan pada Gambar IV-1.

4.2.3.4 Desain Perangkat Lunak

Desain perangkat lunak digambarkan dengan diagram *use case* dan diagram aktivitas.

1. Use Case

Subbab ini menjelaskan gambaran fungsionalitas perangkat lunak yang dibangun dengan menggunakan pemodelan *Use Case*.

a) Diagram Use Case

Diagram *Use Case* menjelaskan secara umum kegiatan yang dilakukan oleh aktor (pengguna) terhadap perangkat lunak yang dapat dilihat pada Gambar IV-2.

b) Tabel Definisi Aktor

Entitas yang dijadikan aktor pada perangkat lunak ini adalah peneliti, yang dijelaskan pada tabel IV-3.

Tabel Definisi Use Case

Daftar definisi dari tiap *use case* pada perangkat lunak dapat dilihat pada Tabel IV-4.

c) Skenario Use Case

Berikut ini adalah skenario dari *Use Case* yang telah didefinisikan pada subbab sebelumnya.

2. Kelas Analisis

Pada sub bab kelas analisis akan digambarkan kelas analisis dari masing-masing use case yang telah didefinisikan sebelumnya berdasarkan perangkat lunak yang dibangun.

a) Kelas Analisis Memuat Data

Gambar IV-3 merupakan kelas analisis yang melakukan proses memuat data jumlah penduduk terdapat 3 kelas, meliputi 1 kelas *Boundary* yaitu HalamanUtama; 1 kelas *Controller* yaitu DokumenManager; 1 kelas *Entity* yaitu Dokumen.

Gambar IV-3. Kelas Analisis Memuat Data

User memulai pada kelas *Boundary* index untuk memuat data, kelas index akan mengakses data jumlah penduduk .

- b) Kelas Analisis Meramalkan jumlah penduduk dengan *FTS* dan algoritma *PSO*

Gambar IV–4 merupakan kelas analisis yang melakukan peramalan dengan *Fuzzy Time Series* dengan algoritma *Particle Swarm Optimization*

Gambar IV-4. Kelas Analisis Optimasi Akurasi Peramalan dengan *fuzzy time series* algoritma *Particle Swarm Optimization*

yang mana terdapat 4 kelas, meliputi 1 kelas *Boundary* yaitu index; 3 kelas *controller* yaitu *fts*, *swarm* dan *util*. *User* memulai pada *interface* index untuk melakukan proses meramalkan jumlah penduduk. Kelas *Boundary* index akan melakukan perhitungan peramalan dengan mengakses kelas *Controller FTS*, *swarm* dan *util*.

- c) Kelas Analisis Akurasi Peramalan

Gambar IV–6 merupakan kelas analisis yang melakukan proses optimasi akurasi peramalan terdapat 4 kelas, meliputi 1 kelas *Boundary* yaitu index; 3 kelas *Controller* yaitu *util*, *swarm* dan *FTS*.

Gambar IV-5. Kelas peramalan akurasi

User memulai pada *interface* index untuk melakukan proses akurasi peramalan jumlah penduduk Kelas *Boundary* index akan melakukan perhitungan peramalan terlebih dahulu dengan mengakses kelas *Controller util*. Selanjutnya mengakses kelas *util* untuk melakukan proses perhitungan peramalan. Pada proses *util* melakukan proses peramalan dengan *FTS* mengoptimasi menggunakan kelas *PSO* untuk mendapatkan hasil peramalan jumlah penduduk.

4.3 Fase Elaborasi

Aktivitas-aktivitas yang dilakukan pada fase ini meliputi perancangan data, perancangan antarmuka, identifikasi kebutuhan sistem, dan permodelan menggunakan *sequence diagram*.

4.3.1 Pemodelan Bisnis

Pada subbab ini akan dibahas mengenai perancangan perangkat lunak yang dibangun. Perancangan dilakukan berdasarkan hasil analisa yang telah dilakukan pada fase insepisi. Perancangan yang dibahas pada subbab ini meliputi perancangan data dan perancangan antar muka.

4.3.1.1 Perancangan Data

Perangkat lunak yang akan dibangun memiliki kemampuan menghasilkan peramalan dan akurasi peramalan terhadap data histori penjualan jumlah penduduk. Adapun data yang akan melalui proses peramalan adalah data histori penjualan produk pakaian yang disimpan dalam file berformat .xlsx yang terdiri dari 3 jenis data yaitu data tahun, bulan, dan jumlah penduduk.

4.3.1.2 Perancangan Antar Muka

Rancangan antarmuka perangkat lunak digambarkan pada Gambar IV-6.

Gambar IV–6. Rancangan Antarmuka Perangkat Lunak

Gambar IV–6 merupakan halaman utama yang berfungsi untuk melakukan memuat data histori, perhitungan hasil peramalan dan akurasi peramalan dengan *Fuzzy Time Series*, dan optimasi akurasi peramalan dengan *Particle Swam Optimization* serta hasil peramalan dan akurasi peramalan yang telah dioptimasi.

4.3.2 Kebutuhan Sistem

Pada subbab ini akan dibahas mengenai kebutuhan sistem dari perangkat lunak yang dibangun berdasarkan hasil analisis dan perancangan pada tahap selanjutnya. Untuk membangun perangkat lunak dalam penelitian ini dibutuhkan

perangkat keras, perangkat lunak dan bahasa pemrograman. Bahasa pemrograman yang digunakan untuk implementasi perangkat lunak adalah Bahasa typescript. Perangkat keras yang digunakan pada tahap pengembangan dan penelitian ini adalah laptop dengan spesifikasi sebagai berikut:

1. Laptop ASUS;
2. Processors Intel(R) Core(TM) i3-3217U CPU @1.80GHz;
3. RAM 4 GB;
4. *Hard Disk* 1 TB .

Sedangkan perangkat lunak yang digunakan untuk implementasi yaitu:

1. Sistem operasi Windows 10 Pro 64-bit ;
2. *Compiler* sublime text 3
3. *Xampp control panel* v3.2.2.

4.3.3 Diagram Sequence

Berdasarkan *use case* yang dibentuk, diagram sequence yang dibentuk pada pengembangan perangkat lunak ini berjumlah tiga buah. Sequence diagram Memuat Data dapat dilihat pada Gambar IV-7, sequence diagram Meramalkan jumlah penduduk dengan *Fuzzy Time Series* dengan algoritma *Particle Swarm Optimization* dapat dilihat pada Gambar IV-8, dan sequence diagram Akurasi Peramalan dapat dilihat pada Gambar IV-9.

4.3.3.1 *Sequence Diagram* Memuat Data

Gambar IV-7. *Sequence Diagram* Memuat Data

4.3.3.2 *Sequence Diagram* peramalan jumlah penduduk dengan *FTS* dengan algoritma *PSO*

Gambar IV-8. *Sequence Diagram* peramalan jumlah penduduk dengan *FTS* dengan algoritma *PSO*

4.3.3.3 *sequence diagram* Akurasi

Gambar IV-9 *Sequence Diagram* akurasi

4.4 Fase Konstruksi

Aktivitas-aktivitas yang dilakukan pada fase ini meliputi pengembangan perangkat lunak berdasarkan rancangan-rancangan yang telah dilakukan pada fase sebelumnya. Perangkat lunak yang dikembangkan akan digunakan sebagai alat penelitian yang nantinya akan digunakan oleh peneliti.

4.4.1 Kebutuhan Sistem

Dalam pengembangan perangkat lunak ini, penulis menggunakan beberapa *library*. Diantaranya adalah *Library* POI untuk membaca excel ke dalam perangkat lunak.

Diagram Kelas

Pada penelitian ini terdapat 7 kelas yang terdiri dari 2 kelas *Entity* (kelas Dokumen, jumlahpenduduk), 4 kelas *Controller* (kelas *FTS*, *PSO*, AkurasiPeramalan dan DokumenManager), dan 1 kelas *Boundary* (kelas HalamanUtama. Di gambar IV-10 menunjukkan hubungan antar kelas tersebut.

Gambar IV-10 diagan kelas perangkat lunak

4.4.2 Implementasi

Pada tahap ini peneliti mengembangkan perangkat lunak berdasarkan diagram kelas dan rancangan antarmuka yang telah dibuat sebelumnya.

4.4.2.1 Implementasi Kelas

Tabel IV-8 menunjukkan implementasi kelas dalam bahasa typescript.

Tabel IV–8. Implementasi Kelas

4.4.2.2 Implementasi Antarmuka

Gambar IV–10 merupakan antarmuka halaman utama perangkat lunak berdasarkan perancangan pada fase elaborasi.

