Jurnal Informatika: Jurnal pengembangan IT

Vol. 11, No. 1, 22026
ISSN : 2477-5126, DOI:10.30591/jpit.v11i1.10043

Optimasi Faktor Friksi dan Dinamis dengan Hibrida GA–ACO pada Estimasi Usaha Perangkat Lunak Agile

Yusril Mahendri [1], Irving Vitra Paputungan [2], Novi Setiani [3]
[123]Magister Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia, Indonesia
[1]23917030@students.uii.ac.id, [2]045230101@uii.ac.id, [3]novi.setiani@uii.ac.id

Info Artikel

Riwayat Artikel:

Received 2025-12-28
Revised 2026-01-31
Accepted 2026-02-06

Abstract *– Effort estimation remains a critical challenge in Agile Software Development due to the high dynamics of requirement changes and the reliance on friction factors (FF) and dynamic factors (DF) that are inherently subjective, often leading to significant deviations between estimated and actual project effort. This study aims to improve the accuracy of Agile software effort estimation by optimizing FF and DF parameters using a hybrid metaheuristic approach based on Genetic Algorithm and Ant Colony Optimization (GACO). The proposed method integrates a pheromone-based guided search mechanism from Ant Colony Optimization to generate high-quality initial populations, which are subsequently refined through the evolutionary process of Genetic Algorithm to achieve more stable and systematic parameter optimization. Experimental evaluation was conducted using two datasets, namely the Ziauddin dataset representing Agile projects and the Maxwell dataset encompassing cross-domain software projects. The results demonstrate that the GACO approach consistently outperforms the conventional Genetic Algorithm, as indicated by a substantial reduction in Mean Absolute Error from 616.38 to 354.81. Furthermore, statistical validation using the Wilcoxon Signed-Rank Test confirms that the performance difference between the two approaches is statistically significant. These findings indicate that integrating Ant Colony Optimization into Genetic Algorithm effectively enhances the accuracy, stability, and robustness of software effort estimation, thereby supporting more reliable resource planning in Agile software development.*

Keywords: *Agile Software Development; Ant Colony Optimization; Effort Estimation; Genetic Algorithm; Hybrid Model; Metaheuristic Optimization.*

Corresponding Author:
Irving Vitra Paputungan
Email: 045230101@uii.ac.id

Abstrak *– Estimasi usaha merupakan permasalahan utama dalam Agile Software Development karena tingginya dinamika perubahan kebutuhan dan ketergantungan pada penentuan faktor friksi dan dinamis yang bersifat subjektif, sehingga sering menyebabkan deviasi signifikan antara estimasi dan usaha aktual proyek. Penelitian ini bertujuan untuk meningkatkan akurasi estimasi usaha perangkat lunak agile melalui optimasi parameter faktor friksi dan dinamis menggunakan pendekatan metaheuristik hibrida algoritma genetika dan Ant Colony Optimization (GACO). Metode yang diusulkan mengintegrasikan mekanisme pencarian terarah berbasis feromon pada Ant Colony Optimization sebagai tahap inisialisasi populasi berkualitas, yang selanjutnya disempurnakan melalui proses evolusi algoritma genetika untuk menemukan kombinasi parameter optimal secara lebih stabil dan sistematis. Evaluasi eksperimental dilakukan menggunakan dua dataset, yaitu dataset Ziauddin yang merepresentasikan proyek agile dan dataset Maxwell yang mencakup proyek perangkat lunak lintas domain. Hasil eksperimen menunjukkan bahwa pendekatan model hibrida algoritma genetika dan Ant Colony Optimization secara konsisten menghasilkan estimasi usaha yang lebih akurat dibandingkan algoritma genetika tunggal, ditunjukkan oleh penurunan nilai Mean Absolute Error dari 616.38 menjadi 354.81. Pengujian statistik menggunakan Wilcoxon Signed-Rank Test juga menunjukkan perbedaan yang signifikan antara kedua pendekatan. Berdasarkan temuan tersebut, dapat disimpulkan bahwa integrasi Ant Colony Optimization ke dalam algoritma genetika efektif dalam meningkatkan akurasi, stabilitas, dan ketahanan estimasi usaha, sehingga berpotensi mendukung perencanaan sumber daya yang lebih andal pada pengembangan perangkat lunak dengan pendekatan agile.*

Kata Kunci*: Agile Software Development; Ant Colony Optimization; Effort Estimation; Genetic Algorithm; Hybrid Model; Metaheuristic Optimization.*

I. PENDAHULUAN

Dalam sektor kehidupan, diantaranya seperti bisnis, pendidikan hingga pemerintahan, pengembangan perangkat lunak telah menjadi elemen krusial pada aspek kehidupan modern saat ini. Pengembangan perangkat lunak umumnya mengacu pada kerangka kerja, yaitu *Software Development Life Cycle (SDLC).* SDLC mencakup beberapa tahapan, diantaranya seperti perencanaan, analisa, desain, implementasi, pengujian dan pemeliharaan. kerangka kerja tersebut dapat menggunakan berbagai jenis pendekatan model yang sistematis dan berurutan, diantaranya seperti waterfall, v-model, inkremental model dan *Agile Software Development* (ASD), dengan tujuan menjamin kualitas dan keberlanjutan perangkat lunak.

Pengembangan perangkat lunak diera sekarang, kompleksitas kebutuhannya sering meningkat. Oleh karena itu model pengembangan tradisional seperti *Waterfall* dianggap kurang adaptif terhadap perubahan yang cepat dan dinamis. Solusi model yang lebih fleksibel, iteratif dan kolaboratif dalam merespon dinamika perubahan kebutuhan pengguna adalah dengan menerapkan pendekatan model agile. Meskipun a*gile* menawarkan fleksibilitas yang tinggi, pendekatan ini menghadirkan tantangan tersendiri, terutama dalam aspek estimasi usaha. Estimasi usaha merupakan proses penting yang memprediksi sumber daya, waktu, dan tenaga kerja yang dibutuhkan untuk menyelesaikan suatu proyek perangkat lunak. Akurasi estimasi sangat menentukan keberhasilan proyek, karena kesalahan dalam estimasi dapat menyebabkan keterlambatan, pembengkakan biaya, dan ketidaksesuaian hasil dengan harapan pengguna.

Pada pengembangan perangkat lunak dengan pendekatan model *agile*, tingkat ketidakpastian yang tinggi dan perubahan kebutuhan yang bersifat dinamis menyebabkan proses estimasi usaha menjadi lebih kompleks. Kompleksitas tersebut dipengaruhi oleh dua parameter utama, yaitu Faktor friksi dan faktor dinamis. Faktor friksi merepresentasikan hambatan dalam pengembangan seperti komunikasi tim dan perubahan kebutuhan, sedangkan faktor dinamis menggambarkan variabilitas proyek yang bersumber dari faktor eksternal dan internal seperti perubahan lingkungan kerja atau evolusi teknologi.

Sejumlah penelitian terdahulu telah mengkajikan estimasi usaha dengan mempertimbangkan parameter faktor friksi dan dinamis. Penelitian terdahulu yang dilakukan oleh Ziauddin[6] telah memperkenalkan model estimasi usaha berbasis parameter faktor friksi dan dinamis pada proyek model pendekatan *agile*, dengan hasil cukup akurat (MMRE 0.32 dan PRED(25) 66,67%). Namun, pendekatan tersebut masih memiliki celah akurasi karena penentuan nilai parameter faktor friksi dan dinamis sangat bergantung pada penilaian subjektif (estimator manusia). Penilaian subjektif tersebut akan berpotensi menurunkan konsistensi pada hasil estimasi usaha. Untuk mengatasi keterbatasan tersebut, Yusril Mahendri[8] dengan menerapkan *Genetic Algorithm* (GA) untuk mengoptimalkan kedua parameter tersebut, dan berhasil meningkatkan akurasi estimasi secara signifikan (MAE menurun dari 56.42 menjadi 12.92, SA meningkat dari 17.15% menjadi 72.35%). Meskipun demikian, penggunaan GA tunggal masih menghadapi kendala berupa konvergensi dini dan sensitivitas terhadap inisialisasi populasi awal yang kurang baik, sehingga hasil optimasi dapat terjebak pada solusi lokal.

