Implementation Of *Tournament* and *Elitism* Selection Techniques in Genetic Algorithm for Software Business Attribute Weight Selection *Analogy*

Nur Fauzi[1], Ardiansyah[2]

Fakultas Teknologi Industri Universitas Ahmad Dahlan

Jl. Ringroad Selatan, Kragilan, Tamanan, Kec. Banguntapan, Kabupaten Bantul, Daerah Istimewa Yogyakarta 55191

[1]nur2000018292@webmail.uad.ac.id, [2]ardiansyah@tif.uad.ac.id

ABSTRACT

Effort estimation plays a role in determining a software development project's resource allocation and cost. One of the effort estimation methods used is the *Analogy* method. The *Analogy* method is based on the principle of reusing data from previous projects by calculating each project's similarity level. However, the problem faced in the *Analogy* method is the difficulty in determining the appropriate weight for each effort attribute, which significantly impacts the estimate's accuracy. Therefore, this study will use a genetic algorithm to determine the similarity weight in the *Analogy* method. The genetic algorithm was chosen because it can produce solutions close to the optimal solution.

This study conducted experiments using three different datasets, namely the Desharnais, Maxwell, and Cocomo datasets. Various variations in the *Analogy* method, such as mean, median, and IRWA, as well as calculations of the Euclidean, Manhattan, and Minkowski distance functions, have also been tried. This study applies the *Tournament* and *Elitism* selection techniques in using genetic algorithms. Different population variations were also tested during the study to find a more optimal solution in effort estimation. The *Analogy* method optimized using a genetic algorithm was run 30 times to obtain accurate estimation results. Evaluation and validation techniques were carried out using absolute error, mean absolute error, MIBRE, MBRE, SA, ES, LOOCV, and the Wilcoxon signed rank test.

This study shows that the Mean K=4 adaptation technique with Manhattan distance for *tournament* selection provides optimal effort estimation results. The *Analogy* method optimized using the Genetic Algorithm produces better MAE than the standard *Analogy* method. The MAE value produced by the optimized *Analogy* method is 0.48, while the MAE value of *Analogy* without optimization is 314.42. The Wilcoxon statistical test shows the Asymp. Sig. $0.00 < 0.05$ for all datasets, indicating a significant difference between the *Analogy* method optimized with Genetic Algorithm and the standard *Analogy* method, as seen from the decrease in MAE values for all datasets.

Keywords: *Analogy*; genetic algorithm; similarity weight; effort estimation; optimization

1. Pendahuluan

Dalam proses pengembangan perangkat lunak, terdapat tahapan-tahapan yang harus dilalui. Software Development Life Cycle (SDLC) merupakan sebuah siklus yang menjelaskan tentang metode dan strategi untuk mengembangkan desain dan memelihara proyek perangkat lunak. Tahapan dari proses Software Development Life Cycle (SDLC) meliputi *planning, analysis, design, implementation, testing and integration,* dan *maintenance*. Salah satu aktivitas utama pada siklus planning adalah melakukan estimasi usaha [1].

Estimasi usaha berperan dalam pengalokasian sumber daya dan penentuan biaya sebuah proyek dalam perencanaan dan manajemen proyek pengembangan perangkat lunak. Estimasi usaha yang terlalu rendah maupun terlalu tinggi terhadap usaha aktual dapat mengakibatkan hilangnya kontrak untuk perusahaan perangkat lunak dan atau kegagalan dalam manajemen proyek perangkat lunak [2].

Untuk mengatasi permasalahan di atas, ada beberapa metode estimasi perangkat lunak yang dapat digunakan yaitu expert judgment, algoritma estimasi usaha, dan estimasi menggunakan *Analogy*. Model estimasi usaha perangkat lunak berbasis *Analogy* merupakan proses mengidentifikasi satu atau lebih proyek serupa yang sudah dikembangkan sebelumnya dengan proyek baru yang akan dikembangkan [3]. Dalam beberapa hal, metode ini merupakan bentuk expert judgment yang sistematis

karena para ahli sering mencari situasi yang serupa untuk menginformasikan pendapat mereka [3]. Teknik ini melibatkan karakterisasi proyek baru yang memerlukan perkiraan. Kemudian karakteristik tersebut digunakan untuk pencarian proyek serupa yang sudah dikembangkan sebelumnya. Nilai hasil identifikasi usaha proyek serupa tersebut digunakan untuk membuat perkiraan estimasi usaha proyek baru yang akan dikembangkan.

