Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) p-ISSN: 2355-7699
Vol. 12, No. 3, Juni 2025, hlm. 671-680 e-ISSN: 2528-6579

GREY WOLF OPTIMIZER TERMODIFIKASI MENGGUNAKAN *CHAOTIC UNIFORM INITIALIZATION* UNTUK ESTIMASI *EFFORT* COCOMO

Ardiansyah[1*], Sri Handayaningsih[2], Deva Fathurrizki[3]

Universitas Ahmad Dahlan, Yogyakarta[1,2,3]
Email: [1]ardiansyah@tif.uad.ac.id, [2]sriningsih@is.uad.ac.id, [3]deva2000018272@webmail.uad.ac.id
[*]Penulis Korespondensi

(Naskah masuk: 26 September 2024, diterima untuk diterbitkan: 3 Juni 2025)

Abstrak

COCOMO merupakan metode estimasi *effort* perangkat lunak berbasis parametrik yang banyak digunakan dan fleksibel diimplementasikan pada organisasi skala kecil hingga besar. Akan tetapi, kedua parameter COCOMO, yaitu multiplikatif dan eksponensial kerap memberikan hasil yang kurang presisi serta tidak realistis untuk diterapkan pada lingkungan pengembangan perangkat lunak saat ini. Untuk mengatasi masalah tersebut, beberapa penelitian mengusulkan pendekatan berbasis pencarian untuk mendapatkan nilai parameter yang tepat dengan menggunakan algoritma optimasi metaheuristik. Grey Wolf Optimizer (GWO) merupakan salah satu algoritma optimasi yang bisa menghindari jebakan optimum lokal yang sering dialami oleh algoritma berbasis *swarm intelligence*. Namun, GWO kurang dalam hal *diversity* populasi yang membuat banyak kandidat solusi tidak mampu menjangkau ruang pencarian secara merata. Untuk mengatasi masalah tersebut, penelitian ini mengusulkan GWO termodifikasi berupa *chaotic uniform initialization* agar bisa meningkatkan *diversity* populasi. Metode yang diusulkan ini membangkitkan dua populasi awal yang masing-masing menggunakan teknik *chaos* dan acak. Setiap kandidat solusi pada kedua populasi tersebut diseleksi berdasarkan nilai *fitness* tertentu yang pada akhirnya akan membentuk satu populasi awal yang memiliki *diversity* yang lebih baik. Eksperimen pada penelitian ini menggunakan tiga himpunan data dari NASA. Untuk mendapatkan teknik *chaos* terbaik, dilakukan komparasi terhadap tujuh teknik *chaos*. Metode yang diusulkan kemudian dikomparasikan dengan algoritma GWO standar dan satu varian GWO terkini. Hasil penelitian menunjukkan bahwa metode yang diusulkan beserta teknik *chaos circle* terbukti mampu memperbaiki *diversity* populasi sehingga meningkatkan performa akurasi estimasi COCOMO. Metode yang diusulkan ini dimungkinkan untuk diintegrasikan pada alat bantu estimasi *effort* perangkat lunak yang biasa digunakan oleh para manajer proyek.

Kata kunci: *estimasi effort, proyek perangkat lunak, grey wolf optimizer, optimasi metaheuristik, COCOMO*

A MODIFIED GREY WOLF OPTIMIZER USING CHAOTIC UNIFORM INITIALIZATION FOR COCOMO EFFORT ESTIMATION

Abstract

COCOMO is a parametric-based software effort estimation method that is widely used and flexible to implement in small to large scale organizations. However, the two COCOMO parameters, namely multiplicative and exponential, often provide results that lack precision and are not realistic to apply to the current software development environment. To overcome this problem, several studies have proposed search-based approach to obtain appropriate parameter values using metaheuristic optimization algorithms. Grey Wolf Optimizer (GWO) is an algorithm that can avoid the local minimum trap that is often experienced by other swarm intelligence algorithms. However, GWO lacks population diversity which makes candidate solutions unable to cover the search space evenly. This study proposes chaotic uniform initialization to increase population diversity. The proposed method generates two initial populations using chaotic and random techniques respectively. Each candidate solution in the two populations is selected based on a certain fitness value which will ultimately produce an initial population that has better diversity. The experiment in this study used three data sets from NASA. To get the best chaos technique, a comparison of seven chaos techniques was carried out. The proposed method is then compared with the standard GWO algorithm and a current GWO variant. The research results show that the proposed method and the chaos circle technique are proven to be able to improve population diversity thereby increasing the accuracy of COCOMO. The proposed method is possible to integrated into software effort estimation tools commonly used by project managers.

Keywords: *effort estimation, software project, grey wolf optimizer, metaheuristic optimization, COCOMO*

1. PENDAHULUAN

Industri perangkat lunak diperkirakan akan bernilai $580 miliar pada tahun 2027 (Amarendra et al. 2023). Angka ini tumbuh hampir 58% dibanding tahun 2023 yaitu sebesar $336.3 miliar. Namun, walau sudah hampir berusia 70 tahun (Sheldon 1994), industri ini masih menghadapi dua tantangan berat yaitu pengerjaan proyek perangkat lunak yang tidak selesai tepat waktu, dan biaya yang selalu melebihi anggaran (Ali et al. 2023).

Para manajer proyek sudah berusaha keras untuk merencanakan anggaran proyek dengan baik. Salah satunya dengan melakukan estimasi *effort*. Estimasi *effort* perangkat lunak merupakan salah satu aktivitas pada fase perencanaan berdasarkan *software development life cycle* (SDLC). Estimasi penting dilakukan karena sangat memengaruhi kinerja atau nama baik perusahaan. Estimasi yang akurat bisa meningkatkan kinerja keuangan perusahaan dan nama baik perusahaan di mata klien. Sebaliknya, estimasi yang tidak akurat bisa mengancam keuangan perusahaan hingga kehilangan klien.

Terdapat dua pendekatan estimasi *effort* perangkat lunak, yaitu berbasis ukuran (*size-based*) dan parametrik. Menurut (Jorgensen & Shepperd 2007), kedua pendekatan tersebut adalah model estimasi formal karena estimasinya berbasiskan spesifikasi kebutuhan perangkat lunak. COCOMO dan SLIM merupakan dua contoh estimasi formal menggunakan pendekatan parametrik. Sedangkan, Function Points (FP), Use Case Points (UCP), dan COSMIC adalah contoh estimasi formal menggunakan pendekatan ukuran.

