Jurnal Inovasi Teknik dan Edukasi Teknologi, 1(9), 2021, 676-684
ISSN: 2797-7196 (online)
DOI: 10.17977/um068v1i92021p676-684

Komparasi Algoritma WOA, MFO dan Genetic pada Optimasi Evolutionary Neural Network dalam Menyelesaikan Permainan 2048

Hendrawan Armanto, Kevin Setiabudi*, C. Pickerling
Institut Sains dan Teknologi Terpadu Surabaya, Jl. Ngagel Jaya Tengah 73-77 Surabaya, Jawa Timur, Indonesia
*Penulis korespondensi, Surel: hendrawan@stts.edu

Paper received: Paper received: 01-9-2021; revised: 11-9-2021; accepted: 17-9-2021

Abstract
Neural network optimization using evolutionary algorithms is an interesting research topic. But right now, there are not much research in this topic that focused on Game, especially 2048. The 2048 game is one of the interesting games to study considering that the level of difficulty of this game will increase when the value of the resulting number increases. In addition, this game is also not limited by time but can be played continuously until the game ends. Neural network and tree are 2 architectures that can be used to play 2048 but require a long training time if you want to play well. In this study, this problem was optimized by an evolutionary algorithm (3 algorithms used in this study: Genetic Algorithm, WOA, and MFO). With this optimization, the best weight will be obtained in either the NN or Tree architecture to produce good intelligence in playing 2048. After going through various trials, it is concluded that the combination with the NN architecture is better than the Tree architecture and the WOA and MFO algorithms have succeeded in optimizing the architecture with better than the genetic algorithm.

Keywords: evolving neural network; genetic algorithm; woa; mfo

Abstrak
Optimasi neural network menggunakan algoritma evolutionary adalah topik penelitian yang menarik akan tetapi tidak banyak penelitian terkait hal ini yang berfokus pada game terutama game 2048. Game 2048 adalah salah satu game yang menarik untuk diteliti mengingat tingkat kesulitan permainan ini akan semakin meningkat disaat nilai angka yang dihasilkan semakin tinggi. Selain itu, permainan ini juga tidak dibatasi oleh waktu melainkan dapat dimainkan terus menerus hingga permainan berakhir. Neural network dan tree adalah 2 arsitektur yang dapat digunakan untuk memainkan 2048 akan tetapi membutuhkan waktu training yang lama jika ingin bermain dengan baik. Lama training tersebut yang pada penelitian ini dioptimasi oleh algoritma evolutionary (3 algoritma yang digunakan pada penelitian ini: Algoritma Genetic, WOA, dan MFO). Dengan adanya optimasi ini maka akan diperoleh bobot terbaik baik pada arsitektur NN ataupun Tree sehingga menghasilkan kecerdasan yang baik dalam memainkan 2048. Setelah melalui berbagai ujicoba maka disimpulkan bahwa kombinasi dengan arsitektur NN lebih baik dibandingkan dengan arsitektur Tree dan algoritma WOA dan MFO berhasil mengoptimasi arsitektur dengan lebih baik dibandingkan algoritma genetic.

Kata kunci: evolving neural network; genetic algorithm; woa; mfo

1. Pendahuluan

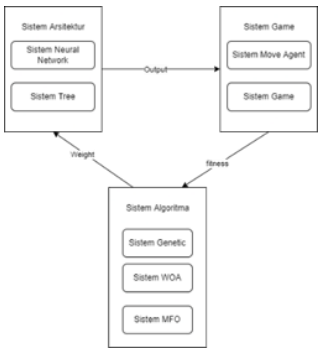
Algoritma genetic sering digunakan sebagai patokan dalam membuat perbandingan kinerja pada algoritma evolutionary lainnya. Hal ini dilakukan karena dari waktu ke waktu algoritma evolutionary semakin bertambah sehingga diperlukan adanya sebuah algoritma yang dapat digunakan sebagai acuan dasar. Pada penelitian ini akan digunakan algoritma Whale Optimation Algorithm (WOA) dan algoritma Moth Flame Optimization (MFO) . Ketiga algoritma tersebut akan di gunakan untuk menyelesaikan game 2048 dan dibandingkan

performanya. Sehingga dapat diketahui algoritma yang paling baik dalam melakukan optimasi sehingga permainan 2048 dapat dimainkan dengan baik dan cepat.

