

MODIFIKASI *FLOWER POLLINATION ALGORITHM* DENGAN *REPLACEMENT* BERBASIS ILS: PERMASALAHAN *QUADRATIC BOUNDED KNAPSACK*

(Modified Flower Pollination Algorithm with ILS-Based Replacement: Quadratic Bounded Knapsack Problem)

Yona Eka Pratiwi^{1*)}, Mohamat Fatekurohman²⁾, Firdaus Ubaidillah³⁾

^{1,2,3)} Magister Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Jember, Jl. Kalimantan No. 37 Kampus Tegal Boto, Jember, Indonesia
e-mail: yonaep04@gmail.com¹, mfatekurohman.fmipa@unej.ac.id²
*)penulis korespondensi

Abstract. Optimization problems are the most interesting problems to discuss in mathematics. Optimization is used to modeling problems in various field to achieve the effectiveness and efficiency of the desired target. One of the optimization problems that are often encountered in everyday life is the selection and packaging of items with limited media or knapsack to get maximum profit. This problem is well-known as knapsack problem. There are various types of knapsack problems, one of them is quadratic bounded knapsack problem. In this paper, the authors proposed a new modified algorithm, which is Flower Pollination Algorithm (FPA) added with Iterated Local Search (ILS)-based Replacement mechanism. Furthermore, the implementation of the proposed algorithm, MFPA, is compared to the original FPA. Based on the results of this study, the proposed MFPA performs better and produces the best solution than the original algorithm on all data used. The advantage obtained by the MFPA algorithm is better and in accordance with the knapsack capacity. In addition, although the computational of the MFPA takes longer time than FPA with the same number of iterations, MFPA is able to find better solutions faster and able to escape from the local optimum.

Keywords: Flower Pollination Algorithm, Iterated Local Search, Knapsack, Optimization, Quadratic Bounded Knapsack.

1. Pendahuluan

Matematika merupakan salah satu bagian dari ilmu pengetahuan yang memiliki peranan penting dalam dunia teknologi dan perusahaan. Perkembangan yang semakin pesat seiring kemajuan teknologi mengakibatkan tingkat persaingan antar industri yang semakin ketat, sehingga perusahaan dituntut untuk memaksimalkan kinerja di berbagai bidang. Salah satunya adalah permasalahan optimasi yang sering dijumpai dalam kehidupan sehari-hari. Perusahaan sering mengalami kesulitan diantaranya yaitu pengemasan barang dengan keterbatasan media atau *knapsack* untuk mengangkut semua barang meskipun jumlah media penyimpanan lebih dari satu.

Permasalahan *knapsack* merupakan permasalahan tentang bagaimana cara memilih barang dari sekian banyak pilihan di mana setiap barang tersebut mempunyai berat dan keuntungan masing-masing dengan memperhatikan kapasitas dari media penyimpanan, sehingga dari pemilihan barang tersebut didapatkan keuntungan maksimal. Permasalahan *knapsack* terdiri dari beberapa persoalan antara lain *binary knapsack*, *bounded knapsack* dan *unbounded knapsack*. Pembagian tersebut berdasarkan pola penyimpanan barang dengan nilai dan bobot yang beragam. Permasalahan *binary knapsack* atau *knapsack 0-1* merupakan permasalahan *knapsack* dimana barang yang dimasukkan ke dalam media penyimpanan harus dimasukkan semua (1) atau tidak sama sekali (0). Permasalahan *bounded knapsack* merupakan permasalahan *knapsack* dimana setiap barang tersedia sebanyak n unit dan jumlah barang yang dimasukkan ke dalam media penyimpanan terbatas, bisa dimasukkan sebagian atau seluruhnya. Permasalahan *unbounded knapsack* merupakan permasalahan *knapsack* dimana setiap barang tersedia lebih dari satu unit dan jumlah barang yang dimasukkan ke dalam media penyimpanan tak terbatas [7].

Berdasarkan permasalahan dasar yang ada pada *knapsack*, muncul beberapa variasi permasalahan *knapsack* yaitu *multiobjective knapsack*, *multiple constraints knapsack*, *multiple knapsack* dan *quadratic knapsack*. Permasalahan *multiobjective knapsack* merupakan permasalahan *knapsack* yang memiliki fungsi tujuan lebih dari satu untuk memaksimalkan keuntungannya. Permasalahan *multiple constraints knapsack* adalah permasalahan *knapsack* yang memiliki kendala lebih dari satu untuk memaksimalkan keuntungannya. Permasalahan *multiple knapsack* merupakan permasalahan *knapsack* yang memiliki media penyimpanan lebih dari satu dimana semua item harus dikemas untuk memaksimalkan keuntungannya. Dan permasalahan *quadratic knapsack* merupakan permasalahan *knapsack* yang tujuannya memaksimalkan fungsi objektif dalam bentuk kuadratik untuk kendala kapasitas biner dan linier [6].

Permasalahan optimasi termasuk permasalahan *knapsack* dapat diselesaikan menggunakan beberapa metode atau algoritma. Salah satu algoritma yang sering digunakan adalah algoritma metaheuristik, banyak penelitian yang menggunakan algoritma ini karena merupakan cara yang efisien untuk menghasilkan suatu solusi. Algoritma metaheuristik merupakan algoritma yang dibuat untuk menyelesaikan permasalahan optimasi melalui pendekatan yang banyak terinspirasi dari alam, seperti biologi, fisika atau perilaku hewan [4].