Gambar IV-11. Antarmuka Halaman Utama Perangkat Lunak

4.5 Fase Transisi

Pada subbab ini dibahas mengenai pengujian dari perangkat lunak yang telah dibangun. Pengujian dilakukan berdasarkan perangkat lunak hasil pengembangan di fase konstruksi.

4.5.1 Pemodelan Bisnis

Pengujian perangkat lunak secara *black box* dengan terlebih dahulu membuat rencana pengujian berdasarkan *use case* yang dibuat pada fase inepsi.

4.5.2 Rencana Pengujian

Rencana pengujian pada perangkat lunak digambarkan dalam tabel-tabel. Kolom pada tabel meliputi identifikasi, pengujian, jenis pengujian, serta tingkat pengujian.

4.5.2.1 Rencana Pengujian *Use Case* Memuat Data

Tabel IV–9 menerangkan rencana pengujian memasukkan dokumen perangkat lunak berdasarkan *Use Case*.

Tabel IV–9. Rencana Pengujian *Use Case* Memuat Data

4.5.2.2 Rencana Pengujian *Use Case* Meramalkan jumlah penduduk dengan

FTS

Tabel IV–10 menerangkan rencana pengujian melakukan Peramalan *Fuzzy Time Series* pada perangkat lunak berdasarkan Use Case.

Tabel IV–10. Rencana Pengujian Use Case Meramalkan jumlah penduduk menggunakan dengan *Fuzzy Time Series dan PSO*

4.5.2.3 Rencana Pengujian Use Case Optimasi Akurasi Peramalan dengan PSO

Tabel IV–10 menerangkan rencana pengujian melakukan optimasi akurasi peramalan dengan algoritma *Particle Swarm Optimization* pada perangkat lunak berdasarkan Use Case.

4.5.2.4 Rencana Pengujian Use Case Akurasi Peramalan

Tabel IV–11 menerangkan rencana pengujian melakukan akurasi peramalan pada perangkat lunak berdasarkan Use Case.

Tabel IV–11. Rencana Pengujian Use Case Akurasi Peramalan

4.5.3 Implementasi

Berikut ini adalah kasus uji yang dilakukan terhadap perangkat lunak yang dibangun. Kasus uji dilakukan berdasarkan rencana uji yang telah dipaparkan sebelumnya.

4.5.3.1. Pengujian *Use Case* Memuat Data

4.5.3.2. Pengujian *Use Case* Meramalkan jumlah penduduk dengan *Fuzzy Time Series* dengan *Particle Swarm Opimize*

Tabel IV–13. Pengujian *Use Case* Meramalkan jumlah penduduk dengan *Fuzzy Time Series* dan *particle swarm optimize*

4.5.3.3. Pengujian *Use Case* Akurasi Peramalan

Tabel IV–14. Pengujian *Use Case* Akurasi Peramalan
IV-29

Berdasarkan hasil pengujian yang telah dilakukan terhadap kesesuaian perancangan perangkat lunak dan implementasi perangkat lunak maka dapat disimpulkan bahwa unit dan antarmuka yang dibangun dapat berjalan dengan baik. Hal ini ditandai dengan kesimpulan hasil skenario pada kasus uji semuanya memberikan kesimpulan yang sama, yaitu diterima.

4.6 Kesimpulan

Pengembangan perangkat lunak dengan metode RUP dilakukan untuk mengembangkan alat penelitian yang dapat digunakan untuk penelitian ini. Aktivitas yang dilakukan pada fase inepsi meliputi identifikasi kebutuhan perangkat lunak, dan perancangan kebutuhan perangkat lunak menggunakan diagram *use case* dan diagram *activity*. Kemudian aktivitas yang dilakukan pada fase elaborasi meliputi perancangan user interface, dan perancangan alur interaksi pada sistem menggunakan diagram *sequence* berdasarkan *use case* yang telah dirancang pada fase inepsi. Lalu aktivitas yang dilakukan pada fase konstruksi yaitu implementasi rancangan perangkat lunak yang telah dirancang sebelumnya ke dalam kode program menggunakan paradigma pemrograman berorientasi objek dan dimodelkan ke dalam *diagram class*. Fase terakhir yaitu transisi meliputi aktivitas pengujian perangkat lunak untuk memastikan bahwa perangkat lunak yang dikembangkan telah memenuhi kebutuhan yang telah ditentukan sebelumnya

agar perangkat lunak dapat digunakan untuk penelitian.

BAB V

HASIL DAN ANALISIS PENELITIAN

5.1 Pendahuluan

Pada bab IV telah dilakukan pengembangan perangkat lunak yang menjadi alat penelitian untuk meramalkan jumlah penduduk dengan menggunakan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*.. Sedangkan pada Bab V ini akan disajikan hasil nilai akurasi peramalan yang didapatkan dari perhitungan meramalkan jumlah penduduk dengan menggunakan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*. Untuk mengetahui hasil nilai akurasi peramalan yang didapatkan dari perhitungan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization* lebih baik atau tidak, maka pada Bab V ini disajikan juga hasil nilai akurasi peramalan yang didapatkan dari perhitungan peramalan jumlah penduduk dengan menggunakan metode *Fuzzy Time Series* tanpa algoritma *Particle Swarm Optimization* yang berfungsi untuk membandingkan hasil nilai akurasi peramalan dengan menggunakan metode *Fuzzy Time Series* dan algoritma *Particle Swarm Optimization*.

5.2 Hasil Percobaan Penelitian

Berdasarkan hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa unit dan antarmuka yang diujikan berjalan dengan baik. Hal tersebut terlihat dari semua kesimpulan skenario pengujian yang sama, yaitu diterima. Selanjutnya, pengujian dilakukan dengan menggunakan 60 data bulanan yang

merupakan data histori jumlah penduduk yang terdiri dari 2 jenis data yaitu data bulan dan jumlah penduduk. Pengujian sistem menghasilkan data hasil peramalan dan akurasi peramalan berdasarkan jenis data masing-masing.

5.2.1 Data Set Pengujian

Tabel data set pengujian pada perangkat lunak ditunjukkan pada Tabel V-1. Terdapat 24 data bulanan yang merupakan data histori jumlah penduduk yang terdiri dari 2 jenis data yaitu data tahun, bulan, jumlah penduduk. Pengujian 60 data bulanan mulai dari Bulan Januari 2016 hingga Desember 2020 yang akan dilakukan proses uji coba untuk mendapatkan hasil peramalan jumlah penduduk

5.2.2 Hasil Pengujian *Fuzzy Time Series* dan *Particle swarm optimization*

Pengujian dilakukan dengan menggunakan 2 jenis data histori jumlah penduduk yang terdiri data bulan, dan jumlah penduduk. *Fuzzy Time Series* pada perangkat lunak ini berfungsi untuk meramalkan jumlah penduduk yang nantinya hasil akurasi peramalan menggunakan metode *Fuzzy Time Series* dengan *Particle Swarm Optimization*. Pada Tabel V-2 menunjukkan hasil dari perhitungan peramalan jumlah penduduk dengan menggunakan 60 data uji untuk mendapatkan hasil peramalan penjualan produk pakaian dan akurasi peramalan.

Tabel V-1. Data Hasil Peramalan dan Hasil Akurasi Peramalan menggunakan metode *Fuzzy Time Series* dan *Particle swarm optimization*

5.2.3 Hasil Pengujian *Fuzzy Time Series* dengan *Particle Swarm Optimization*

Pengujian dilakukan dengan menggunakan 2 jenis data bulan dan jumlah penduduk. *Particle Swarm Optimization* pada perangkat lunak ini berfungsi mengoptimasi interval pada *Fuzzy Time Series* untuk meningkatkan akurasi peramalan sedangkan *Fuzzy Time Series* berfungsi untuk melakukan perhitungan peramalan jumlah penduduk. Sebelum melakukan pengujian hasil percobaan peramalan jumlah penduduk dengan perhitungan metode *Fuzzy Time Series* dilakukan terlebih dahulu menentukan parameter *Particle Swarm Optimization*. Parameter *Particle Swarm Optimization* terdiri dari jumlah partikel, jumlah iterasi dan kombinasi. Grafik pada Gambar V-1 menunjukkan hasil representasi akurasi peramalan jumlah penduduk yang didapatkan dari setiap percobaan pada pengujian jumlah partikel.

