Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi

Vol. 6, No. 1, Januari2011 **ISSN 0216 - 0544**

OPTIMASI FUNGSI MULTI-OBYEKTIF BERKENDALA MENGGUNAKAN ALGORITMA GENETIKA ADAPTIF DENGAN PENGKODEAN *REAL*

[a]Wayan Firdaus Mahmudy, [b]Muh. Arif Rahman
Program Studi Ilmu Komputer, Universitas Brawijaya
Jl. Veteran Malang, 65145
E-Mail: [a]wayanfm@ub.ac.id

Abstrak

Masalah optimasi dengan beberapa fungsi tujuan (multi-obyektif) tidak mudah dipecahkan karena biasanya terjadi konflik di antara fungsi tujuan dan solusinya bukanlah solusi tunggal tetapi berupa himpunan solusi. Algoritma genetika (*Genetic Algorithms/GAs*) yang merupakan salah satu metode meta-heuristis dapat digunakan untuk memecahkan masalah ini tetapi algoritma genetika standar mudah terjebak dalam daerah optimum lokal (konvergensi dini) dan kecepatan pencarian titik solusi akan menurun jika mendekati titik optimum. Penelitian ini memaparkan penggunaan *GAs* dengan tingkat mutasi adaptif untuk menyeimbangkan kemampuan eksplorasi dan eksploitasi daerah pencarian. Sebuah aturan digunakan untuk menentukan apakah nilai tingkat mutasi perlu dinaikkan atau diturunkan. Jika tidak ada perbaikan nilai *fitness* yang signifikan dibandingkan generasi sebelumnya maka nilai tingkat mutasi dinaikkan sehingga memungkinkan *GAs* untuk lebih melakukan eksplorasi daerah pencarian dan sekaligus keluar daerah optimum lokal. Jika ada perbaikan nilai *fitness* yang signifikan maka dilakukan penurunan nilai *mutation rate* sehingga memungkinkan *GAs* untuk lebih mengeksploitasi daerah pencarian lokal. Hasil percobaan menunjukkan bahwa penggunaan mutasi yang adaptif mempercepat pergerakan *GAs* ke daerah *feasible* dan sekaligus mempercepat pencapaian solusi.

Kata kunci: optimasi multi-obyektif, algoritma genetika, mutasi adaptif.

Abstract

Multi-objective optimization problem is difficult to be solved as its objectives generally conflict with each other and its solution is not in the form of a single solution but a set of solutions. Genetic algorithms (GAs) is one of meta heuristic algorithms that may be used to solve this problem. However, a standard GAs is easily trapped in local optimum areas and searching process rate will be lower around the optimum points. This paper proposes a GAs with an adaptive mutation rate to balance the exploration and exploitation on the search space. A simple rule has been developed to determine wheter the mutation rate is increased or decreased. If a significant improvment of the fitness value is not achieved, the mutation rate is increased to enable the GAs exploring search space and escaping the local optimum area. In contrast, the mutation rate is decreased if significant improvment of the fitness value is achieved. This mechanism guide the GAs to exploit the local search area. The experiments show that by using the adaptive mutation, the GAs will move faster toward a feasible search space and achieving solutions on sorter time.

Key words: multi-objective optimization, genetic algorithms, adaptive mutation.

PENDAHULUAN

Hampir semua masalah optimasi di dunia nyata memiliki banyak fungsi obyektif yang harus dipenuhi secara simultan dan seringkali fungsi-fungsi tersebut saling bertentangan. Pada masalah optimasi portofolio saham, peluang keuntungan harus dimaksimumkan dan secara bersamaan resiko harus diminimumkan. Pada optimasi campuran pakan ternak, kandungan nutrisi harus dimaksimumkan dan biaya bahan bakudiminimumkan. Untuk mengoptimumkan (memaksimumkan/meminimumkan) satu fungsi obyektif maka harus mengorbankan fungsi obyektif yang lain. Mendapatkan satu solusi dan mengukur seberapa baik solusi ini dibandingkan solusi-solusi yang lain merupakan tujuan utama penyelesaian masalah ini. Pada hampir semua masalah optimasi fungsi multiobyektif tidak akan didapatkan solusi optimum tunggal tapi berupa kumpulan solusi alternatif. Tidak ada sebuah solusi yang lebih baik terhadap solusi lain jika semua fungsi obyektif dipertimbangkan. Hal ini biasa disebut solusi *pareto-optimal*. Solusi *pareto-optimal* memberikan keleluasaan kepada manusia sebagai pengambil keputusan untuk menentukan tujuannya berdasarkan domain pengetahuan yang dipunyai [1].

