Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) p-ISSN: 2355-7699
Vol. 3, No. 4, Desember 2016, hlm. 285-291 e-ISSN: 2528-6579

PREDIKSI NILAI TUKAR RUPIAH TERHADAP US DOLLAR MENGGUNAKAN METODE GENETIC PROGRAMMING

Daneswara Jauhari[1], Anang Hanafi[2], M. Fahrul Alam Y.[3], Arrofi Reza Satria[4], Luqman Hakim H.[5], Imam Cholissodin[6]

[1,2,3,4,5,6]Fakultas Ilmu Komputer Universitas Brawijaya
Email: [1]daneswarajauhari@gmail.com, [2]ananghanafi13@gmail.com, [3]fahrul.school@gmail.com, [4]arrofirezasatria@gmail.com, [5]hakim.harum@gmail.com, [6]imamcs@ub.ac.id

(Naskah masuk: 1 Oktober 2016, diterima untuk diterbitkan: 26 Desember 2016)

Abstrak

Nilai tukar mata uang mempunyai pengaruh yang sangat luas dalam perekonomian suatu negara, baik dalam negeri ataupun internasional. Pentingnya mengetahui pola nilai tukar *IDR* terhadap *USD* bisa membantu pertumbuhan ekonomi dikarenakan perdagangan luar negeri menggunakan mata uang negara yang berbeda. Maka dari itu, diperlukan suatu aplikasi yang dapat memprediksi nilai tukar Rupiah terhadap *US Dollar* di masa yang akan datang. Pada penelitian ini penulis menggunakan metode *genetic programming(GP)*, yang dapat menghasilkan solusi (*chromosome*) optimum, yang didapat dari evaluasi nilai tukar yang lalu, sehingga solusi ini digunakan sebagai pendekatan atau prediksi terhadap kurs nilai tukar mata uang Rupiah di masa yang akan datang. Solusi ini dibentuk dari kombinasi dari himpunan terminal (*set of terminals*) dan himpunan fungsi (*set of function*) yang dibangkitkan secara *random*. Setelah dilakukan pengujian dengan jumlah *popsize* dan iterasi yang berbeda, didapatkan bahwa Algoritma *GP* dapat melakukan prediksi nilai tukar Rupiah terhadap mata uang *US Dollar* dengan sangat baik, dilihat dari nilai *Mean Absolute Percentage Error (MAPE)* yang dihasilkan sebesar 0,08%. Penelitian ini bisa dikembangkan lebih baik dengan menambahkan parameter terminal dan parameter operasi sehingga bisa menambah variasi hasil perhitungannya.

Kata kunci: *prediksi, nilai tukar mata uang, genetic programming, MAPE*.

Abstract

Exchange currency rate has a wide influence in the economy of a country, both domestically or internationally. The importance of knowing the pattern of exchange rate against the IDR to USD could help the economic growth due to foreign trade involves the use of currencies of different countries. Therefore, we need an application that can predict the value of IDR against the USD in the future. In this research, the authors use genetic programming (GP) method which produces solutions (chromosome) that obtained from the evaluation of exchange rate and then this solution used as an approximation or prediction of currency exchange rate in the future. These solutions formed from the combination of the set terminal and the set of function that generated randomly. After testing by the number popsize and different iterations, it was found that the GP algorithm can predict the value of the rupiah against the US Dollar with a very good, judging from the value of Mean Absolute Percentage Error (MAPE) generated by 0.08%. This research can be developed even better by adding terminal parameters and operating parameters so they can add variation calculation results.

Keywords: *prediction, exchange currency rate, genetic programming, MAPE*.

1. PENDAHULUAN

Dua tahun yang lalu nilai tukar Rupiah terhadap *US Dollar* tidak stabil. Sebab-sebab yang mempengaruhi tidak stabil nya nilai tukar uang yaitu, tingkat inflasi pada negara, suku bunga pada negara, neraca perdagangan antar negara, jumlah permintaan barang ekspor, jumlah impor barang, hutang publik negara, kestabilan politik dan ekonomi negara (Anonymouse, 2015).