Berdasarkan hasil pengujian jumlah partikel yang terdapat pada Gambar V-1 menunjukkan hasil pengujian yang diperoleh bahwa solusi terbaik terletak pada pengujian dengan jumlah partikel 60 dengan nilai rata-rata terendah yaitu 1525945 dibandingkan dengan jumlah partikel yang lain. Dari grafik tersebut dapat dilihat bahwa rata-rata nilai cenderung naik dengan bertambahnya jumlah partikel karena pengaruh inisialisasi partikel secara acak yang menyebabkan hasil pengujian sedikit berbeda antara satu pengujian dengan lainnya. Menurunnya nilai t seiring dengan bertambahnya jumlah partikel, meskipun hal ini tidak sepenuhnya menjamin dapat menurunkan nilai sebab walaupun terdapat banyak partikel tetapi saat inisialisasi berada jauh dari titik optimum maka untuk menuju titik optimum akan memerlukan waktu lebih lama dibanding partikel yang sudah berada di dekat titik optimum.

5.2.4 Analisis Hasil Pengujian Perbandingan Akurasi Peramalan

Berdasarkan dari hasil pengujian peramalan jumlah penduduk yang telah dilakukan, maka dilakukan analisis hasil perbandingan akurasi peramalan pada pengujian yang telah dilakukan. Nilai akurasi terbaik diperoleh berdasarkan nilai akurasi terendah dari masing-masing hasil akurasi peramalan.

Pada tabel V-2 dilakukan pengujian perhitungan peramalan jumlah penduduk dengan menggunakan *Fuzzy Time Series* menggunakan *Optimasi Particle Swarm Optimization* didapatkan hasil rata-rata akurasi dari keseluruhan pengujian sebesar $RMSE = 22182,58487$ dan $AFER$ sebesar $= 0,8175440\%$. Menandakan bahwa rata-rata hasil error minimum yang dihasilkan oleh peramalan jumlah penduduk

dengan *fuzzy time series* dan *particle swarm optimize* kurang dari 1% yang berarti bahwa tingkat kesalahan yang dihasilkan cukup kecil.

5.3 Kesimpulan

Dari hasil pengujian yang telah didapatkan dan dilakukan tahapan analisa hasil peramalan, telah jelas diuraikan hasil masing-masing pengujian yang telah dilakukan. Berdasarkan pada tabel V-3, disimpulkan bahwa hasil akurasi peramalan dengan *Fuzzy Time Series* menggunakan algoritma *Particle Swarm Optimization* dengan rata-rata lebih baik untuk nilai parameter pada algoritma *Particle Swarm Optimization* didapatkan nilai parameter terbaik pada pengujian ini adalah untuk jumlah data 60 Dalam hal ini, besaran nilai parameter *Particle Swarm Optimization* yang berbeda pada tiap kali pengujian mempengaruhi nilai *cost* yang dihasilkan oleh sistem.

BAB VI

KESIMPULAN DAN SARAN

6.1 Pendahuluan

Pada bab sebelumnya telah dilakukan analisis penelitian mengenai prediksi jumlah penduduk di kota Palembang menggunakan metode *fuzzy Time Series* dan *Partical Swarm Optimization (PSO)*. Pada bab VI penulis memberikan kesimpulan dan saran yang diharapkan dapat menjadi acuan bagi penelitian selanjutnya di bidang yang sama.

6.2 Kesimpulan

Berdasarkan penelitian yang telah dilakukan pada penelitian ini, maka kesimpulan yang diperoleh adalah sebagai berikut:

1. Perangkat lunak untuk meramalkan jumlah penduduk di Sumatra Selatan dengan menggunakan metode *Fuzzy time series* dan *particle swarm optimization* telah berhasil dikembangkan. *Particle Swarm Optimization* pada perangkat lunak ini berfungsi mengoptimasi interval pada *Fuzzy Time Series* untuk meningkatkan akurasi peramalan dan parameter yang digunakan untuk optimasi adalah terdiri dari jumlah partikel, jumlah iterasi dan kombinasi.
2. Perhitungan peramalan dengan metode *Fuzzy Time Series* menggunakan *Particle Swarm Optimization* untuk meramalkan jumlah penduduk didapatkan penurunan hasil rata-rata tingkat kesalahan (*error*), rata-rata

akurasi dari keseluruhan pengujian sebesar $RMSE = 22182,58487$ dan $AFER = 0,8175440\%$. Menandakan bahwa rata rata hasil error minimum yang dihasilkan oleh peramalan jumlah penduduk dengan *fuzzy time series* dan *particle swarm optimize* kurang dari 1% yang berarti bahwa tingkat kesalahan yang dihasilkan cukup kecil.

6.3 Saran

Beberapa saran yang dapat dilakukan untuk penelitian selanjutnya adalah sebagai berikut :

1. Perangkat lunak ini dapat dikembangkan dengan menggunakan data historis dengan selain satuan bulanan sebagai masukan untuk meramalkan jumlah penduduk.
2. Perangkat lunak ini dapat dikembangkan dengan membuat selain jumlah penduduk
3. Perangkat lunak ini dapat dikembangkan dengan penggunaan metode peramalan dengan metode optimasi lainnya sebagai perbandingan sehingga mampu menghasilkan peramalan yang lebih baik.

DAFTAR PUSTAKA

- Anwary, A. A. 2011. Prediksi Kurs Rupiah Terhadap Dollar Amerika Menggunakan *Metode Fuzzy Time Series*. Skripsi. Universitas Diponegoro (UNDIP). Semarang.
- Chen, S.-M., Hsu, C.-C., & others. (2004). A new method to forecast enrollments using *fuzzy time series*. *International Journal of Applied Science and Engineering*, 2(3), 234–244.
- Huang L,Y. 2011. A hybrid forecasting model for enrollments based and aggregated *fuzzy time series and particle swarm optimize*. *Vol. 38, No 7, july 2018, pages 8014-8023*
- Irawan, Y. U. (2015). Perbandingan Performansi Metode Peramalan *Fuzzy Time Series* yang Dimodifikasi dan Jaringan Syaraf Tiruan Backpropagation. *JURNAL SAINS DAN SENI ITS Vol. 4, No.2, (2015) 2337-3520 (2301-928X Print)*, 31-36.
- Jumriani, N. (2018). Metode Peramalan dan Peramalan Permintaan. *Metode Peramalan dan Peramalan Permintaan*, 1-27.
- Pambudi, R. A. (2018). Implementasi *Fuzzy Time Series* untuk Memprediksi Jumlah Kemunculan titik api. *Vol. 2, No. 11, November 2018, hlm. 4767-4776, 4767-4776*.
- Prasojo, C. A. (2018). *Optimasi Fuzzy Time Series* Menggunakan Algoritma *Particle Swarm Optimization* Untuk Peramalan Jumlah Penduduk Di Kabupaten Probolinggo. *Vol 2 No 8 (2018)*, 8.

robandi. (2016). Implementasi *Fuzzy Time Series* untuk Peramalan Data. 1-33.SARJANA, P. P. (2013). *PERAMALAN PENJUALAN DENGAN METODE FUZZY TIME SERIES*. semarang.

Satria, I. M. (2015). PERAMALAN JUMLAH WISATAWAN AUSTRALIAYANGBERKUNJUNG KE BALI MENGGUNAKANMULTIVARIAT *FUZZY TIME SERIES*. *E-JurnalMatematika Vol. 4 (3), Agustus 2015, pp. 90-97, 90-97.*

SUMSEL, B. (2021). *BPS kota Palembang*. Dipetik february 5, 2022, dari<https://palembangkota.bps.go.id>

Susetyoko, R. (Vol 13 No 2 (2016)). Peramalan Gabungan Rantai Markov dan ModelDeret Waktu Pada Kasus Peramalan Kurs Nilai Mata Uang. *Peramalan Kurs Nilai Mata Uang*, 50-62.

wardah, s. (2016). ANALISIS PERAMALAN PENJUALAN PRODUK KERIPIKPISANG. 135-142.

Zahro, H. Z. (2020). *OPTIMASI PARTICEL SWARM OPTIMAZATION (PSO)UNTUKPENENTUAN BASE TRANCIVIER SYSTEM (BTS)*. Vol 3,No. 1, *Februari 2020* , 7-10.

