

## Penerapan Algoritma Genetika *Multi-objective* NSGA-II Pada Optimasi Portofolio Saham

*Sherly Isnaeni*<sup>1</sup>, *Deni Saepudin*<sup>2</sup>, *Rian Febrian Umbara*<sup>3</sup>

<sup>1,2,3</sup> Prodi S1 Ilmu Komputasi, Fakultas Informatika, Universitas Telkom

Jalan Telekomunikasi No.1, Dayeuh Kolot, Bandung 40257

[sherlyisnaeni@gmail.com](mailto:sherlyisnaeni@gmail.com)<sup>1</sup>, [denisaepudin@telkomuniversity.co.id](mailto:denisaepudin@telkomuniversity.co.id)<sup>2</sup>, [rianum123@gmail.com](mailto:rianum123@gmail.com)<sup>3</sup>

---

### Abstrak

Algoritma Genetika *Multi-objective* NSGA-II merupakan salah satu *evolutionary algorithm* yang populer digunakan pada permasalahan optimisasi *multi-objective*, yaitu dengan mencari solusi yang mendekati Pareto optimal. Pada penelitian ini NSGA-II digunakan untuk menyelesaikan permasalahan optimisasi portofolio yang terdiri dari saham-saham yang tergabung dalam Indeks LQ45.

Beberapa parameter Algoritma Genetika *Multi-objective* NSGA-II digunakan antara lain ukuran populasi, jumlah generasi, probabilitas *crossover* dan probabilitas mutasi. Hasil akhir dari penerapan Algoritma Genetika *Multi-objective* NSGA-II ini berupa grafik *efficient frontier*, yaitu kumpulan dari pilihan terbaik bagi investor yang mampu menawarkan tingkat *return* yang maksimum untuk tingkat risiko tertentu. Penambahan jumlah saham dalam portofolio berpengaruh terhadap konvergensi Algoritma Genetika *Multi-objective* NSGA-II, semakin banyak jumlah saham yang digunakan maka semakin sulit konvergensinya. Hal tersebut dapat dilihat dari hasil kinerja Algoritma yang memberikan hasil cukup baik untuk 5 sampai 10 saham dan kurang baik untuk lebih dari 10 saham.

**Kata kunci:** *Multi-objective optimization*, Portofolio saham, *Multi-objective Non-dominated Sorting Genetic Algorithm* (NSGA-II)

---

### Abstract

Multi-objective Genetic Algorithm NSGA-II is one of the evolutionary algorithm is popularly used in multi-objective optimization problem that is to find Pareto optimal solution. In this research the Multi-objective Genetic Algorithm NSGA-II is used to solve the optimization problem of the stock portofolio of Indeks LQ45.

Some parameters used are size of population, maximum generation, probability of crossover, and probability of mutation. The final result of the Multi-objective Genetic Algorithm NSGA-II is a form of graph efficient frontier, where it is a set of the best options for the investor which offer the minimum level risk at a given expected return. The number of stock used in portofolio will influence the convergence of the Multi-objective Genetic Algorithm NSGA-II, the more number of stock used in portofolio will more difficult convergence. It can be seen from the result of performance algorithm that gives fairly good result for 5 to 10 stocks and defective for more than 10 stocks.

**Keyword:** *Multi-objective optimization*, Stock Portofolio, *Multi-objective Non-dominated Sorting Genetic Algorithm* (NSGA-II)

---

**1. Pendahuluan**

Dalam praktik banyak ditemukan permasalahan optimisasi, dimana kita harus mencari solusi yang optimal dengan mempertimbangkan banyak tujuan. Permasalahan optimisasi yang berisi lebih dari satu fungsi tujuan dikenal sebagai *Multi-objective optimization*. Seperi halnya tujuan yang dingin dicapai oleh suatu perusahaan dalam memproduksi barang, yaitu ketika waktu produksi barang cepat tetapi kualitas barang yang dihasilkan kurang baik, dan ketika waktu produksi barang lama tetapi kualitas barang yang dihasilkan baik. Tujuan itu tentu saja berlawanan satu sama lain, sehingga tidak memungkinkan adanya solusi tunggal dalam penyelesaiannya.

*Multi-objective optimization* dalam penelitian ini yaitu berupa optimisasi portofolio saham, dimana terdapat dua tujuan yang akan dicapai yaitu berupa meminimalkan risiko dan memaksimalkan *return* portofolio saham. Dalam membentuk portofolio saham, risiko dan *return* merupakan tujuan yang saling *trade-off*, artinya apabila kita menginginkan risiko kecil maka konsekuensi *return* yang di peroleh kecil, dan apabila menginginkan *return* yang diterima besar maka risiko yang diperoleh besar. Pemilihan ini sangat ditentukan oleh tipe investor, oleh karena itu akan ada banyak pilihan terhadap portofolio. Kumpulan dari pilihan tebaik bagi investor yang mampu menawarkan tingkat *return* tertinggi dengan risiko yang sama disebut dengan *efficient frontier*. Salah satu algoritma yang cukup terkenal untuk mendapatkan solusi optimal yaitu dengan menggunakan Algoritma Genetika *Multi-objective*.