Keunggulan sistem estimasi berdasarkan *Analogy* adalah kesederhanaannya, sehingga dapat diimplementasikan dengan cepat dan dapat memberikan performa estimasi yang sangat baik [4]. *Analogy* secara bertahap menghitung tingkat kesamaan antara proyek baru dan semua proyek yang ada menggunakan fungsi jarak seperti jarak Euclidean. Selanjutnya, *Analogy* menghasilkan usaha dengan menggunakan rata-rata nilai usaha dari k proyek yang paling mirip untuk mengestimasi nilai usaha dari proyek baru. Dalam beberapa literatur, nilai k ini biasanya didefinisikan sebagai satu, tiga, atau lima [4].

Similarity adalah faktor penting dalam mengidentifikasi proyek-proyek sebelumnya yang memiliki kemiripan dengan proyek baru dalam *Analogy* [5]. Pada dasarnya setiap atribut usaha dalam *Analogy* memiliki bobot yang sama. Padahal pemberian bobot yang relevan untuk setiap atribut usaha sangat penting karena dapat memengaruhi akurasi model. Oleh karena itu, penting untuk memberikan bobot yang tepat agar atribut usaha lebih relevan dalam proses penentuan kesamaan.

Untuk mendapatkan nilai bobot yang paling sesuai, salah satunya menggunakan teknik optimasi. Teknik optimasi adalah metode pencarian yang dilakukan untuk mendapatkan nilai yang memberikan hasil paling optimal. Ada beberapa algoritma yang bisa digunakan untuk optimasi antara Algoritma Genetika, Particle Swarm Optimization, dan sebagainya. Algoritma Genetika adalah teknik pencarian berdasarkan mekanisme evolusi alami spesies. Algoritma ini telah digunakan untuk memecahkan masalah optimasi di banyak bidang. Pada penelitian ini, Algoritma Genetika digunakan untuk menentukan nilai bobot yang sesuai untuk setiap atribut usaha. Algoritma Genetika digunakan karena unggul dalam penggunaan seleksi alam yang dialami dalam evolusi. Individu secara konstan mengalami perubahan gen untuk beradaptasi dengan lingkungannya, yaitu hanya individu yang kuat yang dapat bertahan hidup untuk menghasilkan keturunan yang lebih baik. Oleh karena itu, algoritma genetika dapat menghasilkan solusi yang mendekati solusi optimal. Algoritma Genetika menggunakan berbagai operator selama proses pencarian. Operator tersebut adalah skema encoding, crossover, mutasi, dan seleksi [6]. Seleksi merupakan langkah penting dalam algoritma genetika untuk memastikan kromosom yang dipilih untuk kawin dan reproduksi serta jumlah keturunan yang dihasilkan setiap kromosom yang dipilih. Ada berbagai mekanisme seleksi yang dapat diterapkan sesuai dengan masalah yang ingin diselesaikan [7].