Ketidakstabilan nilai tukar uang, dapat membuat investor mengurungkan keinginannya untuk melakukan investasi, hal ini akan menyebabkan kemunduran pembangunan di Indonesia karena selama ini peran dari investor asing sangat besar dalam pertumbuhan ekonomi (Cecilya dkk., 2009). Hal ini akan semakin membuat nilai tukar *US Dollar* terhadap Rupiah semakin tinggi, karena sebagian besar transaksi internasional di Indonesia menggunakan *US Dollar*. Sementara itu nilai tukar rupiah akan menentukan indeks harga saham gabungan di Bursa Efek Jakarta (Kho dkk., 2011).

Oleh karena itu perlu adanya suatu aplikasi yang dapat memprediksi nilai tukar Rupiah terhadap *US Dollar* di masa yang akan datang.

Data historis akan dipelajari polanya untuk membuat sebuah prediksi. Metode yang dapat digunakan untuk melakukan prediksi ada banyak, seiring perkembangan teknologi dan informasi, metode untuk melakukan prediksi juga akan semakin berkembang.

Untuk mendapatkan hasil yang akurat dan stabil diperlukan sebuah metode yang tepat. Metode yang ada pada *Artifical Neural Networks*, dapat digunakan untuk mempelajari pola dari data historis yang didapatkan, *Artifical Neural Networks* meniru proses pembelajaran manusia, sehingga tepat untuk digunakan pada kasus sistem *non-linear*.

Penelitian mengenai prediksi dengan menggunakan metode yang ada pada *Artifical Neural Networks* pernah dilakukan oleh Jauhari (2016). Dalam penelitian ini di gunakan metode *Backpropagation Artifical Neural Networks*, penelitian tersebut bertujuan untuk memprediksi distribusi air PDAM Kota Malang, dan didapatkan hasil akurasi terbaik sebesar 97,99%.

Akan tetapi data nilai tukar uang merupakan data *non-linear* dengan banyak *noi*se. Hal ini akan mempersulit pengenalan pola yang dilakukan metode yang ada di *Artifical Neural Networks* (Kho dkk., 2011). Peneliti belum menemukan penelitian lain yang menggunakan metode *Genetic Programming*. Metode *Genetic Programming* merupakan satu metode secara matematis dan otomatis mengevolusi program komputer, berdasarkan sebuah permasalahan yang diberikan, metode ini dapat direpresentasikan dalam bentuk struktur pohon/*tree* (Mahmudy, 2016).

Genetic Programming telah diaplikasikan dalam berbagai bidang, *Genetic Programming* juga pernah digunakan juga untuk melakukan prediksi pada *respon time* sebuah layanan website, penelitian tersebut dilakukan oleh Fanjiang (2016), dan didapatkan hasil yang tidak kalah dengan algoritma pada *Artifical Neural Networks* yang biasa digunakan untuk memprediksi. Pada penelitian tersebut hasil yang didapatkan memiliki nilai rata-rata terbaik dibandingkan metode lain, artinya metode *Genetic Programming* dalam memprediksi mendapatkan hasil prediksi yang baik dan stabil.

Oleh karena itu untuk memprediksi nilai tukar Rupiah terhadap *US Dollar* pada penelitian ini diusulkan menggunakan metode *Genetic Programming*, yang diharapkan mendapatkan hasil prediksi yang baik dan stabil.

2. DASAR TEORI

2.1 Penjelasan Dataset

Data yang digunakan dalam penelitian adalah data nilai tukar mata uang Rupiah terhadap *Dollar* dari website fixer.io dari tahun 2000 sampai 2017 dengan mengambilnya melalui *api server* fixer.io dengan *JSON*. Dataset ini terdapat 5 kolom dengan variabel parameter X1, X2, X3, X4 dan Y. Tabel 2.1 ini berisi penjelasan tiap variabel pada dataset. Dan sampel dari dataset digambarkan pada Tabel 2.2, yang merupakan dataset yang diambil pada tanggal 29/12/2016 dengan jumlah 5.