Sebagai estimasi *effort* berbasis parametrik, COCOMO adalah satu satu metode yang diterima luas di industri dan komunitas perangkat lunak. COCOMO telah terbukti dan teruji digunakan banyak organisasi atau perusahaan selama bertahun-tahun, serta fleksibel untuk diterapkan pada proyek skala kecil hingga besar (Boehm, Abts, & Chulani 2000). COCOMO memiliki dua parameter penting yaitu konstanta multiplikatif dan eksponensial yang berperan dalam perhitungan estimasi *effort* perangkat lunak. Namun, karena kedua parameter tersebut bersifat konstan, membuat COCOMO kerap memberikan hasil yang kurang presisi serta tidak realistis untuk diterapkan pada lingkungan pengembangan perangkat lunak saat ini (Sachan et al. 2016). Kelemahan ini tentu saja memengaruhi performa akurasi estimasi COCOMO secara keseluruhan.

Berbagai penelitian telah dilakukan untuk memperbaiki performa estimasi COCOMO. Salah satunya adalah melalui optimasi terhadap kedua parameter tersebut. Optimasi dilakukan untuk menemukan nilai parameter yang tepat agar bisa meningkatkan performa akurasi estimasi. Performa akurasi yang baik ditunjukkan dengan nilai *error* seminimum mungkin.

Parameter multiplikatif dan eksponensial pada metode estimasi COCOMO telah coba dioptimasi menggunakan Particle Swarm Optimization (PSO) (Chhabra & Singh 2020; Langsari & Sarno 2017; Zakaria et al. 2021), (Zakaria et al. 2021), algoritma Genetika (Sachan et al. 2016), Cuckoo (Parwita, Sarno, & Puspaningrum 2017), dan Grey Wolf Optimizer (GWO) (Putri, Siahaan, & Fatichah 2021). Walau algoritma optimasi tersebut telah terbukti memperbaiki akurasi COCOMO, namun PSO dan algoritma Genetika memiliki dua kelemahan utama yaitu konvergen prematur, dan mudah terjebak pada optimum lokal.

Dari berbagai algoritma optimasi tersebut, GWO merupakan algoritma yang relatif baru. Sejak diusulkan oleh Mirjalili (Mirjalili, Mirjalili, & Lewis 2014), GWO telah mendapat sambutan besar dari komunitas peneliti. GWO merupakan salah satu algoritma optimasi metaheuristik berbasis kecerdasan kelompok (*swarm intelligence*) yang terbukti efektif dan mampu memecahkan permasalahan di berbagai bidang rekayasa, kedokteran, *machine learning*, hingga bioinformatika (Faris et al. 2018). GWO merupakan salah satu algoritma yang mampu menghindar dari jebakan optimum lokal yang selama ini menjadi kelemahan utama optimasi berbasis populasi. Artinya, GWO memiliki keseimbangan yang kuat antara fase eksplorasi dan eksploitasi (Mirjalili, Mirjalili, & Lewis 2014).

Meskipun GWO mampu menghindari jebakan optimum lokal, akan tetapi GWO memiliki satu kelemahan lain yaitu *diversity* populasi. *Diversity* populasi merupakan masalah kurang beragamnya kandidat solusi menempati posisi yang menghasilkan solusi optimum. Kelemahan tersebut memang biasa dialami oleh algoritma optimasi metaheuristik.

Beberapa penelitian berusaha untuk mengatasi kelemahan GWO. Salah satunya adalah yang dilakukan oleh (Putri, Siahaan, & Fatichah 2022) yang mengusulkan *extended gamma*. Penelitian ini menambahkan satu hirarki kandidat solusi terbaik baru yaitu *gamma*. Sebelumnya, pada GWO standar, sudah ada tiga hirarki kandidat solusi terbaik yaitu *alpha*, *beta*, dan *delta*. Tentu saja usulan dari penelitian ini bukan termasuk kategori memperbaiki *diversity* populasi GWO.

Salah satu solusi yang diusulkan para peneliti untuk mengatasi masalah *diversity* tersebut adalah dengan menggunakan teknik gerak *chaos*. Gerak *chaos* memiliki karakteristik berupa keacakan, ergodisitas, dan regularitas yang jika digunakan pada proses pencarian, bisa mempertahankan *diversity* populasi serta meningkatkan kemampuan pencarian global. Gerak *chaos* memiliki banyak teknik, antara lain yaitu *logistic*, *tent*, *sine*, *sinusoidal*, *circle*, *chebyshev*, dan lain sebagainya.

Teknik *chaos* yang digunakan pada berbagai penelitian ternyata berbeda-beda. Sebagai contoh, teknik *sine* dan *tent* digunakan oleh (Cai et al. 2019; Li, Lin, & Liu 2021). Bahkan, (Kohli & Arora 2018) mengevaluasi 10 teknik *chaos* yang digunakan untuk untuk menginisialisasi populasi awal pada GWO. Hasilnya, *chebyshev* menunjukkan performa yang paling baik. Penelitian lain yang dilakukan oleh (Hao & Sobhani 2021; Ibrahim, Elaziz, & Lu 2018), selain menggunakan *logistic* juga menerapkan *oppositional-based learning* (OBL) untuk menghasilkan populasi awal dan eksplorasi yang lebih baik.

Berdasakan penelitian-penelitan sebelumnya telah menunjukkan bahwa penggunaan teknik *chaos* mampu memberikan hasil yang baik. Oleh karena itu, untuk mengatasi masalah *diversity* tersebut, penelitian ini mengusulkan GWO termodifikasi dengan inisialisasi populasi awalnya menggunakan *chaotic uniform initialization*, yang diberi nama CUGWO.

Metode CUGWO yang diusulkan ini memperbaiki performa GWO dengan cara membuat dua kelompok populasi awal. Kelompok pertama dibangkitkan secara acak *uniform*, dan kelompok kedua dibangkitkan menggunakan *chaos*. Selanjutnya, tiap individu pada kedua populasi tersebut diseleksi berdasarkan fungsi *fitness*. Pada akhirnya hasil seleksi tersebut akan menghasilkan satu populasi awal yang memiliki *diversity* yang lebih baik. Perbedaan dengan penelitian sebelumnya adalah penelitian ini terlebih dahulu membandingkan semua teknik *chaos* agar memperoleh teknik yang menghasilkan performa terbaik. Teknik *chaos* terbaik tersebut khusus digunakan pada estimasi *effort* perangkat lunak, yang pada penelitian sebelumnya diterapkan untuk bidang atau permasalahan lain.

Berdasarkan metode yang diusulkan tersebut, maka penelitian ini memiliki kontribusi pada modifikasi teknik pembangkitan populasi awal algoritma GWO. Selain itu, penelitian ini bertujuan untuk mengeksplorasi penggunaan teknik *chaotic* dalam algoritma GWO agar *diversity* populasi dan performa akurasi estimasi COCOMO bisa meningkat.