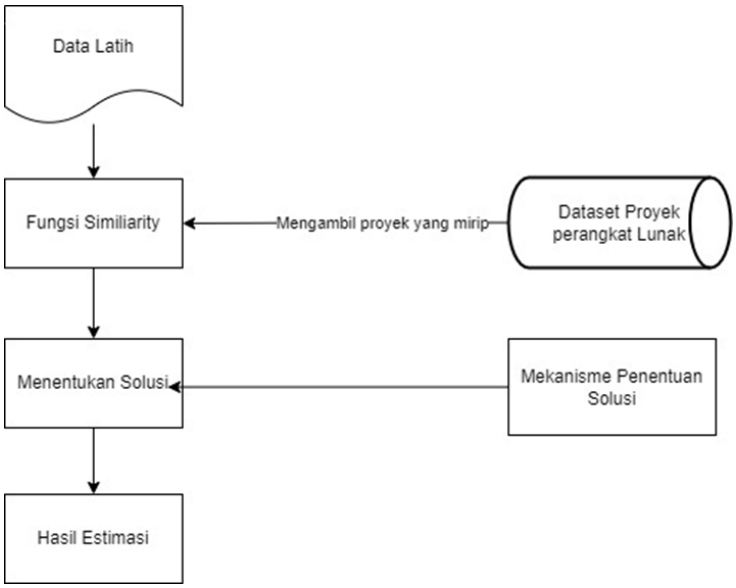
Teknik seleksi yang dipakai dalam penyelesaian algoritma genetika ini adalah teknik seleksi *tournament* dan *elitism*. Dalam seleksi *tournament*, individu dari seluruh populasi bersaing satu sama lain. Individu yang memiliki nilai fitness tertinggi yang akan menang dan diambil untuk diproses lebih lanjut. Teknik seleksi *elitism* menyediakan sarana untuk mengurangi penyimpangan genetik dengan memastikan bahwa kromosom terbaik diizinkan untuk mewariskan atau menyalin sifat mereka ke generasi berikutnya [8]. Melalui penerapan teknik seleksi *tournament* dan *elitism* pada Algoritma Genetika, penelitian ini akan melakukan optimasi proses penentuan nilai bobot atribut usaha perangkat lunak *Analogy*.

2. Metode

Metode yang digunakan adalah metode *Analogy* untuk melakukan estimasi usaha dengan dataset yang digunakan adalah dataset publik yaitu Desharnais, Maxwell, dan Cocomo. Metode ini akan dioptimalkan dengan Algoritma Genetika menggunakan teknik seleksi *tournament* dan *elitism*. Pengembangan metode Algoritma Genetika pada optimasi akan dilakukan pembuatan program dengan bahasa pemrograman *Python*. Penggunaan Algoritma Genetika untuk menentukan bobot yang sesuai untuk setiap atribut usaha agar hasil estimasi menjadi lebih optimal. Untuk mengukur tingkat akurasi estimasi, digunakan beberapa metrik yaitu *Mean Absolute Error* (MAE), *Mean Balanced Residual Error* (MBRE), *Standardize Accuracy* (SA), *Absolute Error* (AE), *Mean Inverted Balanced Residual Error* (MIBRE) dan *Effect Size* (Δ.).

2.1 *Analogy*

Model estimasi usaha perangkat lunak berbasis *Analogy* merupakan proses mengidentifikasi satu atau lebih proyek serupa yang sudah dikembangkan sebelumnya dengan proyek baru yang akan dikembangkan [3]. Metode ini menerapkan beberapa langkah utama seperti pada gambar 1.



Gambar 1. Metode *Analogy*

Langkah pertama untuk melakukan estimasi usaha perangkat lunak adalah mengumpulkan informasi dalam bentuk data pengalaman dari proyek-proyek sebelumnya. Dataset yang berisi informasi tentang pengalaman proyek sebelumnya merupakan bagian penting dari sistem *Analogy*, karena kumpulan data nantinya akan digunakan untuk disesuaikan dengan proyek baru.

Langkah kedua menentukan atribut proyek baru yang sama dengan atribut proyek sebelumnya, dan urutkan proyek sebelumnya berdasarkan peringkat kemiripan proyek baru tersebut. Untuk menentukan peringkat kemiripan, nilai similarity dapat dihitung dengan menggunakan fungsi jarak Euclidean, Manhattan atau Minkowski [9][10]. Euclidean distance adalah metode perhitungan jarak yang digunakan untuk mengukur jarak antara 2 (dua) titik dalam ruang Euclidean (termasuk bidang dua dimensi, tiga dimensi atau bahkan untuk dimensi yang lebih besar) [11]. Manhattan distance digunakan untuk menghitung perbedaan absolute antara koordinat sepasang objek [11]. Minkowski distance adalah metrik dalam ruang vektor di mana norma didefinisikan (ruang vektor standar) dan dianggap sebagai generalisasi jarak Euclidean dan jarak Manhattan. Saat mengukur jarak suatu objek menggunakan jarak Minkowski, nilai p umumnya 1 atau 2 [11]. Rumus ketiga fungsi tersebut dapat dilihat pada persamaan 1, 2, dan 3 di bawah ini:

$$d(x,y) = \sqrt{|x_0 - y_0|^2 + |x_1 - y_1|^2 + \ldots + |x_{n-1} - y_{n-1}|^2 + |x_n - y_n|^2} \quad (1)$$

$$d(x,y) = |x_0 - y_0| + |x_1 - y_1| + \ldots + |x_{n-1} - y_{n-1}| + |x_n - y_n| \quad (2)$$

$$d(x,y) = |x_0 - y_0| + |x_1 - y_1| + \ldots + |x_{n-1} - y_{n-1}| + |x_n - y_n| \quad (3)$$