Terdapat beberapa penelitian yang sudah dilakukan untuk menyelesaikan *Multi-objective optimization*, diantaranya adalah Noel-Ann, Chris, Constantinos, & A. Kevin yang membahas *multi-objektive evolutionary algorithm* pada optimisasi portofolio[2], dan Karoon, Kittipong & Kim-Leng[10] yang membahas *multi-objektive genetic algorithm* pada pasar listrik, dan Kalyanmoy Deb[7] yang membahas *A Fast and Elitist Multiobjective Genetic Algorithm Genetic Algorithm: NSGA-II*. Hasil yang didapat dari penelitian-penelitian yang sudah dilakukan menunjukkan bahwa algoritma genetika *multi-objective* dapat menyelesaikan masalah optimisasi portofolio dengan lebih dari satu fungsi tujuan dan menghasilkan solusi yang mendekati *Pareto optimal*.

Algoritma Genetika *Multi-objective* NSGA-II (*Non-dominated sorting Genetic Algorithm*) merupakan salah satu *evolutionary algorithm* yang sangat populer digunakan pada permasalahan optimisasi *Multi-objective*

yang dikembangkan oleh Kalyanmoy Deb[7]. Algoritma genetika NSGA-II (*Non-dominated sorting Genetic Algorithm*) akan menyimpan solusi optimal dari setiap generasi, kemudian menyeleksi dengan cara mengurutkan berdasarkan *pareto front* dan membandingkan solusi-solusi antar setiap generasi sehingga memungkinkan

untuk mengidentifikasi *feasible solution*. NSGA-II dapat menghasilkan solusi yang lebih baik dengan perhitungan yang lebih sedikit, pendekatan *elitist*, dan pembagian parameter yang lebih sedikit.

Pada penelitian ini Algoritma Genetika *multi-objektive* NSGA-II diterapkan untuk mendapatkan individu-individu terbaik yang berupa bobot masing-masing saham pada portofolio. Bobot-bobot saham yang dihasilkan selanjutnya akan digunakan untuk menghitung nilai *expected return* dan risiko portofolio saham. Hasil akhir dari penerapan Algoritma Genetika *Multi-objective* NSGA-II adalah berupa grafik *efficient frontier*. Kinerja dari Algoritma Genetika *Multi-objective* NSGA-II memberikan hasil yang cukup baik untuk 5 sampai 10 jumlah saham, dan kurang baik untuk lebih dari 10 saham dilihat dari konvergensinya.

**2. Optimasi Portofolio**

**2.1 Saham**

Saham (stock) merupakan salah satu instrument pasar keuangan yang paling populer. Menerbitkan saham merupakan salah satu pilihan perusahaan ketika memutuskan untuk pendanaan perusahaan. Pada sisi yang lain, saham merupakan instrument investasi yang banyak dipilih investor karena saham mampu memberikan tingkat keuntungan yang menarik[3].

Saham dapat didefinisikan sebagai tanda penyertaan modal seseorang atau pihak (badan usaha) dalam suatu perusahaan atau perseroan terbatas. Dengan menyertakan modal tersebut maka pihak tersebut memiliki klaim atas pendapat perusahaan, klaim atas asset perusahaan, dan berhak hadir dalam Rapat Umum Pemegang Saham (RPUS)[3].

**2.2 Return, Expected Return dan Risiko Saham**

*Return* adalah nilai balikan yang diperoleh setelah berinvestasi. *Expected return* adalah *return* yang diharapkan dalam investasi. Risiko merupakan ukuran potensi kerugian yang mungkin dihadapi dalam investasi. Sesuai dengan teori Markowitz untuk mendapatkan portofolio saham yang optimal terlebih dahulu harus dihitung *return* yang diinginkan dan juga menghitung risikonya.

**2.2.1 Return**

*Return* dari saham diperoleh dengan menghitung selisih harga saham pada suatu jangka waktu tertentu. Dapat dituliskan dengan rumus[4]:

$$R_i = \frac{S_{(i)} - S_{(i-1)}}{S_{(i-1)}} \tag{1}$$

Keterangan:

- $R_i$  = *Return* saham pada waktu *i*
- $S_{(i)}$  = Harga saham pada waktu *i*
- $S_{(i-1)}$  = Harga saham pada waktu *i-1*

2.2.2 Expected Return

Expected return didekati nilainya dengan menghitung rata-rata return pada jangka waktu tertentu sesuai dengan strategi atau skenario investasi yang diinginkan. Expected return dapat dituliskan dengan:

$$\mu = \frac{1}{T} \sum_{i=1}^T R_i \tag{2}$$

dan didekati nilainya dengan menghitung rata-rata dari return saham dengan rumus:

Ketrangan:

$\hat{\mu}$  = Expected Return saham

$T$  = Jumlah periode waktu

$R_i$  = Return saham pada waktu i

2.2.3 Risiko

Risiko dapat dinyatakan sebagai ukuran fluktuasi dari return yang diperoleh. Variansi dapat digunakan untuk menghitung resiko karena dengan menghitung variansi kita dapat melihat sebaran harga saham, semakin besar sebarannya maka semakin besar pula resikonya.