Algoritma metaheuristik yang pernah digunakan untuk menyelesaikan permasalahan optimasi adalah *Flower Pollination Algorithm*. FPA pertama kali dikenalkan oleh Yang dalam penelitiannya yang terinspirasi dari proses penyerbukan bunga. Penelitian tersebut membahas tentang perbandingan kinerja algoritma FPA dengan algoritma Genetika (GA) dan algoritma *Particle Swarm Optimization* (PSO). Hasil penelitian yang didapat menunjukkan FPA lebih efisien daripada GA dan PSO [8]. Adamuthe melakukan penelitian tentang penyelesaian masalah *knapsack 0-1* menggunakan *Flower Pollination Algorithm*. Hasil penelitian Adamuthe menunjukkan bahwa FPA dengan variasi

perkawinan silang adalah hasil paling optimal dibandingkan dengan dua variasi lainnya. Akan tetapi, hasil dari variasi ini memiliki kelemahan yaitu mencapai nilai optimal hanya untuk data berskala kecil [1].

Di samping algoritma yang telah diuraikan di atas, akhir-akhir ini jika menemukan kelemahan pada sebuah algoritma banyak para peneliti memodifikasi algoritma atau menggabungkan (*hybrid*) algoritma dengan algoritma lainnya agar algoritma tersebut menjadi lebih baik. ILS merupakan salah satu yang cukup sering digunakan dan memberikan peningkatan performa algoritma yang signifikan. Beberapa algoritma yang di *hybrid* dengan ILS ini adalah MMBO [3], GA dan ILS [5]. Oleh karena itu, pada penelitian ini penulis ingin memperbaiki kelemahan algoritma FPA dengan menambahkan ILS. Adeyemo meneliti permasalahan knapsack 0-1 dengan teknik metaheuristik. Algoritma Genetika dan PSO dibandingkan dengan *Simulated Annealing* dan ILS, hasilnya *Simulated Annealing* dan ILS lebih baik dengan sedikit perbedaan dalam efisiensi dan waktu komputasi daripada algoritma lainnya [2].

Berdasarkan uraian di atas, penulis tertarik untuk meneliti permasalahan baru yaitu *quadratic bounded knapsack with multiple constraints*. Masalah ini muncul ketika didapatkan fungsi tujuan berbentuk kuadrat dengan fungsi kendala lebih dari satu dan batas minimal-maksimal diketahui. Permasalahan tersebut disesuaikan dengan kehidupan sehari-hari, misalnya harga barang yang dapat berubah sewaktu-waktu. Selain itu, penulis ingin memodifikasi dan menggabungkan (*hybrid*) algoritma FPA untuk mengatasi kekurangan yang ditemukan di penelitian-penelitian sebelumnya disebut *Modified Flower Pollination Algorithm* (MFPA). Konsep modifikasi ini didasarkan pada asumsi bahwa dimungkinkan penyerbukan mengalami kegagalan yang direpresentasikan dengan solusi terburuk. Oleh karena itu, diperlukan adanya mekanisme pergantian (*replacement*) untuk menjaga jumlah solusi konstan. Solusi baru yang ditambahkan tersebut diasumsikan sebagai bunga baru. Langkah dari mekanisme pergantian ini, penulis akan mengadopsi langkah-langkah ILS. Penelitian yang akan dilakukan menggunakan data berupa data simulasi. Data yang dibuat akan disesuaikan berdasarkan keadaan sesungguhnya dan sesuai dengan permasalahan penelitian yaitu *quadratic bounded knapsack with multiple constraints*. Dalam penelitian ini, penggunaan data simulasi dimaksudkan agar dapat merepresentasikan jenis data yang lebih bervariasi dan bersifat universal.

Selanjutnya hal menarik yang akan menjadi bahasan pada penelitian ini adalah bagaimana penerapan algoritma FPA dan MFPA pada penyelesaian permasalahan *quadratic bounded knapsack*. Peneliti juga akan membandingkan hasil dari solusi yang diberikan antara kedua algoritma pada permasalahan tersebut. Tujuan penelitian ini adalah menganalisis penerapan dan mengkaji perbandingan algoritma FPA dan MFPA pada penyelesaian permasalahan *quadratic bounded knapsack*. Pada penelitian ini juga diberikan batasan masalah yaitu volume penempatan barang yang diabaikan. Volume yang dimaksud adalah panjang, lebar dan tinggi suatu barang yang diasumsikan bahwa

semua barang tersebut dapat dimasukkan ke media penyimpanan jika besar volume memenuhi kendala kapasitas ruang.

Permasalahan *quadratic bounded knapsack with multiple constraints* merupakan variasi permasalahan berdasarkan parameternya dimana terdapat jumlah ketersediaan barang dari setiap jenisnya dan memiliki kendala lebih dari satu. Tujuan permasalahan *quadratic bounded knapsack with multiple constraints* adalah memilih subset unit yang bobot keseluruhan tidak melebihi kapasitas *knapsack* (C) yang diberikan sehingga dapat menentukan jumlah dari setiap jenis barang dengan memperoleh total keuntungan yang maksimal serta memenuhi semua kendala. Kendala dari permasalahan tersebut yaitu cakupan kapasitas media penyimpanan berupa berat dan ruang serta biaya atau modal yang disediakan. Contoh dari permasalahan tersebut adalah diasumsikan setiap jenis barang memiliki batasan ketersediaan jumlah minimal maupun maksimal yang harus dibeli. Batasan tersebut memiliki tujuan agar jumlah minimal barang untuk mendapatkan keuntungan maksimal dan tidak melebihi kapasitas muatan atau biaya.