Dengan:

x_0 hingga x_n merupakan fitur 0 hingga n pada case x

y_0 hingga y_n merupakan fiture 0 hingga n pada case y

Jika jenis data x_n – y_n adalah data kualitatif seperti biner, nominal, ordinal, dan kategorikal maka dihitung dengan persamaan 2.4 sebagai berikut:

$$x_n - y_n = \begin{cases} 1, & jika\ x_n \neq y_n \\ 0, & jika\ x_n = y_n \end{cases} \quad (4)$$

Kemudian, tahapan selanjutnya adalah menggunakan teknik adaptasi *Analogy* untuk menentukan proyek terdekat. Ada empat adaptasi *Analogy* yang dapat diterapkan yaitu *Analogy* terdekat, rata-rata dari *Analogy* terdekat (Mean), nilai tengah dari *Analogy* terdekat (Median), dan rata-rata terbobot peringkat terbalik dari proyek terdekat (IRWA [9]).

1. *Analogy* terdekat berarti memilih satu (K = 1) dari proyek terdekat.
2. Rata-rata proyek terdekat diperoleh dengan menghitung rata-rata usaha dari K > 1 proyek terdekat yang dipilih.
3. Median dari *Analogy* terdekat diperoleh dengan menghitung nilai tengah usaha dari sebanyak K > 2 proyek terdekat yang dipilih.
4. Rata-rata terbobot peringkat terbalik (IRWA) adalah adaptasi *Analogy* yang memberikan bobot tertinggi pada proyek terdekat yang paling mirip dengan proyek lainnya.

Langkah selanjutnya yang diambil untuk menghitung nilai usaha pada proyek baru. Perhitungan dilakukan dengan cara membagi usaha proyek lama dengan ukuran proyek lama kemudian dikalikan dengan ukuran proyek baru. Rumus untuk menghitung nilai usaha proyek baru tersebut terdapat pada persamaan 5 sebagai berikut:

$$Usaha_{proyekBaru} = \frac{Usaha_{proyekTerdekat}}{Ukuran_{proyekTerdekat}} Ukuran_{proyekBaru} \quad (5)$$

2.2 Algoritma Genetika

Algoritma Genetika termasuk *Evolusioner Algorithms* (EAs), menggunakan evolusi biologis dunia nyata. Teori evolusi diperkenalkan pertama kali oleh Charles Darwin. Algoritma Genetika mewakili satu solusi kandidat untuk satu individu dan sekumpulan kandidat solusi sebagai populasi. Untuk menemukan solusi pemecahan masalah, Algoritma Genetika menggunakan metode seleksi, *crossover*, dan mutasi. Solusi terbaik ditemukan dengan cara terus mengulangi proses pencarian keturunan. *Crossover* dan mutasi bersifat stokastik, yaitu meningkatkan keragaman individu untuk memastikan pembaruan sebuah individu. Sebaliknya, seleksi memiliki peran untuk mengurangi keragaman individu dan sebagai kekuatan koersif individu untuk meningkatkan kualitasnya. Untuk mendapatkan individu terbaik, Algoritma Genetika menggunakan delapan langkah sebagai berikut:

1. Membangkitkan populasi awal yang berisi N kromosom, dimana kromosom mewakili kandidat-kandidat solusi.
2. Membangkitkan nilai gen secara acak pada setiap kromosom. Nilai pada suatu gen dapat berbentuk real atau biner.
3. Setelah populasi awal terbentuk, langkah selanjutnya adalah mengevaluasi populasi dengan fungsi fitness. Semakin tinggi nilai fitness sebuah individu semakin besar kemungkinan dia terpilih.
4. Seleksi individu untuk dijadikan sebagai orang tua secara acak.
5. Perbarui populasi dengan menyilangkan (*crossover*) pasangan induk secara acak untuk menghasilkan keturunan baru (anak-anak) dari persilangan.
6. Mutasi individu keturunan baru yang telah dihasilkan berdasarkan nilai probabilitas mutasi (Pm). Langkah ini akan menyebabkan perubahan genetik relatif kecil.