Secara alami, Algoritma Genetika (*Genetics Algorithms*/[*Ga*s]) tidak hanya memberikan satu solusi tetapi berupa himpunan solusi sehingga sesuai digunakan untuk menyelesaikan masalah optimasi multiobyektif. *GAs* dengan pengkodean biner umum digunakan dalam masalah optimasi fungsi [2]. Kelemahan pengkodean biner adalah jika *range* solusi berada dalam daerah kontinyu maka solusi yang didapatkan dibatasi oleh tingkat ketelitian yang ditentukan sebelumnya. Pengkodean *real* (*Real-Coded Genetic Algorithms, [RCGA]*) dapat digunakan untuk menyelesaikan masalah ini [3]. Dalam banyak kasus, *RCGA* memberikan hasil kurang optimum pada pencarian dalam area yang kompleks. Konvergensi dini sering terjadi karena populasi solusi terjebak dalam *optimum* lokal dan kecepatan pencarian titik solusi akan menurun jika mendekati titik optimum [4]. Mutasi merupakan operator genetika yang digunakan untuk menjaga diversitas dari kromosom sehingga terhindar dari konvergensi dini. Tingkat mutasi (*mutation rate*) yang tinggi (mendekati 1) akan meningkatkan kemampuan eksplorasi solusi pada daerah pencarian tapi menurunkan kemampuan exploitasi solusi pada area lokal.

Penelitian ini memaparkan penggunaan *GAs* dan sekaligus mengeksplorasi beberapa operator *crossover* dan mutasi untuk mengatasi masalah konvergensi dini. Penggunaan aturan (*rule*) untuk mengubah nilai *mutation rate* secara adaptif diharapkan dapat mempercepat proses pencarian solusi.

OPTIMASI MULTIOBYEKTIF

Sebuah permasalahan optimasi (*optimization problem*), yang dimodelkan secara matematis, umumnya terdiri dari fungsi-fungsi tujuan (*objective functions*) dan kendala-kendala (*constraints*). Fungsi tujuan merepresentasikan tujuan yang ingin dioptimalkan. Karena jumlah fungsi tujuannya lebih dari satu, maka solusi optimum dari *multicriteria optimization problem* juga lebih dari satu, yang kesemuanya masuk ke dalam sebuah set yang disebut *Pareto frontier*. Hal ini sejalan dengan prinsip dimana tidak ada satu pun solusi yang mampu memberikan hasil yang lebih optimal dari salah satu fungsi tujuan yang ada tanpa mengorbankan fungsi tujuan lainnya [5].

Optimasi multiobyektif dapat dirumuskan dalam persamaan (1) :

$$\min_{\overline{X} \in C} F(\overline{X}) = \begin{bmatrix} f_1(\overline{X}) \\ f_2(\overline{X}) \\ ... \\ f_n(\overline{X}) \end{bmatrix}, \qquad (1)$$

n≥2 dan

$$C = \left\{ \overline{X} : h(\overline{X}) = 0, g(\overline{X}) \le 0, a_i \le x_i \le b_i \right\}$$

$h(\overline{X})$ dan $g(\overline{X})$ merupakan fungsi kendala, $\overline{X} = (x_1, x_2, ..., x_n)$ merupakan vektor dari variabel keputusan, a_i merupakan batas bawah dan b_i merupakan batas atas [5]. Jika sebuah fungsi kendala mempunyai bentuk $g(\overline{X}) \ge c$ maka dapat diubah menjadi $-g(\overline{X}) + c \le 0$. Konsep skalar dari *optimum* tidak biasa diterapkan secara langsung pada kasus multiobyektif. Konsep penggantinya adalah *optimum pareto*. Vektor $\overline{X}^* \in C$ dikatakan *optimum pareto* jika semua vektor $\overline{X} \in C$ yang

lain mempunyai nilai yang lebih tinggi setidaknya untuk satu fungsi obyektif.

