Penyelesaian Masalah *Mixed Integer Nonlinear Programming* Menggunakan Modifikasi *Salp Swarm Algorithm*

Anton Sudirman dan Ahmad Belva

Fakultas Matematika dan Ilmu Pengetahuan Alam
Institut Teknologi Bandung

2 Maret 2019

Outline

- Latar Belakang
- Masalah *Mixed Integer Nonlinear Programming*
- Metode *Salp Swarm Algorithm*
- Modifikasi Metode *Salp Swarm Algorithm*
- Eksperimen Numerik
- Simpulan dan Saran
- Referensi

Latar Belakang

- Permasalahan dalam bidang *engineering* biasanya dapat dimodelkan sebagai masalah *Mixed Integer Nonlinear Programming* (MINLP) [1].
- Fokus dari permasalahan MINLP adalah menentukan solusi optimal dari suatu fungsi objektif yang dibatasi oleh satu atau lebih kendala.
- Beberapa masalah MINLP bersifat tak-linier yang dapat menghasilkan fungsi objektif multi-modal sehingga metode optimisasi lokal seperti metode *steepest descent* solusinya tidak dapat digunakan.
- Oleh karena itu, metode optimisasi global digunakan untuk mendapatkan solusi dari permasalahan MINLP.

Latar Belakang

- Cagnina *et al.* menggunakan metode *Particle Swarm Optimization* (PSO) untuk menyelesaikan masalah optimisasi dalam bidang *engineering* [2].
- Garg menggunakan metode *Artificial Bee Colony Algorithm* (ABC) untuk menyelesaikan masalah optimisasi desain *engineering* struktural dengan kendala tak-linier [3].
- Yan menggunakan metode *Line-up Competition Algorithm* untuk menyelesaikan masalah *Mixed Integer Nonlinear Programming* [4].

Tujuan

Menyelesaikan masalah *Mixed Integer Nonlinear Programming* (MINLP) menggunakan modifikasi metode *Salp Swarm Algorithm* (SSA).

Masalah *Mixed Integer Nonlinear Programming*

Bentuk umum masalah MINLP dapat dituliskan sebagai:

$$\begin{array}{ll} \text{Minimum} & f(\mathbf{x}) \\ \text{terhadap} & g_i(\mathbf{x}) = 0, i = 1, 2, ..., M \\ & h_j(\mathbf{x}) \leq 0, j = 1, 2, ..., N \\ & \mathbf{x} = (x_1, x_2, ..., x_q, x_{q+1}, ..., x_n)^T \in \mathbb{R}^n \end{array} \tag{1}$$

dengan $x_1, x_2, ..., x_q$ merupakan bilangan bulat untuk suatu konstanta q.

Berdasarkan definisi fungsi penalti, masalah optimisasi dengan kendala dapat ditransformasikan menjadi masalah optimisasi tanpa kendala. Masalah optimisasi (1) dapat dituliskan sebagai:

$$F(\mathbf{x}, \alpha_i, \beta_j) = f(\mathbf{x}) + \sum_{i=1}^{M} \alpha_i g_i(\mathbf{x}) + \sum_{j=1}^{N} \beta_j (\text{maks}\{h_j(\mathbf{x}), 0\})^2 \tag{2}$$

dengan α_i dan β_j merupakan konstanta penalti yang dapat ditentukan sebesar 10^{10} sampai 10^{15} [5]. Kita gunakan $\alpha_i = \beta_j = 10^{15}$ untuk setiap i dan j.

Metode *Salp Swarm Algorithm*

- Metode Salp Swarm Algorithm (SSA) diperkenalkan oleh Mirjalili *et al.* pada tahun 2017. Metode SSA merupakan metode optimisasi metaheuristik yang terinspirasi oleh alam yang dapat digunakan untuk menyelesaikan berbagai masalah optimisasi.
- Metode SSA terinspirasi dari perilaku kawanan salp di laut.
- Salp merupakan anggota dari keluarga *Salpidae* dan sangat mirip dengan ubur-ubur.
- Secara umum, salp hidup berkelompok dan membentuk kawanan yang disebut rantai salp.
- Rantai salp terbagi menjadi dua grup: *Leader* dan *Followers*. *Leader* menempati urutan pertama dari rantai, sedangkan sisanya adalah *Followers*.