Beberapa penjelasan mengenai *quadratic bounded knapsack with multiple constraints* antara lain sebagai berikut setiap jenis barang memiliki jumlah ketersediaan (m_j), keuntungan setiap jenis barang dihitung atau diperoleh dari perkalian jumlah terpilih jenis barang tersebut (y_j) dengan profit satuannya (p_{jj}), terdapat keuntungan tambahan p_{ij} untuk setiap pasangan jenis barang i dan j . $i < j$, jika jumlah terpilih jenis barang i dan jenis barang j keduanya tidak sama dengan nol (0) dan terdapat tiga kendala yang harus terpenuhi, yaitu berat, volume dan modal.

Berdasarkan uraian di atas, maka secara matematis permasalahan *quadratic bounded knapsack with multiple constraints* dapat didefinisikan sebagai berikut:

Fungsi tujuan:

$$\text{Maksimalkan } Z = \sum_{j=1}^n y_j p_{jj} + \sum_{i=1}^{n-1} \sum_{j=i+1}^n t_i t_j p_{ij} \quad (1)$$

Kendala:

$$\sum_{j=1}^n y_j w_j \leq C \quad (2)$$

$$\sum_{j=1}^n y_j v_j \leq S \quad (3)$$

$$\sum_{j=1}^n y_j b_j \leq M \quad (4)$$

$$y_j \in \{0, 1, \dots, m_j\}, j = 1, 2, \dots, n \quad (5)$$

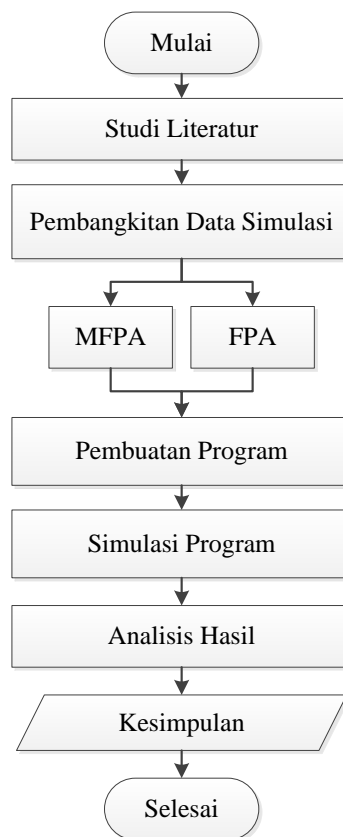
$$t_i \& t_j = \{0, \text{ jika } y_j = 0 \text{ 1, jika lainnya} \quad (6)$$

dengan nilai optimum fungsi tujuan atau total keuntungan (Z), banyaknya jenis barang (n), *profit* atau keuntungan barang jenis i dan j (p_{ij}), variabel keputusan jumlah barang

jenis i, j yang dimasukkan ke media penyimpanan artinya jika 1 jika dipilih atau 0 jika tidak dipilih (y_i, y_j), variabel keputusan jumlah barang jenis i, j yang dimasukkan ke media penyimpanan artinya jika 1 jika dipilih dan mendapat profit tambahan atau 0 jika tidak dipilih dan tidak mendapat profit tambahan (t_i, t_j), *weight* atau berat barang jenis j (w_j), volume barang jenis j dengan dimensi barang diabaikan (v_j), harga beli barang jenis j (b_j), jumlah ketersediaan barang jenis j (m_j), kapasitas berat media penyimpanan satuan kg (C), kapasitas ruang media penyimpanan satuan cm^3 (S), dan modal (M).

2. Metodologi

Dalam penyelesaian masalah pada penelitian ini akan diselesaikan menggunakan beberapa tahapan langkah. Adapun langkah-langkah yang digunakan untuk penyelesaian permasalahan dapat dilihat pada Gambar 1 berikut.



Gambar 1. Skema metode penelitian

Penelitian yang digunakan peneliti kali ini adalah penelitian berjenis eksperimental. Penelitian ini menggunakan data simulasi yang terdiri dari data sejumlah barang, data kendaraan dan modal. Data barang yang digunakan berupa nama barang, batas minimal, batas maksimal, berat, volume, harga beli, harga jual dan keuntungan dari setiap jenis barang.

Pembangkitan data simulasi yang digunakan selanjutnya dibangkitkan menggunakan *software*. Program pembangkitan data acak digunakan untuk membangkitkan data simulasi diantaranya yaitu batas minimal jumlah barang (lb_j), batas maksimal jumlah barang (ub_j), berat, volume, harga beli dan harga jual barang. Ukuran data yang digunakan yaitu 20, 40, 60, 80 dan 100 jenis barang. Masing-masing dari jenis barang tersebut terdapat 5 data yang berbeda yaitu diantaranya interval jumlah barang, kapasitas berat, kapasitas volume dan modal. Program pembangkitan data dibuat dalam *script* dengan beberapa aturan untuk menyesuaikan nilai masing-masing jenis data dan mengidentifikasi data sesuai dengan permasalahan *quadratic bounded knapsack*.

Permasalahan yang menjadi objek pada penelitian ini akan diolah menggunakan algoritma metaheuristik yang meliputi algoritma *Flower Pollination Algorithm* (FPA) dan *Iterated Local Search* (ILS). Adapun bentuk solusi dari masing-masing algoritma metaheuristik yang harus dilakukan adalah sebagai berikut.