Optimasi fungsi multiobyektif mendapatkan perhatian yang signifikan dari para peneliti. Xiaohui & Eberhart [6] menyelesaikan optimasi multiobyektif menggunakan *Particle Swarm Optimization* (*PSO*). Hasil percobaan menunjukkan bahwa PSO dapat menemukan beberapa solusi *pareto-optimal* secara efisien. Doerner dkk [7] menggunakan *Ant Colony Optimization* (*ACO*) untuk menyelesaikan optimasi multiobyektif pada penentuan portofolio proyek. Mereka menunjukkan bahwa *ACO* cukup efisien untuk menangani interaksi antar proyek yang kompleks. Bandyopadhyay dkk [8] mengajukan *Simmulated Annealing* (*SA*) yang juga cukup berhasil dalam menyelesaikan berbagai masalah optimasi fungsi multiobyektif.

ALGORITMA GENETIKA

Algoritma Genetika (*Genetic Algorithms/*[*Gas*]) termasuk dalam kelas algoritma pencarian yang menirukan proses evolusi alami. *GAs* menggunakan kromosom untuk mengkodekan sebuah kemungkinan solusi. Sejumlah kromosom dalam populasi akan mengalami operasi rekombinasi (tukar silang/*crossover* dan mutasi) untuk membentuk generasi berikutnya. Setiap kromosom mempunyai nilai kebugaran (*fitness*) yang menentukan peluangnya untuk tetap bertahan hidup dalam generasi berikutnya. Dengan mekanisme seleksi ini diharapkan nilai *fitness* setiap kromosom akan meningkat pada setiap generasi. Pada akhir generasi, kromosom dengan nilai *fitness* terbaik akan diuraikan menjadi sebuah solusi [9]. Solusi ini mungkin bukan merupakan solusi *optimum* tetapi fakta empirik membuktikan dengan menentukan paramater-parameter seperti ukuran populasi, *crossover-rate* dan *mutation-rate* yang sesuai, *GAs* akan memberikan hasil yang memuaskan (mendekati optimum) dalam waktu yang relatif cepat [10].

Kinerja *GAs* sangat dipengaruhi oleh keseimbangan kemampuan eksplorasi dan eksploitasi yang dilakukan sepanjang generasi. Problem utama yang dialami oleh *GAs* adalah konvergensi dini yang disebabkan oleh kurangnya diversitas populasi setelah melewati sekian generasi. Perubahan paramater *GAs*secara adaptif menggunakan logika fuzzy menghasilkan mekanisme untuk mencegah kondisi ini dan sekaligus keluar dari jebakan *optimum* lokal [11]. Kombinasi (hibridisasi) *GAs* dengan algoritma lain menghasilkan *memetic algorithms* juga terbukti efisien untuk mengatasi masalah ini [2, 12]. Jaszkiewicz [13] memperkaya kemampuan *GAs* dengan *local search* yang mereka sebut *genetic local search* untuk mempercepat pencarian solusi. Deb dkk [14] menggunakan *GAs* yang dimodifikasi. Mereka memperkenalkan pendekatan yang disebut '*improved version of nondominated sorting genetic algorithm*' (*NSGA-II*) yang memungkinkan *GAs* dapat bergerak cepat menuju perbatasan daerah *pareto-optimal*.

Inisialisasi Populasi dan Representasi Kromosom

Siklus perkembangan *GAs* diawali dengan pembuatan himpunan solusi baru (*initialization*) secara acak yang terdiri atas sejumlah string kromosom dan ditempatkan pada penampungan populasi. Penentuan representasi kromosom yang sesuai memegang peranan penting dalam kesuksesan implementasi *GAs*[15]. Pada penelitian ini digunakan representasi kromosom bilangan pecahan (*real*). Panjang string kromosom tergantung pada banyaknya variabel bebas/keputusan (*x*) yang digunakan.

Evaluasi

Untuk mengevaluasi nilai *fitness* dari kromosom dilakukan langkah-langkah berikut :

a. Ambil nilai *real* dari tiap individu dalam populasi. $x^k = \left(x_1^k, ..., x_n^k\right)$;
   k =1,2,..., ukuran_populasi,
   n =banyaknya_variabel_keputusan.
b. Evaluasi nilai fungsi tujuan $f(xk)$
c. Konversi nilai fungsi tujuan menjadi nilai *fitness*. Untuk masalah maksimasi nilai *fitness* = $f(xk)$. Untuk masalah minimasi nilai *fitness* = z - $f(xk)$; z adalah sembarang bilangan *real*.