Salp

Figure 1: *Pegea confoederata*- Copyright Alexander Semenov

Salp

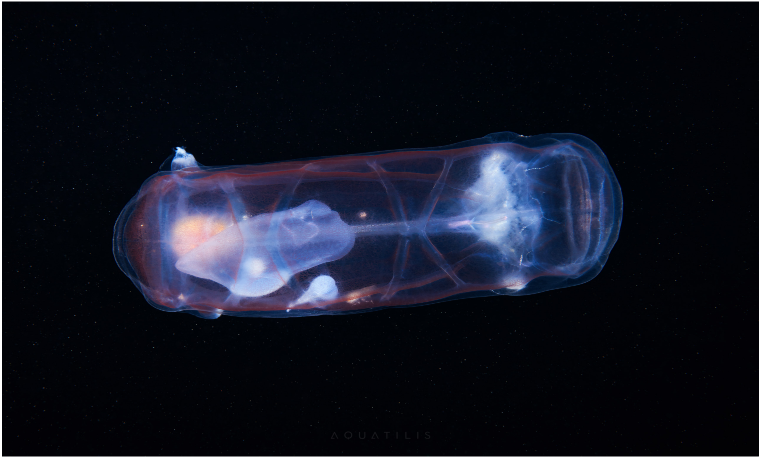

Figure 2: *Pegea confoederata*- Copyright Alexander Semenov

Salp

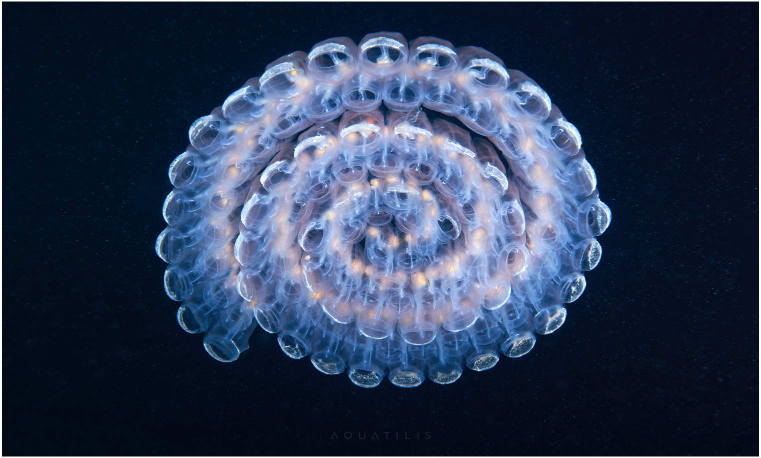

Figure 3: Rantai Salp- Copyright Alexander Semenov

Salp

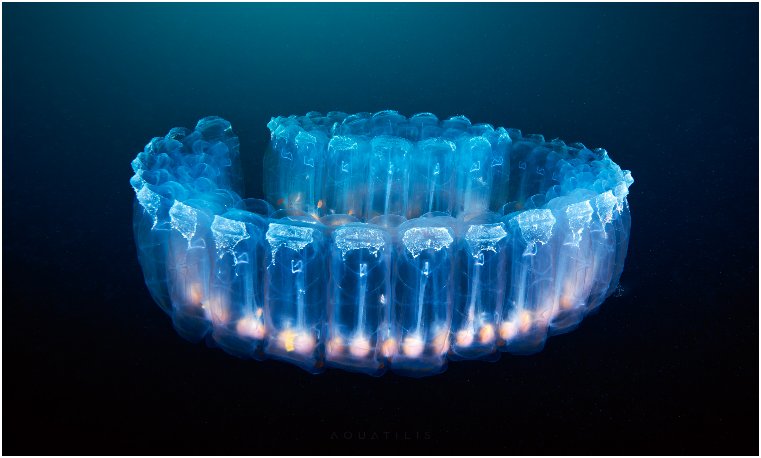

Figure 4: Rantai Salp- Copyright Alexander Semenov

Metode *Salp Swarm Algorithm*

Didefinisikan posisi salp:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{in}) \in \mathbb{R}^n, i = 1, 2, ..., m \tag{3}$$