2. METODE PENELITIAN

Bagian ini akan menjabarkan secara rinci mengenai GWO dan metode yang diusulkan beserta rancangan eksperimennya.

2.1. Grey Wolf Optimizer

GWO terinspirasi dari cara kerja kawanan serigala dalam mencari mangsa. Pada GWO, mangsa yang menjadi target sama saja dengan solusi optimum yang dicari. Kawanan serigala pada GWO merupakan representasi populasi dari kandidat-kandidat solusi.

Proses pencarian solusi terbaik optimum dimulai dengan membangkitkan populasi awal. Setelah populasi awal terbentuk, maka dilakukan pengepungan mangsa (*encircling prey*). Model matematikanya digambarkan oleh persamaan (1) dan (2).

$$\vec{D} = \left|\vec{C} \cdot \vec{X}(t) - \vec{X}(t)\right| \quad (1)$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \quad (2)$$

dimana t menunjukkan iterasi saat ini, $\vec{A}$ dan $\vec{C}$ merupakan koefisien, $\vec{X}_p$ adalah posisi mangsa saat ini, dan $\vec{X}$ menunjukkan posisi *grey wolf*.

Koefisien $\vec{A}$ dan $\vec{C}$ dihitung menggunakan persamaan (3) dan (4).

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \quad (3)$$

$$\vec{C} = 2 \cdot \vec{r}_2 \quad (4)$$

dengan $\vec{a}$ adalah bilangan yang berkurang secara linear dari 2 sampai 0 mengikuti iterasi. Sedangkan $\vec{r}_1$ dan $\vec{r}_2$ adalah bilangan acak antara [0, 1].

Perlu diketahui bahwa $\vec{C}$ merupakan strategi untuk mengatasi jebakan optimum lokal, khususnya di akhir iterasi.

2.1.1 Fase berburu (*hunting*)

Keberadaan mangsa mampu dilacak oleh kawanan serigala mengikuti panduan dari serigala *alpha*. Selain *alpha*, ada juga serigala *beta* dan *delta* yang kadang-kadang membantu perburuan (*hunting*). Pada GWO, *alpha*, *beta*, dan *delta* secara berurutan merupakan tiga serigala terbaik yang paling tahu dengan lokasi mangsa. Dengan kata lain, ketiga serigala tersebut merupakan kandidat solusi yang paling mendekati solusi optimum di iterasi saat itu.

Pada fase ini, di tiap iterasi, tiap kandidat solusi akan memperbarui posisinya masing-masing, mengikuti persamaan (5), (6), dan (7).

$$\vec{D}_\alpha = \left|\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}\right|, \vec{D}_\beta = \left|\vec{C}_2 \cdot \vec{X}_{\boldsymbol{\beta}} - \vec{X}\right|, \vec{D}_\delta = \left|\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}\right| \quad (5)$$

$$\vec{X}_1 = \vec{X}_\alpha - A_1 \cdot \left(\vec{D}_\alpha\right), \vec{X}_2 = \vec{X}_\beta - A_2 \cdot \left(\vec{D}_\beta\right), \vec{X}_3 = \vec{X}_\delta - A_3 \cdot \left(\vec{D}_\delta\right) \quad (6)$$

$$\vec{X}(t-1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (7)$$

2.1.2 Eksploitasi dan eksplorasi

Proses perburuan akan diakhiri dengan penyerangan (*attacking*) mangsa. Perburuan berarti fase eksplorasi, dan penyerangan adalah fase eksploitasi dalam konteks pencarian solusi optimum. Secara matematis, untuk mendekati mangsa perlu dilakukan pengurangan nilai $\vec{a}$. Perlu dicatat bahwa fluktuasi rentang nilai $\vec{A}$ juga disebabkan karena berkurangnya nilai $\vec{a}$. Dengan demikian, rentang nilai $\vec{A}$ adalah antara $[-2a, 2a]$.

Pergerakan kawanan serigala mengikuti posisi serigala *alpha*, *beta*, dan *delta*. Ada masanya mereka

menyebar untuk mencari mangsa, dan ada kalanya berkumpul untuk menyerang mangsa. Ketika menyebar, berarti serigala sedang menjauhi mangsa. Sebaliknya, ketika berkumpul, berarti serigala mendekat untuk menyerang mangsanya.

Secara matematis, pemodelannya adalah jika $|\vec{A}| > 1$ berarti serigala sedang menjauhi mangsa, atau jika $|\vec{A}| < 1$ berarti serigala sedang mendekat untuk menyerang mangsanya.

2.2 Usulan Metode

Metode yang diusulkan adalah *chaotic uniform grey wolf optimizer*, disingkat CUGWO. Seperti yang ditunjukkan oleh Algoritma 1, CUGWO dimulai dengan membangkitkan dua populasi awal yang masing-masing dihasilkan oleh teknik *chaos* dan acak *uniform*. Kandidat solusi (*wolf*) di kedua populasi tersebut diseleksi berdasarkan nilai *fitness* yang telah ditentukan, untuk kemudian membentuk populasi awal terbaik. Dari populasi awal ini akan diinisialisasikan tiga serigala terbaik yaitu *alpha*, *beta*, dan *delta*.

Selanjutnya, dilakukan iterasi untuk memperbarui posisi kandidat solusi atau serigala, koefisien *a*, *A*, dan *C*. Perubahan nilai *a* di tiap iterasi mengikuti persamaan (8).

$$a = 2 - \frac{(2 \times t)}{t_{max}} \quad (8)$$

dimana *t* adalah iterasi saat ini, dan t_{max} adalah iterasi maksimum. Iterasi ini terus berlangsung hingga memenuhi kriteria berhenti tertentu dan memperoleh solusi terbaik optimum.

Algoritma 1. CUGWO

1. Bangkitkan populasi I menggunakan inisialisasi *uniform* $\mathbf{x}_i(i = 1, 2, \dots, n)$
2. Hitung nilai *fitness* tiap *wolf* pada populasi I
3. Bangkitkan populasi II menggunakan inisialisasi *chaotic* $\boldsymbol{y}_i(y = 1, 2, \dots, n)$
4. Hitung nilai *fitness* (pers. 11) tiap *wolf* pada populasi II
5.
6. for i in range(popSize)
7. if fitness_$\mathbf{x}_i$ > fitness_y_i
8. population ← $\mathbf{x}_i$
9. else
10. population ← $\boldsymbol{y_i}$
11.
12. inisialisasi $a = 2$, *A*, dan *C*
13. X_α ← wolf terbaik pertama
14. X_β ← wolf terbaik kedua
15. X_δ ← wolf terbaik ketiga
16.
17. **while** (t < maxIter)
18. **for** each wolf
19. perbarui posisi wolf sekarang (pers. 5, 6, 7)
20. **end for**
21. perbarui a, A, da C (pers. 8, 3, 4)
22. hitung nilai *fitness* (pers. 11) seluruh wolf
23. perbarui X_α, X_β, dan X_δ
24. t=t+1
25. **end while**
26. return X_α

2.3 Himpunan data

Penelitian ini menggunakan tiga himpunan data yaitu COCOMO NASA93 (DS1), NASA81 (DS2), dan NASA10 (DS3). Ketiga himpunan data tersebut masing-masing terdiri dari 93, 63, dan 17 proyek serta sama-sama memiliki 26 atribut atau *effort driver*. Di proyek ke-40 pada DS2, terdapat data yang hilang (*missing data*) sehingga harus dibuang. Dengan demikian yang digunakan pada DS2 adalah sebanyak 62 proyek.