Semakin besar nilai *fitness* dari satu invidu maka semakin baik pula solusi yang didapatkan dari individu tersebut. Nilai *fitness* ini digunakan untuk memilih *n* individu terbaik yang dipertahankan hidup pada generasi selanjutnya. Mekanisme untuk menilai apakah

sebuah individu lebih baik dari yang lain adalah:

a. Jika tidak ada kendala yang dilanggar maka sebuah individu dikatakan lebih baik dari individu yang lain jika semua nilai fitnessnya sama (untuk masing-masing fungsi obyektif) dan minimal satu nilai fitnessnya lebih baik.
b. Jika tidak ada kendala yang dilanggar dan kriteria a tidak dipenuhi maka kedua individu termasuk dalam *pareto-set*. Individu yang lebih baik adalah yang total nilai fitnessnya (dari semua fungsi tujuan) lebih besar.
c. Jika kedua individu melanggar minimal satu kendala maka dipilih yang total nilai kendalanya (untuk nilai $g_i(x)>0$) lebih kecil. Hal ini untuk menjamin solusi yang dipilih memenuhi kendala sebanyak mungkin.

Seleksi

Seleksi digunakan untuk memilih dua individu/kromosom sebagai induk untuk anak pada generasi berikutnya. Pada tulisan ini digunakan metode pemilihan seragam, artinya setiap individu dalam populasi memiliki peluang yang sama untuk terpilih. Metode ini dipilih untuk menghemat waktu yang digunakan pada komputasi nilai probabilitas kumulatif yang digunakan pada metode seleksi *roulette-wheel*.

Crossover

Individu baru (*offspring*) terbentuk dari proses *crossover*. Pada tulisan ini digunakan dua metode *crossover* yaitu *flat crossover*[16] dan *extended intermediate crossover*[17]. Misalkan $C_1 = (c_1^1 \ldots c_n^1)$ dan $C_2 = (c_1^2 \ldots c_n^2)$ adalah dua kromosom yang telah diseleksi untuk melakukan *crossover*, maka perbedaan dua metode *crossover* yang digunakan adalah sebagai berikut

Flat Crossover

$OffspringH = (h_1 \ldots, h_i, \ldots h_n)$ dibangkitkan dan h_i secara acak dipilih pada interval $[c_i^1, c_i^2]$.

Extended Intermediate Crossover

$Offspring\ H = (h_1 \ldots, h_i, \ldots h_n)$ dibangkitkan dan $h_i = c_i^1 + \alpha_i (c_i^2 - c_i^1)$, α_i dipilih secara acak pada *interval* [-0,25; 1,25].

Secara acak dua metode ini digunakan untuk menghasilkan anak. Penggunaan dua metode ini secara bersamaan adalah untuk menghasilkan anak yang lebih bervariasi sehingga diharapkan tidak terjadi konvergensi dini.

Mutasi

Mutasi digunakan untuk mejaga diversitas dari kromosom sehingga terhindar dari konvergensi dini. Pada tulisan ini digunakan *random mutation* [18] dan *M¨uhlenbein's mutation*[17].

Random Mutation

h_i' merupakan bilangan random pada *range* [a,b].

M¨uhlenbein's Mutation

$h_i' = h_i \pm a\ (max_i - min_i)$, a bilangan random pada interval [0, 1], max_i nilai maksimum dari x_i , min_i nilai minimum dari x_i. Pembuatan *rule* untuk mengatur nilai a secara adaptif diperlukan agar proses iterasi berjalan efisien. Setiap anak hasil *crossover* akan terkena mutasi dengan peluang sesuai dengan nilai *mutation rate*.

Pengaturan *Mutation Rate* Secara Adaptif

Pada setiap generasi dihitung nilai *fitness* rata-rata dari populasi (*fitnessAvg*). Jika ada perbaikan nilai *fitness* rata-rata yang signifikan dibanding generasi sebelumnya (*fitnessAvg>>fitnessAvgOld*) maka dilakukan penurunan nilai *mutation rate*. Hal ini memungkinkan *GA* untuk lebih mengeksploitasi daerah pencarian lokal. Jika tidak ada perbaikan nilai *fitness* rata-rata yang signifikan dibanding generasi sebelumnya (*fitnessAvg<<fitnessAvgOld*) maka dilakukan peningkatan nilai *mutation rate*. Hal ini memungkinkan *GA* untuk lebih memperluas (eksplorasi) daerah pencarian dengan melompati daerah *optimum* lokal.

```
procedure UpdateMutationRate;
begin
  if fitnessAvg-fitnessAvgOld>threshold then
    mutationRate := mutationRate * 0.95
  else
    mutationRate := mutationRate * 1.1;

  if mutationRate>mutationRateMax then
```

```
    mutationRate := mutationRateMax
  else if mutationRate<mutationRateMin then
    mutationRate := mutationRateMin;
end;
```