7. Seleksi individu hasil crossover dan mutasi yang digabungkan dengan populasi awal. Langkah ini dilakukan untuk menciptakan populasi baru.
8. Ulangi langkah 3-7 hingga kondisi berhenti didapatkan. Kondisi berhenti bisa berupa jumlah generasi tertentu, jumlah individu tertentu, waktu pemrosesan generasi tertentu, waktu pemrosesan jumlah generasi tertentu, misalnya proses proses dilakukan selama 10 menit dan kriteria-kriteria lainnya [12].

2.3 Deskripsi Dataset

Dataset Desharnais adalah kumpulan 81 proyek perangkat lunak yang dikembangkan oleh perusahaan perangkat lunak Kanada [13]. Namun dari 81 proyek, ada empat proyek yang harus dihapus dari dataset karena nilai beberapa atribut hilang (proyek 38, 44, 66, dan 75). Dengan kata lain, hanya 77 proyek yang memiliki nilai data atribut lengkap dari dataset aslinya dan dapat digunakan sebagai subjek dalam eksperimen penelitian.

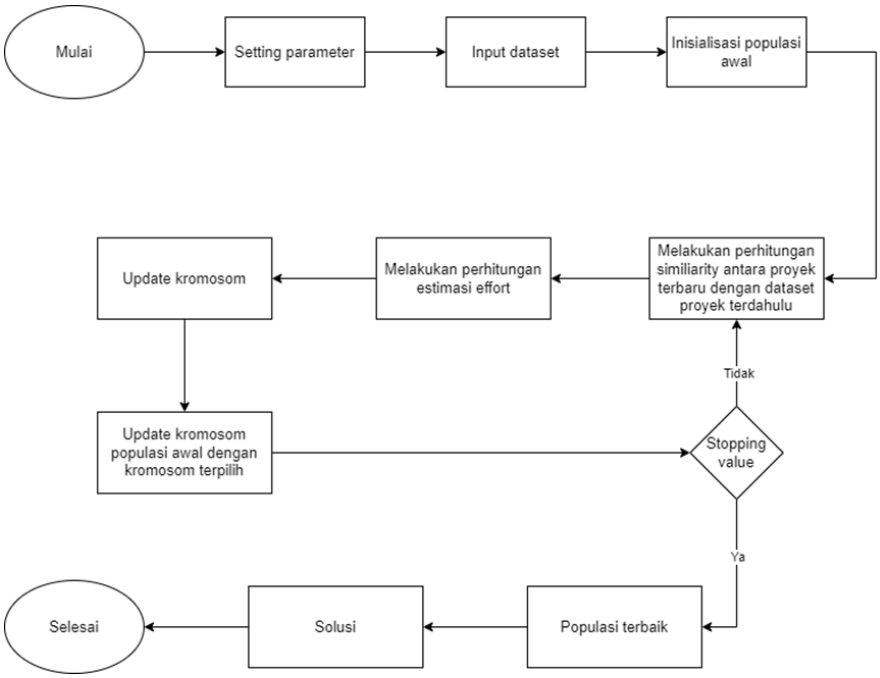
Dataset Maxwell berisi 62 proyek perangkat lunak dari salah satu bank komersial terbesar di Finlandia. Dalam penelitian ini, untuk membuat model estimasi berbasis *Analogy*, dipilih tiga atribut yang mempunyai pengaruh signifikan terhadap proyek, yaitu duration, size, dan effort. Duration adalah atribut numerik yang menunjukkan durasi proyek mulai dari spesifikasi hingga diserahkan ke pelanggan, diukur dalam bulan. Size merupakan atribut numerik yang menunjukkan ukuran proyek perangkat lunak yang diukur dalam satuan titik fungsi (FP). Effort adalah atribut numerik yang mengukur jumlah waktu pengembang perangkat lunak mengerjakan suatu proyek mulai dari spesifikasi hingga diserahkan ke pelanggan, diukur dalam jam.