dengan m adalah banyaknya salp dan n adalah banyaknya dimensi.
Posisi salp $\mathbf{x}_i, i = 1, 2, .., m$ dibagi menjadi dua yaitu posisi *Leader* salp:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{in}) \in \mathbb{R}^n, i = 1, 2, ..., \frac{m}{2} \tag{4}$$

dan posisi *Followers* salp:

$$\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{in}) \in \mathbb{R}^n, i = \left(\frac{m}{2} + 1\right), \left(\frac{m}{2} + 2\right), ..., m. \tag{5}$$

Diasumsikan ada sumber makanan sebagai target yang akan dituju oleh kawanan salp. Sumber makanan salp didefinisikan sebagai

$$\mathbf{s} = (s_1, s_2, \ldots, s_n)^T \in \mathbb{R}^n$$

dan memiliki nilai fungsi objektif terbaik dari semua salp.

Metode *Salp Swarm Algorithm*

Posisi *Leader* salp diperbarui menggunakan:

$$\mathbf{x}_i = \begin{cases} \mathbf{s} + c_1((\mathbf{ub} - \mathbf{lb})c_2 + \mathbf{lb}), & c_3 < 0.5 \\ \mathbf{s} - c_1((\mathbf{ub} - \mathbf{lb})c_2 + \mathbf{lb}), & c_3 \geq 0.5 \end{cases} \tag{6}$$

dengan $\mathbf{x}_i, i = 1, 2, ..., \frac{m}{2}$, merupakan posisi *Leader* salp, m adalah jumlah salp, $\mathbf{s}$ merupakan posisi sumber makanan, $\mathbf{lb}$ dan $\mathbf{ub}$ merupakan batas bawah dan batas atas dari $\mathbf{x}_i$, c_2 dan c_3 adalah bilangan acak pada selang [0,1] dan c_1 dituliskan sebagai

$$c_1 = 2e^{-\left(\frac{4k}{k_{maks}}\right)^2} \tag{7}$$

dengan k merupakan iterasi dan k_{maks} merupakan maksimum iterasi.

Metode *Salp Swarm Algorithm*

Posisi *Followers* salp diperbarui menggunakan:

$$\mathbf{x}_i = \frac{1}{2}\left(\mathbf{x}_i + \mathbf{x}_{i-1}\right), i = \left(\frac{m}{2}+1\right), \left(\frac{m}{2}+2\right), ..., m, \tag{8}$$

dengan $\mathbf{x}_i, i = \left(\frac{m}{2}+1\right), \left(\frac{m}{2}+2\right), ..., m$, merupakan posisi *Followers* salp dan m adalah banyaknya salp.

Metode *Salp Swarm Algorithm*

Algoritma Metode SSA

Input:

$F(\mathbf{x})$, m, k_{maks}, **lb**, **ub**

Proses:

1. Bangun populasi awal salp, $\mathbf{x}_i(0) \in \mathbb{R}^n, i = 1, 2, ..., m$, berdasarkan **ub** dan **lb**.
2. Tetapkan $k = 0$.
3. Tentukan $\mathbf{s} = \mathbf{x}_{i_g}(k)$ dengan $i_g = \arg\min_i F(\mathbf{x}_i(k)), i = 1, 2, ..., m$.
4. Hitung $c_1 = 2e^{-\left(\frac{4k}{k_{maks}}\right)^2}$.
5. Perbarui posisi *Leader* salp dengan menggunakan

$$\mathbf{x}_i(k+1) = \begin{cases} \mathbf{s} + c_1((\mathbf{ub} - \mathbf{lb})c_2 + \mathbf{lb}), & c_3 < 0.5 \\ \mathbf{s} - c_1((\mathbf{ub} - \mathbf{lb})c_2 + \mathbf{lb}), & c_3 \geq 0.5 \end{cases} \tag{9}$$

untuk $i = 1, 2, ..., \frac{m}{2}$, dan $c_2, c_3 \in [0.1]$.