Untuk keperluan penelitian selanjutnya, himpunan data DS1 bisa diakses di http://promise.site.uottawa.ca/SERepository/datasets/cocomonasa_2.arff, DS2 di https://zenodo.org/records/268424, dan DS3 di (Menzies et al. 2017).

2.4 Rancangan eksperimen

Implementasi CUGWO pada estimasi COCOMO dilakukan sebanyak empat tahap sebagaimana ditunjukkan skemanya oleh Gambar 1. Pertama, setiap satu tupel atau *instance* dari himpunan data akan berperan sebagai data uji, sedangkan sisanya menjadi data latih. Pembagian himpunan data seperti ini dikenal sebagai teknik *leave one out cross validation* (LOOCV). LOOCV dipilih karena mampu memberikan hasil dengan *conclusion instability* yang lebih rendah dibanding teknik lain seperti *k-fold*, *3-way*, atau *10-way* (Kocaguneli & Menzies 2013).

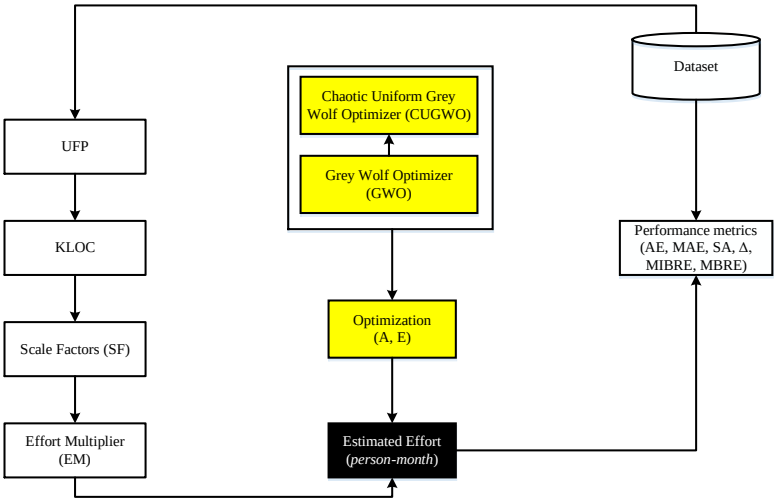
Tahap kedua adalah menghitung nilai *effort* menggunakan persamaan (9) dan (10).

$$Effort = A \times size^E \times \prod_{i=1}^{17} EM_i \quad (9)$$

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_j \quad (10)$$

dimana *A* adalah konstanta multiplikatif bernilai 2.94, *size* adalah ukuran perangkat lunak dalam *kloc*, *EM* adalah *effort multiplier*, dan *SF* adalah *scale factors*.

Baik *size*, *EM*, dan *SF* diambil dari *effort driver* atau atribut pada himpunan data. *Effort multiplier* (*EM*) terdiri dari atribut 'rely', 'data', 'cplx', 'ruse', 'docu', 'time', 'stor', 'pvol', 'acap', 'pcap', 'pcon', 'apex', 'plex', 'ltex', 'tool', 'site', dan 'sced'. *Scale factor* (*SF*) terdiri dari 'prec', 'flex', 'resl', 'team', dan 'pmat'. Sedangkan empat *effort driver* terakhir adalah 'kloc', 'effort', 'defects', dan 'month'.



Gambar 1. Skema rancangan eksperimen CUGWO COCOMO

Sebagai tahap ketiga, setiap kali model akan mengestimasi satu data uji, maka nilai *A* dan *B* akan dioptimasi oleh CUGWO hingga memperoleh hasil optimum. Agar memperoleh hasil terbaik, maka perlu melakukan *setting* parameter terlebih dahulu yaitu: ukuran populasi = 100 dan iterasi maksimum = 20. Adapun variabel desain yang digunakan memiliki rentang [0, 10] untuk parameter *A* dan [0.3, 2] untuk parameter *B*. Nilai *fitness* yang digunakan adalah *absolute error* dan nilai objektif adalah nilai estimasi *effort* yang dihitung menggunakan persamaan (11).

Tahap keempat adalah menghitung performa CUGWO pada estimasi COCOMO. Performa metode CUGWO diukur menggunakan empat teknik seperti yang disajikan oleh persamaan (11) hingga (14) yaitu *absolute error* (AE), *mean absolute error* (MAE), *standardized accuray* (SA), dan *effect size* (ES) (Langdon et al. 2016; Shepperd & MacDonell 2012).

$$\mathrm{AE} = |\mathrm{y}_i - \hat{\mathrm{y}}_i| \tag{11}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{AE}_i \tag{12}$$

$$\mathrm{SA}_{P_j} = 1 - \left(\frac{\mathrm{MAE}_{P_j}}{\overline{MAE}_{P_0}}\right) \mathrm{x}\ 100 \tag{13}$$

$$\Delta = \frac{\mathrm{MAE}_{P_j} - \overline{\mathrm{MAE}}_{P_0}}{\mathrm{S}_{P_0}} \tag{14}$$

dimana y_i adalah *effort* aktual yang merupakan salah satu atribut atau *effort driver* dari himpunan data, $\hat{\mathrm{y}}_i$ adalah estimasi *effort* hasil persamaan (9), dan *n* adalah jumlah *instance* atau tupel himpunan data. MAE_{P_j} dihasilkan dari model estimasi COCOMO + CUGWO. Sedangkan $\overline{MAE}_{P_0}$ diperoleh dari teknik P_0 yaitu *random guessing*, dan S_{P_0} merupakan standar deviasi yang dihasilkan oleh P_0.