FUNGSI UJI

Pada makalah ini digunakan dua fungsi uji yang sudah diketahui nilai optimumnya [1].

Uji Coba 1

Minimumkan $f_1(x_1, x_2)$, $f_2(x_1, x_2)$:

$$f_1(x_1, x_2) = (x_1 - 2)^2 + (x_2 - 1)^2 + 2 \text{ dan} \quad (2)$$

$$f_2(x_1, x_2) = 9x_1 - (x_2 - 1)^2 \quad (2)$$

Dengan dua kendala non linier:

$$x_1^2 + x_2^2 - 255 \le 0 \text{ dan} \quad (3)$$

$$x_1 - 3x_2 + 10 \le 0 \quad (4)$$

Dengan batas pencarian:

$-20 \le x_1 \le 20$ dan $-20 \le x_2 \le 20$

Uji Coba 2

Minimumkan $f_1(x_1, x_2)$, $f_2(x_1, x_2)$:

$$f_1(x_1, x_2) = 4x_1^2 + 4x_2^2 \text{ dan} \quad (5)$$

$$f_1(x_1, x_2) = (x_1 - 5)^2 + (x_2 - 5)^2 \quad (6)$$

Dengan dua kendala non linier:

$$(x_1 - 5)^2 + x_2^2 - 25 \le 0 \text{ dan} \quad (7)$$

$$-(x_1 - 8)^2 - (x_2 + 3)^2 + 7.7 \le 0 \quad (8)$$

Dengan batas pencarian:
$-15 \le x_1 \le 30$ dan $-15 \le x_2 \le 30$

HASIL DAN PEMBAHASAN

Uji Pencapaian Daerah *Feasible*

Untuk mengetahui keoptimalan hasil *GA* adaptif dibandingkan *GA* biasa, masing-masing skenario percobaan dijalankan 10 kali, ukuran populas = 20, nilai *crossover rate* = 0,6 dan *mutation rate* (awal) = 0,2. Tabel 1 dan Tabel 2 menunjukkan iterasi (generasi) yang dibutuhkan untuk menghasilkan populasi yang keseluruhan individunya menempati daerah*feasible* dan menghasilkan solusi pareto optimal.

Dari Tabel 1 dan Tabel 2 didapatkan individu-individu pada *GA* adaptif lebih cepat untuk mencapai daerah *feasible*. Setelah berada pada daerah *feasible*, setiap individu secara adaptif lebih mengeksploitasi daerah tersebut untuk menghasilkan solusi *pareto optimal*.

Perilaku *GA* Adaptif pada Uji Coba 1

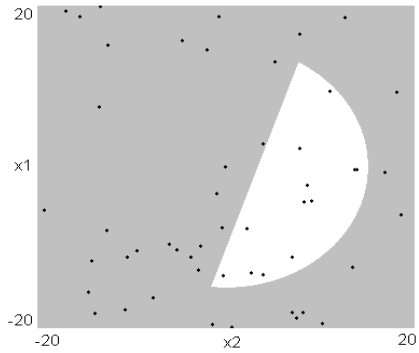
Daerah pencarian dan kendala digambarkan dalam Gambar 1. Area yang *feasible* diwarnai putih. Pada tahap inisialiasi populasi didapatkan individu-individu yang secara acak menempati daerah pencarian (Gambar 1a). Pada tahap ini terlihat banyak individu yang berada di luar daerah *feasible*. Plot nilai fungsi obyektif 1 (f_1) terhadap nilai fungsi obyektif 2 (f_2) menunjukkan sebaran populasi terhadap area perbatasan *pareto optimal* (Gambar 2).

Tabel 1. Iterasi Akhir Uji Coba 1.