- 1) Memasukkan data penelitian meliputi jumlah ketersediaan barang (m_j), berat barang (w_j), volume barang (v_j), harga beli barang (b_j) dan matriks profit (p), serta menentukan batas kendalanya yang meliputi kapasitas berat *knapsack* (C), kapasitas ruang *knapsack* (S) dan modal (M).
- 2) Menentukan nilai parameter yang antara lain: ukuran populasi (N_{pop}), jumlah iterasi (I_{max}), *switch probability* (p), *stepsize* (γ), persentase kandidat terburuk (P_w), dan jumlah iterasi ILS (NC).
- 3) Membangkitkan vektor nilai awal ($x_i; 1 \leq i \leq N_{pop}$) sebagai kandidat solusi, secara acak pada interval $[0, 1]$.

$x_i = [x_{i1}, x_{i2}, \dots, x_{iD}]$ dengan D adalah banyaknya jenis barang

- 4) Mengubah nilai (x) ke bentuk vektor variabel keputusan (y) sebagai jumlah barang terpilih. Untuk permasalahan *bounded knapsack*, konversi vektor x ke vektor y dapat menggunakan Persamaan (7) berikut.

$$y_{i,j} = \text{round}(x_{i,j} * m_j), \quad j = 1, 2, \dots, D \quad (7)$$

- 5) Melakukan pengecekan kendala dari setiap kandidat solusi.

Setiap solusi masing-masing *knapsack* harus memenuhi kendala berikut:

$$\sum_{j=1}^n w_j y_j \leq C, \quad (8)$$

$$\sum_{j=1}^n v_j y_j \leq S, \quad (9)$$

$$\sum_{j=1}^n b_j y_j \leq M, \quad (10)$$

$$y_j \in \{0, 1, \dots, m_j\}, j = 1, 2, \dots, n \quad (11)$$

Apabila dari pengecekan tersebut ditemukan ada kandidat solusi yang tidak memenuhi kendala, maka kandidat solusi tersebut harus dipenalti menggunakan Persamaan (12) berikut.

$$x_{ik} = \left| x_{ik} - \frac{1}{m_k} \right| \quad (12)$$

dengan i adalah indeks kandidat solusi yang tidak memenuhi kendala dan k adalah indeks jenis barang yang harus dikurangi (dipilih acak). Langkah penalti harus

diulang sampai kandidat solusi memenuhi kendala.

- 6) Menghitung total keuntungan (fungsi tujuan)

Nilai keuntungan dari setiap solusi dihitung berdasarkan total profitnya.

$$Z = \sum_{j=1}^n y_j p_{jj} + \sum_{i=1}^{n-1} \sum_{j=i+1}^n t_i t_j p_{ij} \quad (13)$$

dengan t_i, t_j bernilai nol (0) jika tidak ada barang i, j yang terpilih, dan bernilai satu (1) jika ada barang i, j yang terpilih.

- 7) Menyimpan kandidat solusi terbadik dari populasi sebagai g^* .

- 8) Memperbarui nilai (x) setiap kandidat solusi

Langkah-langkah perbaruan nilai (x) adalah sebagai berikut.

- a) Membangkitkan nilai random $rand()$

- b) Jika nilai $rand()$ kurang dari *switch probability* (p), maka dilakukan perbaruan nilai (x) dengan *global pollination* menggunakan Persamaan (14) berikut.

$$x_i = x_i + \gamma L(\lambda)(g^* - x_i) \quad (14)$$

dengan $L(\lambda)$ adalah vektor random dengan distribusi Levy.

- c) Namun, jika nilai $rand()$ lebih besar atau sama dengan *switch probability* (p), maka dilakukan perbaruan nilai (x) dengan *local pollination* menggunakan Persamaan (15) berikut.

$$x_i = x_i + \epsilon(x_j - x_k) \quad (15)$$

dengan ϵ adalah nilai random dan x_j, x_k adalah kandidat solusi yang dipilih acak dari populasi.

- 9) Memeriksa solusi baru dan kendala

Pada langkah ini, kandidat solusi baru diperiksa apakah nilai-nilainya berada pada ruang pencarian (interval [0, 1]). Jika terdapat nilai yang keluar dari batas tersebut, maka kandidat solusi harus ditransformasi menggunakan Persamaan (16) berikut.

$$x_i = \frac{x_i - (x_i)}{(x_i) - \min(x_i)} \quad (16)$$

Jika kandidat solusi telah berada pada ruang pencarian, selanjutnya dilakukan pengecekan kendala seperti pada langkah 5).

- 10) Menghitung total keuntungan (fungsi tujuan) dari kandidat solusi baru

Setiap kandidat solusi baru yang telah memenuhi kendala dapat dihitung total keuntungannya seperti pada langkah 6). Kandidat solusi baru yang memiliki total keuntungan lebih baik dari kandidat solusi sebelumnya diterima dan menggantikan. Namun jika kandidat solusi baru memiliki total keuntungan yang lebih buruk, maka kandidat tersebut ditolak dan tidak digunakan pada iterasi berikutnya.

- 11) Memperbarui kandidat solusi terbaik (g^*) dari populasi.

- 12) Untuk algoritma FPA, pada langkah ini dilakukan pengecekan jumlah iterasi. Apabila jumlah iterasi sudah tercapai, maka perhitungan dihentikan. Namun, jika belum tercapai, perhitungan kembali ke langkah 8). Untuk algoritma MFPA (FPA dan ILS), perhitungan diteruskan pada langkah selanjutnya.