Metode *Salp Swarm Algorithm*

6. Perbarui posisi *Followers* salp dengan menggunakan

$$\mathbf{x}_i(k+1) = \frac{1}{2}\left(\mathbf{x}_i + \mathbf{x}_{i-1}\right), \tag{10}$$

untuk $i = \left(\frac{m}{2}+1\right), \left(\frac{m}{2}+2\right), ..., m$.

7. Jika $\mathbf{x}_i(k+1), i = 1, 2, ..., m$, ada yang keluar dari ruang pencarian, maka tarik $\mathbf{x}_i(k+1)$ ke dalam batas ruang pencarian.
8. Perbarui posisi sumber makanan, $\mathbf{s} = \mathbf{x}_{i_g}(k+1)$ dengan $i_g = \arg\min_i F(\mathbf{x}_i(k+1)), i = 1, 2, ..., m$.
9. Jika $k = k_{maks}$, maka proses berhenti. Lainnya, $k = k+1$ dan kembali ke langkah 4.
10. $\mathbf{x}^* = \mathbf{s}$.

Output:
$\mathbf{x}^*$ adalah titik minimum dari $F(\mathbf{x})$.

Algoritma Modifikasi Metode SSA

Input:

$F(\mathbf{x}), m, k_{maks}, \mathbf{lb}, \mathbf{ub}$.

Proses:

1. Bangun populasi awal salp, $\mathbf{x}_i(0) \in \mathbb{R}^n, i = 1, 2, ..., m$, berdasarkan **ub** dan **lb**.
2. Tetapkan $k = 0$.
3. Hitung nilai $F(\mathbf{x}_i(k))$,

$$\mathbf{x}_i(k) = (x_{i1}(k), x_{i2}(k), ..., x_{in}(k))^T$$

dengan

$$\mathbf{x}_{ij}(k) = \begin{cases} round(\mathbf{x}_{ij}), & \text{jika } \mathbf{x}_{ij} \text{ harus bilangan bulat} \\ \mathbf{x}_{ij}, & \text{lainnya} \end{cases}$$

untuk $i = 1, 2, ..., m$ dan $j = 1, 2, ..., n$.

4. Tentukan $\mathbf{s} = \mathbf{x}_{i_g}(k)$ dengan $i_g = \arg\min_i F(\mathbf{x}_i(k)), i = 1, 2, ..., m$.

5. Hitung $c_1 = 2e^{-\left(\frac{4k}{k_{maks}}\right)^2}$.
6. Perbarui posisi *Leader* salp dengan menggunakan

$$\mathbf{x}_i(k+1) = \begin{cases} \mathbf{s} + c_1((\mathbf{ub} - \mathbf{lb})c_2 + \mathbf{lb}), & c_3 < 0.5 \\ \mathbf{s} - c_1((\mathbf{ub} - \mathbf{lb})c_2 + \mathbf{lb}), & c_3 \geq 0.5 \end{cases} \tag{11}$$

   untuk $i = 1, 2, ..., \frac{m}{2}$, dan $c_2, c_3 \in [0.1]$.
7. Perbarui posisi *Followers* salp dengan menggunakan

$$\mathbf{x}_i(k+1) = \frac{1}{2}\left(\mathbf{x}_i + \mathbf{x}_{i-1}\right), \tag{12}$$

   untuk $i = \left(\frac{m}{2} + 1\right), \left(\frac{m}{2} + 2\right), ..., m$.
8. Jika $\mathbf{x}_i(k+1), i = 1, 2, ..., m$, ada yang keluar dari ruang pencarian, maka tarik $\mathbf{x}_i(k+1)$ ke dalam batas ruang pencarian.