Tabel 2.1 Keterangan Variabel

No	Kode Atribut	Keterangan
1	x1	Nilai aktual Rupiah pada H-4
2	x2	Nilai aktual Rupiah pada H-3
3	x3	Nilai aktual Rupiah pada H-2
4	x4	Nilai aktual Rupiah pada H-1
5	y	Nilai aktual pada hari H

Tabel 2.2 Dataset dari Fixer.io

x1	x2	x3	x4	y
13473	13435	13435	13435	13435
13435	13473	13435	13435	13435
13469	13435	13473	13435	13435
13398	13469	13435	13473	13435
13390	13398	13469	13435	13473

2.2 Nilai Tukar

Nilai tukar atau kurs merupakan suatu perbandingan antara nilai mata uang suatu negara dengan negara lain (Ardra, 2016). Selain itu, nilai Tukar adalah pertukaran antara dua mata uang yang berbeda, yaitu merupakan perbandingan nilai antara kedua mata uang tersebut (Triyono, 2008).

2.3 *Genetic Programming*

Genetic programming (GP) merupakan model dari pemrograman yang menggunakan ide-ide (dan beberapa terminologi) dari evolusi biologis untuk menangani masalah yang kompleks. Dari sejumlah program yang mungkin (biasanya fungsi program kecil dalam aplikasi yang lebih besar), program yang paling efektif akan bertahan dan bersaing atau *cross-breed* dengan program lain untuk terus lebih dekat dengan pendekatan solusi yang dibutuhkan.

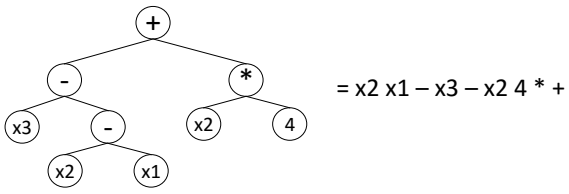
GP merupakan suatu model pendekatan yang terlihat paling sesuai dengan masalah-masalah yang memiliki variable-variable difluktuasi dalam jumlah besar. Berbeda dengan *Genetic Algorithm*, *Genetic Programming* menghasilkan program komputer yang direpresentasikan dalam struktur pohon (*tree*). (Mahmudy dkk., 2016)

GP bisa menggunakan proses rekombinasi dan mutasi dengan probabilitas tertentu, tapi tidak dapat keduanya. Rekombinasi dilakukan dengan cara saling menukar ranting (*sub tree*), sedangkan mutasi dilakukan dengan mungubah pohon secara acak. Sebuah bagian yang sulit dari penggunaan *GP* adalah menentukan fungsi fitness, sejauh mana program membantu untuk sampai ke tujuan yang diinginkan. Berbeda dengan pendekatan yang ada, pendekatan lain menggunakan teknik peramalan *time series* yang konvensional yang terbilang kurang akurat jika dibandingakan dengan *GP*.

Dua parameter yang merepresentasikan solusi dari *Genetic Programming* sebagai berikut:
- Penentuan himpunan terminal (*set of terminals*)
- Penentuan himpunan fungsi (*set of functions*)
(Mahmudy dkk., 2016)

2.3.1 Representasi Kromosom

Setiap kromosom yang mewakili calon solusi untuk memecahkan masalah merupakan struktur pohon yang cocok untuk operasi *GP*. Jika digunakan untuk menemukan program yang valid, maka setiap kromosom berevolusi dan dioperasikan oleh *GP* yang akan menjadi perwakilan program dalam bentuk pohon *(tree)*. Perhatikan representasi kromosom pada Gambar 2.1.



