Menyesuaikan Tingkat Kesulitan Permainan menggunakan Algoritma Genetika dengan Shanon-Wiener Diversity Index sebagai Dasar Pengambilan Keputusan untuk Menginterupsi Konvergensi Prematur

Adjusting the Level of Difficulty of Games using Genetic Algorithms with the Shanon-Weiner Diversity Index as a Basis for Decision Making to Interrupt Premature Convergence

Rachmad Imam Tarecha*[1], Soleman[2], Yan Riyanto[3]
[1,2,3] Program Studi Magister Ilmu Komputer, FTI Universitas Budi Luhur, Jakarta
e-mail: *[1]**ri.tarecha@gmail.com**, [2]solemediagrafik@gmail.com, [3]yanr001@gmail.com

Abstrak

Menyesuaikan tingkat kesulitan dengan perilaku pemain pada permainan menjadi penunjang agar pemain tidak berhenti bermain karena permainan yang dirasa terlalu sulit. Algoritma Genetika dapat digunakan sebagai metode untuk penyesuaianya. Metode ini berkerja dengan menyesuaikan tingkat kesulitan permainan sesuai dengan nilai fitness seorang pemain. Sayangnya, jika target nilai fitness mendekati ambang batas maksimal, algoritma ini membutuhkan proses iterasi yang banyak, bahkan memungkinkan terjadinya konvergensi prematur. Konvergensi prematur ini terjadi karena kurangnya keragaman populasi, sehingga nilai fitness minimal yang ditetapkan tidak memungkinkan untuk dicapai. Penelitian ini mencoba menginterupsi konvergensi prematur pada algoritma genetika agar tidak terjadi pengulangan iterasi yang tak berkesudahan dengan menetapkan dan menghitung batas keragamanya menggunakan Shanon-Wiener Diversity Index. Hasilnya, ketika ambang batas keragaman tercapai, nilai fitness saat keadaan itu diambil sebagai nilai paling optimum, sehingga tidak terjadi pengulangan iterasi yang tak berujung.

Kata kunci—Konvergensi prematur, algoritma genetika, Shanon-Wiener Diversity Index

Abstract

Adjusting the level of difficulty with the behavior of players in the game to be supportive so that players do not stop playing because the game is considered too difficult. Genetic Algorithms can be used as a method for adjustments. This method works by adjusting the difficulty level of the game according to the value of a player's fitness. Unfortunately, if the target fitness value approaches the maximum threshold, this algorithm requires a lot of iteration processes, even allowing premature convergence to occur. This premature convergence occurs because of a lack of population diversity, so the minimum fitness value that is set is not possible to achieve. This study attempts to interrupt premature convergence of genetic algorithms to avoid endless iterations by setting and calculating diversity limits using

the Shanon-Wiener Diversity Index. As a result, when the diversity threshold is reached, the fitness value when the state is taken as the most optimum value, so that there are no endless iterations.

Keywords—Premature convergence, genetic algoritm, Shanon-Wiener Diversity Index

1. PENDAHULUAN

Pada permainan perlu dilakukan upaya penyesuaian tingkat kesulitan permainan dengan kapasitas atau perilaku pemain. Penyesuaian ini dilakukan sebagai upaya agar pemain tidak mudah bosan, mengeluh, atau meninggalkan permainan karena kesulitanya yang dirasa tidak sesuai dengan kapasitas atau perilaku pemain[1]. Tingkat kesulitan permainan yang tepat juga merupakan salah satu penunjang kualitas dari sebuah permaianan[2].

Penyesuaian tingkat kesulitan permainan dapat dilakukan menggunakan algoritma optimasi genetika, sebuah algoritma yang berkerja dengan mencari nilai optimum berdasarkan fungsi nilai *fitness*[3]. Algoritma genetika terbukti cukup cepat untuk mengatasi masalah optimasi kombinatorial. Untuk mengatasi masalah kombinatorial, algoritma genetika dapat menyelesaikan masalah dalam 40 iterasi, dengan kecepatan kurang dari 10 detik saja [4].

Masalah kombinatorial tersebut merupakan masalah yang harus dihadapi sistem untuk proses penyesuaian tingkat kesulitan permainan, dengan skenario menetapkan nilai *fitness* yang menjadi target atau batas toleransi solusi sesuai dengan perolehan skor pemain. Sayangnya, algoritma ini memiliki kelemahan karena adanya potensi terjadi konvergensi prematur[5]. Kondisi tersebut terjadi karena kurangnya keragaman variasi populasi yang menjadi kandidat solusi yang ditawarkan[6]. Kondisi tersebut membuat iterasi algoritma genetika tidak dapat mencapai ambang batas nilai *fitness* yang ditetapkan.