Persamaan umum variansi dapat dituliskan seperti:

$$\sigma^2 = \frac{1}{T} \sum_{i=1}^T (R_i - \hat{\mu})^2$$

Variansi akan diestimasi nilainya sebagai:

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^T (R_i - \hat{\mu})^2}{T} \tag{3}$$

Ketrangan:

$\hat{\sigma}^2$  = variansi return saham

$R_i$  = Return saham pada waktu ke i

$\hat{\mu}$  = Expected return saham

T = Jumlah periode waktu

2.3 Porfolio Saham

Portofolio dalam keuangan sering diartikan sebagai kumpulan investasi yang dimiliki oleh investor. Portofolio sangat berkaitan erat dengan pengurangan risiko investasi. Portofolio saham sendiri adalah kumpulan investasi saham yang dimiliki oleh investor sehingga dengan banyaknya saham yang dijadikan investasi diharapkan dapat menghasilkan keuntungan yang maksimal dan dapat mengurangi risiko.

bobot portofolio dijumlahkan akan berjumlah total 100% atau 1. Artinya seluruh dana telah diinvestasikan dalam portofolio.

2.3.1 Expected Return dan Risiko Portofolio

Expected return portofolio saham dengan bobot w dapat ditulis[4]:

$$\mu_p = \sum_{i=1}^n w_i \mu_i \tag{4}$$

Keterangan:

$\mu_p$  = Expected return portofolio

$w_i$  = Bobot saham ke i

$\mu_i$  = Expected return saham i

$n$

= Banyaknya saham dalam portofolio

Sedangkan untuk menghitung risiko portofolio harus diperhatikan korelasi atau hubungan antara saham-saham pembentuk portofolio. Sehingga risiko portofolio dapat dituliskan dengan rumus[4]:

$$\begin{aligned} \sigma_p^2 &= \left( \sum_{i=1}^n w_i \mu_i \right)^2 - \left( \sum_{i=1}^n w_i \mu_i \right)^2 \\ &= \left( \sum_{i=1}^n w_i \mu_i \right)^2 - \left( \sum_{i=1}^n w_i \mu_i \right)^2 \\ &= \sum_{i=1}^n w_i^2 \mu_i^2 + 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^n w_i w_j \mu_i \mu_j \end{aligned} \tag{5}$$

Proporsi nilai portofolio yang terdiri dari n aset diinvestasikan dalam setiap aset-aset individual dalam

$$w =$$

Keterangan

:

$\sigma^2$  = Risiko portofolio

$\rho_{ij}$  = Bobot saham i dan j

$w$  = Vektor bobot

$\sigma_{ij}$  = Kovariansi antara saham i dan j

portofolio disebut sebagai bobot portofolio yang dilambangkan dengan  $(w_1, w_2, w_3, \dots, w_n)$ . Jika seluruh

$C$  = Matriks kovariansi

### 2.4 Metode Portofolio Mean Variance

Pada tahun 1959, Harry Markowitz membentuk suatu pendekatan investasi saham baru dengan metode *Mean Variance*. Metode *Mean Variance* menekankan pada usaha memaksimalkan ekspektasi *return (mean)* atau meminimumkan ketidakpastian/risiko (*variance*) untuk memilih dan menyusun portofolio. Pada metode *Mean Variance*, penentuan bobot saham dengan variansi minimum akan menggunakan persamaan sebagai berikut:

$$\frac{wC^{-1}r}{wC^{-1}1} \tag{6}$$

Keterangan:

$w$  = Vektor bobot masing-masing saham dalam portofolio,  $w = (w_1, w_2, \dots, w_n)$  dengan  $w_i$  adalah bobot saham ke- $i$

$\mu$  = Vektor baris yang semua entrinya berukuran  $(1 \times n)$ , dimana  $n$  adalah jumlah saham

$C$  = Matriks kovariansi  $(n \times n)$ .

Selanjutnya untuk menentukan bobot saham dengan variansi minimum dan dengan memasukkan *expected return* portofolio yang diinginkan dapat dirumuskan sebagai berikut:

$$w = \frac{\mu^{-1} C^{-1} (1 - \mu C^{-1} \mu^{-1})^{-1} (\mu^{-1} m - 1)}{\mu^{-1} C^{-1} \mu^{-1} (1 - \mu C^{-1} \mu^{-1})^{-1} \mu^{-1} C^{-1} \mu^{-1}} \quad (7)$$

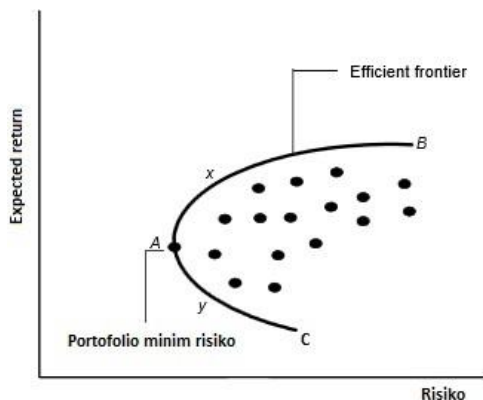
Keterangan:

$\mu_y$  = *Expected return* portofolio yang diinginkan

$m$  = Vektor  $(n \times 1)$  *expected return* saham.