Terakhir, dilakukan pengukuran dan analisis *diversity* terhadap metode yang diusulkan beserta pembandingnya. Pengukuran *diversity* mengikuti persamaan (15) dan (16) berikut yang diadopsi dari (Shen, Wei, & Zeng 2015; Shi & Eberhart 2008).

$$Div(S(t)) = \frac{1}{N_s}\sum_{i=1}^{N_s}\sqrt{\sum_{j=1}^{D_x}\left(x_{ij} - \overline{x_j(t)}\right)^2} \tag{15}$$

$$\overline{x_j(t)} = \frac{\left(\sum_{i=1}^{N_s} x_{ij}(t)\right)}{N_s} \tag{16}$$

dimana *Div* adalah *diversity*, dan *S* adalah *swarm* atau populasi. Sehingga $Div(S(t))$ adalah ukuran *diversity* populasi pada iterasi ke-*t*. Sedangkan $N_s = |S|$ adalah ukuran populasi, D_x adalah ukuran dimensi, x_{ij} adalah nilai ke-*j* dari kandidat solusi ke-*i*, dan $\overline{x_j(t)}$ adalah rerata nilai dimensi ke-*j* dari seluruh kandidat solusi.

3. HASIL DAN PEMBAHASAN

Metode yang diusulkan pada penelitian ini adalah *chaotic uniform grey wolf optimizer* (CUGWO). CUGWO membangkitkan dua kelompok populasi awal. Populasi pertama dibangkitkan secara acak dengan distribusi *uniform*, dan kelompok kedua dibangkitkan dengan menggunakan *chaos map*. Selanjutnya, kedua populasi tersebut diseleksi sehingga terbentuklah satu populasi awal yang lebih baik. Dengan pendekatan ini, *diversity* populasi diharapkan menjadi lebih baik sehingga bisa meningkatkan performa pencarian solusi optimum.

Untuk mengetahui *chaotic maps* yang terbaik, maka terlebih dahulu dilakukan komparasi antara tujuh *chaotic maps* yaitu, *chebyshev*, *singer*, *sine*, *logistic*, *circle*, *gauss*, dan *sinusoidal map*. Tiap *chaotic map* akan digunakan pada CUGWO dengan jumlah *runs* sebanyak 30 kali. Kinerja *chaotic map* terbaik ditentukan berdasarkan metrik akurasi MAE.

Tabel 1. Hasil komparasi nilai MAE tiap *chaotic maps* setelah dijalankan 30 kali

Chaotic Map	Mean			Best			Worst			St. Dev			Friedman Test		
	DS1	DS2	DS3	DS1	DS2	DS3	DS1	DS2	DS3	DS1	DS2	DS3	DS1	DS2	DS3
Chebyshev	38.7	33	62.8	36.3	29.80	57.9	41.2	35.71	79.8	1.0	1.71	4.08	6.9	5.8	6.9
Singer	19.7	24.8	34.1	18.6	21.50	29.5	20.6	26.27	36.4	0.4	0.95	1.75	3.9	3.9	3.6
Sine	19.2	21.8	31.6	17.8	20.69	29.6	21.2	24.97	42.6	0.7	1.10	2.92	2.8	2.2	2.2
Logistic	18.4	21.9	32.2	17.3	15.23	24.6	20.0	26.14	34.4	0.6	2.05	1.75	2.3	2.9	3.0
Circle	13.3	13.9	24.3	12.8	12.97	23.2	15.0	14.51	27.2	0.6	0.43	0.91	1.0	1.0	1.0
Gauss	31.0	32.2	58.1	30.1	27.17	44.3	33.2	36.43	68.9	0.6	2.13	4.02	5.5	5.2	5.4
Sinusoidal	33.2	36.2	57.5	31.6	34.78	50.9	40.5	43.89	59.4	1.9	1.75	2.05	5.5	6.9	5.6

Tabel 1 menyajikan hasil komparasi nilai MAE tiap *chaotic map* setelah dijalankan 30 kali. Bisa dilihat bahwa *circle map* merupakan teknik *chaos* terbaik untuk ketiga himpunan data. Estimasi COCOMO+CUGWO dengan teknik *circle map* yang memberikan rerata MAE sebesar 13.3 (DS1), 13.9 (DS2), dan 24.3 (DS3) merupakan kombinasi terbaik untuk ketiga himpunan data yang digunakan pada penelitian ini. Hasil ini juga diperkuat oleh uji statistik Friedman Test seperti yang ditunjukkan oleh tabel tersebut

Hasil yang diperoleh ini ternyata berbeda dengan penelitian lain yang menggunakan *logistic map* (Ibrahim, Elaziz, & Lu 2018), *tent map*, hingga *sinusoidal map* (Ardiansyah, Ferdiana, & Permanasari 2022; Mirjalili & Gandomi 2017). Perbedaan ini bisa dimaklumi karena karakteristik data dan permasalahan yang diangkat berbeda-beda. Sebagai contoh, penelitian (Ibrahim, Elaziz, & Lu 2018) tersebut menguji algoritmanya menggunakan data citra galaksi untuk keperluan klasifikasi citra dengan fungsi objektif berupa *accuracy* dan *stability*. Penelitian yang cukup mirip adalah yang dilakukan oleh (Ardiansyah, Ferdiana, & Permanasari 2022), yaitu sama-sama estimasi *effort* perangkat lunak. Namun, perbedaan mendasarnya adalah pada metode estimasinya yaitu *use case points* (UCP) dan himpunan datanya yang memiliki karakteristik berbeda. Hal ini perlu ditekankan, karena untuk metode estimasi dan himpunan data yang berbeda belum tentu cocok dengan teknik *chaotic* tertentu. Oleh karena itu, mengevaluasi seluruh teknik *chaotic* untuk mendapatkan teknik yang terbaik adalah hal yang penting dilakukan.

Setelah memperoleh *chaotic map* terbaik, selanjutnya dilakukan eksperimen untuk membandingkan akurasi estimasi COCOMO+CUGWO dengan penelitian-penelitian sebelumnya. Ada dua algoritma pembanding yang digunakan yaitu COGWO2D (Ibrahim, Elaziz, & Lu 2018), dan GWO (Mirjalili, Mirjalili, & Lewis 2014). Kedua algoritma tersebut bersama dengan CUGWO juga digunakan pada metode estimasi *effort* COCOMO, sehingga ada tiga algoritma yang akan dianalisis.

Analisis, perbandingan dan pembahasan ketiga algoritma tersebut dilakukan dengan menggunakan *baseline* yaitu metode *random guessing*. Tabel 2 menunjukkan hasil perbandingannya. Berdasarkan metrik *mean absolute error* (MAE), terlihat bahwa ketiga metode menunjukkan performa yang unggul dibandingkan dengan *random guessing*. Hasil ini menunjukkan bahwa ketiganya memang benar-benar melakukan estimasi dan bukan secara kebetulan (*coincidence*). Perlu diingat bahwa nilai MAE yang makin kecil adalah makin baik.