Dari 5 kali percobaan, algoritma genetika mengalami 2 kali konvergensi prematur, atau sekitar 40%. Saat mengalami konvergensi prematur, algoritma genetika akan melakukan perulangan iterasi terus menerus jika tidak pernah sampai pada ambang batas nilai *fitness* yang ditentukan. Hal ini akan membuat sistem berkerja dengan percuma, jika kandidat solusi dari algoritma genetika tidak memungkinkan lagi untuk menghasilkan solusi yang lebih baik. Karena permasalahan konvergensi prematur tersebut, sistem harus melakukan interupsi agar sistem tetap dapat bekerja. Untuk itu penelitian ini diharapkan mampu mengatasi permasalahan tersebut..

2. METODE PENELITIAN

Sebagai upaya penanggulangan agar iterasi algoritma genetika dalam permainan tidak mengalami kondisi pengulangan tanpa henti karena tidak mampu mencapai ambang batas nilai *fitness* yang ditentukan, kondisi ini perlu dideteksi dan segera dilakukan interupsi agar sistem dapat tetap berjalan tanpa perulangan iterasi tanpa henti. Karena penyebab utama adalah kurangnya keragaman populasi. Maka, tingkat keragaman perlu diukur agar dapat dilakukan interupsi untuk mencegah perulangan iterasi yang tanpa henti. Salah satu metode untuk mengukur tingkat keragaman pada populasi kandidat calon solusi menggunakan Shanon-Weiner Diversity Index[7].

Setelah tingkat keragaman dihitung pada populasi kandidat calon solusi, tingkat keragaman tersebut dijadikan batas untuk melakukan interupsi pada algoritma genetika. Jika tingkat keragaman mencapai nilai tertentu, maka iterasi algoritma genetika dapat diinterupsi dengan segera untuk menghindari perulangan yang tanpa henti. Kemudian, nilai *fitness* dalam posisi terakhir dikondisikasn sebagai nilai paling optimum yang mungkin dicapai. Setelah itu, nilai optimum digunakan untuk mengenerasi nilai untuk tingkat kesulitan pada level berikutnya.

Pengujian pada penilitan ini dilakukan dengan cara menghitung nilai *fitness* solusi, jumlah iterasi, indeks keragaman, memori yang digunakan, kecepatan proses, hingga berapa kali interupsi aktif pada saat iterasi algoritma genetika berjalan untuk menyesuaikan tingkat kesulitan permainan. Untuk lebih detailnya, penelitian ini dijabarankan kedalam beberapa subbab sebagai berikut:

2.1 Desain Permainan

Permainan yang dibuat adalah permainan melawan monster. Sebelum melawan monster, pemain perlu menyiapkan atau melatih karakter dalam permainan untuk meningkatkan kekuatan. Saat berlatih untuk meningkatkan kemampuan, uang yang tersedia akan berkurang untuk biaya pelatihan, juga untuk membeli item. Jika uang habis, pemain dapat mendapatkan uang tambahan dengan menjawab pertanyaan. Jika kekuatan telah melebih monster, maka pemain akan menang jika bertarung. Sebaliknya jika kekuatan pemain lebih lemah, maka akan kalah. Jika kalah, sistem akan memberikan lawan yang lebih sesuai dengan pencapaian pelatihan pemain menggunakan algoritma genetika dengan batasan *Shanon-Weiner Diversity Index* yang telah ditentukan.

2. 2 Tampilan Permaian

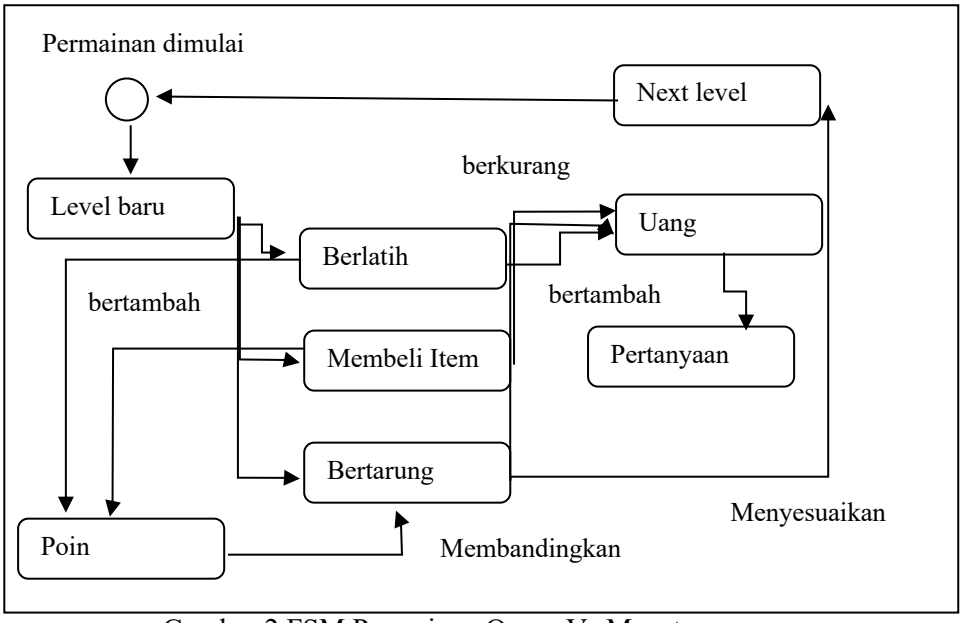
Tampilan permainan ini memuat tombol untuk berlatih, bertarung, dan menjawab pertanyaan untuk mengumpulkan uang. Untuk lebih jelasnya dapat dilihat pada gambar 1 berikut ini:

Gambar 1 Tampilan Permaian Orang Vs Monster (Semua gambar dalam permainan ini didukung oleh https://game-icons.net)

Permainan Orang VS Monster ini, pada laman utamanya terdapat tampilan level, uang, poin kekuatan pemain, dan poin kekuatan monster, serta tombol untuk berlatih dan membeli item. Jika uang habis maka akan muncul tampilan pertanyaan di bawah judul, jika jawaban benar akan muncul notifikasi uang telah bertambah. Kemudian jika pertarungan menang, akan nada notifikasi naik level. Jika kalah akan memulai permainan baru dengan monster lain yang lebih sepadan dengan perilaku atau poin pemain.

2. 3 Finite State Machine

Dalam permainan melawan monster ini, aktivitas yang dapat dilakukan adalah menambah melatih karakter pamain, membeli item, mengumpulkan uang dengan menjawab pertanyaan, dan melawan monster. Sebagai gambarang yang lebih jelas, perhatikan gambar Finite State Machine (FSM) pada gambar 1 berikut ini yang memuat *state (*keadaan), *event* (kejadian), dan *action* (aksi)[8] :



Gambar 2 FSM Permainan Orang Vs Monster

Pada gambar 2 FSM Permainan Orang Vs Monster di atas mengambarkan bahwa pemain dapat melakukan aktivitas berlatih, membeli item dan bertarung. Saat berlatih dan membeli item uang pemain akan berkurang, jika habis dapat menjawab pertanyaan untuk menambah uang. Setelah itu pemain bias bertarung untuk mendapatkan uang dan naik ke level berikutnya. Jika kalah, level akan disesuaikan sesuai kapasitas atau kondisi perolehan poin pemain sebelumnya. Tabel 1 berikut ini adalah susunan poin setiap itemnya:

Tabel 1 Poin Pemainan

No.	Gambar Menu	Atribusi	Perolehan Poin	Kebutuhan Uang
1.		Images by Skoll under CC BY 3.0	+1 poin persenjataan	50
2.		Images by Lorc under CC BY 3.0	+2 poin pertahanan	50
3.		Images by Delapouite under CC BY 3.0	+2 poin kecepatan	50
4.		Images by Delapouite under CC BY 3.0	+1 poin serangan	10
5.		Images by Delapouite under CC BY 3.0	+2 poin serangan	150

Pada awal permianan, atau pada kondisi level 1 pemain akan dibekali uang sebesar 400. Jika uang habis pemain dapat menjawab pertanyaan untuk menambah uang tambahan sebesar 100. Setelah poin terkmpul dan bertarung dengan monster, pemain akan mendapat uang sebesar 200 dan mendapat kenaikan level 1 poin.

2. 4 Algoritma Genetika

Algoritma ini berbasis pada mekanisme seleksi alam dan genetika[9]. Algoritmanya bekerja dengan cara membangkitkan populasi secara acak[10]. Populasi yang dibangkitkan merupakan kandidat solusi. Kandidat yang ada dievaluasi dengan nilai *fitness* pemain. Nilai *fitness* adalah nilai yang menyatakan seberapa baik sebagai solusi[11]. Setelah itu dilakukan seleksi menggunakan metode *roullete whell,* dimana nilai *fitness* terbaik berpeluang lebih besar untuk terpilih[12]. Kemudian dilakukan *on cut crossover* dengan probabilitas crossover sebesar 1 untuk meningkatkan keragaman populasi. Juga dilakukan mutasi kromosom dengen tingkat probabilitas mutasi sebesar 0.1. Semakin tinggi *rate crossover* akan menghasilkan solusi yang mirip dengan nilai induk, dan jika *rate* mutase lebih rendah dari *crossover* maka algoritma genetika performanya menurun[13]. Perhitungan dilakukan perulangan hingga mencapai nilai *fitness* pemain. Fungsi *fitness* terbagi menjadi dua macam, untuk optimasi dan minimalisasi[14]. Dalam penelitian ini digunakan untuk optimasi dengan fungsi optimasi sebagai berikut*:*

$$f(x) \quad ;(10 * \sum x) - (x\max - xperolehan) \quad (1)$$

Pada persamaan 1 diatas, nilai f(x) didapat dari pengurangan selisih nilai maksimal dan nilai perolehan dengan nilai maksimum setiap poin (10) dikalikan dengan jumlah perolehan poin. Setelah didapati nilai *fitness*nya barulah dijadikan ambang batas untuk proses iterasi algoritma genetika.