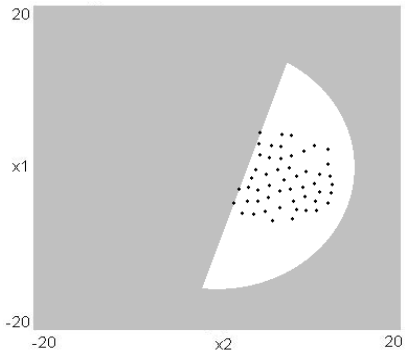
Percobaan	Iterasi Akhir GA Adaptif	Iterasi Akhir GA
1	20	40
2	19	34
3	30	37
4	25	42
5	40	58
6	34	35
7	39	49
8	20	45
9	33	57
10	29	41
Rata-Rata	28,9	43,8

Tabel 2. Iterasi Akhir Uji Coba 2.

Percobaan	Iterasi Akhir GA Adaptif	Iterasi Akhir GA
1	43	65
2	39	73
3	49	49
4	56	57
5	62	58
6	48	63
7	59	69
8	61	70
9	47	81
10	51	72
Rata-Rata	51,5	65,7

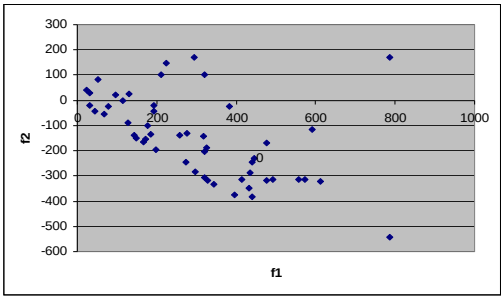


(a) awal iterasi

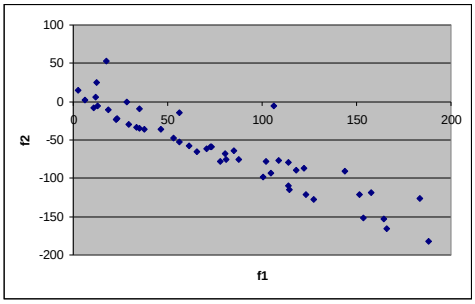


(b) akhir iterasi

Gambar 1. Sebaran Populasi Fungsi Uji ke-1.

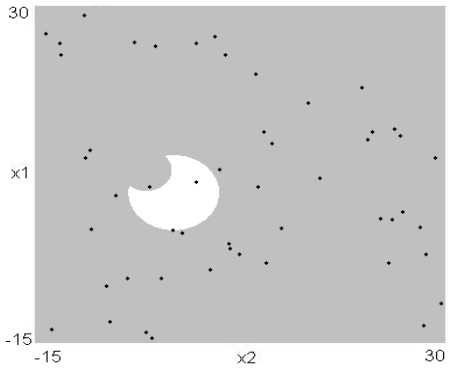


(a) awal iterasi

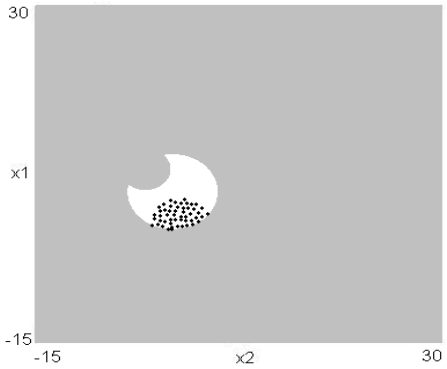


(b) akhir iterasi

Gambar 2. Sebaran Populasi Terhadap Batas *Pareto-optimal.*

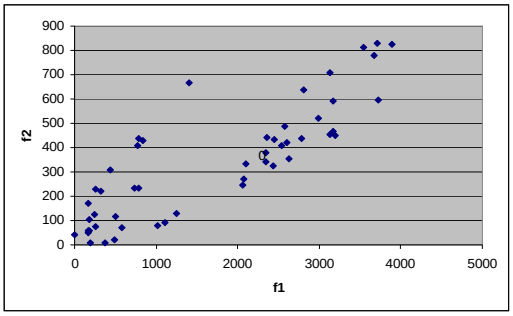


(a) awal iterasi

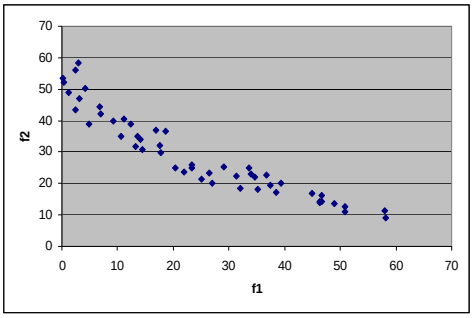


(b) akhir iterasi

Gambar 3. Sebaran Populasi Fungsi Uji ke-2.