Dalam ranah optimasi, pendekatan algoritma hibrida telah banyak diteliti untuk mengatasi kelemahan algoritma tunggal. Terdapat penelitian yang menunjukkan potensi pendekatan hibrida untuk meningkatkan performa optimasi, yaitu penelitian dari Zukhri dan Paputungan[9]. Pada penelitian tersebut mengusulkan kombinasi antara *Genetic Algorithm* dan *Ant Colony Optimization* (GACO). Hibrida algoritma tersebut guna untuk menyelesaikan *Travelling Salesman Problem (TSP)*, dan memperoleh peningkatan kualitas solusi 5–10% dibandingkan algoritma tunggal. Pendekatan ini membuktikan bahwa integrasi mekanisme eksplorasi ACO dengan mekanisme evolusi GA mampu menjaga keragaman solusi dan menghindari konvergensi prematur. Selain itu, pendekatan dengan metode algoritma *Ant Colony Optimization* juga telah terbukti efektif sebagai mekanisme pencarian terarah (*guided search*) dalam berbagai permasalahan optimasi, termasuk penentuan solusi optimal berbasis graf [24]. Dalam konteks estimasi usaha perangkat lunak, Fachruddin dan Pratama [23] menegaskan bahwa pemilihan dan pengolahan parameter proyek memiliki pengaruh signifikan terhadap akurasi estimasi, sehingga optimasi parameter menjadi aspek yang sangat penting.

Berdasarkan kajian tersebut, dapat diidentifikasi bahwa penelitian mengenai optimasi parameter friksi dan dinamis dalam estimasi usaha perangkat lunak dengan pendekatan *agile* masih memiliki keterbatasan. Secara khusus, belum terdapat penelitian yang menerapkan pendekatan hibrida GA-ACO untuk mengoptimalkan kedua parameter tersebut. Selain itu, permasalahan inisialisasi populasi awal pada GA yang kurang optimal masih menjadi tantangan yang belum sepenuhnya teratasi. Inisialisasi populasi awal yang buruk pada GA sering kali menyebabkan hasil yang tidak optimal.

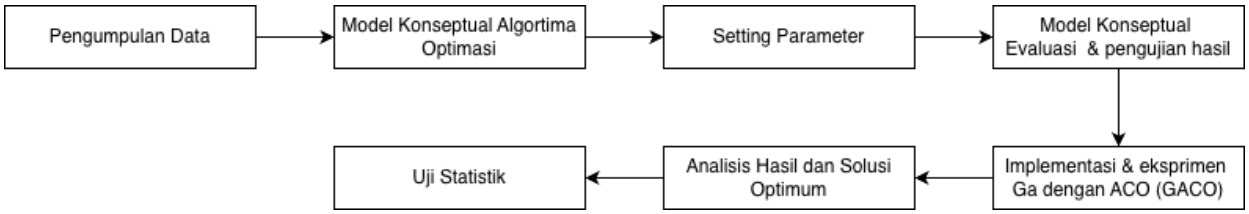
Oleh karena itu, penelitian ini bertujuan untuk mengembangkan dan mengevaluasi model estimasi usaha perangkat lunak dengan pendekatan *agile* berbasis optimasi parameter friksi dan dinamis menggunakan pendekatan hibrida algoritma genetika (GA) dan *Ant Colony Optimization* (ACO). Pendekatan ini memanfaatkan mekanisme eksplorasi ACO untuk membentuk populasi awal GA yang lebih beragam, sehingga diharapkan mampu memperluas ruang pencarian solusi dan mengurangi risiko konvergensi dini.

Kebaruan penelitian ini terletak pada penerapan integrasi GA-ACO secara khusus untuk optimasi parameter friksi dan dinamis dalam estimasi usaha perangkat lunak dengan pendekatan *agile*. Berbeda dengan penelitian-penelitian terdahulu yang menggunakan GA secara tunggal atau mengombinasikan GA dan ACO untuk domain optimasi lain, penelitian ini memanfaatkan ACO digunakan sebagai mekanisme pembentukan populasi awal berbasis jejak feromon, yang belum pernah diterapkan dalam konteks estimasi usaha perangkat lunak dengan pendekatan *agile*.

Kontribusi penelitian ini meliputi, diantaranya ialah pengusulan model estimasi usaha perangkat lunak dengan pendekatan *agile* berbasis optimasi parameter friksi dan dinamis menggunakan pendekatan hibrida GA-ACO, pengembangan mekanisme inisialisasi populasi awal yang adaptif berbasis ACO untuk meningkatkan performa GA, serta penyajian bukti empiris bahwa pendekatan hibrida yang diusulkan mampu meningkatkan akurasi estimasi usaha secara signifikan dibandingkan dengan metode GA tunggal.

II. METODE

Eksperimen kuantitatif komputasional adalah metode yang digunakan pada penelitian ini. Metode pendekatan tersebut digunakan untuk mengembangkan dan mengevaluasi model optimasi estimasi usaha perangkat lunak berbasis kombinasi *Genetic Algorithm* (GA) dan *Ant Colony Optimization* (ACO). Fokus utama penelitian ini adalah peningkatan akurasi estimasi usaha dalam pengembangan perangkat lunak dengan pendekatan *Agile Software Development* (ASD) melalui optimasi dua parameter utama, yaitu faktor friksi dan dinamis. Secara sistematis rancangan penelitian ini ditunjukkan pada gambar 1, yang menggambarkan tahapan penelitian dari pengumpulan data hingga uji statistik.



Gambar 1. Tahapan Penelitian

Pengumpulan data adalah proses tahapan pertama pada penelitian ini, yang dilakukan dengan menelaah berbagai literatur dan sumber data terdahulu yang relevan dengan estimasi usaha perangkat lunak berbasis metodologi agile. Karena penelitian ini bersifat eksperimen komputasional, maka data yang digunakan berasal dari sumber sekunder, bukan hasil pengumpulan data lapangan. Penelitian ini menggunakan dua dataset publik, yakni dataset Ziauddin[6] dan dataset Maxwell[15] Kedua dataset tersebut dipilih karena telah digunakan secara luas dalam penelitian estimasi usaha dan merepresentasikan berbagai karakteristik proyek perangkat lunak, baik yang bersifat *agile* maupun konvensional. Model konseptual algoritma optimasi adalah proses tahapan ke-dua, yang mendefinisikan rancangan integrasi antara algoritma genetika dan *Ant Colony Optimization*. Setelah tahapan model konseptual tersebut, dilakukan setting parameter. Setting parameter adalah tahapan ke-tiga, yang bertujuan untuk memastikan model bekerja secara optimal saat diimplementasikan. Untuk menghasilkan performa estimasi usaha yang sesuai dengan harapan, maka perlu dilakukan model konseptual evaluasi. Model konseptual evaluasi adalah tahapan ke-empat, yang terdiri dari beberapa teknik, diantaranya seperti *Absolute Error* (AE), *Mean Absolute Error* (MAE), *Mean Balanced Relative Error* (MBRE), *Mean Inverted Balanced Relative Error* (MIBRE), *Standardized Accuracy* (SA), dan *Effect Size* (ES). Implementasi dan eksprimen algoritma genetika dengan *Ant Colony Optimization* adalah tahapan ke-lima. Proses tahapan ini akan menjalankan kedua algoritma tersebut untuk mencari kombinasi parameter yang optimal dan estimasi usaha yang akurat. Hasil yang didapatkan dari proses tahapan ke-lima dilakukan Analisa dan mengidentifikasi solusi optimum. Hasil analisa dan solusi optimum adalah tahapan ke-enam, yang dilakukan dengan cara membandingkan perfoma algoritma genetika tunggal, algoritma genetika dengan *Ant Colony Optimization* (GACO) berdasarkan metrik evaluasi. Proses Terakhir adalah tahapan ke-tujuh, yaitu uji statistik. Uji statistik yang digunakan pada penelitian ini adalah *wilcoxon signed test rank*.