Lampiran 1: data jumlah penduduk Dinas Kependudukan Dan Catatan Sipil Kota Palembang

DATA JUMLAH PENDUDUK KOTA PALEMBANG 2016 - 2020

berdasarkan catatan **DISDUKCAPIL Kota Palembang** tahun 2021:

NO	TAHUN	BULAN												JUMLAH
		JANUARI	FEBRUARI	MARET	APRIL	MEI	JUNI	JULI	AGUSTUS	SEPTEMBER	OKTOBER	NOVEMBER	DESEMBER	
1	2016	1.525.945	1.526.174	1.528.229	1.527.213	1.530.175	1.529.990	1531.731	1.532.102	1.532.000	1.534.673	1.535.980	1.555.980	1.555.980
2	2017	1.559.132	1.562.540	1.567.909	1.570.187	1.585.009	1.593.840	1.594.601	1.590.321	1.585.582	1.587.119	1.579.000	1.573.898	1.573.898
3	2018	1.578.689	1.575.221	1.579.991	1.574.185	1.577.540	1.580.312	1.583.221	1.588.139	1.589.003	1.587.841	1.589.187	1.592.248	1.592.248
4	2019	1.621.641	1.618.967	1.620.082	1.622.219	1.617.298	1.615.658	1.613.935	1.611.112	1.613.923	1.616.256	1.616.990	1.619.533	1.619.533
5	2020	1.623.146	1.628.435	1.626.213	1.632.645	1.637.908	1.641.668	1.645.758	1.642.932	1.659.174	1.667.214	1.665.546	1.668.164	1.668.164
Jumlah keseluruhan													6.320.290	

Kesimpulan:

1. Dalam kurun waktu 5 tahun banyak peningkatan dari jumlah penduduk kota palembang
2. Jumlah penduduk paling sedikit
3. Jumlah penduduk paling banyak pada bulan Desember 2020 (Semester II)

Lampiran 2: *source program*

index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, shrink-to-
    fit=no">
  <title>
    FTS-PSO
  </title>
  <link rel="stylesheet"
    href="assets/bootstrap/css/bootstrap.min.css?h=c25e0cf726e61fe1945
    be655ed94b205">
  <link rel="stylesheet" href="assets/css/Navbar-Centered-Brand-
    Dark.css?h=befd8a398792e305b7ffd4a176b5b585">
  <link rel="stylesheet"
    href="assets/css/styles.css?h=b57243cc2e71ef3c66bc178bffd4805">
</head>
<body>
  <nav class="navbar navbar-dark navbar-expand-md fixed-top bg-dark py-3">
    <div class="container">
      <a class="navbar-brand d-flex align-items-center" href="#">
        <span>
          FTS-PSO
        </span>
      </a>
      <button data-bs-toggle="collapse" class="navbar-toggler" data-bs-target="#navcol-
        5">
        <span class="visually-hidden">
          Toggle navigation
        </span>
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navcol-5">
        <ul class="navbar-nav ms-auto">
          <li class="nav-item">
            <button id="ctaRetry" class="btn btn-outline-light d-none" type="button"
              title="Retry">
              <svg
                xmlns="http://www.w3.org/2000/svg" width="1em" height="1em"
                fill="currentColor" viewBox="0 0 16 16"
                class="bi bi-arrow-counterclockwise">
                <path fill-rule="evenodd" d="M8 3a5 5 0 1 1-4.546 2.914.5.5 0 0-.908-
                  .417A6 6 0 1 0 8 2v1z"></path>
                <path d="M8 4.466V.534a.25.25 0 0 0-.41-.192L5.23 2.308a.25.25 0 0 0
                  .384.236 1.966A.25.25 0 0 0 8 4.466z"></path>
              </svg>
            </button>
          </li>
          <li class="nav-item">
            <div id="stageControl" class="btn-group d-none" role="group">
              <input type="radio" class="btn-check" name="stage" id="stageClient"
```

```

        autocomplete="off" value="client">
<label class="btn btn-outline-primary" for="stageClient" data-bs-
toggle="tooltip" data-bs-placement="bottom" title="Run in Client
Side">
    Client
</label>

<input type="radio" class="btn-check" name="stage" id="stageServer"
autocomplete="off" value="server">
<label class="btn btn-outline-primary" for="stageServer" data-bs-
toggle="tooltip" data-bs-placement="bottom" title="Run at Server
Side">
    Server
</label>
</div>
</li>
</ul>
</div>
</div>
</nav>
<div class="container-fluid floor-down" id="mainContainer" style="height:
calc(100vh - 72px);margin-top: 72px;">
<form id="configForm" class="d-flex flex-column w-100 px-3 d-none">
<fieldset class="mb-3">
<legend>Dataset Source</legend>
<input class="form-control" type="file" name="dataset" required=""
accept="text/csv">
<a href="assets/dataset.csv" target="_blank">Download example dataset</a>
</fieldset>
<div class="row flex-fill mb-3">
<div class="col-md-12 col-lg-6 mb-md-3 mb-lg-0">
<div class="card h-100">
<div class="card-header">
<h5 class="mb-0">
    Fuzzy Time Series
</h5>
</div>
<div class="card-body floor-down">
<div class="mb-3">
<label class="form-label" for="ftsMinMargin">
    Lower Margin
</label>
<input class="form-control" type="number" id="ftsMinMargin"
name="ftsMinMargin" placeholder="Minimum Margin" required=""
step="0.0001">
</div>
<div class="mb-3">
<label class="form-label" for="ftsMaxMargin">
    Upper Margin
</label>
<input class="form-control" type="number" id="ftsMaxMargin"
name="ftsMaxMargin" placeholder="Maximum Margin" required=""
step="0.0001">
</div>
<div class="mb-3">
<label class="form-label" for="ftsInterval">
    Interval

```



```

        </label>
        <input class="form-control" type="number" id="ftsInterval"
            name="ftsInterval" placeholder="Interval" required=""
            step="0.0001">
    </div>
</div>
</div>
</div>
<div class="col-md-12 col-lg-6">
    <div class="card h-100">
        <div class="card-header">
            <h5 class="mb-0">
                Particle Swarm Optimization
            </h5>
        </div>
        <div class="card-body floor-down">
            <div class="mb-3">
                <label class="form-label">
                    Spaces Configuration
                </label>
                <div class="row mx-1">
                    <div class="col-sm-12 col-md-6">
                        <div class="row">
                            <div class="col-3">
                                <label class="col-form-label" for="psoSpaceMin">
                                    Min
                                </label>
                            </div>
                            <div class="col">
                                <input class="form-control" type="number" id="psoSpaceMin"
                                    name="psoSpaceMin" placeholder="Minimum Value"
                                    step="0.0001">
                            </div>
                        </div>
                    </div>
                    <div class="col-sm-12 col-md-6">
                        <div class="row">
                            <div class="col-3">
                                <label class="col-form-label" for="psoSpaceMax">
                                    Max
                                </label>
                            </div>
                            <div class="col">
                                <input class="form-control" type="number" id="psoSpaceMax"
                                    name="psoSpaceMax" placeholder="Maximum Value"
                                    step="0.0001">
                            </div>
                        </div>
                    </div>
                </div>
            </div>
            <div class="mb-3">
                <label class="form-label" for="psoWeight">
                    Weight
                </label>
                <input class="form-control" type="number" id="psoWeight"
                    name="psoWeight" placeholder="Weight" required=""

```

```
        step="0.0001">
</div>
<div class="mb-3">
  <label class="form-label" for="psoMaxIteration">
    Maximum Iteration Count
  </label>
  <input class="form-control" type="number" id="psoMaxIteration"
    name="psoMaxIteration" placeholder="Maximum Iteration Count"
    min="1" step="1" required="">
</div>
<div class="mb-3">
  <label class="form-label" for="psoParticleCount">
    Spawned Particle Count
  </label>
  <input class="form-control" type="number" id="psoParticleCount"
    name="psoParticleCount" placeholder="Spawned Particle Count"
    min="1" step="1" required="">
</div>
<div class="mb-3">
  <label class="form-label" for="psoConfidence">
    Swarm Confidence
  </label>
  <input class="form-control" type="number" id="psoConfidence"
    name="psoConfidence" placeholder="Swarm Confidence"
    required="" step="0.0001">
</div>
<div class="mb-3">
  <label class="form-label" for="psoSelfConfidence">
    Self Confidence
  </label>
  <input class="form-control" type="number" id="psoSelfConfidence"
    name="psoSelfConfidence" placeholder="Self Confidence"
    required="" step="0.0001">
</div>
<div class="mb-3">
  <label class="form-label">
    Stopping Criteria
  </label>
  <div class="row mx-1">
    <div class="col-sm-12 col-md-6">
      <div class="row">
        <div class="col-3">
          <label class="col-form-label" for="psoStopMse">
            MSE
          </label>
        </div>
        <div class="col">
          <input class="form-control" type="number" id="psoStopMse"
            name="psoStopMse" placeholder="Minimum MSE" min="0"
            required="" step="0.0001">
        </div>
      </div>
    </div>
  </div>
  <div class="col-sm-12 col-md-6">
    <div class="row">
      <div class="col-3">
        <label class="col-form-label" for="psoStopAfer">
```