2. 5 Shanon-Wiener Diversity Index

Shanon-Wiener Diversity Index ini untuk mengukur tingkat keragaman sebuah populasi. Fungsi *Shanon-Wiener* dapat dinyatakan dalam persamaan berikut[15]:

$$H' = -\sum_{i=1}^{n} (pi * \ln(p_i)) \quad (2)$$

Persamaan berikut dapat berjalan dimana p_i merupakan proporsi dari semua spesies. Nilai H' maksimum sama dengan logaritma natural dari jumlah spesies n. Jika H' kurang dari 1 maka termasuk kategori keragaman rendah, jika lebih dari 1 dan kurang dari 3 masuk kategori keragaman rendah, jika lebih dari 3, maka masuk kategori keragaman tinggi[16].

2. 6 Pengujian

Pengujian dilakukan dengan menghitung waktu proses, dan jumlah iterasi, dan jumlah interupsi saat terjadi konvergensi prematur. Adapun satuan ukur sebagai berikut:

Tabel 2 Satuan Ukur

Kecepatan	Memori	Jumlah Iterasi	Jumlah Interupsi	*Fitness*	*H'*
Detik	MB	Buah	Buah	x dari f(x)	H'<1= Keragaman Rendah 1 < H' <3 = Keragaman Sedang H' > 3 = Keragaman Tinggi

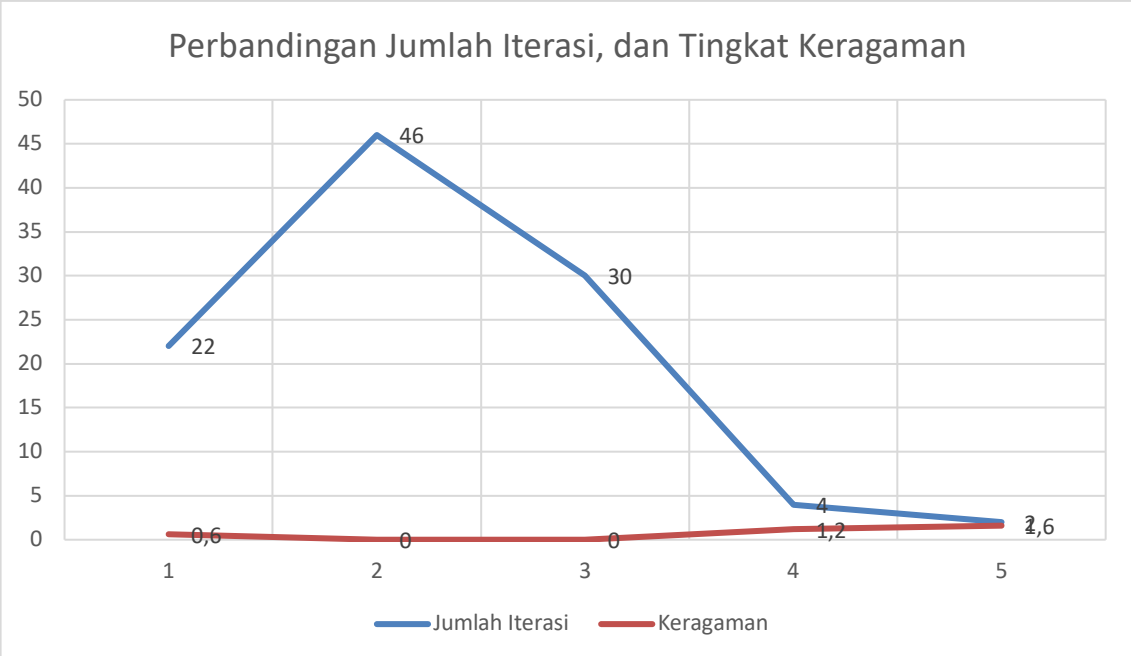
3. HASIL DAN PEMBAHASAN

Pada penelitian ini dilakukan sepuluh kali percobaan perhitungan algoritma genetika dengan hasil seperti pada table 3 berikut ini:

Tabel 3 Hasil Pengujian

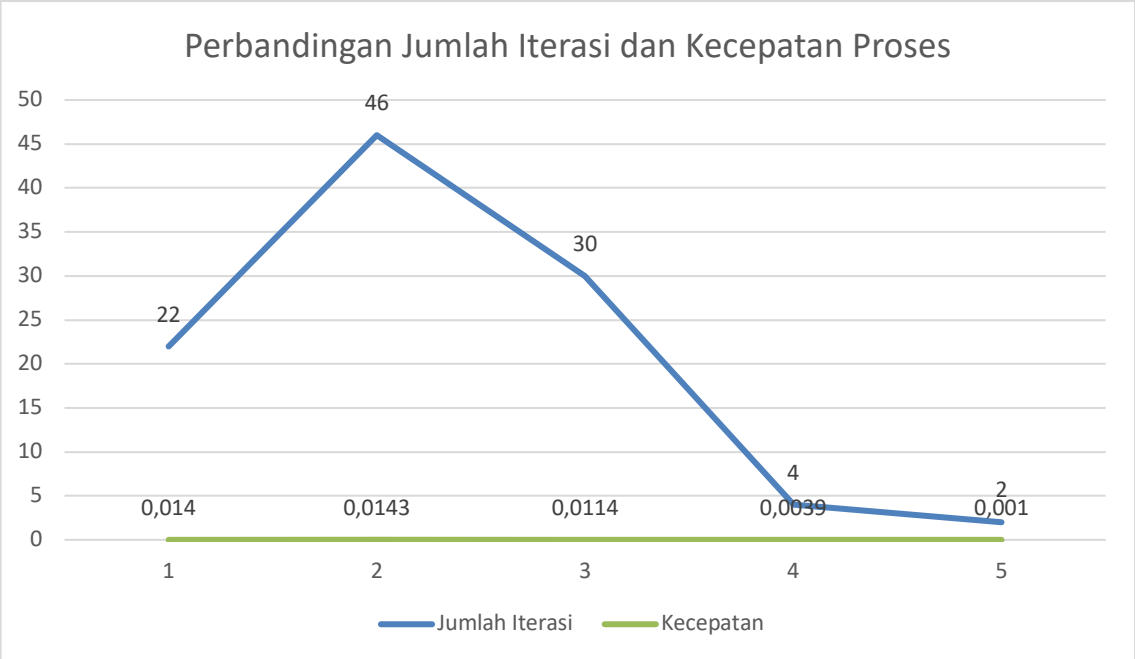
No	Kecepatan	Memori	Jumlah Iterasi	Jumlah Interupsi	*Fitness*	*H'*
1	0.0140 s	0.39 MB	22	0	36 dari 36	0.6
2	0.0143 s	0.39 MB	46	1	32 dari 36	0
3	0.0114 s	0.39 MB	30	1	26 dari 30	0
4	0.0039 s	0.39 MB	4	0	32 dari 32	1.2
5	0.0010 s	0.39 MB	2	0	28.5 dari 26	1,6

Tabel 3 menunjukkan, dalam 5 kali percobaan perhitungan algoritma genetika didapati 2 kali mengalami konvergensi prematur hingga harus dilakukan interupsi agar dapat berjalan dengan baik meski nilai *fitness* belum memenuhi target.



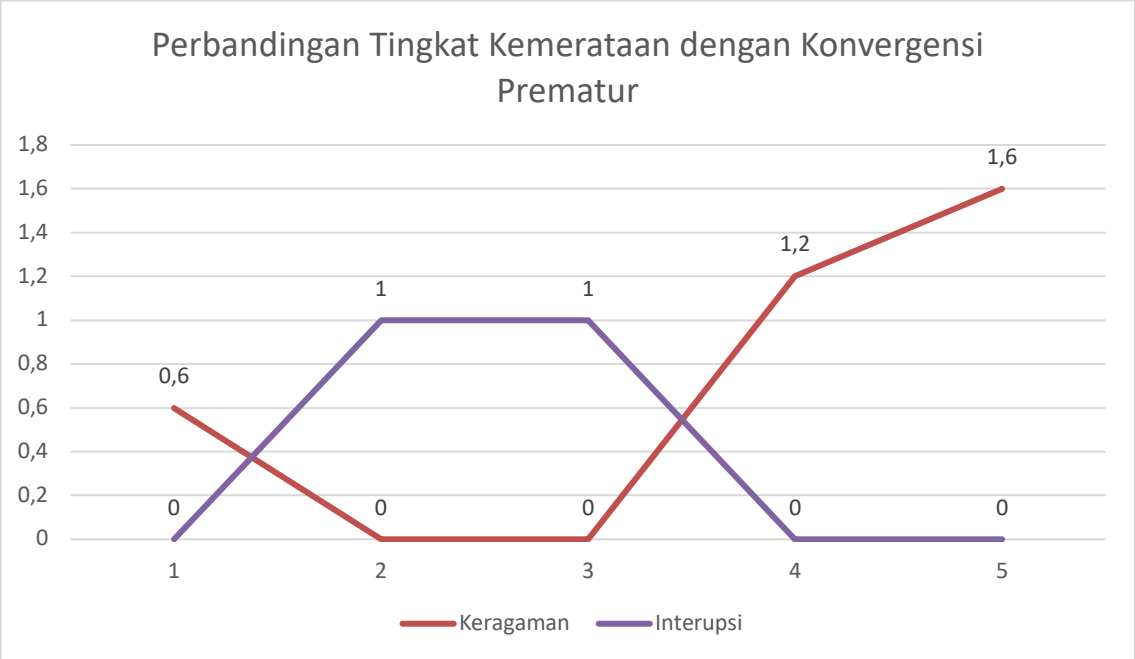
Gambar 3 Grafik Perbandingan Jumlah Iterasi, dan Tingkat Keragaman