A. Pengumpulan Data

Pengumpulan data dilakukan melalui metode dokumentasi, yaitu pemanfaatan catatan atau dokumen historis yang relevan dengan penelitian[4] Data yang digunakan berasal dari dua dataset industri, yaitu Ziauddin[6] dan Maxwell[15] Dataset Ziauddin mencakup 21 proyek perangkat lunak berbasis *agile* dengan 13 variabel, antara lain *Effort, Vi, D, V, Sprint Size, Work Days, Team Salary, Actual Time,* dan *Cost MRE*. Sementara itu, dataset Maxwell berisi 62 proyek perangkat lunak dari berbagai domain dengan variabel seperti *Size, Effort, Duration,* serta faktor tim dan proyek. Dalam penelitian ini, dataset Maxwell digunakan sebagai pembanding untuk mengevaluasi kinerja metode estimasi usaha berbasis algoritma optimasi, khususnya pendekatan hibrida antara *Genetic Algorithm* (GA) dan *Ant Colony Optimization* (ACO).

B. Model Konseptual Algoritma Optimasi

Model konseptual algoritma optimasi yang diusulkan pada penelitian ini dikembangkan untuk menjawab permasalahan akurasi estimasi usaha perangkat lunak dengan pendekatan model agile, yang dipengaruhi oleh penentuan nilai parameter faktor friksi dan dinamis secara subjektif. Meskipun algoritma genetika telah terbukti efektif dalam melakukan optimasi parameter tersebut, keterbatasan pada tahap inisialisasi populasi awal berpotensi menyebabkan konvergensi dini dan solusi yang kurang optimal. Oleh karena itu, penelitian ini mengusulkan pendekatan algoritma hibrida dengan memanfaatkan *Ant Colony Optimization* sebagai mekanisme pembentukan populasi awal pada algoritma genetika, sementara proses optimasi parameter faktor friksi dan faktor dinamis tetap dilakukan menggunakan algoritma genetika guna memperoleh estimasi usaha yang lebih akurat dan objektif.

Faktor friksi merepresentasikan tingkat hambatan atau efisiensi kerja tim dalam proses pengembangan perangkat lunak. Faktor ini mencakup aspek-aspek seperti komposisi tim, proses kerja, lingkungan proyek, serta dinamika kolaborasi antara anggota tim. Nilai faktor friksi yang tinggi menunjukkan adanya hambatan besar dalam tim, seperti kurangnya koordinasi atau kemampuan adaptasi, yang berimplikasi pada meningkatnya total usaha pengembangan. Sebaliknya, nilai faktor friksi yang rendah mengindikasikan kondisi tim yang efisien dengan tingkat friksi yang minimal. Nilai parameter faktor friksi yang digunakan dalam penelitian ini disajikan pada tabel 1.

Sementara itu, faktor dinamis menggambarkan tingkat dinamika dan perubahan yang terjadi selama pelaksanaan proyek, seperti pergantian anggota tim, penggunaan alat baru, keterlambatan pemangku kepentingan, maupun perubahan kebutuhan pengguna. Nilai faktor dinamis yang tinggi menunjukkan proyek dengan tingkat ketidakpastian dan kompleksitas yang lebih besar, sehingga memerlukan usaha tambahan untuk menyesuaikan diri terhadap perubahan tersebut.

Kedua parameter ini memiliki peran signifikan dalam menentukan akurasi estimasi usaha, dikarenakan secara langsung memengaruhi waktu, sumber daya, serta stabilitas proses pengembangan perangkat lunak dengan pendekatan model *agile*. Nilai parameter faktor dinamis selengkapnya disajikan pada tabel 2.

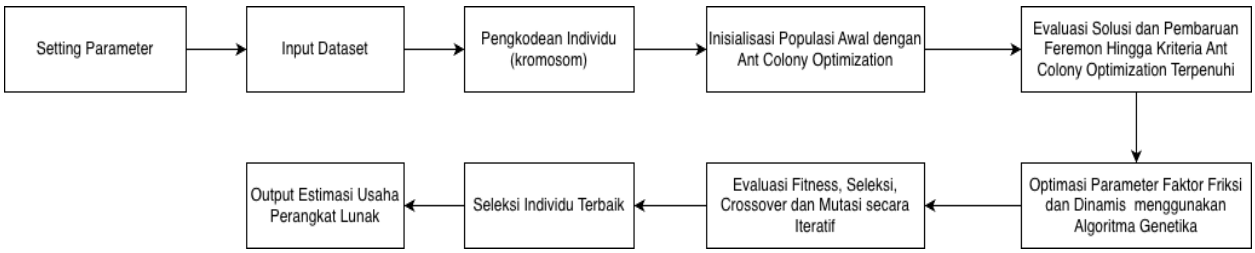
TABEL 1
FAKTOR FRIKSI

Friction Factor	Stable	Volatile	Highly Volatile	Very Highly Volatile
Team Composition	1	0.98	0.95	0.91
Process	1	0.98	0.94	0.89
Enviromental	1	0.99	0.98	0.96
Team Dynamic	1	0.98	0.91	0.85

TABEL 2
FAKTOR DINAMIS

Variabel Factor	Normal	High	Very High	Extra High
Expected team changes	1	0.98	0.95	0.91
Introduction of new tools	1	0.99	0.97	0.96
Vendor's defect	1	0.98	0.94	0.95
Team members responsibilities outside the project	1	0.99	0.98	0.98
Persolan issues	1	0.99	0.99	0.98
Expected delay in stakeholder	1	0.99	0.98	0.96
Expected ambiguity in detail	1	0.98	0.97	0.95
Expected changes in environment	1	0.99	0.98	0.97
Expected relocation	1	0.99	0.99	0.98

Diagram alir algoritma pada gambar 2 menggambarkan tahapan eksprimen yang dilakukan dalam penelitian ini. Tahapan proses eksprimen dimulai dengan setting parameter, input dataset dan pembentukan populasi awal menggunakan *Ant Colony Optimization*, kemudian diikuti oleh optimasi parameter faktor friksi dan dinamis menggunakan algoritma genetika hingga diperoleh solusi terbaik.



Gambar 2. Diagram Alir Algoritma

C. Setting Parameter

Dalam implementasi algoritma genetika dan *Ant Colony Optimization*, diperlukan beberapa parameter yang harus disiapkan terlebih dahulu. Parameter-parameter tersebut disajikan pada tabel 3.

TABEL 3
SETTING PARAMETER 1

No	Parameter	Nilai	Keterangan	Justifikasi
1	Popsize	40	Jumlah individu dalam setiap generasi. Semakin besar populasi, eksplorasi lebih baik, namun waktu komputasi meningkat.	Nilai *popsize* 40 ditetapkan berdasarkan hasil percobaan dengan variasi ukuran populasi dari 10 hingga 40. Hasil menunjukkan bahwa *popsize* 40 memberikan nilai *Absolute Error* terendah serta konvergensi yang stabil pada proses pencarian solusi. *Popsize* yang semakin tinggi tidak menjamin menghasilkan *Absolute Error* yang lebih baik, karena nilai *popsize* yang terlalu besar dapat menyebabkan waktu komputasi yang dibutuhkan menjadi semakin lama [18].
2	crossoverRate	0.7	Probabilitas pasangan individu melakukan crossover. Nilai tinggi meningkatkan eksplorasi kombinasi genetik.	Nilai *crossoverRate* 0.7 dipilih untuk memberikan keseimbangan antara eksplorasi dan eksploitasi ruang solusi. Berdasarkan eksperimen awal dengan nilai 0.5 hingga 0.9, probabilitas 0.7 menghasilkan kombinasi genetik yang paling stabil dan menghindari konvergensi prematur. Nilai ini juga sejalan dengan praktik umum dalam algoritma genetika, di mana rentang 0.6–0.8 sering digunakan untuk mempertahankan keragaman populasi. [19] semakin tingginya nilai *crossover rate*, maka akan semakin banyak kromsom yang mengalami crossover, sehingga semakin banyak kromosom yang ter-crossover. Jika terlalu bannyaj kromosom yang ter-crossover maka dapat merusak kromosom yang sudah memiliki nilai fitness yang tinggi.
3	numOfDimension	22	Panjang kromosom (7 faktor friction + 15 faktor dynamic).	Nilai *numOfDimension* merepresentasikan panjang kromosom, terdiri dari 4 faktor friksi dan 9 faktor dinamis pada dataset Ziauddin [6]. Nilai ini ditentukan berdasarkan jumlah parameter *effort driver* yang terlibat dalam model estimasi usaha perangkat lunak. Pemilihan jumlah dimensi tidak dilakukan secara eksperimen, melainkan mengikuti struktur faktor pada dataset yang digunakan.
4	mutationRate	0.05	Probabilitas tiap gen mengalami mutasi.	Nilai *mutationRate* 0.05 dipilih untuk menjaga keseimbangan antara stabilitas dan kemampuan eksplorasi. Percobaan awal menunjukkan bahwa nilai mutasi di bawah 0.03 menyebabkan konvergensi terlalu cepat dan terjebak pada solusi lokal, sedangkan di atas 0.1 menghasilkan variasi yang terlalu acak. Nilai 0.05 memberikan hasil *Absolute Error* yang konsisten rendah dan stabil [18]
5	mutationSigma	0.05	Skala perubahan mutasi Gaussian relatif terhadap rentang nilai gen.	Nilai *mutationSigma* 0.05 digunakan untuk mengatur besar perubahan yang terjadi akibat mutasi berbasis distribusi Gaussian. Nilai ini diperoleh melalui percobaan empiris yang menunjukkan bahwa skala 5% dari rentang nilai gen menghasilkan perbaikan moderat tanpa