```

    &quot;,&quot;5300&quot;,&quot;6250&quot;,&quot;7800&quot;,&quot;9800&quot;,&quot;15000&quot;]]},&quot;options&quot;:{&quot;
    ot;maintainAspectRatio&quot;:true,&quot;legend&quot;:{&quot;disp
    lay&quot;:true,&quot;labels&quot;:{&quot;fontStyle&quot;:&quot;n
    ormal&quot;},&quot;position&quot;:&quot;right&quot;},&quot;title
    &quot;:{&quot;fontStyle&quot;:&quot;bold&quot;,&quot;display&q
    uot;:true,&quot;text&quot;:&quot;Fuzzy Time Series Result
    Optimized using Particle Swarm
    Optimization&quot;},&quot;scales&quot;:{&quot;xAxes&quot;:[{&q
    uot;ticks&quot;:{&quot;fontStyle&quot;:&quot;normal&quot;}}],&q
    uot;yAxes&quot;:[{&quot;ticks&quot;:{&quot;fontStyle&quot;:&quo
    t;normal&quot;}}]}]}"></canvas>
  </div>
</div>
</div>
</div>
</div>
</div>
<script src="https://unpkg.com/systemjs@6.12.1/dist/system.min.js"></script>
<script
  src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bu
  ndle.min.js"></script>
<script
  src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.8.0/Chart.bundle
  .min.js"></script>
<script src="assets/js/bs-init.js?h=e2b0d57f2c4a9b0d13919304f87f79ae"></script>
<script src="scripts/fts-pso/types.ts"></script>
<script src="scripts/fts-pso/utills.ts"></script>
<script src="scripts/fts-pso/fuzzy.ts"></script>
<script src="scripts/fts-pso/FTS.ts"></script>
<script src="scripts/fts-pso/particle.ts"></script>
<script src="scripts/fts-pso/swarm.ts"></script>
<script src="scripts/fts-pso/main.ts"></script>
</body>
</html>

```

ETS.ts

```

import { FuzzyTriangleGate } from './fuzzy';
import {
  FTSTOptions,
  FTSTestOptions,
  KeyValuePair,
} from './types';

export class FTS<TKey, TValue extends number> {
  private _dataset: Array<KeyValuePair<TKey, TValue>>;
  private _minMargin: number;
  private _maxMargin: number;
  private _marginMultiplier: number;
  private _partitionInterval: number;
  private _partitionCount: number;
  private _partitionRef: Array<FuzzyTriangleGate>;
  private _ruleset: {
    [precedent: number]: Set<number>;
  };
};

```

```

public get maxValue() {
    return Math.max(...this._dataset.map(v => v.value));
}
public get minValue() {
    return Math.min(...this._dataset.map(v => v.value));
}
public get lowerBound() {
    return this.minValue - this._minMargin;
}
public get upperBound() {
    return this.maxValue + this._maxMargin;
}
public get partitionCount() {
    return this._partitionCount;
}
public get partitionLength() {
    return this._partitionInterval;
}

constructor(dataset: Array<KeyValuePair<TKey, TValue>>, options?: FTOptions)
{
    this._dataset = dataset;
    this._marginMultiplier = options?.marginMultiplier || 0.1;
    this._minMargin = options?.minMargin || (this.minValue * this._marginMultiplier);
    this._maxMargin = options?.maxMargin || (this.maxValue *
        this._marginMultiplier);
    if (options?.interval) {
        this._partitionInterval = options.interval;
        this._partitionCount = Math.ceil((this.upperBound - this.lowerBound) /
            options.interval);
    } else {
        this._partitionCount = options?.partitionCount || 10;
        this._partitionInterval = (this.upperBound - this.lowerBound) /
            this._partitionCount;
    }
    this._partitionRef = new Array<FuzzyTriangleGate>();
    this._ruleset = {};
    for (let i = 0; i < this._partitionCount; i++) {
        const prevPoint = i === 0 ? 0 : (this.lowerBound + (this._partitionInterval * (i -
            1)));
        const maxPoint = this.lowerBound + (this._partitionInterval * i);
        const nextPoint = this.lowerBound + (this._partitionInterval * (i + 1));
        this._partitionRef.push(new FuzzyTriangleGate(prevPoint, maxPoint,
            nextPoint));
        this._ruleset[i] = new Set<number>();
    }
    const _fuzzySet = new Array<number>();
    for (let i = 1; i < this.partitionCount; i++) {
    }
}

private nearestPartition(value: number) {
    const degrees = this._partitionRef.map(x => x.degree(value));
    const highestDegree = Math.max(...degrees);
    return degrees.findIndex(x => x === highestDegree);
}

```

```

public train() {
  const generatedPattern = this._dataset.map(v => this.nearestPartition(v.value));
  for (let i = 1; i < generatedPattern.length; i++) {
    const precedent = generatedPattern[i - 1];
    const consequent = generatedPattern[i];
    this._ruleset[precedent].add(consequent);
  }
}

public test(options?: FTSTestOptions<TKey, TValue>) {
  const baseData = options?.dataset || this._dataset;
  const predicted = baseData.map(({ key, value }) => {
    const partitionIndex = this.nearestPartition(value);
    const partitionConsequent = [...(this._ruleset[partitionIndex] || new
      Set<number>()).values()];
    const predictedValue = partitionConsequent.length === 0 ?
      (this._partitionRef[partitionIndex].median) :
      ([...partitionConsequent].map(x => this._partitionRef[x].median).reduce((p, c)
        => p + c, 0) / partitionConsequent.length);
    return {
      key,
      value: value,
      predicted: predictedValue,
    };
  });
  return predicted;
}
}

```

App.ts

```

import { FTS } from './FTS';
import { Swarm } from './swarm';
import { KeyValuePair } from './types';
import { averageForecastingErrorRate, meanSquaredError } from './utils';

```

```

async function main() {
  const dataset: Array<KeyValuePair<string, number>> = [
    { key: '2016-Januari', value: 1525945 },
    { key: '2016-Februari', value: 1526174 },
    { key: '2016-Maret', value: 1528229 },
    { key: '2016-April', value: 1527213 },
    { key: '2016-Mei', value: 1530175 },
    { key: '2016-Juni', value: 1529990 },
    { key: '2016-Juli', value: 1531731 },
    { key: '2016-Agustus', value: 1532102 },
    { key: '2016-September', value: 1532000 },
    { key: '2016-Oktober', value: 1534673 },
    { key: '2016-November', value: 1535980 },
    { key: '2016-Desember', value: 1555980 },
    { key: '2017-Januari', value: 1559132 },
    { key: '2017-Februari', value: 1562540 },
    { key: '2017-Maret', value: 1567909 },
    { key: '2017-April', value: 1570187 },
    { key: '2017-Mei', value: 1585009 },
    { key: '2017-Juni', value: 1593840 },
    { key: '2017-Juli', value: 1594601 },
  ];
}

```