9. Hitung nilai $F(\mathbf{x}_i(k+1))$,

$$\mathbf{x}_i(k+1) = (x_{i1}(k+1), x_{i2}(k+1), ..., x_{in}(k+1))^T$$

dengan

$$\mathbf{x}_{ij}(k+1) = \begin{cases} round(\mathbf{x}_{ij}), & \text{jika } \mathbf{x}_{ij} \text{ harus bilangan bulat} \\ \mathbf{x}_{ij}, & \text{lainnya} \end{cases}$$

untuk $i = 1, 2, ..., m$ dan $j = 1, 2, ..., n$.

10. Perbarui posisi sumber makanan, $\mathbf{s} = \mathbf{x}_{i_g}(k+1)$ dengan $i_g = \arg\min_i F(\mathbf{x}_i(k+1)), i = 1, 2, ..., m$.
11. Jika $k = k_{maks}$, maka proses berhenti. Lainnya, $k = k + 1$ dan kembali ke langkah 5.
12. $\mathbf{x}^* = \mathbf{s}$.

Output:

$\mathbf{x}^*$ adalah titik minimum dari $F(\mathbf{x})$.

Masalah Desain Pereduksi Kecepatan

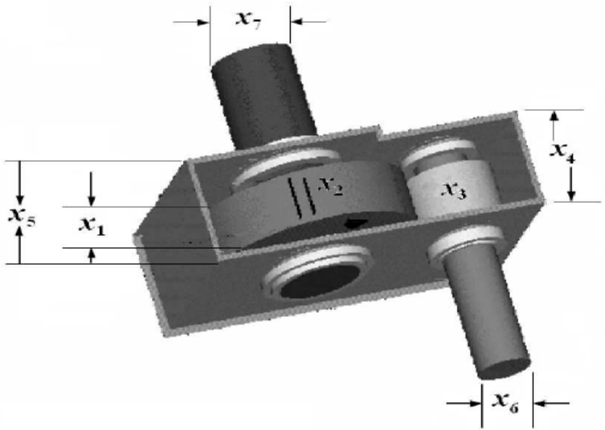

Figure 5: Desain Pereduksi Kecepatan

Keterangan: x_1 adalah *face width*, x_2 adalah modul gigi, x_3 adalah jumlah gigi pada pinion, x_4 adalah panjang poros pertama antar bantalan, x_5 adalah panjang poros kedua antar bantalan, x_6 adalah diameter poros pertama dan x_7 adalah diameter poros kedua (semua variabel kontinu kecuali x_3 yang berupa bilangan bulat).

Masalah optimisasi desain pereduksi kecepatan adalah meminimumkan kecepatan terhadap kendala-kendala: tekanan dari gigi, tegangan permukaan, defleksi melintang dari poros dan tekanan diporos.

Secara matematis, masalah desain pereduksi kecepatan dapat diformulasikan sebagai:

$$
\begin{aligned}
\text{Min } f(\mathbf{x}) &= 0.7854x_1x_2^2(3.333x_3^2 + 14.9334x_3 - 43.0934) \\
&\quad - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) \\
&\quad + 0.7854(x_4x_6^2 + x_5x_7^2) \\
\text{terhadap } g_1(\mathbf{x}) &= \frac{27}{x_1x_2^2x_3} - 1 \leq 0 \quad g_2(\mathbf{x}) = \frac{397.5}{x_1x_2^2x_3^3} - 1 \leq 0 \\
g_3(\mathbf{x}) &= \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0 \quad g_4(\mathbf{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0 \\
g_5(\mathbf{x}) &= \frac{1}{110x_6^3}\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6} - 1 \leq 0 \\
g_6(\mathbf{x}) &= \frac{1}{85x_7^3}\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6} - 1 \leq 0
\end{aligned}
$$

$$g_7(\mathbf{x}) = \frac{x_2 x_3}{40} - 1 \le 0$$
$$g_8(\mathbf{x}) = \frac{5x_2}{x_1} - 1 \le 0 \; g_9(\mathbf{x}) = \frac{x_1}{12x_2} - 1 \le 0$$
$$g_{10}(\mathbf{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$$
$$g_{11}(\mathbf{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0$$

dengan $2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.8 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9$, dan $5.0 \le x_7 \le 5.5$ (x_3 berupa bilangan bulat)