### 2.4.1 Efficient frontier

*Efficient frontier* pertama kali dikemukakan oleh Harry Markowitz pada tahun 1995. Portofolio-portofolio pada *efficient frontier* adalah portofolio optimal dimana tujuannya menawarkan *expected return* maksimal untuk beberapa tingkat risiko tertentu. Bagian dari *efficient frontier* merupakan pilihan terbaik bagi investor karena mampu menawarkan *return* tertinggi dengan risiko yang sama.



Gambar 1 Grafik *Efficient frontier*

- Titik A merupakan portofolio pada *efficient frontier* yang memberikan standar deviasi paling kecil.
- Titik A disebut dengan *global minimum variance portfolio*.

tingkat risiko yang sama, portofolio X memiliki *expected return* lebih besar dari portofolio Y.

### 2.5 Optimasi Multi-objective

Masalah optimasi portofolio saham dengan pendekatan *Mean variance* dapat dituliskan sebagai masalah optimasi *Multi-objective* yaitu:

$$\min wC \quad (8)$$

$$\max \sum_{i=1}^n w_i \mu_i \quad (9)$$

Dengan kendala,  $\sum_{i=1}^n w_i = 1$ .

Optimasi *Multi-objective* adalah Permasalahan optimisasi

yang berisi lebih dari satu tujuan fungsi tujuan. Optimisasi *multi-objective* dapat didefinisikan dalam persamaan[1]:

$$f_1(x), f_2(x), \dots, f_n(x) \quad (10)$$

Keterangan:

$n$  = Jumlah fungsi tujuan

$x$  = Vektor dari variable keputusan  $[w_1, w_2, \dots, w_n]^T$ .

Pada penelitian ini fungsi yang akan dioptimalkan adalah *expected return* dan risiko, dimana tujuan yang ingin dicapai adalah dengan memaksimalkan *expected return* serta meminimumkan risiko. Optimisasi portofolio *multi-objective* dapat dirumuskan sebagai berikut:

$$\min_{x \in X} f_1 \quad (11)$$

$$\min_{x \in X} f_2 \quad (12)$$

Dengan kendala,  $\sum_{i=1}^n w_i = 1$

Keterangan:

$$f_1 = wC \quad (\text{Fungsi objektif 1})$$

$$f_2 = \frac{1}{\sum_{i=1}^n w_i \mu_i + 1} \quad (\text{Fungsi objektif 2})$$

- Daerah *efficient frontier* (*efficient set*) atau bagian AB adalah segmen yang berada diatas *global minimum variance portfolio*. Oleh karena itu, portofolio X dominan terhadap portofolio Y yang artinya pada

$w_i$  = Bobot saham ke  $i$

$r_i$  = *Expected return* saham  $i$

C = Matriks kovariansi saham

### 2.5.1 *Dominance Test*

Dalam masalah optimisasi pada *single-objective* untuk mendapatkan solusi terbaik ditentukan dengan membandingkan nilai fungsi objektifnya, sedangkan dalam masalah optimisasi pada *multi-objective*, solusi terbaik ditentukan dengan *dominance*. Berikut adalah gambaran dari *Dominance Test*[6]:

dominasi jika:

- Solusi  $p$  adalah lebih baik dibanding  $q$  dalam semua hasil fungsi objektifnya.
- Solusi  $p$  lebih baik daripada  $q$  dalam sedikitnya satu fungsi tujuan.

*Non-dominated solutions set* adalah suatu set solusi yang merupakan solusi yang tidak didominasi oleh solusi manapun dari kumpulan anggota solusi yang ada, dan kumpulan dari *non-dominated solution set* disebut *Pareto-Optimal set*. Sedangkan batas yang digambarkan dengan kumpulan semua titik-titik yang dipetakan dari *Pareto-Optimal set* disebut *Pareto-optimal Front*.

## 2.6 Multi-objective Non-dominated Sorting Genetic Algoritim (NSGA-II)

*Non-dominated sorting algorithm* (NSGA-II) merupakan algoritma genetika yang sangat populer yang digunakan pada permasalahan *multi objective*. NSGA-II merupakan versi yang dimodifikasi dan dikembangkan, dimana lebih baik dalam *sorting algorithm* yang disertai dengan *elitism* dan tidak membutuhkan prioritas pembagian parameter yang harus dipilih[7].

Dalam NSGA-II semua individu dalam populasi gabungan (*parents and child*) diranking berdasarkan pada solusi *non-dominat* pada setiap *front*. *Front* pertama yang terbentuk didasarkan pada kumpulan *non-dominant* dalam populasi awal dan *Front* yang kedua akan didominasi oleh individu-individu yang berada dalam *front* yang pertama dan seterusnya.

Selain diberikan nilai *fitness*, parameter baru yang disebut *crowding distance* juga dihitung oleh masing-masing individu. *Crowding distance* merupakan pengukuran mengenai kedekatan antara individu dengan individu di sampingnya. Nilai *crowding distance* yang semakin besar akan menghasilkan populasi yang beragam.