No	Parameter	Nilai	Keterangan	Justifikasi
				menyebabkan ketidakseimbangan nilai antar gen.
6	Elite_k	1	Jumlah individu terbaik (elit) yang disalin langsung ke generasi berikut.	Jumlah elit (*elite_k*) sebesar 1 ditetapkan agar solusi terbaik dari generasi sebelumnya tetap dipertahankan di generasi berikutnya. Hasil uji coba menunjukkan bahwa mempertahankan lebih dari satu elit cenderung menurunkan keberagaman populasi, sedangkan tanpa elit menyebabkan hilangnya solusi terbaik.
7	patience	10	Mekanisme *early stopping*. Algoritma berhenti jika tidak ada perbaikan dalam 10 generasi berturut-turut.	Nilai *patience* 10 digunakan sebagai mekanisme *early stopping*, di mana algoritma berhenti jika tidak ada perbaikan selama 10 generasi berturut-turut. Nilai ini dipilih berdasarkan pengamatan bahwa setelah 10 generasi tanpa peningkatan, kurva konvergensi *Absolute Error* cenderung datar, menandakan bahwa algoritma telah mencapai kondisi stabil.
8	maxIter	60	Batas maksimum jumlah generasi.	Nilai *maxIter* 60 ditetapkan sebagai batas iterasi maksimum setelah dilakukan percobaan dengan variasi iterasi dari 30 hingga 60. Hasil menunjukkan bahwa *Absolute Error* mulai stabil pada iterasi ke-50 dan mencapai konvergensi penuh pada iterasi ke-60. Oleh karena itu, iterasi ke-60 dipilih karena menghasilkan keseimbangan terbaik antara stabilitas konvergensi dan waktu komputasi [20].
9	stoppingFitenss	0.03	Nilai ambang *absolute error (AE)* sebagai kriteria penghentian.	Nilai *stoppingFitness* 0.03 digunakan sebagai ambang batas penghentian algoritma. Nilai ini diperoleh dari pengujian yang menunjukkan bahwa ketika Absolute Error mencapai nilai mendekati 0.03, perubahan antar generasi berikutnya tidak lagi signifikan. Nilai ini juga sesuai dengan target akurasi pada penelitian-penelitian serupa di bidang estimasi usaha perangkat lunak.
10	seed	42	Nilai awal generator bilangan acak untuk menjamin reproducibility.	Nilai *seed* 42 ditetapkan untuk memastikan replikasi hasil eksperimen. Penggunaan nilai tetap pada generator bilangan acak menjamin bahwa hasil eksperimen dapat diulang dengan keluaran yang konsisten.
11	ρ (rho)	0.1	Tingkat evaporasi feromon pada ACO.	Nilai *rho* 0.1 merepresentasikan tingkat evaporasi feromon pada *Ant Colony Optimization*. Nilai ini diadopsi dari hasil eksperimen dan literatur, yang menyarankan tingkat evaporasi 0.05–0.2 untuk menjaga keseimbangan antara eksploitasi jalur terbaik dan eksplorasi jalur baru [21].
12	α (alpha)	1.0	Bobot pengaruh feromon dalam pemilihan solusi ACO.	Nilai alpha 1.0 digunakan untuk menentukan bobot pengaruh feromon dalam pemilihan solusi. Berdasarkan percobaan, nilai ini memberikan kestabilan konvergensi yang baik tanpa mendominasi faktor heuristic [21].
13	β (beta)	2.0	Bobot pengaruh heuristik dalam pemilihan solusi ACO.	Nilai *beta* 2.0 menetapkan pengaruh heuristik yang lebih kuat dibandingkan feromon. Nilai ini dipilih karena menghasilkan jalur pencarian yang lebih efisien dengan konvergensi cepat, sesuai dengan panduan empiris Dorigo (2019) bahwa

No	Parameter	Nilai	Keterangan	Justifikasi
				rasio $\beta > \alpha$ meningkatkan kinerja algoritma pada permasalahan optimasi kontinu [21].
14	q0	0.5	Probabilitas eksploitasi (memilih solusi terbaik) dalam ACO.	Nilai *q0* 0.5 digunakan untuk menyeimbangkan antara eksplorasi dan eksploitasi pada pemilihan jalur semut. Percobaan menunjukkan bahwa nilai di atas 0.7 menyebabkan konvergensi terlalu cepat, sedangkan nilai di bawah 0.3 memperlambat proses pencarian. Nilai tengah 0.5 memberikan performa yang stabil dan adaptif.
15	τ_init	1.0	Nilai awal feromon pada setiap jalur.	Parameter τ_init digunakan sebagai nilai feromon awal yang diberikan secara seragam pada seluruh jalur solusi dalam algoritma *Ant Colony Optimization*. Penetapan nilai awal yang sama bertujuan untuk mencegah bias pencarian pada iterasi awal serta menjaga keseimbangan eksplorasi ruang solusi sebelum proses pembaruan feromon dilakukan. Nilai feromon awal konstan, seperti τ_init = 1.0, efektif dalam menghasilkan keragaman solusi awal dan stabilitas proses pencarian, khususnya pada penerapan ACO sebagai bagian dari algoritma hibrida untuk optimasi parameter.
16	τ_min	1e-6	Batas minimum nilai feromon agar tidak hilang seluruhnya	Nilai batas minimum feromon ditetapkan 1e-6 agar jalur yang kurang optimal tidak kehilangan peluang sepenuhnya. Nilai ini mencegah stagnasi prematur pada jalur terbaik[22].
17	τ_max	100.0	Batas maksimum nilai feromon agar tidak mendominasi total.	Nilai maksimum feromon sebesar 100 digunakan untuk menghindari dominasi berlebihan pada satu jalur solusi. Hasil eksperimen menunjukkan bahwa nilai lebih tinggi menyebabkan *over-exploitation* sehingga mengurangi keragaman solusi.
18	vi_idx	0	Indeks kolom untuk nilai *vi* pada dataset.	Nilai indeks *vi_idx* ditetapkan berdasarkan posisi kolom *vi* dalam dataset, yaitu kolom pertama (indeks 0). Nilai ini mengikuti struktur data tanpa melalui proses eksperimental.
19	actual_idx	2	Indeks kolom untuk nilai *actual effort* pada dataset.	Indeks kolom *actual_effort* berada pada kolom ke-3 (indeks 2). Penetapan ini sesuai dengan struktur dataset yang digunakan untuk menghitung selisih antara hasil estimasi dan nilai aktual.
20	effort_idx	4	Indeks kolom untuk nilai *effort* pada dataset.	Nilai indeks *effort_idx* ditetapkan pada kolom ke-5 (indeks 4) sesuai dengan posisi atribut *effort* dalam dataset Maxwell[15], yang digunakan sebagai basis estimasi dalam perhitungan *Absolute Error*

D. Model Konseptual Evaluasi

Evaluasi pada penelitian ini dilakukan secara sistematis terhadap dua pendekatan algoritma optimasi, yaitu algoritma genetika tunggal dan integrasi algoritma genetika dengan *Ant Colony Optimization* (GACO). Tujuan evaluasi dilakukan adalah menilai tingkat akurasi estimasi usaha pada pengembangan perangkat lunak berbasis *agile* dengan menggunakan enam metrik pengukuran, yaitu *Absolute Error* (AE), *Mean Absolute Error* (MAE), *Mean Balanced Relative Error* (MBRE), *Mean Inverted Balanced Relative Error* (MIBRE), *Standardized Accuracy* (SA), dan *Effect Size* (ES). Metrik *Absolute Error* mengukur selisih absolut antara nilai aktual dan estimasi pada setiap proyek, sedangkan *Mean Absolute Error* menghitung rata-rata keseluruhan error untuk menilai akurasi model secara umum. Kedua metrik ini menjadi dasar dalam menilai efektivitas pendekatan optimasi yang diusulkan.