- Jika masalah desain pereduksi kecepatan diselesaikan dengan menggunakan metode SSA maka solusi yang diperoleh adalah

$$\mathbf{x} = (3.5, 0.7, 17.008, 7.30273, 7.82697, 3.35025, 5.28669)$$

  dengan waktu eksekusi 7.71 detik dan nilai fungsi objektif sebesar $f(\mathbf{x}) = 2998.3598$.
- Namun, jika dilihat dari solusi yang diperoleh, variabel x_3 bukan berupa bilangan bulat, hal ini tidak sesuai dengan syarat bahwa x_3 harus bilangan bulat.
- Oleh karena itu, modifikasi SSA akan diterapkan pada masalah ini.

- Masalah optimisasi desain pereduksi kecepatan dapat diselesaikan menggunakan modifikasi SSA sebanyak 100 kali eksekusi dengan populasi salp sebanyak 50 dan iterasi maksimum 1000, kemudian dipilih nilai fungsi objektif yang paling minimum.
- Solusi yang diperoleh dengan menggunakan modifikasi SSA adalah

  $$\mathbf{x} = (3.5, 0.7, 17, 7.31831, 7.80461, 3.35025, 5.28668)$$

  dengan waktu eksekusi 7.74 detik.
- Nilai fungsi objektif berdasarkan solusi tersebut adalah $f(\mathbf{x}) = 2996.62$.
- Jika dilihat dari solusi yang diperoleh, variabel x_3 berupa bilangan bulat sehingga solusi yang diperoleh sudah optimal dengan nilai fungsi objektif $f(\mathbf{x}) = 2996.62$ hampir sama dengan yang diperoleh pada [2] sebesar $f(\mathbf{x}) = 2996.35$.

Masalah pemilihan pelari dalam lari estafet 4 $\times$ 100 m.

- Ada 4 pelari yang masing-masing berada pada setiap bagian dari lintasan 4×100 m.
- Para pelari tersebut dipilih dari grup yang beranggotakan 6 pelari untuk memperoleh tim yang memiliki waktu tercepat.
- Pemilihan harus memenuhi beberapa persyaratan yaitu: setiap atlet hanya boleh ditempatkan pada satu bagian; setiap bagian hanya ada satu pelari.

Tabel 1. Waktu (detik) dari setiap pelari pada masing-masing bagian.

Atlet	Bagian 1	Bagian 2	Bagian 3	Bagian 4
Pelari 1	12.27 s	11.57 s	11.54 s	12.07
Pelari 2	11.34 s	11.45 s	12.45 s	12.34 s
Pelari 3	11.29 s	11.50 s	11.45 s	11.52 s
Pelari 4	12.54 s	12.34 s	12.32 s	11.57 s
Pelari 5	12.20 s	11.22	12.07 s	12.03 s
Pelari 6	11.54 s	11.48 s	11.56 s	12.30 s

Masalah pemilihan pelari dalam lari estafet 4×100 m:

$$\begin{aligned} \text{Minimumkan} & \sum_{j=1}^{4}\sum_{i=1}^{6} \tau_{ij}x_{ij} \\ \text{terhadap} & \sum_{i=1}^{6} x_{ij} = 1, \ \forall i, 1 \leq j \leq 4 \\ & \sum_{j=1}^{4} x_{ij} \leq 1, \ \forall j, 1 \leq i \leq 6 \end{aligned}$$

dengan $\mathrm{x}_{ij} = \begin{cases} 1, & \text{jika pelari ke-i berlari pada bagian lintasan ke-j} \\ 0, & \text{lainnya} \end{cases}$

dan τ_{ij} adalah waktu yang ditempuh pelari ke-i pada bagian ke-j.