- 13) Menghitung jumlah dan menentukan kandidat terburuk berdasarkan persentase kandidat terburuk (P_w) dan total keuntungan kandidat solusi.

14) Untuk kandidat-kandidat terburuk yang terpilih, dilakukan langkah ILS sebagai berikut.

- a) Menerapkan mekanisme *local search* dengan memilih salah satu jenis barang (j) secara random. Kemudian jumlah barang ditambah, atau dapat menggunakan Persamaan (17) berikut.

$$x_{ij} = x_{ij} + \frac{1}{m_j} \quad (17)$$

Selanjutnya diperiksa kendala dan dihitung total keuntungannya. Jika hasilnya lebih baik maka diterima.

- b) Menerapkan mekanisme *perturbation* menggunakan Persamaan (18) berikut.

$$x_i = x_i + \text{rand} \cdot (g^* - x_j) \quad (18)$$

dengan x_j adalah kandidat solusi yang dipilih acak dari populasi.

- c) Menerapkan mekanisme *local search* pada hasil langkah sebelumnya dan kemudian diperiksa kendalanya serta dihitung total keuntungannya. Jika hasilnya lebih baik dari sebelumnya maka solusi baru diterima.

- d) Langkah b dan c diulangi hingga jumlah iterasi ILS (NC) dicapai.

15) Memperbarui kandidat solusi terbaik (g^*) dari populasi untuk algoritma MFPA.

16) Melakukan pengecekan kriteria pemberhentian. Apabila jumlah iterasi (I_{max}) telah dicapai, maka iterasi dihentikan. Namun apabila belum tercapai, perhitungan MFPA kembali ke langkah 8).

Setelah penerapan algoritma selesai dikerjakan, dilanjutkan dengan pembuatan dan simulasi program. Percobaan dijalankan sebanyak sepuluh kali karena algoritma metaheuristik memuat nilai acak atau bersifat stokastik yang memungkinkan solusi algoritma dapat berubah-ubah. Uji parameter algoritma MFPA dan FPA terdiri dari enam parameter yaitu ukuran populasi (N_{pop}), presentase gagalnya penyerbukan (Pw), jumlah iterasi ILS NC), probabilitas sakelar (p), faktor penskalaan ukuran langkah (γ) dan maksimal iterasi (I_{max}). Selanjutnya, program diujikan untuk menyelesaikan keseluruhan data penelitian. Langkah selanjutnya yaitu analisis hasil lalu penarikan kesimpulan.

3. Hasil dan Pembahasan

Pada subbab ini akan diuraikan mengenai hasil, penerapan permasalahan *quadratic bounded knapsack* menggunakan data simulasi serta pembahasannya. Pada bagian pembahasan akan dijelaskan tentang pengaruh parameter, perbandingan algoritma MFPA dan FPA, serta hasil terbaik dari algoritma tersebut. Penyelesaian tersebut dilakukan menggunakan bantuan *software* MATLAB yang dijalankan pada laptop dengan CPU Intel(R) Core(TM) i7-4510U @2,00GHz, RAM 4GB dan 64-bit OS. Adapun hasil penelitian algoritma FPA dan MFPA yang sudah dilakukan sebagai berikut.

3.1 Uji parameter

Program penerapan algoritma FPA dan MFPA yang telah dibuat diujikan pada data yang telah dikumpulkan. Pada penelitian ini, percobaan dilakukan sesuai data yang diambil sebanyak 25 macam data. Dari 25 data yang digunakan terdiri dari kombinasi ketersediaan barang yaitu dengan interval 1-5, 1-10, 1-15, 1-20 dan 1-25. Setiap nilai parameter tersebut diuji dengan jumlah populasi 100 dan maksimum iterasi 1000.

Pada uji parameter γ , nilai p yang digunakan adalah 1. Dari setiap nilai parameter yang digunakan, program simulasi dijalankan sebanyak sepuluh kali. Hasil terbaik yang didapatkan dari uji parameter γ adalah 0,5. Parameter γ memiliki pengaruh terhadap rata-rata keuntungan pada algoritma FPA.

Selanjutnya untuk uji parameter p , nilai parameter lain yang digunakan yaitu $\gamma = 0.5$ dan $NC = P_w = 0$. Dari setiap nilai parameter yang digunakan, program simulasi dijalankan sebanyak lima kali untuk masing-masing algoritma. Dapat diketahui bahwa semakin besar nilai p maka semakin besar pula rata-rata keuntungannya. Akan tetapi, dari interval uji yang diberikan nilai terbaik yang diperoleh parameter p adalah 0,8. Parameter p berfungsi untuk mengukur tingkat terjadinya suatu kejadian acak yaitu penyerbukan bunga itu sendiri. Parameter p memiliki pengaruh terhadap rata-rata keuntungan pada algoritma FPA dan algoritma MFPA terlihat dari hasil fluktuatif seiring perubahan nilai parameter.