(a) awal iterasi



(b) akhir iterasi

Gambar 4. Sebaran populasi terhadap batas *pareto-optimal*.

Pada generasi berikutnya (generasi 5-10), lebih banyak individu-individu yang menempati daerah *feasible*. Secara alami individu-individu yang berada jauh dari daerah *feasible* akan tereliminasi. Pada akhir generasi, semua individu menempati daerah *feasible* (Gambar 1b). Plot nilai fungsi obyektif 1 (f_1) terhadap nilai fungsi obyektif 2 (f_2) menunjukkan sebaran populasi yang menempati area perbatasan *pareto optimal* (Gambar 2b).

Perilaku GA Adaptif pada Uji Coba 2

Pada tahap inisialiasi populasi didapatkan individu-individu yang secara acak menyebar menempati daerah pencarian (Gambar 3a). Pada tahap ini terlihat banyak individu yang berada di luar daerah *feasible*.Pola yang sama juga ditunjukkan pada uji coba ke-2. Pada akhir generasi, semua individu menempati daerah *feasible* (Gambar 3b). Plot nilai fungsi obyektif 1 (f_1) terhadap nilai fungsi obyektif 2 (f_2) menunjukkan sebaran populasi yang menempati area perbatasan *pareto optimal* (Gambar 4b).

Perilaku Pengaturan *Mutation Rate* Secara Adaptif

Keseimbangan antara eksplorasi dan eksploitasi daerah pencarian merupakan hal penting dalam Algoritma Heuristis. Dengan operator *crossover* pada *GAs*, hal ini dapat dicapai. Pada tahap awal generasi terbentuk individu-individu yang sangat divergen seperti terlihat pada Gambar 1a dan 3a. Anak yang dihasilkan dari operator *crossover* berfungsi untuk melakukan proses eksplorasi daerah pencarian. Pada generasi-generasi berikutnya secara alami individu yang mempunyai *fitness* tertinggi akan bertahan dalam populasi dan secara perlahan terbentuk individu-individu yang semakin konvergen seperti terlihat pada Gambar 1b dan 3b. Anak yang dihasilkan pada tahap ini lebih berfungsi untuk melakukan proses eksploitasi daerah pencarian.

Penggunaan *mutation rate* yang konstan hanya dimaksudkan untuk menahan supaya tidak terjadi konvergensi dini. Pada kasus seperti ini mutasi hanya berfungsi untuk eksplorasi. Dengan menggunakan *mutation rate* yang adaptif, jika ada perbaikan nilai *fitness* rata-rata yang signifikan dibanding generasi sebelumnya maka dipastikan pada tahap ini proses *crossover* lebih berorientasi pada eksplorasi sehingga nilai *mutation rate* diturunkan. Hal ini dimaksudkan agar operator mutasi lebih berorientasi untuk eksploitasi. Sebaliknya jika tidak ada perbaikan nilai *fitness* rata-rata yang signifikan dibanding generasi sebelumnya maka dipastikan pada tahap ini proses *crossover* lebih berorientasi pada eksploitasi sehingga nilai *mutation rate* dinaikkan supaya proses eksplorasi tetap terjaga. Mekanisme yang sederhana ini terbukti dapat menjaga keseimbangan antara eksplorasi dan eksploitasi daerah pencarian sehingga proses pencarian solusi berjalan lebih cepat seperti ditunjukkan pada Tabel 1 dan 2.

SIMPULAN DAN SARAN

Penelitian ini menunjukkan bahwa *GAs* dengan pengkodean real dapat menghasilkan solusi *pareto optimal* pada masalah optimasi multiobyektif dengan kendala nonlinier.Pengaturan *mutation rate* secara adaptif dapat mempercepat proses pencarian titik optimum. Pada uji coba ke-1, *GAs* adaptif mencapai daerah *pareto optimum* rata-rata pada generasi ke-28,9, jauh lebih baik daripada *GAs* standar dengan rata-rata 43,8. Pada uji coba ke-2 pola yang sama juga terjadi *GAs* adaptif dengan rata-rata 51,5 dan *GAs* standar rata-rata 65,7.