Induk-induk (*parents*) akan diseleksi dari suatu populasi dengan menggunakan *binary tournament selection* berdasarkan nilai *rank* dan *crowding distance*. Individu yang akan terpilih merupakan individu yang memiliki nilai *rank* lebih kecil dibandingkan individu lainnya, atau memiliki nilai *crowding distance* yang lebih besar dari individu lainnya. *Crowding distance* dibandingkan jika nilai *rank* dari kedua individu sama. Populasi yang diseleksi akan membangkitkan keturunan baru (*offspring*) melalui proses *crossover* dan mutasi. Populasi awal yang berisikan induk (*parents*) dan populasi anak (*offspring*) diurutkan kembali berdasarkan *non-dominat* dan hanya N individu yang terbaik yang akan terpilih, dimana N adalah ukuran populasi.

### 2.6.1 Inisialisasi Populasi

Pada tahap awal algoritma, pertama yang dilakukan adalah membangkitkan secara acak sebuah populasi yang berisi sejumlah kromosom. Inisialisasi populasi ini akan direpresentasikan dalam gen-gen yang bernilai *real*.

Populasi yang telah dibangkitkan kemudian diurutkan berdasarkan *non-dominat*.

### 2.6.2 Non-dominated Sort

Dalam permasalahan *multi-objective optimization* solusi terbaik ditentukan oleh *dominance*. Kemudian kumpulan dari dengan *non-dominated solution set* akan dipetakan dalam *Pareto Optimal Front*. Algoritma pengurutannya adalah sebagai berikut[7]:

- a. Untuk setiap individu  $p$  dalam populasi utama  $P$ , dilakukan :
  1. Inisialisasi  $F_p = \emptyset$ , yang nantinya akan berisikan semua individu-individu yang didominasi oleh  $p$
  2. Inisialisasi  $n_p = 0$ , yang merupakan jumlah individu-individu yang mendominasi  $p$
  3. Untuk setiap individu  $q$  didalam  $P$ 
    - Jika  $p$  mendominasi  $q$  maka :  
Tambahkan  $q$  pada kumpulan  $F_p$  yaitu  $F_p = F_p \cup \{q\}$
    - Jika  $q$  mendominasi  $p$  maka :  
Sehingga  $n_p = n_p + 1$
  4. Jika  $n_p = 0$  yaitu tidak terdapat individu yang mendominasi  $p$  sehingga  $p$  merupakan *front* yang pertama dan kemudian individu  $p$  diberi *rank 1* seperti  $rank_p = 1$ . Update *front* yang pertama dengan menambahkan  $p$  pada *front* yang pertama yaitu  $F = F \cup \{p\}$
- b. Inisialisasi *front* = 1  $\rightarrow i=1$
- c. Dilakukan pada *front* ke- $i$  ketika  $i \neq 0$ 
  1.  $F_i = \emptyset$  merupakan penyimpanan kumpulan individu-individu untuk *front* ke  $(i+1)$ .
  2. Untuk setiap individu  $p$  di dalam *front*  $F_{i-1}$ 
    - Setiap individu  $q$  yang berada dalam  $F_{i-1}$  berisikan semua individu-individu yang didominasi oleh  $p$
    - $n_p = n_p - 1$ , pengurangan individu  $q$
    - Jika  $n_p = 0$ , tidak terdapat individu-individu pada *front* yang berikutnya mendominasi  $q$ . akibatnya ubah  $rank_q = i + 1$  dan update  $F_i$  dengan menambahkan individu  $q$  yaitu  $F_i = F_i \cup \{q\}$
  3. Tambahkan *count front* dengan 1
  4. Kemudian ubah  $F_i$  menjadi *front* berikutnya sehingga  $F_i = F_{i-1}$

### 2.6.3 Crowding Distance

Setelah pengurutan *non-dominated* selesai dilakukan, kemudian langkah berikutnya adalah menghitung *crowding distance*. Perhitungan nilai *crowding distance* ini hanya dilakukan pada sepanjang nilai *front* yang sama. Untuk perhitungan *crowding distance* dilakukan sebagai berikut[7]:

- a. Inisialisasi jarak (*distance*) untuk semua individu-individu dengan nilai 0
- b. Untuk setiap fungsi tujuan  $m$ 
  - Urutan semua individu-individu di dalam *front* berdasarkan fungsi tujuan  $m$  yaitu  $I = \text{sort}(I)$
  - Untuk pertama kali, beri nilai jarak (*distance*) untuk setiap individu dalam *front* sama dengan tak terhingga yaitu  $I(1) = \infty$  dan  $I(2) = \infty$
  - Untuk  $k=2$  sampai  $(n-1)$

$$I(k+1).m - I(k-1).m$$

- $I(d_k) = I(d_k) + \frac{f_{\max} - f_{\min}}{m}$  (13)
- $I(k).m$  adalah nilai fungsi tujuan ke- $m$  dari individu ke- $k$  di  $I$ .

**2.6.4 Binary Tournament Selection**

Ketika pengurutan berdasarkan *non-domination* dan nilai *crowding distance* selesai dilakukan, maka semua individu diseleksi dengan menggunakan *binary tournament selection*. Terdapat dua kriteria dalam seleksi yaitu:

- a. *Non-domination rank*.  
Jika terdapat solusi dengan nilai *rank* yang berbeda maka solusi yang terpilih adalah solusi dengan nilai *rank* terkecil.
- b. *Crowding distance*  
Jika terdapat solusi dengan nilai *rank* yang sama maka solusi yang terpilih adalah solusi dengan nilai *crowding distance* terbesar.