Metrik *standardize accuracy* (SA) juga menunjukkan bahwa ketiga metode memiliki akurasi yang jauh lebih besar dibanding *random guessing*. Terlihat bahwa ketiga algoritma memiliki akurasi yang sama-sama hampir 100. Sedangkan *effect size* (ES) untuk ketiga metode berdasarkan tiga himpunan data adalah relatif sama yaitu 1.7. Hasil ini menunjukkan bahwa ketiga model memberikan perbaikan estimasi yang substansial.

Setelah mengetahui performa CUGWO terhadap *random guessing*, maka selanjutnya dilakukan komparasi antara COGWO2D dan CUGWO dengan GWO sebagai *baseline*-nya. Berdasarkan Tabel 3, terlihat bahwa CUGWO memiliki SA dan ES lebih besar dibanding COGWO2D menggunakan ketiga himpunan data. Bahkan, SA untuk COGWO2D bernilai negatif pada DS2 yang menunjukkan bahwa performa akurasinya masih di bawah *baseline* yaitu GWO.

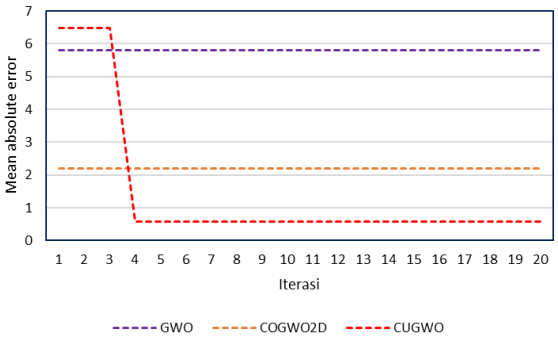
Performa ES yang dihasilkan CUGWO pada ketiga himpunan data menunjukkan skala medium ($ES \approx 0.5$). Untuk COGWO2D, hanya DS3 yang masuk skala medium, sedangkan DS1 dan DS2 masuk skala *small* ($ES \approx 0.2$). Artinya, pada DS1 dan DS2, estimasi yang dilakukan oleh COGWO2D kemungkinan besar dihasilkan secara kebetulan.

Tabel 2. Akurasi estimasi COCOMO dengan tiga algoritma metaheuristik dengan *random guessing* sebagai *baseline*

Algoritma	SA			ES			MAE		
	DS1	DS2	DS3	DS1	DS2	DS3	DS1	DS2	DS3
GWO	99.99	99.99	99.97	1.74	1.73	1.72	31.50	38.47	138.42
COGWO2D	99.99	99.99	99.99	1.74	1.74	1.73	25.68	45.89	31.99
CUGWO	100	100	99.99	1.74	1.72	1.73	**12.77**	**16.01**	**30.02**

Tabel 3. Akurasi estimasi COCOMO menggunakan COGWO2D dan CUGWO, dengan GWO sebagai *baseline*

Algoritma	SA			ES		
	DS1	DS2	DS3	DS1	DS2	DS3
COGWO2D	18.48	-19.28	76.89	0.13	0.10	0.45
CUGWO	59.45	58.38	78.31	0.41	0.30	0.45



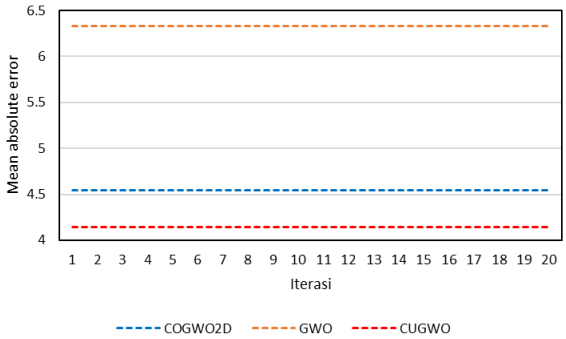
Gambar 2. Perbandingan konvergensi tiga metode optimasi untuk estimasi COCOMO pada DS1

Hasil estimasinya berarti tidak menarik, dan tidak memberi pengaruh yang baik dari sisi praktikal di dunia nyata. Performa yang dihasilkan oleh CUGWO pada DS2 terlihat sedikit lebih rendah dibanding pada saat menggunakan DS1 dan DS3. Apalagi melihat nilai ES-nya sebesar 0.30 yang hanya sedikit lebih besar dari batas marjinal skala *small*, menunjukkan bahwa GWO sebagai *baseline* memberikan perlawanan cukup sengit dan sanggup bersaing dengan CUGWO. Namun, karena modifikasi yang dilakukan terhadap pembentukan populasi oleh CUGWO, pada akhirnya membuatnya tetap lebih unggul dibanding *baseline*-nya.

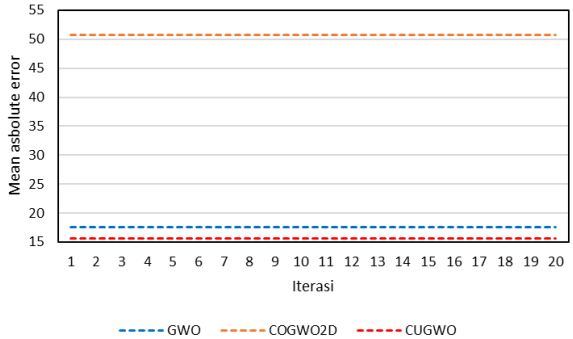
Analisis berikutnnya adalah efisiensi ketiga metode untuk menemukan solusi optimal dari fungsi objektif estimasi COCOMO. Gambar 2, 3, dan 4 menunjukkan grafik perbandingan konvergensi ketiga metode dengan menggunakan data proyek ke-2 dari masing-masing himpunan data.

Berdasarkan Gambar 2 tersebut bisa dilihat tahwa GWO dan COGWO2D langsung konvergen sejak iterasi awal dengan masing-masing MAE bernilai 5.8 dan 2.19. Sedangkan CUGWO, pada empat iterasi awal memiliki MAE 6.47, namun setelahnya langsung konvergen ke 0.58 hingga terasi selesai.

Untuk Gambar 3 dan 4 menunjukkan pola konvergensi yang sama, yaitu ketiga metode langsung konvergen sejak iterasi pertama. Bisa dilihat bahwa GWO merupakan algoritma yang memiliki konvergensi ke nilai MAE paling buruk. Sedangkan CUGWO dan CUGWO2D berturut-turut merupakan dua algoritma yang memiliki konvergensi ke nilai MAE yang baik.