Uji parameter P_w dan NC , nilai parameter lain yang digunakan adalah $\gamma = 0,5$ dan $p = 0,8$. Dari setiap nilai parameter yang digunakan, program simulasi dijalankan sebanyak sepuluh kali. Ketika nilai P_w kecil nilai NC sangat berpengaruh untuk rata-rata keuntungan. Akan tetapi, saat nilai $P_w \geq 0,5$ nilai NC sudah tidak berpengaruh signifikan. Dalam hal ini, analisis waktu komputasi juga perlu diperhatikan. Dan hasilnya semakin besar nilai P_w , maka waktu komputasi akan semakin lama karena semakin banyak yang di proses oleh ILS. Hal tersebut juga berlaku untuk nilai NC , bahwa semakin besar nilai NC juga semakin lama waktu komputasi yang diperlukan karena iterasi ILS semakin banyak. Jadi, dapat diambil kesimpulan nilai terbaik yang diuji sebagai simulasi akhir adalah saat $P_w = 0,8$ dan $NC = 10$.

Tabel 1. Simulasi Akhir Profit Terbaik FPA dan MFPA

No	Data	FPA		MFPA	
		Profit Terbaik	Rata-Rata	Profit Terbaik	Rata-Rata
1	20A	1076000	1074700	1076000	1076000
2	20B	1440000	1439600	1447000	1447000
3	20C	2730000	2726800	2733000	2733000
4	20D	3584000	3577000	3590000	3589700
5	20E	4194000	4187400	4201000	4200700
6	40A	2286000	2273375	2325000	2323000
7	40B	3943000	3926200	4021000	4017300
8	40C	4405000	4377300	4516000	4508100
9	40D	8100000	8068300	8265000	8259300
10	40E	6566000	6523700	6695000	6691000
11	60A	2845000	2832300	2986000	2980100
12	60B	5452000	5412700	5723000	5709700
13	60C	7281000	7223700	7734000	7717100
14	60D	9652000	9574200	10065000	10029400
15	60E	12736000	12654700	13294000	13255900
16	80A	4220000	4193900	4487000	4469300
17	80B	7148000	7102300	7673000	7651500
18	80C	10692000	10645400	11438000	11387400
19	80D	15100000	15038200	15898000	15857400
20	80E	18346000	18211000	19378000	19313400
21	100A	4741000	4704000	5022000	4986600
22	100B	9757000	9712100	10391000	10355300
23	100C	12284000	12227400	13396000	13328600
24	100D	15800000	15755900	17219000	17093100
25	100E	17913000	17821900	19227000	19172500

3.2 Simulasi akhir

Setelah uji parameter selesai, selanjutnya dilakukan simulasi akhir untuk menguji algoritma FPA dan MFPA dalam penyelesaian 25 data permasalahan *quadratic bounded knapsack*. Nilai parameter-parameter yang digunakan di simulasi akhir ini didasarkan pada hasil uji parameter, yaitu nilai yang mampu menghasilkkan atau meningkatkan solusi menjadi lebih baik. Nilai parameter tersebut antara lain: $N = 100$; $I_{max} = 10000$, $\gamma = 0,5$; $p = 0,8$; $P_w = 0,5$ dan $NC = 10$. Hasil simulasi akhir yang didapatkan dari uji parameter terbaik untuk masing-masing algoritma disajikan dalam Tabel 1.

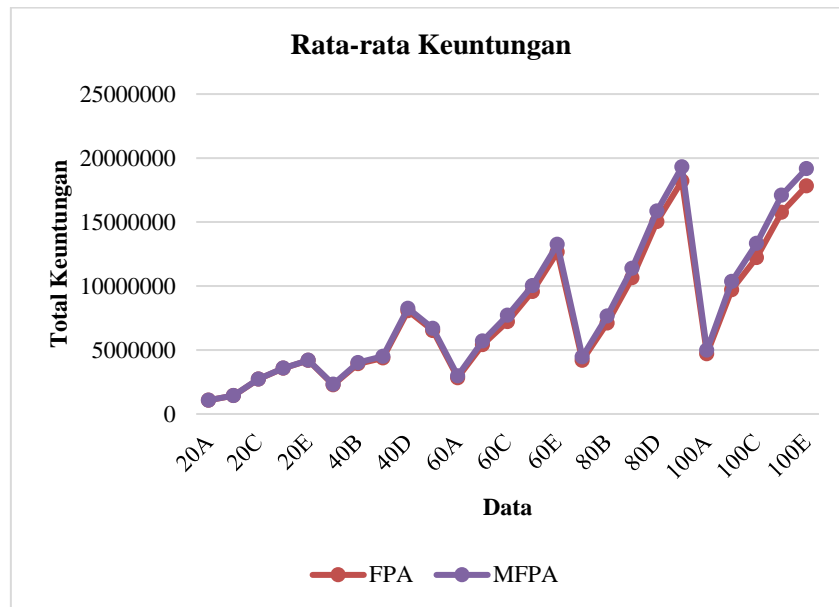
Tabel 2. Simulasi Akhir Iterasi Konvergen dan Waktu Komputasi FPA dan MFPA

No	Data	FPA		MFPA	
		Rata-rata Iterasi Konvergen	Rata-rata Waktu Komputasi	Rata-rata Iterasi Konvergen	Rata-rata Waktu Komputasi
1	20A	775	7,1145	152	36,8741
2	20B	848	6,2525	311	30,2710
3	20C	784	6,5608	269	30,2186
4	20D	799	6,6812	408	33,1490
5	20E	898	8,0139	531	35,6400
6	40A	918	11,2071	881	56,6253
7	40B	940	12,8279	873	61,2865
8	40C	917	12,8223	876	61,5946
9	40D	952	16,2973	924	73,7340
10	40E	954	15,7444	913	61,7161
11	60A	950	18,9825	936	92,6933
12	60B	941	22,7374	949	112,3042
13	60C	925	24,9703	981	112,5489
14	60D	947	26,5475	963	111,4062
15	60E	970	33,3961	990	126,2825
16	80A	916	27,4652	952	144,7927
17	80B	895	32,8927	978	163,8601
18	80C	898	35,8582	980	171,7827
19	80D	935	39,3859	978	176,8052
20	80E	924	43,2826	988	193,2795
21	100A	864	36,9576	964	192,1292
22	100B	942	44,0092	983	233,5686
23	100C	943	47,2491	988	215,6645
24	100D	959	52,5439	986	226,2839
25	100E	961	59,3272	990	251,634