```

    { key: '2017-Agustus', value: 1590321 },
    { key: '2017-September', value: 1585582 },
    { key: '2017-Oktober', value: 1587119 },
    { key: '2017-November', value: 1579000 },
    { key: '2017-Desember', value: 1573898 },
    { key: '2018-Januari', value: 1578689 },
    { key: '2018-Februari', value: 1575221 },
    { key: '2018-Maret', value: 1579991 },
    { key: '2018-April', value: 1574185 },
    { key: '2018-Mei', value: 1577540 },
    { key: '2018-Juni', value: 1580312 },
    { key: '2018-Juli', value: 1583221 },
    { key: '2018-Agustus', value: 1588139 },
    { key: '2018-September', value: 1589003 },
    { key: '2018-Oktober', value: 1587841 },
    { key: '2018-November', value: 1589187 },
    { key: '2018-Desember', value: 1592248 },
    { key: '2019-Januari', value: 1621641 },
    { key: '2019-Februari', value: 1618967 },
    { key: '2019-Maret', value: 1620082 },
    { key: '2019-April', value: 1622219 },
    { key: '2019-Mei', value: 1617298 },
    { key: '2019-Juni', value: 1615658 },
    { key: '2019-Juli', value: 1613935 },
    { key: '2019-Agustus', value: 1611112 },
    { key: '2019-September', value: 1613923 },
    { key: '2019-Oktober', value: 1616256 },
    { key: '2019-November', value: 1616990 },
    { key: '2019-Desember', value: 1619533 },
    { key: '2020-Januari', value: 1623146 },
    { key: '2020-Februari', value: 1628435 },
    { key: '2020-Maret', value: 1626213 },
    { key: '2020-April', value: 1632645 },
    { key: '2020-Mei', value: 1637908 },
    { key: '2020-Juni', value: 1641668 },
    { key: '2020-Juli', value: 1645758 },
    { key: '2020-Agustus', value: 1642932 },
    { key: '2020-September', value: 1659174 },
    { key: '2020-Oktober', value: 1667214 },
    { key: '2020-November', value: 1665546 },
    { key: '2020-Desember', value: 1668164 },
  ];
  const min = Math.min(...dataset.map(v => v.value));
  const max = Math.max(...dataset.map(v => v.value));
  const minBorder = min * 0.1;
  const maxBorder = max * 0.1;
  const engine = new FTS(dataset, {
    minMargin: minBorder,
    maxMargin: maxBorder,
    interval: ((max * 1.1) - (min * 0.9)) / 10,
  });
  engine.train();
  const singleResult = engine.test();
  const forecasted = singleResult.slice(0, singleResult.length - 1).map(v =>
    v.predicted);
  const actual = singleResult.slice(1, singleResult.length).map(v => v.value);
  const mse = meanSquaredError(actual, forecasted);

```

```

const afer = averageForecastingErrorRate(actual, forecasted);
console.table(singleResult);
console.log({ mse, afer });

const swarm = new Swarm<{ mse: number; afer: number; }>({
  spaces: { min: 10, max: 1000 },
  weight: 0.1,
  maxIteration: 10000,
  particleCount: 100,
  swarmConfidence: 2,
  selfConfidence: 2,
  fitnessFunction: (n) => {
    const subFts = new FTS(dataset, {
      minMargin: minBorder,
      maxMargin: maxBorder,
      interval: ((max * 1.1) - (min * 0.9)) / Math.abs(n),
    });
    subFts.train();
    const subResult = subFts.test();
    const subForecasted = subResult.slice(0, subResult.length - 1).map(v =>
      v.predicted);
    const subActual = subResult.slice(1, singleResult.length).map(v => v.value);
    return {
      mse: meanSquaredError(subActual, subForecasted),
      afer: averageForecastingErrorRate(subActual, subForecasted),
    };
  },
  selectorFunction: (results) => {
    let bestIndex = 0;
    for (let i = 1; i < results.length; i++) {
      if (results[i].mse < results[bestIndex].mse || results[i].afer <
        results[bestIndex].afer) {
        bestIndex = i;
      }
    }
    return bestIndex;
  },
  stopCriteria: (n) => n.mse <= 10 || Math.abs(n.afer) <= 0.000001,
});
const lastIteration = swarm.optimize();
console.log('Best interval count:', swarm.bestPosition);
console.log('Iteration passed:', lastIteration);
const optimizedEngine = new FTS(dataset, {
  minMargin: minBorder,
  maxMargin: maxBorder,
  interval: ((max * 1.1) - (min * 0.9)) / swarm.bestPosition,
});
optimizedEngine.train();
const optimizedResult = optimizedEngine.test();
const optimizedForecast = optimizedResult.slice(0, optimizedResult.length -
  1).map(v => v.predicted);
const optimizedActual = optimizedResult.slice(1, optimizedResult.length).map(v =>
  v.value);
console.table(optimizedResult);
console.log({
  mse: meanSquaredError(optimizedActual, optimizedForecast),
  afer: averageForecastingErrorRate(optimizedActual, optimizedForecast),
});

```



```

    });
}

main().catch(e => {
  console.trace(e);
});

```

Main.ts

```

import { FuzzyTriangleGate } from './fuzzy';
import {
  FTSTOptions,
  FTSTestOptions,
  KeyValuePair,
} from './types';

export class FTS<TKey, TValue extends number> {
  private _dataset: Array<KeyValuePair<TKey, TValue>>;
  private _minMargin: number;
  private _maxMargin: number;
  private _marginMultiplier: number;
  private _partitionInterval: number;
  private _partitionCount: number;
  private _partitionRef: Array<FuzzyTriangleGate>;
  private _ruleset: {
    [precedent: number]: Set<number>;
  };

  public get maxValue() {
    return Math.max(...this._dataset.map(v => v.value));
  }
  public get minValue() {
    return Math.min(...this._dataset.map(v => v.value));
  }
  public get lowerBound() {
    return this.minValue - this._minMargin;
  }
  public get upperBound() {
    return this.maxValue + this._maxMargin;
  }
  public get partitionCount() {
    return this._partitionCount;
  }
  public get partitionLength() {
    return this._partitionInterval;
  }

  constructor(dataset: Array<KeyValuePair<TKey, TValue>>, options?: FTSTOptions)
  {
    this._dataset = dataset;
    this._marginMultiplier = options?.marginMultiplier || 0.1;
    this._minMargin = options?.minMargin || (this.minValue * this._marginMultiplier);
    this._maxMargin = options?.maxMargin || (this.maxValue *
      this._marginMultiplier);
    if (options?.interval) {
      this._partitionInterval = options.interval;
      this._partitionCount = Math.ceil((this.upperBound - this.lowerBound) /

```

```

        options.interval);
    } else {
        this._partitionCount = options?.partitionCount || 10;
        this._partitionInterval = (this.upperBound - this.lowerBound) /
            this._partitionCount;
    }
    this._partitionRef = new Array<FuzzyTriangleGate>();
    this._ruleset = {};
    for (let i = 0; i < this._partitionCount; i++) {
        const prevPoint = i === 0 ? 0 : (this.lowerBound + (this._partitionInterval * (i -
            1)));
        const maxPoint = this.lowerBound + (this._partitionInterval * i);
        const nextPoint = this.lowerBound + (this._partitionInterval * (i + 1));
        this._partitionRef.push(new FuzzyTriangleGate(prevPoint, maxPoint,
            nextPoint));
        this._ruleset[i] = new Set<number>();
    }
    const _fuzzySet = new Array<number>();
    for (let i = 1; i < this.partitionCount; i++) {

    }
}

private nearestPartition(value: number) {
    const degrees = this._partitionRef.map(x => x.degree(value));
    const highestDegree = Math.max(...degrees);
    return degrees.findIndex(x => x === highestDegree);
}

public train() {
    const generatedPattern = this._dataset.map(v => this.nearestPartition(v.value));
    for (let i = 1; i < generatedPattern.length; i++) {
        const precedent = generatedPattern[i - 1];
        const consequent = generatedPattern[i];
        this._ruleset[precedent].add(consequent);
    }
}

public test(options?: FTSTestOptions<TKey, TValue>) {
    const baseData = options?.dataset || this._dataset;
    const predicted = baseData.map(({ key, value }) => {
        const partitionIndex = this.nearestPartition(value);
        const partitionConsequent = [...(this._ruleset[partitionIndex] || new
            Set<number>()).values()];
        const predictedValue = partitionConsequent.length === 0 ?
            (this._partitionRef[partitionIndex].median) :
            ([...partitionConsequent].map(x => this._partitionRef[x].median).reduce((p, c)
                => p + c, 0) / partitionConsequent.length);
        return {
            key,
            value: value,
            predicted: predictedValue,
        };
    });
    return predicted;
}
}

```