Gambar 3. Perbandingan konvergensi tiga metode optimasi untuk estimasi COCOMO pada DS2



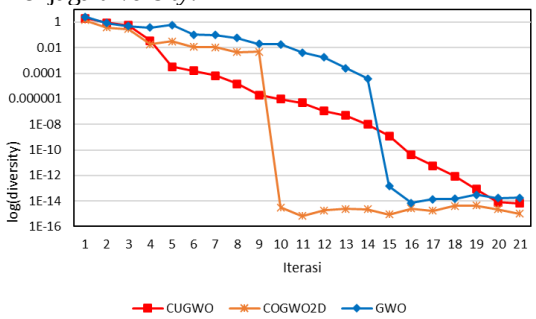
Gambar 4. Perbandingan konvergensi tiga metode optimasi untuk estimasi COCOMO pada DS3

Analisis terakhir adalah berupa analisis *diversity* terhadap solusi optimum yang dihasilkan oleh ketiga algoritma untuk metode estimasi *effort* COCOMO.

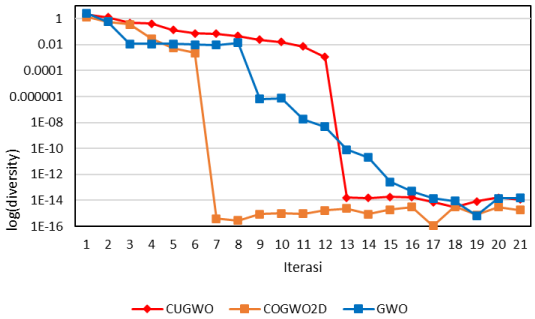
Pada Gambar 5, terlihat bahwa *diversity* COGWO2D paling cepat menuju nilai terkecil, disusul GWO dan CUGWO. GWO sebenarnya relatif bagus mempertahankan *diversity*-nya hingga iterasi ke-14. Namun, GWO gagal mempertahankannya karena *diversity*-nya langsung jatuh hampir mendekati COGWO2D sejak iterasi ke-15. Sedangkan CUGWO, seperti yang ditunjukkan oleh gambar, sangat stabil menjaga *diversity*-nya dari awal hingga akhir iterasi. Hasil ini menunjukkan bahwa CUGWO mampu mengeksplorasi solusi-solusi lain sehingga bisa terhindar dari jebakan optimum lokal. Dengan

demikian, pada DS1 ini CUGWO memiliki *diversity* yang unggul dibanding GWO dan COGWO2D.

Pada Gambar 6 juga menunjukkan tren yang hampir sama seperti Gambar 5. Terlihat bahwa *diversity* CUGWO2D langsung mengecil di iterasi ke-7 yang menunjukkan ketidakmampuannya menjaga *diversity*.



Gambar 5. Diversity kandidat solusi yang dihasilkan oleh tiga metode optimasi pada estimasi COCOMO pada DS1

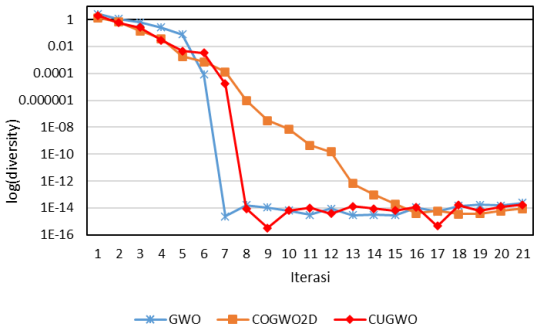


Gambar 6. Diversity kandidat solusi yang dihasilkan oleh tiga metode optimasi pada estimasi COCOMO pada DS2

Adapun *diversity* GWO dan CUGWO cukup bersaing pada DS2 ini. Terlihat bahwa *diversity* GWO relatif stagnan pada 8 iterasi pertama, dan mulai menurun perlahan hingga akhir iterasi. *Diversity* CUGWO menunjukkan tren penurunan *diversity* yang relatif stabil mulai iterasi pertama hingga ke-12. Namun, sejak iterasi ke-13 *diversity*-nya langsung jatuh mendekati COGWO2D dan relatif stagnan hingga iterasi berakhir.

Kondisi berkebalikan dialami oleh COGWO2D pada DS3. Seperti yang ditunjukkan oleh Gambar 7, COGWO2D mampu menjaga *diversity*-nya sejak iterasi awal, dan mulai stabil menjelang akhir iterasi. Begitu pula yang dialami oleh GWO dan CUGWO. Keduanya memiliki pola yang hampir mirip seperti pada Gambar 6. Namun, CUGWO relatif sedikit lebih stabil dibanding GWO.

Keunggulan CUGWO menjaga *diversity*-nya pada DS1, dan mampu dipertahankannya bersama GWO di DS2, kemungkinan besar karena kedua himpunan data tersebut memiliki data proyek yang relatif besar. Sedangkan, kondisi sebaliknya yang dialami oleh GWO dan CUGWO pada DS3, kemungkinan disebabkan karena kecilnya data proyek yang dimiliki oleh himpunan data tersebut. Pada kondisi seperti ini, maka COGWO2D relatif lebih unggul



Gambar 7. Diversity kandidat solusi yang dihasilkan oleh tiga metode optimasi pada estimasi COCOMO pada DS3

4. KESIMPULAN

Penelitian ini bertujuan untuk mengeksplorasi berbagai teknik *chaotic* pada algoritma optimasi yang diusulkan yaitu CUGWO agar *diversity* populasi dan performa akurasi estimasi bisa meningkat. Eksplorasi ini dilakukan karena adanya permasalahan *diversity* populasi yang dialami GWO selama ini. Untuk mengatasi masalah tersebut, penelitian ini memodifikasi GWO dengan membangkitkan dua populasi awal menggunakan *chaotic uniform initialization*.

Berdasarkan hasil eksperimen dapat diketahui bahwa metode yang diusulkan berhasil meningkatkan *diversity* dan performa akurasi estimasi COCOMO. Hasil ini menunjukkan bahwa *diversity* yang baik akan turut memberikan konvergensi yang baik pula. Peningkatan performa akurasi estimasi dicapai ketika teknik *chaotic* yang digunakan adalah *circle map*, dan karakteristik himpunan datanya seperti pada DS1, DS2, dan DS3.

Metode CUGWO memang telah terbukti berhasil meningkatkan performa akurasi estimasi COCOMO. Akan tetapi, keberhasilan ini memunculkan tantangan lain terkait performa CUGWO khususnya pada DS2 yang memiliki *missing value*. Oleh karena itu, penelitian berikutnya perlu mengkaji teknik-teknik penanganan *missing value* agar performa akurasi estimasi COCOMO tetap menjanjikan.

Selain saran untuk penelitian berikutnya, hasil penelitian ini bisa dijadikan pertimbangan bagi para manajer proyek untuk diterapkan dalam proyek-proyek perangkat lunak mereka selanjutnya.

DAFTAR PUSTAKA

ALI, S.S. et al. 2023. “Heterogeneous Ensemble Model to Optimize Software Effort Estimation Accuracy.” *IEEE Access* 11(February): 27759–92.