1. Nilai mutlak dari hasil pengurangan antara *effort actual* dengan *estimated effort* adalah *absolute error*, yang dihitung menggunakan persamaan (1). Nilai ini menunjukkan seberapa besar deviasi hasil estimasi yang telah dioptimalkan menggunakan algoritma optimasi terhadap nilai aktual dalam suatu proyek.

$$\text{AE} = |\ \text{Effort Aktual} - \text{Estimated Effort}\ | \qquad (1)$$

2. Sementara itu, mean absolute error (MAE) dihitung berdasarkan persamaan (2) sebagai rata-rata kesalahan *Absolute Error* dari seluruh proyek dalam dataset. Mean absolute error merepsentasikan rata-rata kesalahan estimasi secara keseluruhan dan digunakan sebagai indikator akurasi model estimasi usaha.

$$MAE = \frac{1}{N}\sum_{i}^{n} \quad |AE| \qquad (2)$$

3. Menghitung nilai *Mean Balanced Relative Error* (MBRE), dihitung menggunakan persamaan (3) untuk mengurangi bias estimasi terhadap proyek berskala besar maupun kecil.

$$\text{MBRE} = \sum_{i=1}^{n} \quad \frac{\hat{y}_I - y_i}{min(\hat{y}_I, y_i)} \qquad (3)$$

4. Sementara itu, *Mean Inverted Balanced Relative Error* (MIBRE) dihitung berdasarkan persamaan (4) sebagai metrik pelengkap *Mean Balanced Relative Error*.

$$\text{MIBRE} = \sum_{i=1}^{n} \quad \frac{\hat{y}_I - y_i}{min(\hat{y}_I, y_i)} \qquad (4)$$

5. *Standardized Accuracy* (SA) merupakan ukuran akurasi yang digunakan untuk mengevaluasi, apakah metode estimasi algoritma genetika (misalnya P_i) ini memberikan hasil estimasi yang bermakna atau tidak. Untuk dikategorikan sebagai metode yang bermakna, maka P_i harus membuktikan bahwa performanya harus mengalahkan metode baseline yaitu *random guessing* atau P_0 adalah penentuan nilai usaha terestimasi $\hat{y}$ hasil dari target data yang dipilih secara acak dengan frekuensi keterpilihan paling besar. Setiap kali menentukan nilai usaha terestimasi, maka pengacakan dijalankan biasanya sebanyak 1000 kali. Setiap target data yang terpilih akan dihitung frekuensi terpilihnya secara tabulasi. Setelah dijalankan sebanyak 1000 kali maka dipilihlah target data yang paling banyak frekuensi keterpilihannya, Nilai usaha terestimasi $\hat{y}$ adalah sama dengan nilai usaha aktual ($\hat{y} = y_i$) pada target data yang terpilih tersebut. Untuk mendapatkan nilai *Standardize Accuracy* (SA) dihitung menggunakan persamaan (5).

$$SA_{pi} = 1 - \left(\frac{MAE_{Pi}}{MAE_{P_0}}\right) \times 1000 \qquad (5)$$

6. *Effect size* (ES) adalah metrik yang digunakan untuk menginterpretasi signitifikasikan hasil estimasi secara partikal atau dunia nyata[14]. *Effect size* digunakan untuk memastikan bahwa hasil estimasi metode algoritma genetika (P_i) tidak dihasilkan secara kebetulan semata. Nilai Effect Size dihitung berdasarkan persamaan (6).

$$ES = \frac{MAE_{P_i} - \underline{MAE_{P_0}}}{S_{P_0}} \qquad (6)$$

E. Uji Statistik

Uji Statistik yang digunakan dalam penelitian ini adalah *Wilcoxon Signed Rank Test*, yaitu metode non-parametrik yang bertujuan menguji ada atau tidaknya perbedaan signifikan pada dua kelompok data yang bersifat berpasangan. Uji ini sesuai diterapkan pada data berskala ordinal atau interval yang tidak tidak memenuhi asumsi

distribusi normal. *Wilcoxon Signed Rank Test* sangat tepat diterapkan ketika suatu sampel mengalami dua perlakuan berbeda sehingga hasilnya dapat dibandingkan secara lebih objektif.

Dasar pengambilan keputusan pada uji ini mengacu pada nilai probabilitas (Asymp. Sig) sebagaimana ditunjukkan pada persamaan (7), di mana hipotesis diterima apabila nilai < 0,05 sedangkan apabila nilai > 0,05 maka hipotesis ditolak[8].

$$\begin{matrix} Probabilitas\ (Asymp.Sig) \\ < \ 0{,}05\ maka\ Hipotesis\ diterima. Probabilitas\ (Asymp.Sig) \\ > \ 0{,}05\ maka\ Hipotesis\ ditolak. \end{matrix} \quad (7)$$

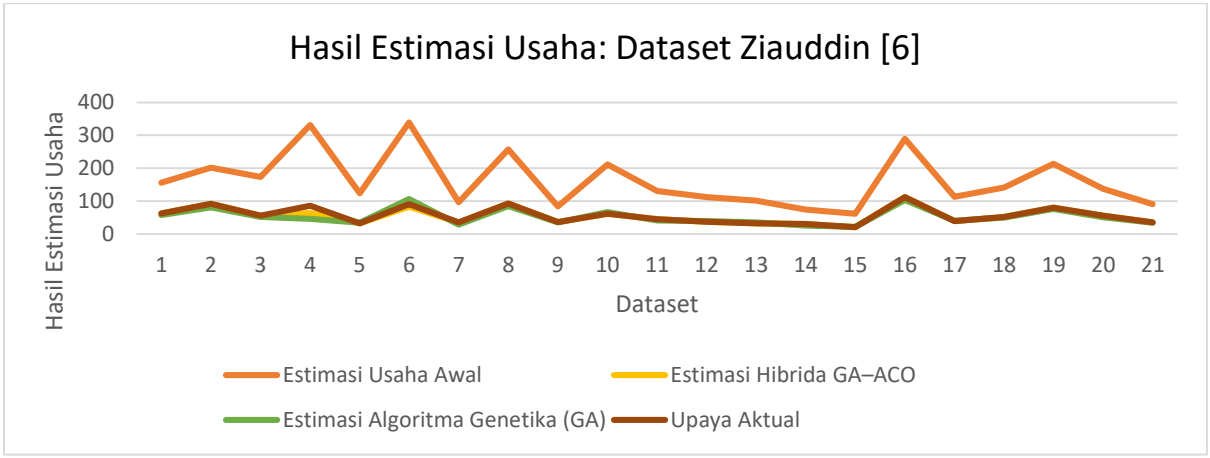
III. HASIL DAN PEMBAHASAN

Penelitian ini menghasilkan model estimasi usaha perangkat lunak berbasis model agile dengan dua pendekatan algoritma optimasi, yaitu algoritma genetika dan *Ant Colony Optimization* (GACO). Evaluasi menggunakan dataset Ziauddin[6] dan maxwell[15] untuk mengukur efektivitas integrasi algoritma *Ant Colony Optimization* sebagai mekanisme inisialisasi populasi awal dalam meningkatkan akurasi estimasi usaha. Penggunaan dua dataset dengan karakteristik dan skala proyek yang berbeda dimaksudkan untuk menguji konsistensi, stabilitas, dan keandalan model estimasi yang diusulkan.

Subbab A menyajikan perbandingan kinerja estimasi pada dataset Ziauddin[6] dengan menekankan kedekatan nilai estimasi terhadap usaha aktual berdasarkan metrik evaluasi yang digunakan. Subbab B membahas hasil pada dataset Maxwell[15] untuk mengamati konsistensi kinerja model pada karakteristik data yang berbeda. Selanjutnya, Subbab C menguraikan solusi optimum yang diperoleh melalui analisis stabilitas hasil estimasi pada berbagai konfigurasi iterasi dan ukuran populasi. Subbab D menyajikan pengujian statistik, untuk menegaskan signifikansi perbedaan kinerja antar pendekatan.