Gambar 2.1 Representasi Kromosom

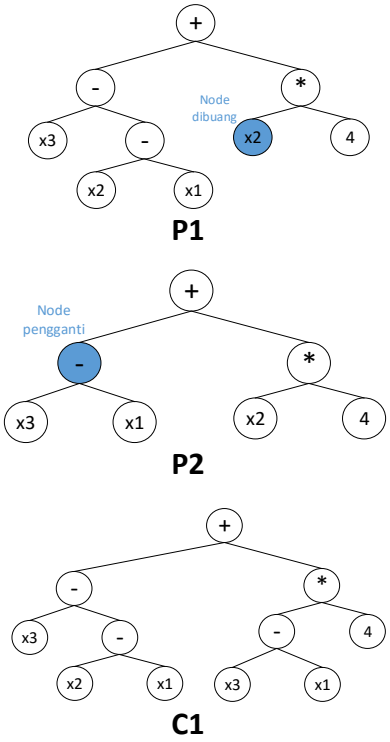
2.3.2 Inisialisasi dan Evaluasi

Sebelum menjalankan GP, operator harus menentukan elemen-elemen apa yang seharusnya ada dalam fungsi dan terminal set. Setelah itu GP mengambil beberapa *sample* atau *parents* secara *random* untuk diinisialisasi dan dievaluasi dengan menghitung nilai fitness dari individu tersebut.

Evaluasi individu dijalankan dengan menyusun fungsi non-linear dari binary tree. Sedangkan nilai fitness dihitung dari total selisih antara output fungsi *non-linear* dengan nilai y pada tabel.

2.3.3 Crossover

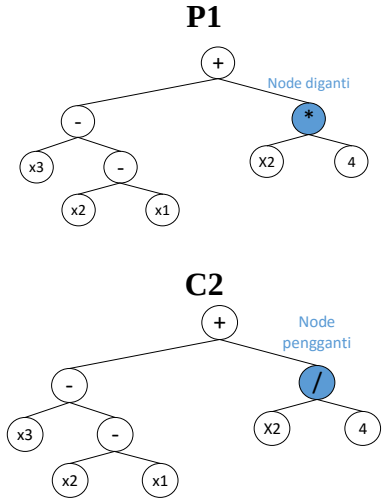
Operator menukar satu bagian (*gen*) dari satu induk kromosom dengan induk lainnya yang akan dicrossover. Setelah dua operasi seleksi selesai, kedua induk akan menghasilkan 2 *childs* yang akan dipilih dengan syarat memiliki kemampuan bertahan hidup yang *relative* lebih baik (bisa kita lihat dari nilai fitnessnya) dan akan diproses/digunakan untuk generasi berikutnya. Perhatikan proses crossover pada Gambar 2.2.



Gambar 2.2 Proses Crossover

2.3.4 Mutasi

Secara acak *GP* melakukan perubahan node dari sebuah kromosom menjadi node yang berbeda. Mutasi bisa meningkatkan keragaman evolusi pada suatu populasi dan meningkatkan eksplorasi *GP* dari ruang pencarian. Dimana, probabilitas mutasi biasanya ditetapkan jauh lebih rendah dari probabilitas crossover. Perhatikan proses mutasi pada Gambar 2.3.



Gambar 2.3 Proses Mutasi

2.3.5 Seleksi

Operator melakukan seleksi dengan mengambil kromosom dengan nilai fitness yang tertinggi, sehingga bisa bertahan hidup di generasi berikutnya. Proses dalam seleksi pada *GP* ini sama dengan *GAs*

yang biasanya menggunakan *binary tournament* dan *roulette-wheel*. Kromosom dengan fitness yang tertinggi, memiliki kesempatan yang besar untuk menjadi parent pada generasi berikutnya. Pada penelitian ini digunakan jenis seleksi *elitism* yang mengambil kromosom terbaik dari populasi.

2.4 Perhitungan Nilai Evaluasi

Untuk melakukan perhitungan evaluasi pada penelitian ini menggunakan persamaan *Mean Absolute Percentage Error (MAPE)* yang dihitung menggunakan kesalahan absolut pada tiap periode dibagi nilai observasi yang nyata. Setelah itu, dihitung rata-rata kesalahan persentase absolut. Pendekatan ini berguna ketika ukuran atau besar variabel ramalan itu penting dalam mengevaluasi ketepatan ramalan.