```

import { FTS } from "./FTS";
import { Swarm } from "./swarm";
import { averageForecastingErrorRate, meanSquaredError } from "./utils";

let strategy: 'client' | 'server' = 'client';
const parsedDataset = new Array<{ key: string; value: number; }>();

const stageControlRef = document.getElementById('stageControl') as
    HTMLDivElement;
const stageClientRef = document.getElementById('stageClient') as
    HTMLInputElement;
const stageServerRef = document.getElementById('stageServer') as
    HTMLInputElement;
const retryRef = document.getElementById('ctaRetry') as HTMLButtonElement;
const configFormRef = document.getElementById('configForm') as
    HTMLFormElement;
const resultRef = document.getElementById('resultGraph') as HTMLDivElement;

const dataSourceInputRef = document.querySelector('input[name=dataset]') as
    HTMLInputElement;
const ftsMinMarginInputRef = document.querySelector('input[name=ftsMinMargin]')
    as HTMLInputElement;
const ftsMaxMarginInputRef = document.querySelector('input[name=ftsMaxMargin]')
    as HTMLInputElement;
const ftsIntervallInputRef = document.querySelector('input[name=ftsInterval]') as
    HTMLInputElement;

const psoSpaceMinInputRef = document.querySelector('input[name=psoSpaceMin]') as
    HTMLInputElement;
const psoSpaceMaxInputRef = document.querySelector('input[name=psoSpaceMax]')
    as HTMLInputElement;
const psoWeightInputRef = document.querySelector('input[name=psoWeight]') as
    HTMLInputElement;
const psoMaxIterationInputRef =
    document.querySelector('input[name=psoMaxIteration]') as
    HTMLInputElement;
const psoParticleCountInputRef =
    document.querySelector('input[name=psoParticleCount]') as
    HTMLInputElement;
const psoConfidenceInputRef = document.querySelector('input[name=psoConfidence]')
    as HTMLInputElement;
const psoSelfConfidenceInputRef =
    document.querySelector('input[name=psoSelfConfidence]') as
    HTMLInputElement;
const psoStopMseInputRef = document.querySelector('input[name=psoStopMse]') as
    HTMLInputElement;
const psoStopAferInputRef = document.querySelector('input[name=psoStopAfer]') as
    HTMLInputElement;

const ctaExecuteButtonRef = document.getElementById('ctaExecute') as
    HTMLButtonElement;
const ctaFakeLoadingButtonRef = document.getElementById('ctaFakeLoading') as
    HTMLButtonElement;

const ftsChartRef = document.getElementById('ftsChart') as HTMLCanvasElement;
const ftsPsoChartRef = document.getElementById('ftsPsoChart') as
    HTMLCanvasElement;

```

```

function formRequirementCheck() {
  if (!configFormRef.checkValidity()) {
    ctaExecuteButtonRef.disabled = true;
    ctaExecuteButtonRef.classList.add('disabled');
  } else {
    ctaExecuteButtonRef.disabled = false;
    ctaExecuteButtonRef.classList.remove('disabled');
  }
}

stageClientRef.checked = true;
stageClientRef.addEventListener('click', () => {
  strategy = 'client';
  stageClientRef.checked = true;
});
stageServerRef.addEventListener('click', () => {
  strategy = 'server';
  stageServerRef.checked = true;
});
ftsMinMarginInputRef.addEventListener('input', () => {
  ftsMaxMarginInputRef.min = ftsMinMarginInputRef.value;
});
ftsMaxMarginInputRef.addEventListener('input', () => {
  ftsMinMarginInputRef.max = ftsMaxMarginInputRef.value;
});
psoSpaceMinInputRef.addEventListener('input', () => {
  psoSpaceMaxInputRef.min = psoSpaceMinInputRef.value;
});
psoSpaceMaxInputRef.addEventListener('input', () => {
  psoSpaceMinInputRef.max = psoSpaceMaxInputRef.value;
});
dataSourceInputRef.addEventListener('input', () => {
  try {
    const files = dataSourceInputRef.files;
    if (!files) {
      return;
    }
    let target = [...files].find(current => current.type === 'text/csv');
    if (!target) {
      throw new TypeError('Only accepting csv file as dataset source.');
```

```

    if (keyIndex < 0) {
      throw new EvalError("'Key" header is not exists.');
```

```

    }
    const valueIndex = splitHeader.findIndex(head => head.toLowerCase() ===
      'value');
```

```

    if (valueIndex < 0) {
      throw new EvalError("'Value" header is not exists.');
```

```

    }
    const rawDataset = new Array<{ key: string; value: string; }>();
    for (const line of pairs) {
      const splitted = line.split(',');
      rawDataset.push({
        key: splitted[keyIndex],
        value: splitted[valueIndex],
      });
    }
    parsedDataset.length = 0;
    rawDataset.forEach((rawPair, index) => {
      if (!rawPair.key) {
        throw new EvalError(`Key at line ${index + 1} were empty.`);
      }
      const tryCast = Number(rawPair.value);
      if (isNaN(tryCast)) {
        throw new EvalError(`Value for key "${rawPair.key}" at line ${index +
          1} was not a number.`);
      }
      parsedDataset.push({
        key: rawPair.key,
        value: tryCast,
      });
    });
    const min = Math.min(...parsedDataset.map(v => v.value));
    const max = Math.max(...parsedDataset.map(v => v.value));
    ftsMinMarginInputRef.valueAsNumber = Number((min * 0.1).toFixed(4));
    ftsMaxMarginInputRef.valueAsNumber = Number((max * 0.1).toFixed(4));
    ftsIntervalInputRef.valueAsNumber = Number((((max * 1.1) - (min * 0.9)) /
      10).toFixed(4));
  } catch (e) {
    alert(e.toString());
    dataSourceInputRef.value = "";
  }
});
reader.readAsText(target);
} catch (e) {
  alert(e.toString());
  dataSourceInputRef.value = "";
}
});
ctaExecuteButtonRef.addEventListener('click', async () => {
  const valid = configFormRef.reportValidity();
  if (valid) {
    ctaExecuteButtonRef.classList.add('d-none');
    ctaFakeLoadingButtonRef.classList.remove('d-none');
    const config = {
      fts: {
        minMargin: ftsMinMarginInputRef.valueAsNumber,
        maxMargin: ftsMaxMarginInputRef.valueAsNumber,

```

```

        interval: ftsIntervalInputRef.valueAsNumber,
    },
    pso: {
        spaces: {
            min: psoSpaceMinInputRef.valueAsNumber,
            max: psoSpaceMaxInputRef.valueAsNumber,
        },
        weight: psoWeightInputRef.valueAsNumber,
        maxIteration: psoMaxIterationInputRef.valueAsNumber,
        particleCount: psoParticleCountInputRef.valueAsNumber,
        swarmConfidence: psoConfidenceInputRef.valueAsNumber,
        selfConfidence: psoSelfConfidenceInputRef.valueAsNumber,
        stopCriteria: {
            mse: psoStopMseInputRef.valueAsNumber,
            afer: psoStopAferInputRef.valueAsNumber,
        },
    },
};

try {
    const fts = {
        result: new Array<{ key: string; value: number; predicted: number; }>(),
        mse: 0,
        afer: 0,
    };
    const ftsPso = {
        result: new Array<{ key: string; value: number; predicted: number; }>(),
        mse: 0,
        afer: 0,
        lastIteration: 0,
        best: 0,
    };
    switch (strategy) {
        case 'client': {
            const min = Math.min(...parsedDataset.map(v => v.value));
            const max = Math.max(...parsedDataset.map(v => v.value));
            const engine = new FTS(parsedDataset, {
                minMargin: config.fts.minMargin,
                maxMargin: config.fts.maxMargin,
                interval: config.fts.interval,
            });
            engine.train();
            fts.result = engine.test();
            const forecasted = fts.result.slice(0, fts.result.length - 1).map(v =>
                v.predicted);
            const actual = fts.result.slice(1, fts.result.length).map(v => v.value);
            fts.mse = meanSquaredError(actual, forecasted);
            fts.afer = averageForecastingErrorRate(actual, forecasted);

            const swarm = new Swarm<{ mse: number; afer: number }>({
                spaces: config.pso.spaces,
                weight: config.pso.weight,
                maxIteration: config.pso.maxIteration,
                particleCount: config.pso.particleCount,
                swarmConfidence: config.pso.swarmConfidence,
                selfConfidence: config.pso.selfConfidence,
                fitnessFunction: (n) => {

```

```

const subFts = new FTS(parsedDataset, {
  minMargin: config.fts.minMargin,
  maxMargin: config.fts.maxMargin,
  interval: ((max * 1.1) - (min * 0.9)) / Math.abs(n),
});
subFts.train();
const subResult = subFts.test();
const subForecasted = subResult.slice(0, subResult.length - 1).map(v
=> v.predicted);
const subActual = subResult.slice(1, subResult.length).map(v =>
v.value);
return {
  mse: meanSquaredError(subActual, subForecasted),
  afer: averageForecastingErrorRate(subActual, subForecasted),
};
},
selectorFunction: (results) => {
  let bestIndex = 0;
  for (let i = 1; i < results.length; i++) {
    if (results[i].mse < results[bestIndex].mse || results[i].afer <
results[bestIndex].afer) {
      bestIndex = i;
    }
  }
  return bestIndex;
},
stopCriteria: (n) => n.mse <= config.pso.stopCriteria.mse ||
  Math.abs(n.afer) <= config.pso.stopCriteria.afer,
});
ftsPso.lastIteration = await new Promise(resolve => {
  setTimeout(() => resolve(swarm.optimize()), 1);
});
const optimizedEngine = new FTS(parsedDataset, {
  minMargin: config.fts.minMargin,
  maxMargin: config.fts.maxMargin,
  interval: ((max * 1.1) - (min * 0.9)) / swarm.bestPosition,
});
optimizedEngine.train();
const optimizedResult = optimizedEngine.test();
const optimizedForecast = optimizedResult.slice(0, optimizedResult.length
- 1).map(v => v.predicted);
const optimizedActual = optimizedResult.slice(1,
  optimizedResult.length).map(v => v.value);
ftsPso.result = optimizedResult;
ftsPso.mse = meanSquaredError(optimizedActual, optimizedForecast);
ftsPso.afer = averageForecastingErrorRate(optimizedActual,
  optimizedForecast);
ftsPso.best = swarm.bestPosition;
break;
}
case 'server': {
  break;
}
}
}

```