- Jika masalah optimisasi tersebut diselesaikan dengan metode SSA maka solusi yang diperoleh adalah

$$\begin{aligned}\mathbf{x} = (&0.24139, 0.5215, 0.09201, 0.0940, 0.0103, 0.0408, 0.0080, 0.0499,\\ &0.1870, 0.0044, 0.4771, 0.2737, 0.4450, 0.1674, 0.1135, 0.0022,\\ &0.0947, 0.1772, 0.2998, 0.2327, 0.0450, 0.0002, 0.4076, 0.0150)\end{aligned}$$

  dengan waktu eksekusi 1.02 detik.
- Total waktu pelari yang diperoleh dari solusi tersebut adalah $f(\mathbf{x}) = 51.34$ detik.
- Namun, dapat dilihat bahwa solusi yang diperoleh bukan berupa bilangan bulat, hal ini tidak sesuai dengan syarat bahwa solusi harus bilangan bulat.
- Oleh karena itu, modifikasi metode SSA akan diterapkan pada masalah ini.

- Masalah optimisasi tersebut dapat diselesaikan menggunakan modifikasi SSA sebanyak 100 kali eksekusi dengan populasi salp sebanyak 50 dan iterasi maksimum 1000, kemudian dipilih nilai fungsi objektif yang paling minimum.
- Solusi yang diperoleh adalah $\mathbf{x} = (0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)$ dengan waktu eksekusi 1.56 detik dan nilai fungsi objektif $f(\mathbf{x}) = 45.58$.
- Berdasarkan solusi yang diperoleh sudah berupa bilangan bulat sehingga solusi sudah optimal dan nilai fungsi objektifnya sebesar **45.58** lebih baik jika dibandingkan dengan yang diperoleh pada [8] sebesar 45.62.

Tabel 2. Waktu minimum yang diperoleh dari pelari-pelari yang terpilih.

Lintasan	Atlet
Lintasan 1	Pelari 2 (11.34 s)
Lintasan 2	Pelari 5 (11.22 s)
Lintasan 3	Pelari 3 (11.45 s)
Lintasan 4	Pelari 4 (11.57 s)
Total waktu	**45.58** s

Simpulan dan Saran

- Modifikasi metode SSA untuk masalah MINLP terletak pada penggunaan fungsi *round* dalam algoritma.
- Modifikasi metode SSA dapat digunakan untuk menyelesaikan masalah MINLP lebih baik dibandingkan dengan metode SSA tanpa modifikasi.
- Untuk penelitian selanjutnya, dapat dilakukan perbandingan dengan metode-metode optimisasi yang lain dalam menyelesaikan masalah MINLP.

Referensi

[1] A. Kania, K. A. Sidarto, "Solving Mixed Integer Nonlinear Programming Problems Using Spiral Dynamics Optimization Algorithm," AIP Conference Proceedings, vol. 1716, 2016.
[2] L. C. Cagnina, S. C. Esquival, C. A. Coello Coello, "Solving Engineering Optimization Problems with the Simple Constrained Particle Swarm Optimizer," Informatica, vol. 32, pp. 319-326, 2008.
[3] H. Garg, "Solving Structural Engineering Design Optimization Problems Using Artificial Bee Colony Algorithm," Journal of Industrial and Management Optimization, vol. 10, pp. 777-794, 2014.
[4] L. Yan, K. Shen, S. Hun, "Solving Mixed Integer Nonlinear Programming Problems with Line-up Competition Algorithm," Computers Chemical Engineering, vol. 28(12), pp.2647-2657, November 2004.
[5] X. S. Yang, Engineering Optimization, John Wiley Sons Inc., New Jersey 2010.

Referensi

[6] H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A. Z. Alam, S. Mirjalili, H. Fujita, “An efficient binary Salp Swarm algorithm with crossover scheme for feature selection problems,” Knowledge Based System, vol. 154, pp. 43-67, 2018.

[7] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, S. M. Mirjalili, “Salp Swarm algorithm: a bio-inspired optimizer for engineering design problems,” Advanced Engineering Software, vol. 114, pp. 163-191, 2017.

[8] F. Masedu and M. Angelozzi, “Modelling Optimum Fraction Assigment in the 4x100 m Relay Race by Integer Linear Programming,” Italian Journal of Sports Sciences, vol. 13, pp. 74-77, 2008.

- Terima kasih.