Dataset Cocomo berisi 63 proyek model estimasi biaya perangkat lunak prosedural yang dikembangkan oleh Barry W. Boehm.

3. Hasil dan Pembahasan

3.1 Implementasi Algoritma

Pada tahap implementasi, Algoritma Genetika bertujuan untuk menentukan bobot atribut usaha yang relevan pada estimasi usaha perangkat lunak *Analogy*. Metode ini dikembangkan menggunakan bahasa pemrograman Python. Untuk tahapan implementasi dapat dilihat pada gambar 2.



Gambar 2. Implementasi Algoritma Genetika pada Metode *Analogy*

3.2 Hasil Pengujian

Penelitian ini mengevaluasi performa Algoritma Genetika dalam mengoptimalkan estimasi usaha perangkat lunak menggunakan *Analogy*. Berikut adalah rangkuman hasil dan pembahasan dari penelitian ini:

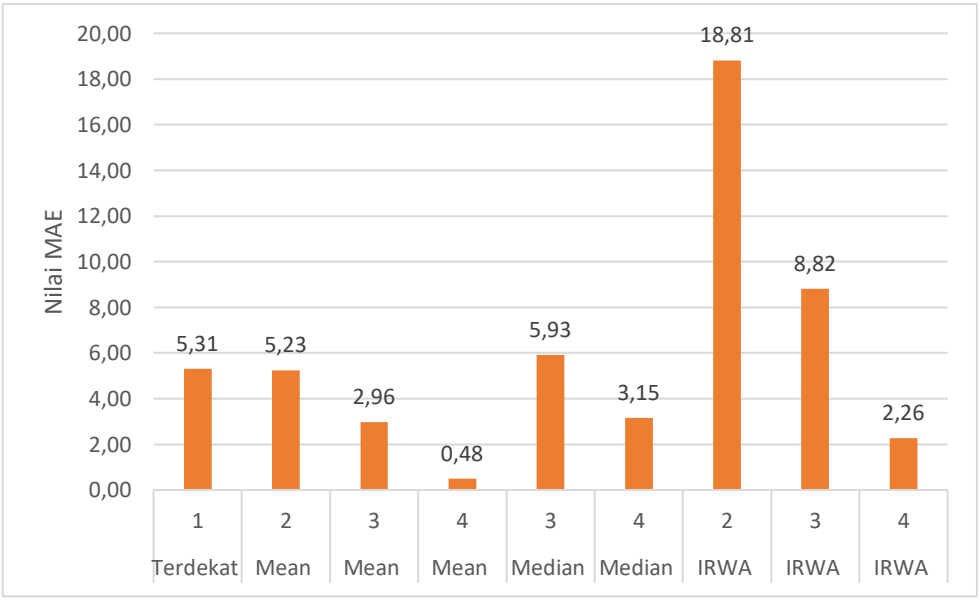
3.2.1 Mean Absolute Error (MAE)

Evaluasi performa Algoritma Genetika dilakukan dengan menghitung nilai *Mean Absolute Error* (MAE) dari hasil estimasi. Proses evaluasi diulang sebanyak 30 kali dengan iterasi yang ditingkatkan secara bertahap. Hasil pengujian dapat dilihat pada tabel 1 berikut.