AMARENDRA et al. 2023. *Forecast Analysis: Enterprise Application Software, Worldwide*. https://www.gartner.com/en/documents/5012231.

ARDIANSYAH, A., Ferdiana, R., & Permanasari, A.E., 2022. “MUCPSO: A Modified Chaotic Particle Swarm Optimization with Uniform

Initialization for Optimizing Software Effort Estimation." *Applied Sciences* 12(3): 1081. https://www.mdpi.com/2076-3417/12/3/1081.

BOEHM, B., ABTS, C., & CHULANI, S., 2000. "Software Development Cost Estimation Approaches – A Survey." *Annals of Software Engineering* 10(2000): 177–205.

CAI, Z. et al. 2019. "Quadrotor Trajectory Tracking and Obstacle Avoidance by Chaotic Grey Wolf Optimization-Based Active Disturbance Rejection Control." *Mechanical Systems and Signal Processing* 128: 636–54. https://doi.org/10.1016/j.ymssp.2019.03.035.

CHHABRA, S., & SINGH, H., 2020. "Optimizing Design of Fuzzy Model for Software Cost Estimation Using Particle Swarm Optimization Algorithm." *International Journal of Computational Intelligence and Applications* 19(1): 1–16.

FARIS, H. et al. 2018. "Knowledge-Based Systems An Efficient Binary Salp Swarm Algorithm with Crossover Scheme for Feature Selection Problems." *Knowledge-Based Systems* 154(January): 43–67. https://doi.org/10.1016/j.knosys.2018.05.009.

HAO, P., & SOBHANI, B., 2021. "Application of the Improved Chaotic Grey Wolf Optimization Algorithm as a Novel and Efficient Method for Parameter Estimation of Solid Oxide Fuel Cells Model." *International Journal of Hydrogen Energy* 46(73): 36454–65. https://doi.org/10.1016/j.ijhydene.2021.08.174.

IBRAHIM, R.A., ELAZIZ, M.A., & LU, S., 2018. "Chaotic Opposition-Based Grey-Wolf Optimization Algorithm Based on Differential Evolution and Disruption Operator for Global Optimization." *Expert Systems with Applications* 108: 1–27. https://doi.org/10.1016/j.eswa.2018.04.028.

JORGENSEN, M., & SHEPPERD, M., 2007. "A Systematic Review of Software Development Cost Estimation Studies." *IEEE Transactions on Software Engineering* 33(1): 33–53.

KOCAGUNELI, E., & MENZIES, T., 2013. "Software Effort Models Should Be Assessed via Leave-One-out Validation." *Journal of Systems and Software* 86(7): 1879–90. http://dx.doi.org/10.1016/j.jss.2013.02.053.

KOHLI, M., & ARORA, S., 2018. "Chaotic Grey Wolf Optimization Algorithm for Constrained Optimization Problems." *Journal of Computational Design and Engineering* 5(4): 458–72. https://doi.org/10.1016/j.jcde.2017.02.005.

LANGDON, W.B., DOLADO, J., SARRO, F., & HARMAN, M., 2016. "Exact Mean Absolute Error of Baseline Predictor, MARP0." *Information and Software Technology* 73: 16–18. http://dx.doi.org/10.1016/j.infsof.2016.01.003.

LANGSARI, K., & SARNO, R., 2017. "Optimizing Effort and Time Parameters of Cocomo Ii Estimation Using Fuzzy Multi-Objective PSO." *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)* 4(September): 466–71.

LI, Y., LIN, X., & LIU, J., 2021. "An Improved Gray Wolf Optimization Algorithm to Solve Engineering Problems." *Sustainability (Switzerland)* 13(6): 1–23.

MENZIES, T. et al. 2017. "Negative Results for Software Effort Estimation." *Empirical Software Engineering* 22(5): 2658–83. http://dx.doi.org/10.1007/s10664-016-9472-2.

MIRJALILI, S., & GANDOMI, A.H., 2017. "Chaotic Gravitational Constants for the Gravitational Search Algorithm." *Applied Soft Computing Journal* 53: 407–19. http://dx.doi.org/10.1016/j.asoc.2017.01.008.

MIRJALILI, S., MIRJALILI, S.M., & LEWIS, A., 2014. "Grey Wolf Optimizer." *Advances in Engineering Software* 69: 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

PARWITA, I.M.M., SARNO, R., & PUSPANINGRUM, A., 2017. "Optimization of COCOMO II Coefficients Using Cuckoo Optimization Algorithm to Improve the Accuracy of Effort Estimation." In *2017 11th International Conference on Information & Communication Technology and System (ICTS)*, IEEE, 99–104. http://ieeexplore.ieee.org/document/8265653/.

PUTRI, R.R., SIAHAAN, D.O., & FATICHAH, C., 2022. "Improving the Accuracy of COCOMO II Using Extended Gamma GWO." *2022 International Seminar on Intelligent Technology and Its Applications: Advanced Innovations of Electrical Systems for Humanity, ISITIA 2022 - Proceeding*: 145–50.

PUTRI, R.R., SIAHAAN, D.O., & FATICHAH, C., 2021. "Improve the Accuracy of Software Project Effort and Cost Estimates in COCOMO II Using GWO." *Proceedings - International Conference on Informatics and Computational Sciences* 2021-Novem: 128–33.

SACHAN, R.K. et al. 2016. "Optimizing Basic COCOMO Model Using Simplified Genetic Algorithm." *Procedia Computer Science* 89: 492–98. http://dx.doi.org/10.1016/j.procs.2016.06.107

.

SHELDON, J.W., 1994. "Recollections of the First Software Company." *IEEE Annals of the History of Computing* 16(2): 65–71.

SHEN, Y., WEI, L., & ZENG, C., 2015. 9140 Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) *Swarm Diversity Analysis of Particle Swarm Optimization*. eds. Ying Tan et al. Cham: Springer International Publishing. https://link.springer.com/10.1007/978-3-319-20466-6.

SHEPPERD, M., & MACDONELL, S., 2012. "Evaluating Prediction Systems in Software Project Estimation." *Information and Software Technology* 54(8): 820–27. https://linkinghub.elsevier.com/retrieve/pii/S095058491200002X.

SHI, Y., & EBERHART, R.C., 2008. "Population Diversity of Particle Swarms." *2008 IEEE Congress on Evolutionary Computation, CEC 2008*: 1063–67.

ZAKARIA, N.A. et al. 2021. "Optimized COCOMO Parameters Using Hybrid Particle Swarm Optimization." *International Journal of Advances in Intelligent Informatics* 7(2): 177. http://ijain.org/index.php/IJAIN/article/view/583.