Gambar 3 merupakan bentuk perbandingan jumlah iterasi algoritma genetika dengan tingkat keragaman populasi. Grafik pada gambar 3 menunjukan jika nilai keragaman semakin tinggi, semakin sedikit proses iterasi yang diperlukan.



Gambar 4 Grafik Perbandingan Jumlah Iterasi dan Kecepatan Proses

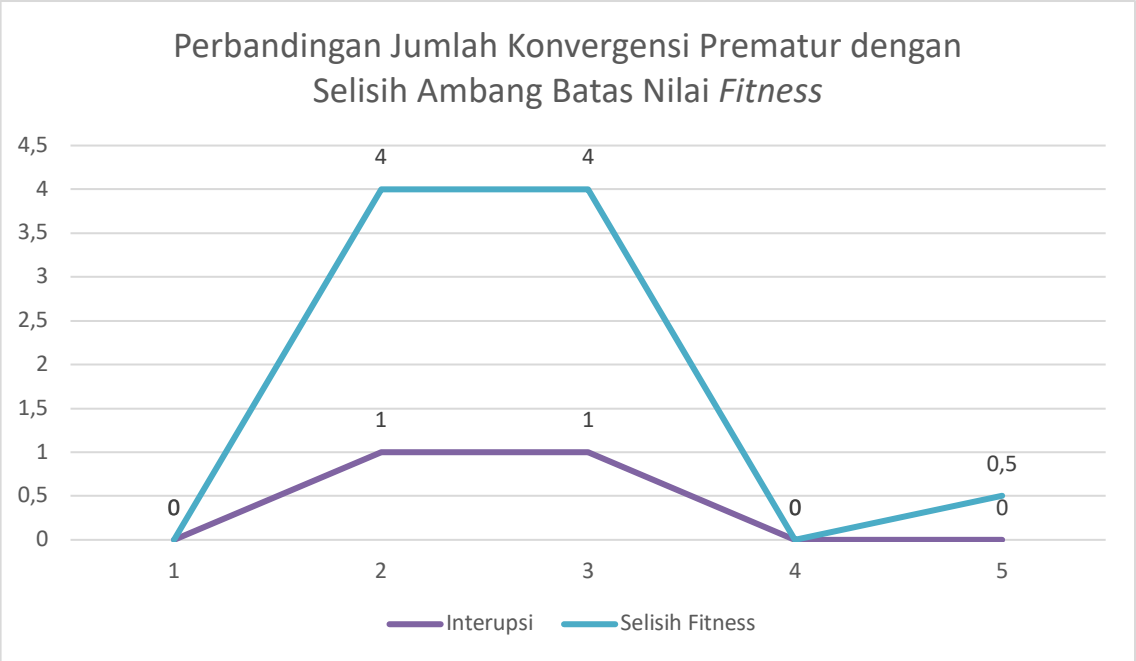
Dari gambar 4, menunjukkan bahwa semakin banyak jumlah iterasi, maka semakin tinggi pula waktu proses yang dibutuhkan. Artinya semakin sedikit proses iterasi, maka proses perhitungan semakin cepat. Dari percobaan pada Tabel 3 didapati proses tercepat ada pada percobaan ke-5 dengan jumlah iterasi sebanyak 2 buah, ditempuh dalam waktu 0.001 detik. Sedangkan proses yang memakan waktu paling lama pada percobaan kedua dengan jumlah iterasi sebanyak 46 buah, ditempuh dengan waktu 0.0143 detik.