MAPE mencerminkan seberapa besar kesalahan dalam memprediksi yang dibandingkan dengan nilai nyata (Jauhari dkk., 2016). Rumus yang digunakan untuk menghitung *MAPE* terdapat pada persamaan 2.1 :

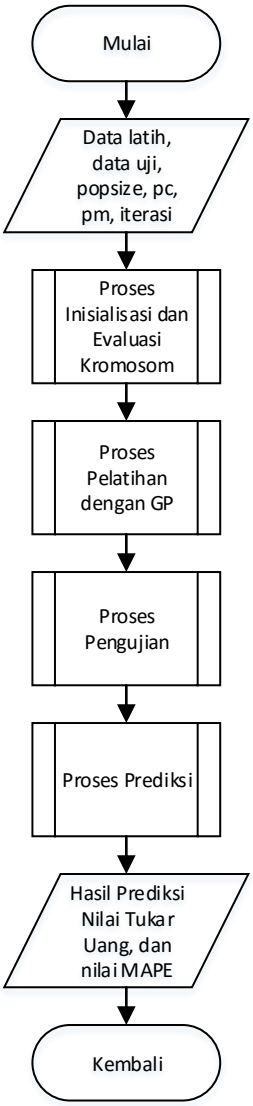
$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - yp_i|}{y_i} \times 100\% \qquad (2.1)$$

3. PERANCANGAN DAN IMPLEMENTASI

Proses dari prediksi nilai tukar uang dengan menggunakan metode *GP* melalui beberapa tahapan. Tahapan yang pertama yaitu melakukan proses inisialisasi dan evaluasi pada kromosom, kemudian tahap dua yaitu melakukan pelatihan dengan metode *GP*, pelatihan ini akan melewati beberapa proses yaitu:

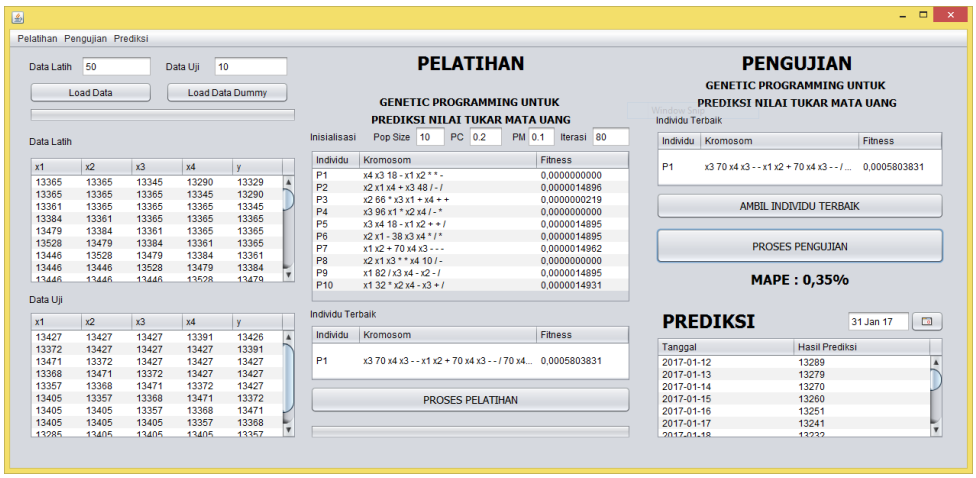
1. Crossover
2. Mutasi
3. Seleksi dengan metode *elitism*
4. Mendapatkan individu terbaik

Setelah dilakukan pelatihan, akan didapatkan individu terbaik, individu terbaik ini pada tahap ketiga digunakan untuk proses pengujian, tahap yang terakhir yaitu tahap 4 merupakan tahap prediksi nilai tukar uang. Tahapan-tahapan yang dilalui dapat digambarkan pada Gambar 3.1.



Gambar 3.1. *Flowchart* proses prediksi nilai tukar uang

Implementasi Antarmuka halaman utama, pada Gambar 3.2 merupakan tampilan halaman utama yang berisi input data, pelatihan, pengujian, dan prediksi.