Tabel 1. Nilai MAE Terendah Setiap Pengujian Model

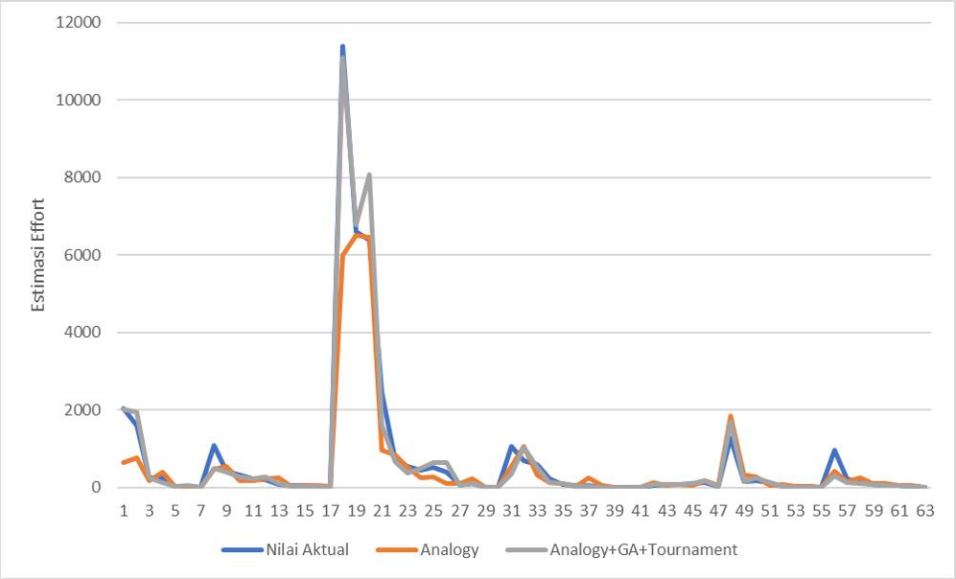
Model	Desharnais-Manhattan		Maxwell-Manhattan		Cocomo-Manhattan	
	Tournament	*Elitism*	*Tournament*	*Elitism*	*Tournament*	*Elitism*
Analogy Terdekat (K1)	9,65	12,08	12,60	12,60	5,31	16,40
Mean Terdekat (K2)	4,27	4,66	17,04	24,24	5,23	7,11
Mean Terdekat (K3)	4,71	7,94	8,33	9,62	2,96	7,17
Mean Terdekat (K4)	1,06	1,27	17,10	18,92	0,48	7,55
Median Terdekat (K3)	7,53	8,64	15,98	16,26	5,93	7,06
Median Terdekat (K4)	0,60	1,32	20,03	21,45	3,15	7,31
IRWA (K2)	152,94	206,93	159,09	203,19	18,81	20,29
IRWA (K3)	27,35	28,47	49,09	52,30	8,82	9,11
IRWA (K4)	0,92	1,89	5,25	5,43	2,26	4,23

Penggunaan 30 individu dan 30 iterasi pada perhitungan Manhattan memberikan hasil estimasi paling optimal pada penelitian ini. Nilai MAE terendah yang dihasilkan adalah 0,48 dengan teknik seleksi *tournament* dapat dilihat pada gambar 3 berikut.



Gambar 3. Perbandingan Nilai MAE

Pengujian terakhir adalah melakukan perbandingan antara estimasi effort menggunakan metode *Analogy* dengan metode *Analogy* yang dioptimasi menggunakan Algoritma Genetika. terdapat peningkatan akurasi ketika estimasi effort dilakukan menggunakan metode *Analogy* yang dioptimasi menggunakan Algoritma Genetika terlihat pada garis berwarna abu-abu yang menunjukkan estimasi *Analogy* dengan Algoritma Genetika mendekati garis berwarna biru yang merupakan nilai effort aktual seperti pada gambar 4 berikut.



Gambar 4. Perbandingan *Analogy* dengan *Analogy*+GA

3.2.2 Wilcoxon Signed Ranked Test

Untuk menguji signifikansi perbedaan antara hasil estimasi AnaIogy dan estimasi *Analogy* yang dioptimalkan dengan Algoritma Genetika digunakan uji statistik Wilcoxon. Hasil uji yang terdapat pada tabel 2 menunjukkan nilai Z sebesar -5.908 dengan nilai signifikansi (Asymp. Sig.) sebesar 0.000, yang berarti perbedaan tersebut sangat signifikan secara statistik (kurang dari alpha 0.05). Hal ini menunjukkan bahwa penerapan Algoritma Genetika dapat memberikan dampak signifikan pada hasil estimasi *Analogy*.