A. *Perbandingan Kinerja Model pada Dataset Ziauddin*

Perbandingan hasil estimasi usaha antara metode algoritma optimasi, yaitu algoritma genetika tunggal dan hibrida algoritma genetika dengan *Ant Colony Optimization* ditunjukkan pada gambar 3.



Gambar 3. Hasil Perbandingan Estimasi Usaha Ziauddin

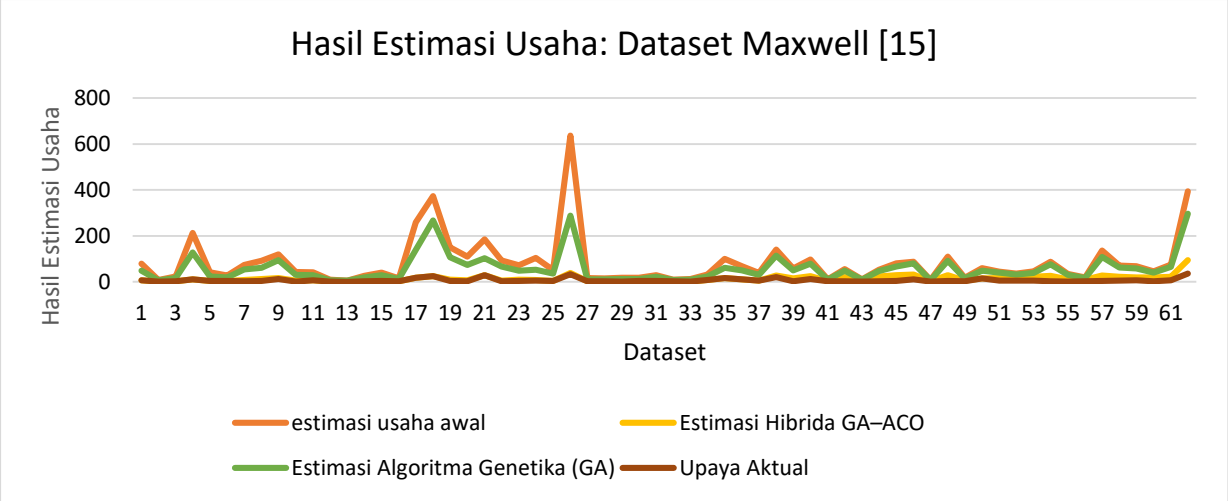
Berdasarkan gambar 3, hasil perbandingan estimasi usaha menunjukkan bahwa nilai Estimasi Hibrida algoritma genetika dengan *Ant Colony Optimization* (garis kuning) lebih mendekati upaya aktual (garis cokelat) dibandingkan dengan estimasi usaha menggunakan algoritma genetika tunggal (garis hijau) maupun Estimasi Usaha awal (garis orange). Hal ini menandakan bahwa metode hibrida algoritma genetika dengan *Ant Colony Optimization* menghasilkan prediksi usaha yang lebih akurat dan presisi di seluruh data (1–21). Nilai Estimasi Usaha Awal terlihat jauh lebih tinggi dan fluktuatif, terutama pada data tertentu seperti dataset ke-4, ke-6, dan ke-16, yang menunjukkan deviasi atau penyimpangan besar terhadap nilai aktual.

B. *Perbandingan Kinerja Model pada Dataset Maxwell*

Hasil perbandingan estimasi usaha menggunakan metode algoritma optimasi, yaitu algoritma genetika tunggal dan hybrid algoritma genetika dengan *Ant Colony Optimization* disajikan pada gambar 4. Hasil perbandingan menunjukkan bahwa nilai estimasi usaha dengan metode hibrida algoritma genetika dengan *Ant Colony Optimization* (garis kuning) secara signifikan lebih mendekati upaya aktual (garis cokelat) dibandingkan dengan hasil estimasi

usaha menggunakan metodea algoritma genetika tunggal (garis hijau) maupun hasil estimasi usaha awal (garis orange). Hal ini menandakan bahwa metode hibrida algoritma genetika dengan *Ant Colony Optimization* menghasilkan prediksi usaha yang lebih akurat, terutama pada dataset Maxwell yang memiliki rentang nilai usaha yang sangat luas dan kompleks.

Nilai estimasi usaha awal terlihat jauh lebih tinggi dan fluktuatif dibandingkan nilai aktual. Deviasi atau penyimpangan yang sangat besar terlihat jelas pada titik data tertentu, seperti pada data ke-18, ke-26, dan ke-62, di mana estimasi awal mengalami lonjakan drastis (overestimation). Meskipun algoritma genetika tunggal (garis hijau) telah berhasil mereduksi kesalahan tersebut, namun integrasi algoritma genetika dan *Ant Colony Optimization* (garis kuning) terbukti jauh lebih efektif dalam menekan nilai estimasi hingga mendekati garis upaya aktual.