**2.6.5 Crossover**

Rekombinasi untuk representasi *real* bisa dilakukan dengan dua cara yaitu, *discrete* dan *intermediate crossover*. *Intermediate crossover* adalah rekombinasi yang paling umum digunakan. Kedua anak dihasilkan berdasarkan rumus:

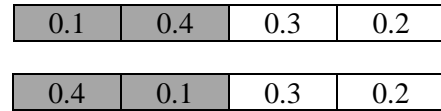
$$\begin{matrix} \text{Anak 1} \\ \text{Anak 2} \end{matrix} = \begin{matrix} \text{Orangtua 1} \\ \text{Orangtua 2} \end{matrix} + \alpha \cdot \begin{matrix} \text{Orangtua 1} \\ \text{Orangtua 2} \end{matrix} - \begin{matrix} \text{Orangtua 1} \\ \text{Orangtua 2} \end{matrix} \quad (14)$$

Dimana  $\alpha$  adalah orangtua. Dan  $0 \leq \alpha \leq 1$  merupakan parameter yang bisa dibuat konstan, atau ditentukan secara acak pada setiap saat.

**2.6.6 Mutasi**

Mutasi merupakan proses pergantian suatu gen dalam individu dengan menggunakan probabilitas mutasi. Mutasi yang digunakan adalah *swap mutation*. *Swap mutation* dilakukan dengan menentukan jumlah kromosom yang akan mengalami mutasi dalam satu

populasi melalui parameter *mutation rate* ( $pm$ ) dimana proses mutasi dilakukan dengan cara menukar gen yang telah dipilih secara acak dengan gen sesudahnya.



Gambar 2 Swap Mutation

**3. Analisis Hasil Pengujian**

Tujuan dari pengujian sistem adalah menguji kinerja Algoritma Genetika Multi-objective NSGA-II untuk

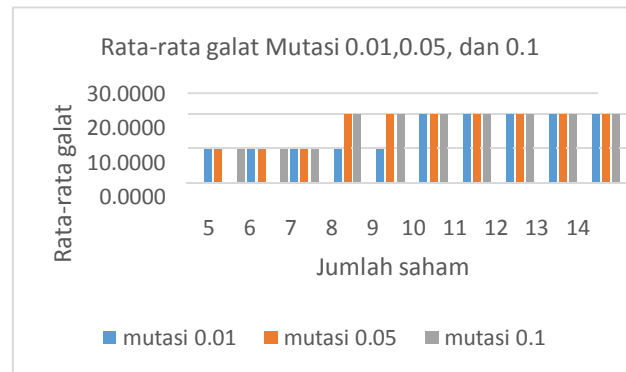
mencari parameter Algoritma genetika *Multi-objective*

yang terbaik. Parameter NSGA-II yang digunakan yaitu ukuran populasi 100, maksimum generasi 1000, probabilitas crossover 0.99, dan probabilitas mutasi 0.01, 0.05, dan 0.1.

Pengujian ini dilakukan *running* sebanyak empat kali untuk setiap data yang sama. Setiap kali *running*, hasil dari Algoritma genetika *Multi-objective* NSGA-II dan *Mean variance* akan disimpan untuk mendapatkan nilai rata-rata galat.

**3.1 Hasil rata-rata dari nilai galat**

Hasil pengaruh probabilitas mutasi ( $Pm$ ) 0.01, 0.05, dan 0.1 dengan probabilitas *crossover* berturut-turut 0.99, 0.95, dan 0.9 terhadap rata-rata galat dapat dilihat pada grafik berikut ini:



Gambar 3 Grafik rata-rata galat mutasi 0.01, 0.01, dan 0.1

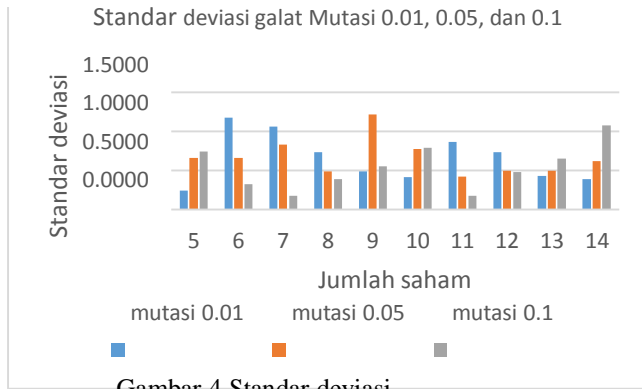
Berdasarkan gambar 3 dapat dilihat bahwa semakin besar probabilitas mutasi yang digunakan berpengaruh terhadap nilai rata-rata galat untuk setiap portofolio saham. Probabilitas mutasi 0.01 lebih baik dibandingkan probabilitas mutasi 0.05 dan 0.1 dilihat dari nilai rata-rata galat.