	Analogy GA - *Analogy*
Z	-5.908
Asymp. Sig. (2-tailed)	0.000

4. Kesimpulan

Penelitian ini bermaksud untuk mengoptimalkan bobot similarity estimasi usaha perangkat lunak *Analogy*. Penelitian sebelumnya menunjukkan bahwa metode *Analogy* biasa tidak dapat memberikan hasil estimasi yang optimal dikarenakan setiap atribut usaha memiliki bobot yang sama, padahal pemilihan bobot yang tepat untuk setiap atribut usaha dapat memengaruhi akurasi model. Oleh karena itu, penelitian ini menerapkan Algoritma Genetika dengan teknik seleksi *elitism* dan *tournament* untuk membantu menentukan bobot similarity yang sesuai pada metode *Analogy*.

Penelitian ini menunjukkan bahwa teknik adaptasi Mean K=4 dengan jarak Manhattan untuk seleksi *tournament* memberikan hasil estimasi usaha optimal. Metode *Analogy* yang dioptimasi menggunakan Algoritma Genetika memberikan nilai Mean Absolute Error (MAE) yang lebih baik sebesar 0,482737245. Sementara itu metode *Analogy* tanpa dioptimasi dengan Algoritma Genetika menghasilkan nilai MAE lebih besar sebesar 314,4224241. Hasil uji statistik non-parametrik Wilcoxon menunjukkan nilai Signifikansi Asimptotik (Asymp. Sig.) sebesar 0.00 menunjukkan adanya perbedaan yang signifikan antara hasil estimasi usaha *Analogy* yang dioptimalkan dengan Algoritma Genetika dan *Analogy* dalam bentuk standar.

5. Referensi

[1] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," 2012, *Elsevier B.V.* doi: 10.1016/j.infsof.2011.09.002.

[2] L. L. Minku and X. Yao, "Software effort estimation as a multiobjective learning problem," *ACM Transactions on Software Engineering and Methodology*, vol. 22, no. 4, Oct. 2013, doi: 10.1145/2522920.2522928.

[3] M. Shepperd and C. Schofield, "Effort Estimation".

[4] P. Phannachitta, "On an optimal *Analogy*-based software effort estimation," *Inf Softw Technol*, vol. 125, Sep. 2020, doi: 10.1016/j.infsof.2020.106330.

[5] S. J. Huang and N. H. Chiu, "Optimization of *Analogy* weights by genetic algorithm for software effort estimation," *Inf Softw Technol*, vol. 48, no. 11, pp. 1034–1045, Nov. 2006, doi: 10.1016/j.infsof.2005.12.020.

[6] W. Mustafa, "Optimization of production systems using genetic algorithms," vol. 3, pp. 233–248, 2003.

[7] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimed Tools Appl*, vol. 80, no. 5, pp. 8091–8126, Feb. 2021, doi: 10.1007/s11042-020-10139-6.

[8] J. J. Grefenstette, "Optimization of Control Parameters for Genetic Algorithms," vol. 16, no. 1.

[9] Ardiansyah, M. M. Mardhia, and S. Handayaningsih, "*Analogy*-based model for software project effort estimation," *International Journal of Advances in Intelligent Informatics*, vol. 4, no. 3, pp. 251–260, Nov. 2018, doi: 10.26555/ijain.v4i3.266.

[10] A. Idri, M. Hosni, and A. Abran, "Improved estimation of software development effort using Classical and Fuzzy *Analogy* ensembles," *Applied Soft Computing Journal*, vol. 49, pp. 990–1019, Dec. 2016, doi: 10.1016/j.asoc.2016.08.012.

[11] M. Nishom, "Perbandingan Akurasi Euclidean Distance, Minkowski Distance, dan Manhattan Distance pada Algoritma K-Means Clustering berbasis Chi-Square," *Jurnal Informatika: Jurnal Pengembangan IT*, vol. 4, no. 1, pp. 20–24, Jan. 2019, doi: 10.30591/jpit.v4i1.1253.

[12] Suyanto, *Machine Learning : Tingkat Dasar dan Lanjut*, 1st ed. Bandung: INFORMATIKA, 2018.

[13] A. Bou Nassif, L. Fernando Capretz, and D. Ho, "Analyzing the Non-Functional Requirements in the Desharnais Dataset for Software Effort Estimation," 2012. [Online]. Available: https://ir.lib.uwo.ca/electricalpubhttp://ir.lib.uwo.ca/electricalpub