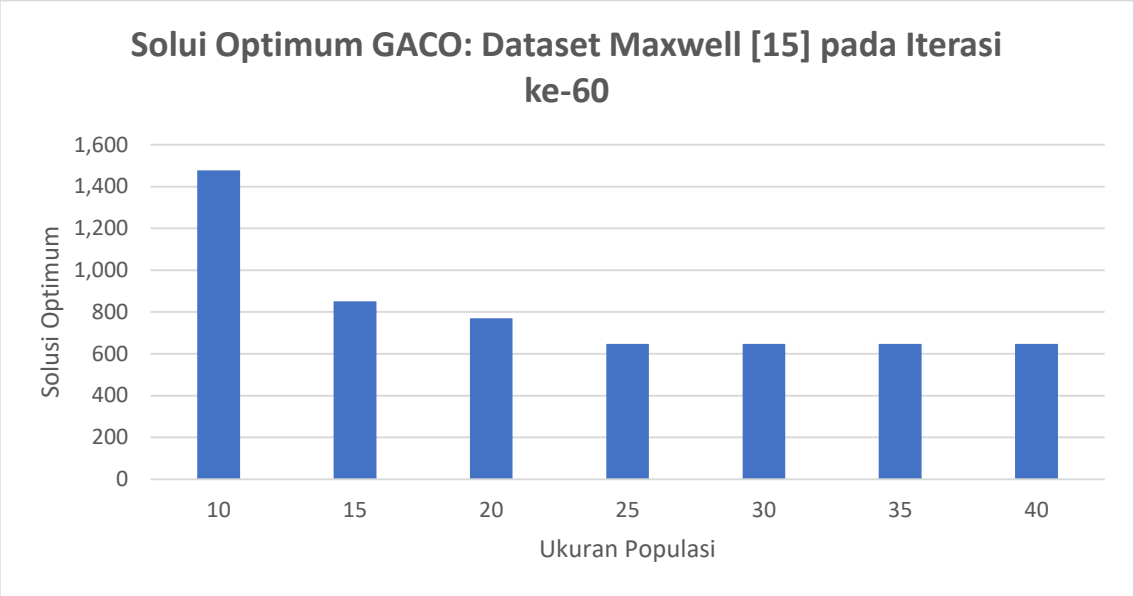


Gambar 4. Hasil Perbandingan Estimasi Usaha Maxwell

C. *Solusi Optimum*

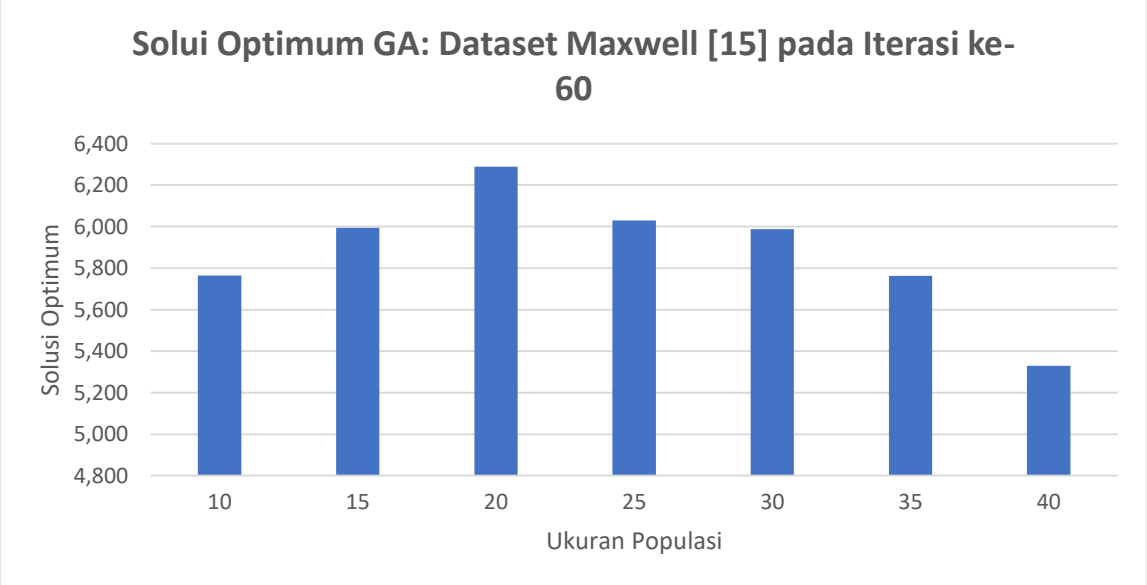
Proses pencarian solusi optimum dilakukan melalui dua langkah utama, yaitu iterasi percobaan menggunakan algoritma genetika tunggal dan hibrida algoritma genetika dengan *Ant Colony Optimization*. Iterasi percobaan ini bertujuan untuk menganalisa perilaku konvergensi dan kestabilan solusi yang dihasilkan dari masing-masing pendekatan algoritma optimasi yang diusulkan, guna untuk memastikan konsistensi konvergensi pada skala data yang berbeda.

Hasil iterasi percobaan menunjukkan bahwa pendekatan hibrida algoritma genetika dengan *Ant Colony Optimization* (GACO) mencapai konvergensi yang lebih cepat dan stabil dibandingkan algoritma genetika tunggal, terutama pada fase iterasi akhir. Pola konvergensi hibrida algoritma genetika dengan *Ant Colony Optimization* pada dataset Maxwell[15] ditunjukkan pada gambar 5, di mana kestabilan estimasi mulai tercapai sejak iterasi ke-50 dan solusi optimum diperoleh pada iterasi ke-60 dengan ukuran populasi 40 kromosom. Stabilitas ini mengindikasikan bahwa mekanisme pembentukan populasi awal berbasis pheromone pada *Ant Colony Optimization* mampu mengarahkan proses evolusi secara lebih efektif sejak generasi awal.



Gambar 5. Solusi Optimum Hybrid Algoritma

Sebaliknya, pola konvergensi algoritma genetika tunggal yang ditampilkan pada gambar 6 menunjukkan fluktuasi nilai estimasi yang relatif lebih besar, meskipun peningkatan jumlah iterasi tetap menghasilkan perbaikan akurasi secara bertahap. Solusi terbaik algoritma genetika tunggal juga dicapai pada iterasi ke-60, namun dengan tingkat kestabilan yang lebih rendah dibandingkan hibrida algoritma genetika dengan *Ant Colony Optimization*.



Gambar 6. Solusi Optimum Algoritma Genetika Tunggal

Pola konvergensi yang sejalan juga diperoleh pada dataset Ziauddin[6], di mana kedua pendekatan menunjukkan kecenderungan konvergensi pada iterasi akhir yang sama dengan dataset Maxwell. Oleh karena karakteristik konvergensi yang dihasilkan bersifat konsisten, visualisasi grafik iterasi untuk dataset Ziauddin tidak ditampilkan guna menghindari redundansi penyajian dan keterbatasan ruang publikasi.

Perbedaan ini menegaskan bahwa ketiadaan mekanisme pengarah pada tahap inisialisasi populasi menyebabkan proses pencarian solusi pada algoritma genetika berlangsung kurang konsisten. Sebaliknya, integrasi *Ant Colony Optimization* berfungsi sebagai mekanisme pencarian terarah yang memperkaya kualitas populasi awal dan mengarahkan proses evolusi algoritma genetika secara lebih sistematis, sehingga menghasilkan estimasi usaha yang lebih stabil dan mendekati nilai aktual.

D. *Uji Statistik*

Estimasi usaha dengan hibrida algoritma genetika dan *Ant Colony Optimization* (ACO) pada tabel 6, menunjukkan adanya perbedaan yang signifikan. Dari hasil pengujian terhadap 62 pasangan data, seluruh observasi menghasilkan *Positive Ranks* dengan total peringkat 1953.00, sedangkan *Negative Ranks* bernilai nol. Temuan ini menunjukkan bahwa nilai estimasi usaha algoritma genetika secara konsisten lebih tinggi dibandingkan dengan hibrida algoritma genetika dengan *Ant Colony Optimization* (GACO). Hasil uji Wilcoxon memberikan *p-value* < 0,05, sehingga Hipotesis Nol (H_0) ditolak. Dengan demikian, terdapat perbedaan signifikan antara kedua metode, di mana metode hibrida algoritma menghasilkan estimasi usaha yang lebih rendah dan akurat dibandingkan algoritma genetika tunggal.

TABEL 6
UJI STATISTIK

		Ranks		
		N	**Mean Rank**	**Sum Of Ranks**
Estimasi usaha Agile GA-Estimasi Usaha Agile GACO	Negative Ranks	0	.00	0.00
	Positive Ranks	62	31.50	1953.00
	Ties	0	.00	.00
	Total	62		

Rangkaian pengujian pada dataset Maxwell[15] dan Ziauddin[6] membuktikan bahwa integrasi *Ant Colony Optimization* sebagai mekanisme inisialisasi populasi dalam algoritma genetika secara signifikan meningkatkan akurasi estimasi upaya dibandingkan model algoritma genetika tunggal. Model hibrida algoritma genetika dengan *Ant Colony Optimization* terbukti menghasilkan nilai estimasi yang lebih presisi dan konsisten mendekati nilai aktual melalui penguatan fase awal pencarian solusi.

Secara analisis komparatif, temuan ini memperkuat urgensi optimasi parameter dalam estimasi perangkat lunak sebagaimana ditekankan oleh Ziauddin [6] dan Yusril Mahendri [8], namun dengan kontribusi tambahan berupa ketahanan (robustness) yang lebih unggul saat menghadapi data ekstrem (outlier). Keberhasilan ini didorong oleh kemampuan *Ant Colony Optimization* dalam menyediakan populasi awal berkualitas yang mencegah risiko lokal optimal. Kendati memiliki kompleksitas komputasi yang lebih tinggi, model ini menawarkan implikasi strategis bagi industri dalam meminimalisasi risiko kegagalan proyek melalui perencanaan sumber daya yang lebih akurat.

IV. SIMPULAN

Penelitian ini bertujuan untuk meningkatkan akurasi estimasi usaha perangkat lunak berbasis *agile* melalui optimalisasi parameter faktor friksi dan dinamis menggunakan pendekatan hibrida algoritma genetika dan *Ant Colony Optimization* (GACO). Berdasarkan hasil eksperimen dan analisis inferensial, pendekatan hibrida algoritma genetika dan *Ant Colony Optimization* terbukti secara signifikan lebih akurat dibandingkan algoritma genetika tunggal. Temuan ini didukung oleh hasil uji *Wilcoxon Signed-Rank* yang menghasilkan p-value < 0,05, dengan seluruh 62 observasi berada pada positive ranks dan tidak ditemukan negative ranks. Selain itu, nilai *mean absolute error* (MAE) yang dihasilkan oleh hibrida algoritma genetika dan *Ant Colony Optimization* adalah 354,91 lebih rendah dibandingkan algoritma genetika tunggal 618,38 pada dataset Ziauddin[6], yang mengindikasikan bahwa estimasi usaha menggunakan algoritma genetika tunggal cenderung menghasilkan nilai yang lebih besar dibandingkan pendekatan hibrida. Keunggulan pendekatan hibrida algoritma genetika dan *Ant Colony Optimization* terutama terletak pada integrasi mekanisme pencarian berbasis pheromone dari *Ant Colony Optimization* yang berperan sebagai mekanisme pengarah dalam tahap inisialisasi populasi. Integrasi ini memperkaya kualitas solusi awal, mengurangi risiko konvergensi prematur, serta mengarahkan proses evolusi algoritma genetika secara sistematis, sehingga menghasilkan kombinasi parameter faktor friksi dan dinamis yang lebih stabil serta mendekati nilai aktual. Konsistensi pola konvergensi dan peningkatan akurasi yang diperoleh pada dataset Maxwell[15] dan Ziauddin[6] menunjukkan bahwa model GACO memiliki ketahanan (robustness) yang baik terhadap variasi skala dan karakteristik data proyek perangkat lunak. Sebagai arah penelitian lanjutan, pengembangan model hibrida algoritma genetika dan *Ant Colony Optimization* dapat difokuskan pada penerapan skema adaptasi parameter secara dinamis, khususnya pada pengaturan laju pembaruan pheromone dan operator genetika, serta analisis efisiensi komputasi untuk menyeimbangkan peningkatan akurasi dengan kompleksitas waktu eksekusi pada lingkungan pengembangan perangkat lunak berskala industri.