Gambar 5 Grafik Perbandingan Tingkat Kemerataan dengan Konvergensi Prematur

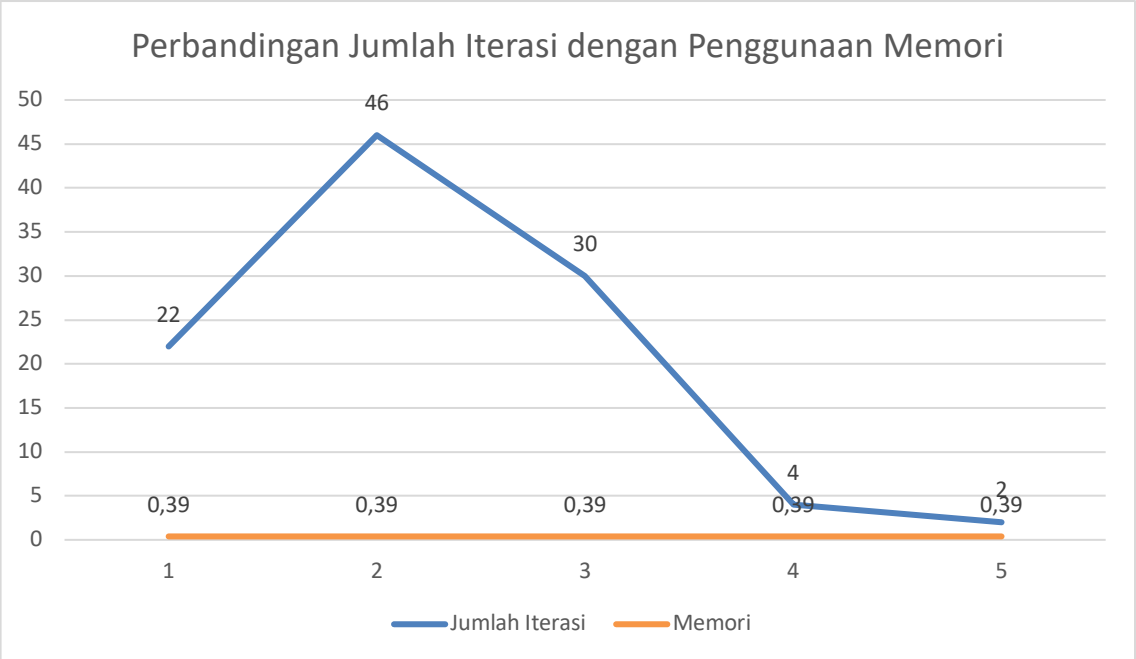
Dari gambar 5, menunjukkan bahwa semakin tinggi tingkat keragaman populasi yang dihitung dengan rumus Shanon-Wiener Diversity Index, maka semakin rendah kemungkinan

terjadinya konvergensi prematur. Dari percobaan pada Tabel 3, terjadi 2 kali konvergensi prematur pada populasi kandidat calon solusi yang tidak memiliki keragaman dengan nilai keragaman 0 yang termasuk pada kategori keragaman rendah.



Gambar 6 Grafik Perbandingan Jumlah Konvergensi Prematur dengan Selisih Ambang Batas Nilai *Fitness*.

Dari gambar 6, menunjukkan bahwa saat terjadi konvergensi prematur, selisih antara nilai *fitness* optimum yang mungkin dicapai dengan ambang batas minimal lebih tinggi daripada saat tidak terjadi konvergensi prematur. Dari percobaan Tabel 3, terdapat 2 kali konvergensi prematur dengan selisih nilai *fitness* optimum yang mungkin dicapai dengan ambang batas minimal sebesar 4 poin di bawah ambang batas minimal. Sementara itu, 2 percobaan mendapatkan selisih 0 dari ambang batas minimal, dan 1 percobaan selisih 0.5 melebihi ambang batas minimal.



Gambar 7 Grafik Perbandingan Jumlah Iterasi dengan Penggunaan Memori

Dari gambar 7, menunjukkan bahwa jumlah iterasi algoritma genetika tidak terlalu berpengaruh pada besaran penggunaan memori. Dari 5 kali percobaan pada Tabel 3, memori yang digunakan tetap pada nilai 0.39 MB.

4. KESIMPULAN

Dari 5 kali percobaan pada Tabel 2, terjadi 2 kali konvergensi prematur atau sekitar 40%. Hasil penelitian ini dapat ditarik kesimpulan sebagai berikut:

1. Konvergensi prematur cenderung terjadi pada populasi kandidat calon solusi yang tidak memiliki kergaman, dibuktikan dengan nilai keragaman rendah sebesar 0.
2. Karena tingkat keragaman berbanding lurus dengan jumlah konvergensi prematur, maka Shanon-Weiner Diversity Index juga dapat digunakan untuk deteksi dini konvergensi prematur pada algoritma genetika.
3. Konvergensi prematur terjadi saat populasi kandidat calon solusi tidak memiliki keragaman atau bernilai sama, sehingga saat proses *crossover*, dan *mutasi* nilai yang diolah sama, dan tidak mungkin untuk menghasilkan kandidat calon sulusi lain yang lebih baik.
4. Konvergensi prematur menyebabkan kandidat solusi tidak mencapai nilai ambang batas nilai *fitness* minimal yang telah ditentukan,
5. Selain itu semakin tinggi nilai *fitness* yang harus dicapai maka cenderung semakin tinggi pula jumlah iterasi algoritma.
6. Untuk kecepatan, berbanding lurus dengan jumlah iterasi algoritma genetika.
7. Besaran memori yang diperlukan tidak terlalu terpengaruh pada jumlah iterasi algoritma genetika.