Gambar 3.2. Antarmuka Halaman Utama

Implementasi Proses Postfix ini merupakan proses yang digunakan untuk membuat kromosom baru dengan cara random. Diawali dengan menentukan terminal dan operasi. Kemudian melakukan random sesuai dengan syarat postfix. Keseluruhan proses tersebut ditunjukkan dari potongan Kode Program 3.1.

```
1  public static String randomPostfix() {
2  String hasil = "";
3  Random rand = new Random();
4  // inisialisasi operator
5  String[] operator = {"+", "-", "*", "/"};
6  // inisialisasi terminal
7  String[] terminal = new String[5];
8  String[] tampungan = {"x1", "x2", "x3",
9  "x4", "0"};
10 tampungan[tampungan.length - 1] =
11 String.valueOf(rand.nextInt(100) + 1);
12
13 ArrayList<Integer> random = new
14 ArrayList<Integer>();
15 for (int r = 0; r < tampungan.length; r++)
16 {
17 random.add(new Integer(r));
18 }
19 Collections.shuffle(random);
20 for (int j = 0; j < tampungan.length; j++)
21 {
22 terminal[j] = tampungan[(int)
23 random.get(j)];
24 }
25
26 // buat postfix random
27 int A = 0;
28 int O = 0;
29 for (int i = 0; i < ((operator.length +
30 terminal.length)); i++) {
31 if (i < 2) {
32 hasil += terminal[A] + " ";
33 A++;
34 } else if (i == (operator.length +
35 terminal.length) - 1) {
36 hasil += operator[rand.nextInt(4)];
37 } else {
38 if (A >= terminal.length) {
39 hasil += operator[rand.nextInt(4)] + " ";
40 O++;
41 } else if (A == (O + 1)) {
42 hasil += terminal[A] + " ";
43 A++;
44 } else {
45 if ((rand.nextInt(2) + 1) % 2 == 1) {
46 hasil += operator[rand.nextInt(4)] + " ";
47 O++;
48 } else {
49 hasil += terminal[A] + " ";
50 A++;
51 }
52 }
53 }
54 }
55 return hasil;
56 }
```

Kode Program 3.1. Proses Random Postfix

Penjelasan dari Kode Program 3.1:

1. Baris 2-12 merupakan proses inisialisasi operator, terminal, dll.
2. Baris 13-24 merupakan proses pembentukan random untuk terminal.
3. Baris 27-28 merupakan proses inisialisasi angka dan operator.
4. Baris 29-30 melakukan perulangan sampai jumlah operator + jumlah terminal
5. Baris 31-33 jika jumlah i masih kurang dari 2 maka isi dengan angka
6. Baris 34-37 jika jumlah i merupakan urutan terakhir
7. Baris 38-41 jika jumlah A lebih besar dari panjang terminal maka isi dengan operator
8. Baris 42-44 jika jumlah O = A + 1 maka isi dengan terminal
9. Baris 45-48 jika nilai random ganjil maka isi operator
10. Baris 49-51 jika nilai random genap maka isi terminal
11. Baris 55 kembalikan nilai hasil

4. PENGUJIAN DAN ANALISIS

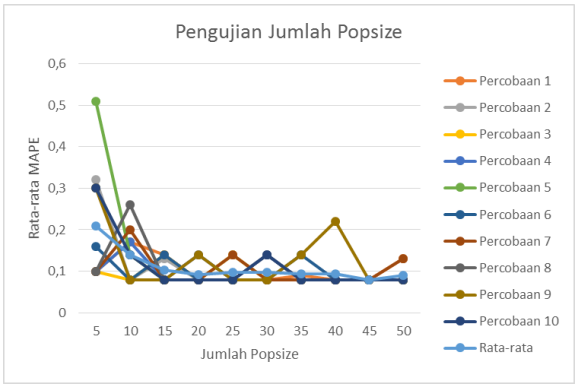
4.1 Pengujian Jumlah Popsize

Pengujian *popsize* digunakan untuk menentukan *popsize* yang optimal untuk mendapatkan nilai *MAPE* yang minimum. Parameter yang digunakan pada pengujian ini adalah jumlah data latih 50, jumlah data uji 10, jumlah *popsize* pada range 5-50, nilai pc 0,2, nilai pm, 0.1, dan jumlah iterasi 100. Pengujian jumlah *popsize* ini dilakukan sebanyak 10 kali. Hasil pengujian dapat dilihat pada Tabel 4.1 berikut ini:

Tabel 4.1 Pengujian Jumlah *Popsize*

Jumlah *Pop Size*	*MAPE* (%) Nilai Percobaan Ke-i										Rerata *MAPE* (%)
	1	2	3	4	5	6	7	8	9	10	
5	0,1	0,32	0,1	0,1	0,51	0,16	0,1	0,1	0,3	0,3	0,209
10	0,17	0,08	0,08	0,17	0,14	0,08	0,2	0,26	0,08	0,14	0,14
15	0,14	0,13	0,14	0,08	0,08	0,14	0,08	0,08	0,08	0,08	0,103
20	0,08	0,08	0,08	0,14	0,08	0,08	0,08	0,08	0,14	0,08	0,092
25	0,08	0,14	0,14	0,08	0,08	0,08	0,14	0,08	0,08	0,08	0,098
30	0,08	0,08	0,08	0,14	0,08	0,08	0,08	0,14	0,08	0,14	0,098
35	0,09	0,08	0,08	0,08	0,08	0,14	0,08	0,08	0,14	0,08	0,093
40	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,22	0,08	0,094
45	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
50	0,13	0,08	0,08	0,08	0,08	0,08	0,13	0,08	0,08	0,08	0,09

Berdasarkan grafik hasil pengujian pada Gambar 4.1, rata-rata nilai *MAPE* terkecil didapatkan pada popsize 45 dengan nilai 0.08, yang artinya semakin banyak jumlah *popsize*, maka akan didapatkan nilai *MAPE* yang minimum.



Gambar 4.1. Hasil Pengujian Jumlah *Popsize*

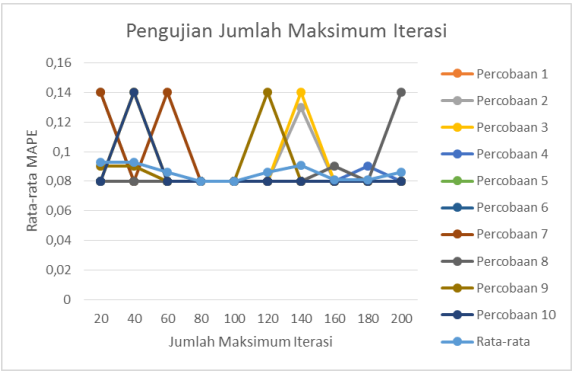
4.2 Pengujian Jumlah Maksimum Iterasi

Pengujian jumlah maksimum iterasi digunakan untuk menentukan jumlah maksimum iterasi yang optimal untuk mendapatkan nilai *MAPE* yang minimum. Tentunya dengan menggunakan jumlah *popsize* yang optimal pada pengujian sebelumnya. Parameter yang digunakan pada pengujian ini adalah jumlah data latih 50, jumlah data uji 10, jumlah *popsize* 45, nilai pc 0,2, nilai pm, 0.1, dan jumlah iterasi pada range 20-200. Pengujian jumlah maksimum iterasi ini dilakukan sebanyak 10 kali. Hasil pengujian dapat dilihat pada Tabel 4.2 berikut ini:

Tabel 4.1 Pengujian Jumlah *Popsize*

Jumlah Maksimum Iterasi	*MAPE* (%) Nilai Percobaan Ke-i										Rerata *MAPE* (%)
	1	2	3	4	5	6	7	8	9	10	
20	0,08	0,14	0,08	0,08	0,08	0,08	0,14	0,08	0,09	0,08	0,093
40	0,08	0,08	0,14	0,08	0,08	0,08	0,08	0,08	0,09	0,14	0,093
60	0,08	0,08	0,08	0,08	0,08	0,08	0,14	0,08	0,08	0,08	0,086
80	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
100	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08
120	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,14	0,08	0,086
140	0,08	0,13	0,14	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,091
160	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,09	0,08	0,08	0,081
180	0,08	0,08	0,08	0,09	0,08	0,08	0,08	0,08	0,08	0,08	0,081
200	0,08	0,08	0,08	0,08	0,08	0,08	0,08	0,14	0,08	0,08	0,086

Berdasarkan grafik hasil pengujian pada Gambar 4.2, rata-rata nilai *MAPE* terkecil didapatkan pada jumlah maksimum iterasi sebesar 80 dan 100 dengan nilai *MAPE* 0,08. Ketika jumlah maksimum iterasi 80 dan 100 sudah didapatkan jumlah maksimum iterasi yang optimal, hal ini dapat simpulkan karena nilai *MAPE* tidak ada yang lebih kecil lagi di jumlah maksimum iterasi setelahnya.