**3.2 Hasil standar deviasi dari nilai galat**

Nilai standar deviasi dari hasil galat *running* program untuk jumlah saham yang berbeda dengan mutasi 0.01,



0.05 dan 0.1 dengan probabilitas *crossover* berturut-turut 0.99, 0.95, dan 0.9 dengan tujuan mengukur seberapa luas penyimpangan nilai data tersebut dari nilai rata-rata nya. Hasil pengaruh probabilitas mutasi (Pm) 0.01, 0.05, dan 0.1 dengan probabilitas *crossover* berturut-turut 0.99, 0.95, dan 0.9 terhadap standar deviasi galat dapat dilihat pada gambar berikut ini:



Berdasarkan Gambar 4 dapat dilihat bahwa nilai standar deviasi galat untuk 5 sampai 14 saham dengan mutasi 0.01, 0.05, dan 0.1 dengan probabilitas *crossover* berturut-turut 0.99, 0.95, dan 0.9 menunjukkan bahwa nilai dari galat yang ada pada himpunan tersebut cukup banyak tersebar jauh dari rata-rata galat himpunan tersebut, sehingga sebaran galatnya cukup besar yang menyebabkan nilai standar deviasi bervariasi.

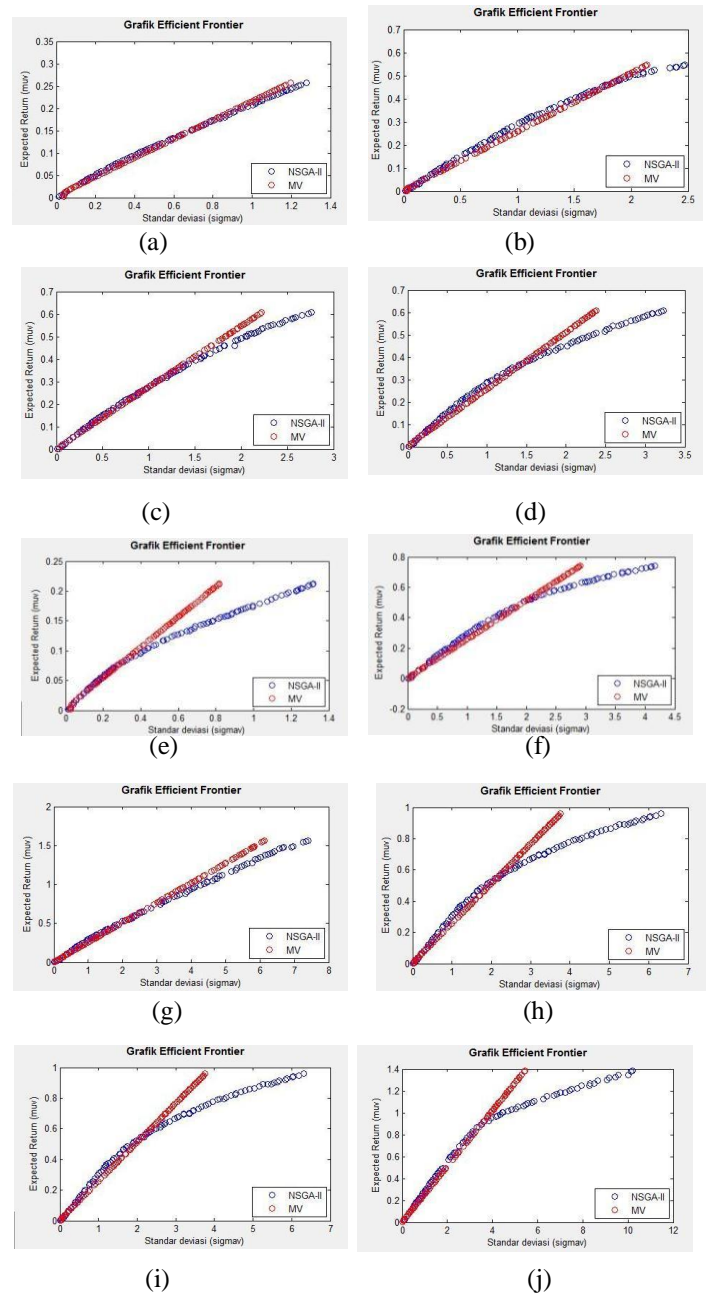
### 3.3 Hasil dan Analisis *Efficientr frontier*

Keluaran yang dihasilkan dari Algoritma Genetika *Multi-objective* NSGA-II yaitu berupa solusi-solusi yang terbentuk dalam *efficient frontier*. Hasil dari pengujian yang telah dilakukan probabilitas mutasi yang terpilih adalah 0.01 dengan probabilitas *crossover* 0.99. Berikut daftar tabel dati nilai galat paling kecil dari setiap *running* program yang dilakukan pada mutasi 0.01:

Tabel 1 Galat terpilih

Jumlah saham	Galat
5	9.0978
6	10.6805
7	13.7309
8	13.1111
9	16.3367
10	16.8531
11	17.8422
12	18.1828
13	19.6624
14	22.5506

Berikut merupakan grafik *efficient frontier* dari hasil *running* dengan probabilitas mutasi (Pm) = 0.01:



Gambar 5 Efficient frontier: (a) 5 Saham, (b) 6 Saham, (c) 7 Saham, (d) 7 Saham, (e) 9 Saham, (f) 10 Saham, (g) 11 Saham, (h) 12 Saham, (i) 13 Saham, (j) 14 Saham

Kumpulan dari gambar diatas adalah kumpulan grafik *efficient frontier* terpilih untuk 5 sampai 14 saham. Terdapat dua garis pada gambar tersebut yaitu garis berwarna biru yang menyatakan solusi dari NSGA-II dan garis berwarna merah menyatakan solusi dari *Mean variance*. Sumbu x merupakan fungsi objective 1 yang menyatakan nilai standar

deviasi ( $\sigma$ ), dan sumbu  $y$  merupakan fungsi 2 yang menyatakan nilai *expected return* ( $\mu$ ).