```

const ftsChart = ftsChartRef['chart'];
ftsChart.options.title.text = `Fuzzy Time Series Result: (interval = 10)`;

```

```

ftsChart.data.labels = fts.result.map(v => v.key);
ftsChart.data.datasets = [
  { label: 'Data', borderColor: '#4e73df', data: fts.result.map(v => v.value) },
  { label: 'Prediction', borderColor: '#5bbf21', data: fts.result.map(v =>
    v.predicted) },
];
ftsChart.update();
const ftsPsoChart = ftsPsoChartRef['chart'];
ftsPsoChart.options.title.text = `Fuzzy Time Series Result Optimized using
  Particle Swarm Optimization: (interval = ${ftsPso.best})`;
ftsPsoChart.data.labels = ftsPso.result.map(v => v.key);
ftsPsoChart.data.datasets = [
  { label: 'Data', borderColor: '#4e73df', data: ftsPso.result.map(v => v.value) },
  { label: 'Prediction', borderColor: '#5bbf21', data: ftsPso.result.map(v =>
    v.predicted) },
];
ftsPsoChart.update();

configFormRef.classList.add('d-none');
resultRef.classList.remove('d-none');
retryRef.classList.remove('d-none');
ctaFakeLoadingButtonRef.classList.add('d-none');
ctaExecuteButtonRef.classList.remove('d-none');
} catch (e) {
  ctaFakeLoadingButtonRef.classList.add('d-none');
  ctaExecuteButtonRef.classList.remove('d-none');
  alert(e.toString());
}
}
});
retryRef.addEventListener('click', () => {
  configFormRef.classList.remove('d-none');
  resultRef.classList.add('d-none');
  retryRef.classList.add('d-none');
  ctaFakeLoadingButtonRef.classList.add('d-none');
  ctaExecuteButtonRef.classList.remove('d-none');
});

configFormRef.classList.remove('d-none');

```

swarm.ts

```

import { Particle } from "./particle";
import { FitFuncDelegate, SelectorFuncDelegate, StopCriteria } from "./utils";

```

```

interface SwarmOption<TFit = any> {
  spaces: {
    min: number;
    max: number;
  };
  weight?: number;
  maxIteration: number;
  particleCount: number;
  swarmConfidence?: number;
  selfConfidence?: number;
  fitnessFunction: FitFuncDelegate<TFit>;
  selectorFunction: SelectorFuncDelegate<TFit>;
}

```



```

    stopCriteria: StopCriteria<TFit>;
}

export class Swarm<TFit = any> {
  private _maxIteration: number;
  private _particles: Particle<TFit>[];
  private _fitFn: FitFuncDelegate<TFit>;
  private _selectFn: SelectorFuncDelegate<TFit>;
  private _stopCriteria: StopCriteria<TFit>;
  private _bestFit: TFit;
  private _bestPosition: number;
  private _swarmConfidence: number;
  private _w: number;

  public get bestPosition() {
    return this._bestPosition;
  }

  public get bestFit() {
    return this._bestFit;
  }

  constructor(options: SwarmOption<TFit>) {
    if (options.weight && (options.weight >= 1 || options.weight <= 0)) {
      console.warn('Weight out of boundaries, generating random weight...');
    }
    this._w = options.weight && options.weight > 0 && options.weight < 1 ?
      options.weight : Math.random();
    this._maxIteration = options.maxIteration;
    this._fitFn = options.fitnessFunction;
    this._selectFn = options.selectorFunction;
    this._stopCriteria = options.stopCriteria;
    this._swarmConfidence = options.swarmConfidence || Math.random();
    this._particles = new Array<null>(options.particleCount).fill(null).map(() => new
      Particle<TFit>({
        min: options.spaces.min,
        max: options.spaces.max,
        fitFn: this._fitFn,
        selectorFn: this._selectFn,
        selfConfidence: options.selfConfidence,
        weight: this._w,
      }));
    const bestParticle = this._selectFn.call(this, this._particles.map(v =>
      v.bestFitResult));
    this._bestFit = this._particles[bestParticle].bestFitResult;
    this._bestPosition = this._particles[bestParticle].bestPosition;
  }

  public optimize() {
    let stop = this._stopCriteria.call(this, this._bestFit);
    let currentIteration = 0;
    while (!stop && currentIteration < this._maxIteration) {
      for (const particle of this._particles) {
        particle.update(this._swarmConfidence, this._bestPosition);
      }
      const bestParticle = this._selectFn.call(this, this._particles.map(v =>
        v.bestFitResult));
    }
  }
}

```

```

    this._bestFit = this._particles[bestParticle].bestFitResult;
    this._bestPosition = this._particles[bestParticle].bestPosition;
    stop = this._stopCriteria.call(this, this._bestFit);
    currentIteration += 1;
  }
  return currentIteration;
}
}

```

Utils.ts

```

function baseLookup(interval: number) {
  return interval <= 0.1 ? 0.1 : (10 ** Math.ceil(Math.log10(interval) - 1));
}

export function averageInterval(dataset: number[]) {
  const range = dataset.map((v, i, a) => i === 0 ? null : Math.abs(v - a[i - 1])).filter(v
    => v !== null) as number[];
  const rangeAverage = range.reduce((p, c) => p + c, 0) / range.length;
  const halfAverage = rangeAverage / 2;
  return baseLookup(halfAverage);
}

export function meanSquaredError(actual: number[], forecast: number[]) {
  if (actual.length !== forecast.length) {
    throw new RangeError('Actual data and forecast data has different length');
  }
  return actual.map((v, i) => (v - forecast[i]) ** 2).reduce((p, c) => p + c, 0) /
    actual.length;
}

export function averageForecastingErrorRate(actual: number[], forecast: number[]) {
  if (actual.length !== forecast.length) {
    throw new RangeError('Actual data and forecast data has different length');
  }
  return actual.map((v, i) => Math.abs(v - forecast[i]) / v).reduce((p, c) => p + c, 0) /
    actual.length;
}

export type FitFuncDelegate<TResult = any> = (position: number) => TResult;
export type SelectorFuncDelegate<TResult = any> = (fitResult: TResult[]) => number;
export type StopCriteria<TResult = any> = (bestResult: TResult) => boolean;

```

CSS

```

bs-icon {
  --bs-icon-size: .75rem;
  display: flex;
  flex-shrink: 0;
  justify-content: center;
  align-items: center;
  font-size: var(--bs-icon-size);
  width: calc(var(--bs-icon-size) * 2);
  height: calc(var(--bs-icon-size) * 2);
  color: var(--bs-primary);
}

.bs-icon-xs {

```

```
--bs-icon-size: 1rem;
width: calc(var(--bs-icon-size) * 1.5);
height: calc(var(--bs-icon-size) *
1.5);
}

.bs-icon-sm {
  --bs-icon-size: 1rem;
}

.bs-icon-md {
  --bs-icon-size: 1.5rem;
}

.bs-icon-lg {
  --bs-icon-size: 2rem;
}

.bs-icon-xl {
  --bs-icon-size: 2.5rem;
}

.bs-icon.bs-icon-primary {
  color: var(--bs-white);
  background: var(--bs-
primary);
}

.bs-icon.bs-icon-primary-light
{ color: var(--bs-primary);
background: rgba(var(--bs-primary-rgb), .2);
}

.bs-icon.bs-icon-semi-white
{ color: var(--bs-primary);
background: rgba(255, 255, 255, .5);
}

.bs-icon.bs-icon-rounded
{ border-radius: .5rem;
}

.bs-icon.bs-icon-circle
{ border-radius: 50%;
}
```