Gambar 4.2. Hasil Pengujian Jumlah Maksimum Iterasi

5. KESIMPULAN DAN SARAN

Berdasarkan hasil uji coba parameter algoritma *Genetic Programming* untuk melakukan prediksi nilai tukar Rupiah terhadap *US Dollar*. Didapatkan bahwa Algoritma *GP* dapat melakukan prediksi nilai tukar Rupiah terhadap *US Dollar* dengan sangat baik, dengan parameter algoritma *GP* yang digunakan terdapat 4 operasi yakni "+", "-", "*", "/", "," dan juga 5 terminal yakni x1, x2, x3, x4, x, yang merupakan data nilai tukar uang Rupiah terhadap *US Dollar*.

Untuk mengukur solusi dari permasalahan prediksi nilai tukar Rupiah terhadap *US Dollar*, pada penelitian ini menggunakan perhitungan nilai *MAPE*. Penghitungan menggunakan parameter terbaik dapat menghasil nilai *MAPE* sebesar 0,08%. Parameter terbaik dengan rata-rata nilai *MAPE* terendah adalah sebagai berikut :

a. Jumlah data latih : 50
b. Jumlah data uji : 10
c. Jumlah popsize : 45
d. Nilai PC : 0.2
e. Nilai PM : 0.1
f. Jumlah Iterasi : 80 atau 100

Penelitian ini dapat dikembangkan dengan menambahkan jumlah parameter terminal, dan juga menambah variasi dari parameter operasi. Penggunaan nilai terminal yang lebih banyak akan mempengaruhi hasil variasi dari perhitungannya, penambahan variasi dari operasi juga akan mengatasi permasalahan ketika kromosom terdapat operasi pembagian. Selain itu bisa dicoba untuk menerapkan algoritma pengembangan dari *GP* yang telah ada saat ini, sehingga dapat mendapatkan hasil yang lebih baik.

6. DAFTAR PUSTAKA

ANONYMOUSE, 2015. Mengapa Mata Uang Dunia dan Nilai Kurs Negara Beda-beda. Tersedia di: http://uangindonesia.com/mengapa-mata-uang-dunia-dan-nilai-kurs-negara-beda/ [Diakses 05 Desember 2016].

ARDRA. 2016. Pengertian Nilai Tukar. Tersedia di: https://ardra.biz/ekonomi/valuta-asing/kurs-valuta-asing/ [Diakses 29 Desember 2016].

CECILYA, ARANTIKA. dkk., 2009. Prediksi Nilai Tukar US Dollar Terhadap Rupiah Menggunakan Algoritma Genetika Dan Elman Recurrent Neural Network, Universitas Telkom.

MAHMUDY, F.M. 2016. Modul Algoritma Evolusi Semester Ganjil 2016-2017, Universitas Brawijaya.

FANJIANG, YONG-YI. dkk. 2016. Search based approach to forecasting QoS attributes of web services using genetic programming. *Information and Software Technology 80 (2016) 158-174*.

JAUHARI, DANESWARA. dkk. 2016. Prediksi Distribusi Air PDAM Menggunakan Metode Jaringan Syaraf Tiruan Backpropagation di PDAM Kota Malang. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), 3(2), 85-89.*

KHO, DICKY KHOSMAN. dkk. 2011. Prediksi Nilai Tukar IDR / USD Menggunakan Artificial Neural Networks Dengan Metode Pembelajaran Ant Colony Optimization, Universitas Telkom.

TRIYONO. 2008. Analisis Perubahan Kurs Rupiah Terhadap Dollar Amerika. Jurnal Ekonomi Pembangunan. Vol.9 No. 2, Desember 2008: 156-167. Universitas Muhammadiyah Surakarta.