Pada grafik *efficient frontier* beberapa informasi dapat diketahui bahwa garis *efficient frontier* menggambarkan garis yang efisien yaitu dengan tingkat nilai risiko yang sama terdapat *return* yang lebih besar. Semua titik-titik solusi dalam *efficient frontier* mempunyai kedudukan yang sama satu sama lain. Dari grafik *efficient frontier* dapat dilihat bahwa ketika nilai risiko portofolio kecil maka nilai *expected return* kecil, demikian juga ketika nilai risiko meningkat maka *expected return* juga meningkat. Perbedaan yang ada pada grafik *efficient frontier* untuk setiap penambahan jumlah saham adalah pada titik-titik solusi yang dihasilkan dari NSGA-II dan *Mean variance*. Ketika 5 - 10 saham solusi-solusi yang dihasilkan NSGA-II mendekati solusi *Mean variance*, dan ketika 11 - 14 saham solusi yang dihasilkan mulai menjauh dari solusi *Mean variance*, namun jarak yang diberikan antara NSGA-II dan *Mean variance* tidak terlalu besar. Sehingga dapat disimpulkan bahwa untuk penambahan lebih dari 10 saham masih belum menunjukkan hasil yang konvergen.

#### 4. Kesimpulan

Berdasarkan hasil analisis terhadap implementasi dan pengujian sistem Algoritma Genetika *Multi-objective* NSGA-II dengan metode pembandingan *Mean variance*, maka dapat ditarik kesimpulan sebagai berikut:

1. Dari hasil pengujian dengan menggunakan parameter probabilitas mutasi 0.01, 0.05, dan 0.1 pada penerapan Algoritma Genetika *Multi-objective* NSGA-II, dengan banyak generasi 1000, ukuran populasi 100, dan probabilitas *crossover* 0.99 diperoleh parameter probabilitas mutasi 0.01 lebih baik dibandingkan dengan probabilitas mutasi 0.05 dan 0.1 dilihat dari segi nilai *galat* yang dihasilkan.
2. Penambahan jumlah saham berpengaruh terhadap konvergensi Algoritma Genetika *Multi-objective* NSGA-II, semakin banyak jumlah saham yang digunakan maka semakin sulit konvergensinya.
3. Algoritma Genetika *Multi-objective* NSGA-II masih memberikan hasil *running* yang berbeda-beda atau belum stabil, hal tersebut dikarenakan jumlah generasi yang dilakukan masih belum cukup untuk mendapatkan hasil *running* yang stabil.

#### 5. Daftar pustaka

- [1] Abraham, A., Jain, L., & Goldberg, R., (2005) Evolutionary multiobjective optimization: *Theoretical Advances and Applications*. London: Springer
- [2] Bradshaw, N., Walshaw, C., & Parrott, A.K, 2009, A multi-objective evolutionary algorithm for portfolio optimization.
- [3] Bursa Efek Indonesia, "IDX,"[Online]. Available: <http://www.idx.co.id/id-id/beranda/informasi/bagiinvestor/saham.aspx>. [Diakses 16 10 2014].
- [4] Capinski, M., & Zastawniak, T. (2003). *Mathematics for Finance: An Introduction to Financial Engineering*. London: Springer.
- [5] Coelleo Coelleo, C.A., Landa Becerra, R., 2009, Evolutionary multi-objective optimization in materials science and engineering. *Mater. Manuf. Process.* 24, 199-129.
- [6] Deb, K., 2001, Multi-objective optimization using evolutionary algorithms. New York: John Wiley & Son, Inc.
- [7] Deb, K, Pratap, A, Argawal, S, & Meyarivan, T, 2002, A fast elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans Evol Comput*, 182-97.
- [8] Konak, A., Coit D.W., & Smith A.E., Multi-objective optimization using genetic algorithms: A Tutorial.
- [9] Indonesia Stock Exchange, Buku Panduan : Indeks Harga Saham Efek Indonesia, Jakarta : PT Bursa Efek Indonesia, 2008.
- [10] Suksonghong, K., Boonglong, K., & Goh, K.L., 2014, Multi-objective genetic algorithm for solving optimization problem in the electricity market.
- [11] Suyanto, (2008), *Evolutionary Computation*, Bandung: Informatika Bandung.
- [12] Tandelilin, Eduardus. 2010, *Portofolio dan Investasi (Teori dan Aplikasi)*, Yogyakarta: Kanisius.