Untuk pengembangan penelitian selanjutnya perlu dikembangkan penggabungan *GAs* dengan algoritma pencarian lain seperti Algoritma *Hill-Climbing* untuk melakukan perbaikan solusi lokal. *GAs* terdistribusi dapat digunakan untuk memelihara kumpulan solusi *pareto-optimal*. Dengan adanya kumpulan solusi *pareto-optimal*, manusia sebagai pengambil keputusan dapat lebih leluasa memilih solusi yang harus dijalankan.

DAFTAR PUSTAKA

[1] SarkerR, AbbassHA and KarimS, An Evolutionary Algorithm forConstrained Multiobjective Optimization Problems,In *The 5th Australasia-Japan JointWorkshop*.

University of Otago, Dunedin, New Zealand,pp. 1-10, 2001.

[2] Mahmudy WF, Optimasi Fungsi Tanpa Kendala Menggunakan Algoritma Genetika dengan Kromosom Biner dan Perbaikan Kromosom *Hill-Climbing*,*Kursor*, 4 (1) : 23-29, 2008.

[3] Mahmudy WF, Optimasi Fungsi Tak Berkendala Menggunakan Algoritma Genetika Terdistribusi dengan Pengkodean Real. In *Seminar Nasional Basic Science VI*, Universitas Brawijaya, Malang, hal. 1-5, 2009.

[4] Herrera F, LozanoM, and VerdegayJL, Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. *Artificial Intelligence Review*, 12: 265–319, 1998.

[5] Sleesongsom S, Multiobjective Optimization with Even Pareto Filter,In *Fourth Natural ComputationInternational Conference onNatural Computation*, pp. 92-96, 2008.

[6] Xiaohui H, and EberhartR, Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization. In Proceedings of *the 2002 Congress on Evolutionary Computation*, pp. 1677-1681, 2002.

[7] Doerner K, Gutjahr W, Hartl R, Strauss C, and Stummer C, Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection.*Annals of Operations Research*. 131(1): 79-99, 2004.

[8] Bandyopadhyay S,S. Saha, U. Maulik, and K. Deb, A Simulated Annealing-Based Multiobjective Optimization Algorithm: AMOSA.*Evolutionary Computation, IEEE Transactions on* 12(3): 269-283, 2008.

[9] Gen M, and ChengR, *Genetic Algorithms and Engineering Design*, New York: John Wiley & Sons, Inc. 1997.

[10] Gen M, and ChengR, *Genetic Algorithms and Engineering Optimization*, New York: John Wiley & Sons, Inc. 2000.

[11] Lozano M, and HerreraF, Fuzzy Adaptive Genetic Algorithms: Design, Taxonomy,*Soft Computing*. 7: 545–562. 2003.

[12] Lozano, M., F. Herrera, N. Krasnogor, and D. Molina, Real-Coded Memetic Algorithms with Crossover Hill-Climbing.*Evolutionary Computation* **12**(3): 273-302, 2004.

[13] Lozano M, HerreraF, KrasnogorN, and MolinaD, Real-Coded Memetic Algorithms With Crossover *Hill-Climbing*,*Evolutionary Computation* 12(3): 273-302, 2004.

[14] Jaszkiewicz A, Genetic Local Search for Multi-Objective Combinatorial Optimization,*European Journal of Operational Research*, 137(1) : 50-71, 2002.

[15] Deb K, Pratap A, Agarwal S, and Meyarivan T, A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II, Evolutionary Computation, *IEEE Transactions on*, 6(2) : 182-197, 2002.

[16] Baraka HA,Eid S, Kamal H, and Wahab AHA, Unified Chromosome Representation for Large Scale Problems, in Multiple Approaches to Intelligent Systems, ImamI, et al, Editors, Springer Berlin / Heidelberg,pp. 753-760, 2004.

[17] Radcliffe NJ, Equivalence Class Analysis of Genetic Algorithms,*Complex Systems*, 5(2) : 183–205, 1991.

[18] M¨uhlenbein H, and VoosenDS, Predictive Models for The Breeder Genetic Algorithm; Continuous Parameter Optimization,*Evolutionary Computation*,1 : 25–49, 1993.

[19] Michalewicz Z, *Genetic Algorithms + Data Structures = Evolution Programs*, Heidelberg: Springer-Verlag, 1996.