UCAPAN TERIMAKASIH

Penulis menyampaikan terima kasih kepada Program Studi Magister Informatika Universitas Islam Indonesia atas dukungan akademik dan fasilitas penelitian yang diberikan selama pelaksanaan studi ini. Penghargaan juga ditujukan kepada dosen pembimbing atas bimbingan dan masukan berharga yang telah membantu penyempurnaan penelitian ini.

DAFTAR PUSTAKA

[1] Turban E, Rainer R K dan Potter R E (2001 Pengantar teknologi informasi (hal. 550) (New York, NY: John Wiley & Sons)
[2] Kualitatif, 17, 43. http://repository.unpas.ac.id/30547/5/BAB III.pdf A. Trendowicz and R. Jeffery, Software Project Effort Estimation: Founda-tions and Best Practice Guidelines for Success. Berlin, Germany: Springer, 2014
[3] C. Larman dan V. R. Basili, ‘‘Pengembangan iteratif dan inkremental. Sejarah singkat,’’ Computer, vol. 36, no. 6, pp. 47–56, Juni 2003, doi:10.1109/MC.2003.1204375.
[4] Ahmad Suryana. (2017). Metode Penelitian Metode Penelitian. Metode Penelitian
[5] Diyar, Itmam. “Optimasi estimasi effort perangkat lunak berbasis use case point menggunakan algoritma genetika.” Skripsi. Yogyakarta: Program Studi Teknik Informatika UAD.
[6] Ziauddin, et al. “An Effort Estimation Model for Agile Software Development.” Advances in Computer Science and its Applications (ACSA), vol. 2, 2012. https://www.researchgate.net/publication/268186219_An_Effort_Estimation_Model_for_Agile_Software_Development
[7] Adani, M. R. (2020, August). Metode Agile: Pengertian, Tujuan, Jenis, Manfaat, dan Prinsip. https://www.sekawanmedia.co.id/met ode-agile-development
[8] Yusril Mahendri. “Estimasi effort perangkat lunak agile dengan pemilihan prameter frcition dan dynamic factors menggunakan algoritma genetika.” Skripsi. Yogyakarta: Program Studi Teknik Informatika UAD.
[9] Zukhri, Z., & Paputungan, I. V. (2013). A Hybrid Optimization Algorithm based on *Genetic Algorithm* and *Ant Colony Optimization*. International Journal of Artificial Intelligence & Applications, 4(5), 63–75. https://doi.org/10.5121/ijaia.2013.4505
[10] E. Mendes, teknik estimasi biaya untuk proyek web. H yS, PA: IGI Pub, 2007. https://doi.org/10.4018/978-1-59904-135-3
[11] S. Rc, M. Sánchez-Gordón, R. Colomo-Palacios & M. Kristiansen, “Estimasi Usaha dalam Pengembangan Perangkat Lunak Agile: Sebuah Studi Eksplorasi dari Perspektif Praktisi,” dalam LASD 2022: Pengembangan Perangkat Lunak Lean dan Agile, Przybyłek, A., Jarzębowicz, A., Luković, I., Ng, Y. (Eds)., Cham, CH: Springer, 2022, vol. 428, hlm. 136–149. https://doi.org/10.1007/978-3-030-94238- 0_8
[12] Bagheri, A., Akbarzadeh, T. M., Saraee, M., (2008), "Menemukan Jalur Terpendek dengan Algoritma Pembelajaran", J. Int. Kecerdasan Buatan, Vol. 1, No. A8, hlm. 86-95
[13] Dorigo, M., Maniezzo, V., Colorni, A., (1996), "Sistem Ant: Optimisasi oleh koloni agen yang bekerja sama", J. IEEE Sistem, Man, dan Sibernetika, Vol. 26, No. 1, hlm. 1-13
[14] Rahman, Hossan Abdel, and Mohamed Al-Ansary. “A Proposed *Genetic Algorithm* Model to Improve Effort Estimation in Agile Software Development.” Computer Science and Information Security, vol. 22, 2024.
[15] Rahman, Mizanur Rahman, et al. “Review of Existing Datasets Used for Software Effort Estimation.” (IJACSA) International Journal of Advanced Computer Science and Applications, vol.14, 2023, https://www.researchgate.net/publication/372922008_Review_of_Existing_Datasets_Used_for_Software_Effort_Estimation.
[16] Ardata. “Mengenal Agile Development: Sifat, Metode, Cara Kerja.” *Ardata Indonesia*, 2023. Diakses dari https://ardata.co.id/agile-development-adalah/
[17] Ali, Asad, and Carmine Gravino. “Improving software effort estimation using bio-inspired algorithms to select relevant features: An empirical study.” Science of Computer Programming, 2021, https://www.sciencedirect.com/science/article/pii/S0167642321000149.
[18] Pradini, Risqy Siwi, et al. “OPTIMASI WEIGHT AHP MENGGUNAKAN *GENETIC ALGORITHM* UNTUK REKOMENDASI PLATFORM MEDIA SOSIAL SEBAGAI SARANA PROMOSI DIGITAL.” Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), vol. 11, 2024, https://jtiik.ub.ac.id/index.php/jtiik/article/view/8011.
[19] Yunita, et al. “APPLICATION OF *GENETIC ALGORITHM*S SUBJECT SCHEDULING SYSTEM AT SMA BINA JAYA PALEMBANG.” TEKNOMATIKA, vol. 13, 2023, https://ojs.palcomtech.ac.id/index.php/teknomatika/article/view/617.
[20] Mas’ud, Muhammad Faris, et al. “Optimasi Algoritme Genetika Untuk Memaksimalkan Laba Pembangunan Perumahan.” Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 3, 2019, https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/4262.
Setyati, Endang, and Ine Juniwati. “*Ant Colony Optimization* untuk Menyelesaikan Perutean Distribusi Snack dengan Vehicle Routing Problem.” Jurnal Teknologi Informasi dan Terapan, vol. 9, 2022. https://jtit.polije.ac.id/index.php/jtit/article/view/296.
[21] Suryana, Fitra, et al. “Implementation of *Ant Colony Optimization* (ACO) Algorithm for Route Optimization of Tourist Paths in Takengon.” Journal of Applied Informatics and Computing, vol. 4, 2025, https://jurnal.polibatam.ac.id/index.php/JAIC/article/view/9706.
[22] Setyati, Endang, and Ine Juniwati. “*Ant Colony Optimization* untuk Menyelesaikan Perutean Distribusi Snack dengan Vehicle Routing Problem.” Jurnal Teknologi Informasi dan Terapan, vol. 9, 2022. https://jtit.polije.ac.id/index.php/jtit/article/view/296.
[23] Fachruddin, and Yogi Pratama. “Eksperimen Seleksi Fitur pada Parameter Proyek untuk Software Effort Estimation dengan K-Nearest Neighbor.” Jurnal Informatika: Jurnal Pengembangan IT (JPIT), vol. 10, no. 1, 2025. https://ejournal.poltekharber.ac.id/index.php/informatika/article/view/510.
[24] Utomo, Dwi Wahyu, Deni Kurniawan, and Nanda Kartika Ningrum. “Implementasi Traveling Salesman Problem pada Pemilihan Jalur ATM Locator Menggunakan *Ant Colony Optimization*.” Jurnal Informatika: Jurnal Pengembangan IT (JPIT), vol. 9, 2024. https://ejournal.poltekharber.ac.id/index.php/informatika/article/view/2265.