5. SARAN

Pada penelitian ini telah berhasil mengidentifikasi kapan konvergensi prematur terjadi saat proses berjalan. Namun, pada penelitian ini masih terbatas pada tahap untuk menginterupsi dan menghentikan iterasi secara paksa meski belum memenuhi nilai *fitness* minimal untuk menghindari perulangan iterasi tanpa henti. Penelitian ini bisa dikembangkan dengan mengatasi konvergensi prematur yang telah dapat teridentifikasi dengan menghitung indeks keragamanya menggunakan Shanon-Weiner Diversity Index.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada semua pihak yang telah memberi dukungan terhadap penelitian ini.

DAFTAR PUSTAKA

[1] R. Sutoyo, D. Winata, K. Oliviani, and D. M. Supriyadi, "Dynamic Difficulty Adjustment in Tower Defence," *Procedia Comput. Sci.*, vol. 59, no. Iccsci, pp. 435–444, 2015.

[2] P. W. Atmaja, D. O. Siahaan, and I. Kuswardayan, "Game Design Document Format For Video Games With Passive Dynamic Difficulty Adjustment," *Regist. J. Ilm. Teknol. Sist. Inf.*, vol. 2, no. 2, p. 86, 2016.

[3] R. Rismala and M. D. Sulistyo, "PENERAPAN TEKNIK KLASIFIKASI PADA SISTEM REKOMENDASI MENGGUNAKAN ALGORITMA GENETIKA," *J. Ilm. Teknol. Inf. Terap.*, vol. II, no. 3, 2016.

[4] A. W. Widodo and W. F. Mahmudy, "Penerapan Algoritma Genetika Pada Sistem Rekomendasi Wisata," *Ilm. KURSOR*, vol. 5, no. 4, pp. 205–211, 2010.

[5] M. R. Kamal, R. Satria, A. Syukur, F. I. Komputer, and U. D. Nuswantoro, "Integrasi Kromosom Buatan Dinamis untuk Memecahkan Masalah Konvergensi Prematur pada Algoritma Genetika untuk Traveling Salesman Problem," vol. 1, no. 2, pp. 61–66, 2015.

[6] C. Science and S. Engineering, "Preventing Premature Convergence in Genetic Algorithm Using DGCA and Elitist Technique," vol. 4, no. 6, pp. 410–418, 2014.

[7] E. Elejalde, L. Ferres, E. Herder, and J. Bollen, "Quantifying the ecological diversity and health of online news," *J. Comput. Sci.*, vol. 27, pp. 218–226, 2018.

[8] M. F. Rahadian, A. Suyatno, and S. Maharani, "Penerapan Metode Finite State Machine Pada Game 'The Relationship,'" *Inform. Mulawarman J. Ilm. Ilmu Komput.*, vol. 11, no. 1, p. 14, 2016.

[9] T. K. S. Okol Sri Suharyo, Bambang Suhardjo, "ASRO JURNAL- STTAL Vol. 8 ; Juli-Des 2017," *Asro J. - STTAL*, vol. 8, 2017.

[10] F. Purwanto, E. C. Djamal, and A. Komarudin, "Optimalisasi Penempatan Halte Trans

Metro Bandung Menggunakan Algoritma Genetika," *Semin. Nas. Apl. Teknol. Inf.*, pp. 36–38, 2016.

[11] R. Firmansyah *et al.*, "PERAKITAN MOBIL RADIO CONTROL BUGGY NITRO OFFROAD," pp. 1–7.

[12] S. M. Lim, A. B. Sultan, N. Sulaiman, A. Mustapha, and K. Y. Leong, "Crossover and Mutation Operators of Genetic Algorithms," *Int. J. Mach. Learn. Comput.*, vol. 7, no. 1, pp. 10–13, 2017.

[13] E. Nur Azizah, I. Cholissodin, and W. Firdaus Mahmudy, "Optimasi Fungsi Keanggotaan Fuzzy Tsukamoto Menggunakan Algoritma Genetika Untuk Penentuan Harga Jual Rumah," *J. Enviromental Eng. Sustain. Technol.*, vol. 2, no. 2, pp. 79–82, 2015.

[14] Muliadi, "Pemodelan Algoritma Genetika Pada Sistem Penjadwalan Perkuliahan Prodi Ilmu Komputer Universitas Lambungmangkurat," *Kumpul. J. Ilmu Komput.*, vol. 01, no. 01, pp. 67–78, 2014.

[15] E. N. N. Riris Aryawati, T. Zia Ulqodry, Heron Subakti, "Populasi Fitoplankton Skeletonema di Estuaria Banyuasin, Sumatera Selatan," vol. 10, no. 2, pp. 269–276, 2018.

[16] N. Thomas Hidayat, "Biodiversity Indices of Demersal Fish," *J. Penelit. Perikan. Indones.*, vol. 6, no. April, pp. 47–53, 2014.