Selanjutnya, dari hasil simulasi akhir diperoleh rata-rata iterasi konvergen (*iteration of non-improved solution*) dan rata-rata waktu komputasi (*exexution time*) dari program yang dijalankan sebanyak sepuluh kali. Hasil dari rata-rata iterasi konvergen dan waktu komputasi untuk algoritma FPA dan algoritma MFPA disajikan dalam Tabel 2.

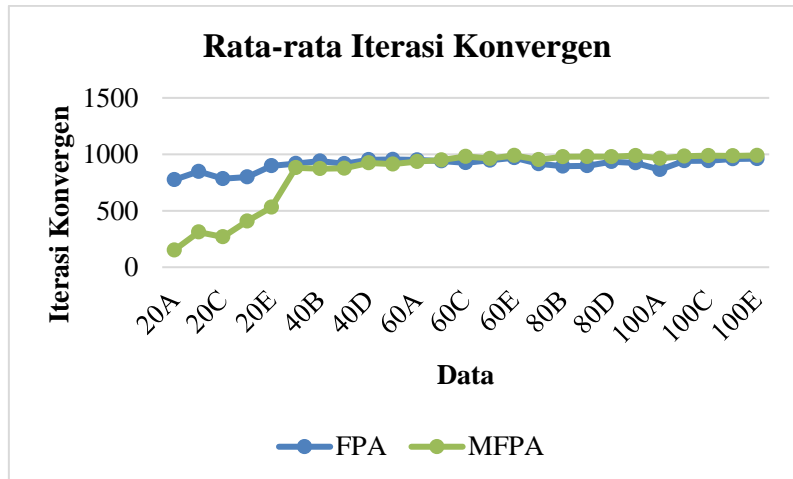
Berdasarkan hasil simulasi akhir (Tabel 1) yang telah dilakukan, dihitung profit terbaik dari semua data yang sudah ditentukan menggunakan parameter-parameter terbaik yang sudah diujikan. Untuk setiap data simulasi, program dijalankan sebanyak sepuluh kali. Dapat diketahui bahwa algoritma MFPA yang diajukan mampu memberikan solusi yang lebih baik dibandingkan algoritma FPA. Dari 25 data *quadratic bounded knapsack* diperoleh hasil bahwa semakin besar data yang digunakan maka semakin besar pula keuntungan yang didapatkan. Dengan adanya perbedaan kapasitas berat, kapasitas volume dan modal yang disajikan pada setiap data juga dapat mempengaruhi keuntungan

yang akan didapatkan. Perbandingan keuntungan yang didapatkan oleh algoritma FPA dan MFPA dapat dilihat pada Gambar 2.



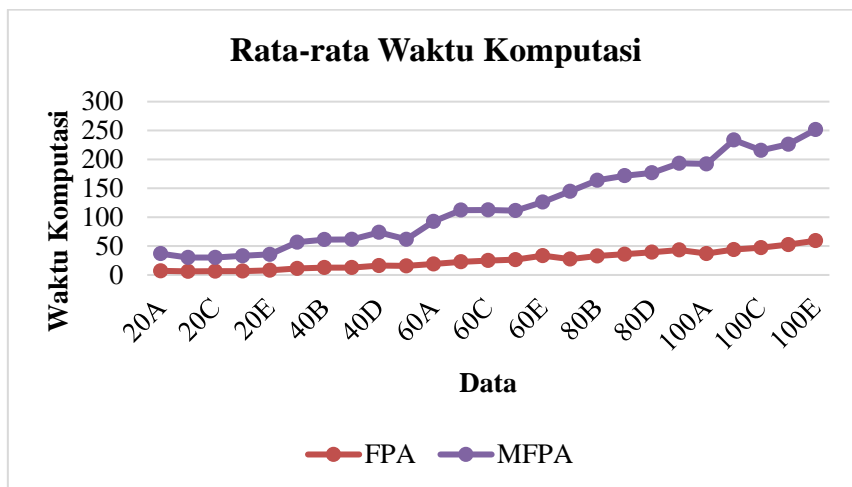
Gambar 2. Grafik profit terbaik FPA dan MFPA

Selanjutnya, berdasarkan simulasi akhir dengan kombinasi nilai yang digunakan (lihat Tabel 1), dapat diketahui bahwa semakin besar data yang digunakan maka semakin cepat algoritma dapat menemukan solusi yang lebih baik atau cepat memenuhi batas kekonvergenannya. Iterasi konvergen merupakan iterasi yang menandakan algoritma tidak mampu menemukan solusi lebih baik hingga iterasinya mencapai batas maksimal yang ditentukan. Berdasarkan rata-rata iterasi konvergen, dapat diketahui bahwa algoritma MFPA mampu menghindari terjebak lokal optimum. Hal ini terlihat dari nilai rata-rata iterasi yang selalu mendekati batas iterasinya. Berbeda dengan algoritma FPA yang rata-rata iterasi konvergenya relatif mendekati batas maksimal dari 25 data yang diberikan. Grafik rata-rata iterasi konvergen dapat dilihat pada Gambar 3.



Gambar 3. Iterasi konvergen FPA dan MFPA

Berdasarkan iterasi konvergen dan waktu komputasi algoritma yang disajikan pada Tabel 2 dapat diketahui bahwa semakin besar data yang diselesaikan, semakin besar pula rata-rata iterasi konvergen dan waktu komputasi dari algoritma FPA dan algoritma MFPA. Waktu komputasi algoritma MFPA relatif lebih besar dibandingkan dengan algoritma FPA seiring dengan ukuran data yang semakin besar. Grafik rata-rata waktu komputasi algoritma FPA dan MFPA dapat dilihat pada Gambar 4.



Gambar 4. Waktu komputasi FPA dan MFPA

Berdasarkan hasil penelitian yang telah diperoleh, dapat dikatakan bahwa dengan adanya mekanisme pergantian untuk menjaga jumlah solusi konstan dari algoritma FPA ke algoritma MFPA dengan sedikit modifikasi, mampu meningkatkan kualitas solusi yang dihasilkan secara signifikan. Namun, dengan konsekuensi waktu komputasi yang relatif lebih lama. Oleh karena itu, mekanisme pergantian menggunakan langkah-langkah ILS sangat membantu untuk mempercepat algoritma FPA mencapai kekonvergenannya. Sehingga dapat ditarik kesimpulan bahwa algoritma MFPA yang diajukan dapat dinilai lebih baik dalam menentukan solusi optimal.

4. Kesimpulan

Berdasarkan hasil dan pembahasan pada sub bab sebelumnya, didapatkan kesimpulan:

1. Permasalahan *quadratic bounded knapsack* dapat diselesaikan dengan menerapkan algoritma FPA dan algoritma MFPA. Dari sejumlah data yang telah disajikan, algoritma FPA dan algoritma MFPA mampu menghindari terjebak di lokal optimum. Meskipun dari hasil yang diperoleh mengalami keuntungan yang fluktuatif. Solusi keuntungan terbaik diperoleh dengan input beberapa parameter yang dikombinasikan diantaranya banyaknya kandidat solusi (N) = 100; *switch probability* (p) = 0,8; *stepsize* (γ) = 0,5; persentase kandidat terburuk (P_w) = 0,5; jumlah iterasi ILS (NC) = 10; dan banyaknya iterasi (I_{max}) = 1000.
2. Algoritma MFPA dari sisi pencarian nilai optimum lebih baik dibandingkan dengan algoritma FPA karena menghasilkan total keuntungan yang lebih besar. Selain itu, algoritma MFPA lebih cepat konvergen daripada algoritma FPA. Meskipun dari segi waktu komputasi, algoritma MFPA membutuhkan waktu yang lebih lama dari algoritma FPA. Algoritma MFPA dapat dikatakan efektif dalam permasalahan *quadratic bounded knapsack*, karena hasil yang diperoleh lebih baik dibandingkan dengan algoritma FPA.

Ucapan Terima Kasih

Atas penyusunan makalah ini, kami mengucapkan terima kasih kepada seluruh pembimbing dan seluruh anggota Laboratorium Matematika Terapan dan Komputasi. Selain itu penulis juga mengucapkan terima kasih kepada jurusan Matematika fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Daftar Pustaka

Pustaka diurutkan secara alpabetika dan semua harus dirujuk ke dalam naskah.

- [1] Adamuthe, A. C. dan A. Sundarkar. 2021. Solving 0-1 knapsack problem with variations of flower pollination algorithm. *Neural Computing and Applications* 31(7), <https://doi.org/10.1007/s00521-018-3375-7>
- [2] Adeyemo, H. Dan M. Ahmed. 2017. Solving 0-1 Knapsack Problem Using Metaheuristic Techniques. *IEEE-GCC Conference and Exhibition (GCCCE)*, <https://doi.org/10.1109/IEEEGCC.2017.8448239>
- [3] Asmuni, H., L. W. Shen dan F. C. Weng. 2015. A Modified Migrating Bird Optimization for University Course Timetabling Problem. *Jurnal Teknologi*

(*Sciences & Engineering*) 72(1): 89-96, <https://doi.org/10.11113/jt.v72.2949>

- [4] Boussaid, I., J. Lepageant dan P. Siarry. 2013. A survey on Optimization Metaheuristics. *Information Sciences* 237: 82-117, <https://doi.org/10.1016/j.ins.2013.02.041>
- [5] Chen, M., T. Song, S. Liu, X. Tang dan X. Peng. 2018. An iterated local search algorithm for the University Course Timetabling Problem. *Applied soft computing* 68: 597-608, <https://doi.org/10.1016/j.asoc.2018.04.034>
- [6] Kellerer, H., D. Pisinger dan U. Pferschy. 2004. *Knapsack Problem*. Berlin: Springer.
- [7] Supriadi, D. 2016. Perbandingan penyelesaian knapsack problem secara matematika, kriteria greedy dan algoritma greedy. *Indonesian Journal on Computer and Information Technologi*, **1(2)**: 91, <https://doi.org/10.31294/ijcit.v1i2.1835>
- [8] Yang, X. S. 2012. *Flower Pollination Algorithm for Global Optimization*. UK: Department of Engineering, University of Cambridge.