Saat ini permainan 2048 (Dickey, 2014; Ferri-Benedetti, 2014) sudah memiliki berbagai macam kecerdasan buatan menggunakan minimax, expectimax, alpha beta pruning atau berbagai algoritma lain. Tetapi kecerdasan buatan 2048 menggunakan evolutionary neural network masih jarang diteliti dan merupakan topik penelitian yang menarik. Tujuan akhir permainan 2048 adalah untuk mencapai tile dengan nilai 2048, tetapi tentu permainan akan tetap dilanjutkan hingga game over yaitu kondisi di mana sudah tidak ada tile yang dapat digeser atau dimerge. Dimana pada map 4x4 nilai tertinggi yang mungkin untuk didapatkan yaitu 131.072 (Neller, 2015). Permainan ini dapat dikatakan relatif mudah untuk dimainkan karena hanya memiliki 4 aksi yaitu kiri, kanan, atas, atau bawah akan tetapi untuk menyelesaikan permainan adalah hal yang berbeda, tanpa mengetahui strategi bermainnya hampir mustahil untuk menyelesaikan permainan. Berikut adalah beberapa strategi untuk menyelesaikan permainan 2048 (OLSON, 2015) yaitu selalu menaruh tile dengan nilai tertinggi di bagian corner, mengurutkan tile dari yang nilai tertinggi menuju terendah secara bersebelahan, memenuhi sebagian map dengan tile yang memiliki nilai sekitar 32, menjaga row atau kolom di mana tile dengan nilai tertinggi berada selalu padat, dan menyimpan tile yang memungkinkan konsekutif merge.

Penelitian ini akan menyimulasikan permainan 2048 dengan memberikan beberapa atribut tambahan seperti lama waktu bermain, generasi dan populasi ke berapa AI, jenis arsitektur yang digunakan, dan jenis algoritma evolutionary yang digunakan. Terdapat 2 jenis arsitektur yang disediakan dalam penelitian ini yaitu arsitektur Neural Network (NN) dan Tree, dengan pilihan algoritma adalah algoritma genetic, WOA dan MFO. Sedangkan untuk area permainan sendiri terdapat 3 ukuran yaitu 4x4, 5x5 dan 6x6. Pada akhirnya, hasil performa yang akan dibandingkan dan diharapkan dari penelitian ini adalah jumlah generasi, waktu bermain AI dalam menyelesaikan permainan, dan nilai score maksimum yang dihasilkan.

2. Metode



Gambar 1. Arsitektur Kecerdasan Buatan 2048

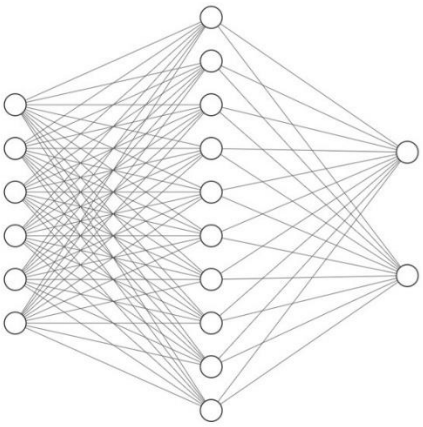
- Pertama kali diusulkan oleh Mirjalilis dan Lewis pada tahun 2016
- Pertama kali diusulkan oleh Mirjalili pada tahun 2015

Gambar 1 adalah arsitektur kecerdasan buatan 2048 yang kami gunakan. Pada arsitektur tersebut dapat dilihat bahwa algoritma evolutionary akan selalu melakukan update bobot dari arsitektur yang digunakan (baik neural network atau tree). Arsitektur yang

dihasilkan digunakan untuk memainkan permainan 2048 hingga selesai dan menghasilkan output berupa nilai fitnes. Nilai fitnes tersebut akan digunakan untuk meningkatkan kinerja dari algoritma evolutionary di iterasi/generasi berikutnya.

2.1. Sistem Neural Network (ENN(D. Fogel, Fogel, & Porto, 1990; Hintz & Spofford, 1990)) pada 2048

Untuk representasi ENN pada 2048 ini akan digunakan kumpulan angka yang berisi bobot hubungan antar neuron pada neural network. Pada penelitian ini hidden layer akan diujicoba terus menerus hingga ditemukan yang terbaik/tepat. Gambar 2 adalah contoh representasi arsitektur neural network untuk 1 hidden layer dengan 10 neuron.



W1	W2	W3	W4	W5		W80
0.154	0.958	0.462	0.356	0.418	...	0.264

Gambar 2. Representasi Neural Network pada Algoritma Evolutionary

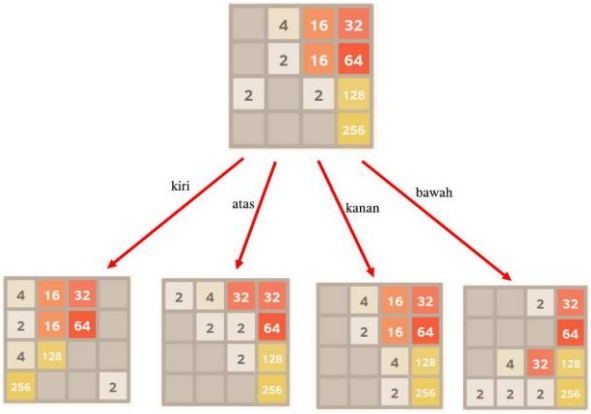
Input layer memiliki 6 neuron yang berasalkan dari:

- Highest value tile: nilai tile yang paling tinggi pada map saat ini, karena tujuan untuk memenangkan permainan adalah mencapai nilai 2048 maka semakin tinggi highest tile maka semakin tinggi juga tingkat dalam memenangkan permainan
- Count of sequence descending value tile: jumlah tile yang nilainya urut dari tertinggi ke terendah
- Is the highest tile in the corner: input layer boolean yang akan mengembalikan 1 atau 0, apakah nilai tile tertinggi berada di corner atau pojok map.
- Highest count merge that could happen: memiliki konsep yang sama dengan input layer 2, input ini adalah jumlah tile bersebelahan yang dapat melakukan merge tetapi harus dimulai dari tile dengan nilai tertinggi.
- Count of small tile: jumlah tile – tile dengan nilai kecil yaitu nilai di bawah 32 (salah satu strategy yang digunakan)
- Is area around highest tile dense: strategi untuk menjaga area sekitar highest tile untuk tetap padat dengan tujuan untuk menjaga highest tile tetap berada di corner dan tidak tergeser.

- Sedangkan output layer adalah 2 neuron yang akan dijadikan biner dengan kombinasi [0,0], [0,1], [1,0] atau [1,1] dimana masing – masing akan merepresentasikan kiri, atas, kanan dan bawah.

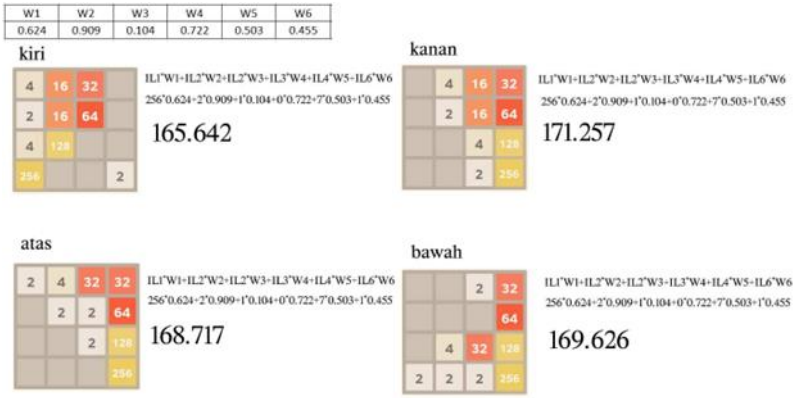
2.2. Sistem Tree(Boumaza, 2009) pada 2048

Arsitektur tree yang digunakan pada penelitian ini karena hanya membutuhkan 4 aksi maka tree akan memiliki 4 child seperti Gambar 3.



Gambar 3. Arsitektur Tree pada 2048

Representasi Tree yang digunakan dalam penelitian ini berupa array angka sepanjang 6 elemen (sesuai dengan jumlah input layer yang digunakan), sedangkan output diperoleh berdasarkan total dari perkalian antara bobot dengan nilai input.

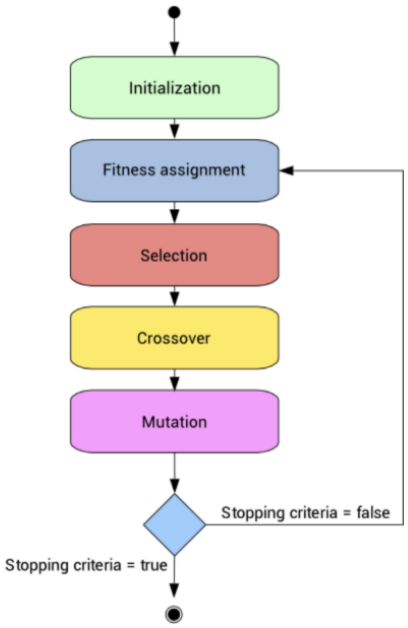


Gambar 4. Representasi Tree pada Algoritma Evolutionary

Gambar 4 merupakan contoh representasi dimana setiap arah akan dihitung nilainya dengan rumus IL1*W1+IL2*W2+IL3*W3+IL4*W4+IL5*W5+IL6*W6 di mana IL adalah input layer dan W adalah weight. Nilai dari 4 arah simulasi ini akan dibandingkan untuk mendapatkan nilai yang paling besar. Berdasarkan gambar tersebut dapat kita lihat bahwa pergerakan ke kanan memiliki nilai tertinggi yaitu 171.257 maka output yang dilakukan adalah melakukan slide ke kanan.

2.3. Algoritma Genetic

Algoritma Genetic paling awal pernah disinggung oleh Alan Turing saat mengutarakan idenya untuk membuat mesin berdasarkan prinsip evolusi(Turing, 1950). Kemudian pada tahun 1960an, cukup banyak paper yang mulai mengimplementasikan metode – metode utama genetic seperti selection, recombination dan mutation, yang kemudian di print ulang oleh Fogel pada tahun 1998(D. B. Fogel, 1998). Algoritma Genetic sendiri dipopularkan oleh John Holland pada tahun 1975, di mana ia mengeluarkan buku berjudul Adaptation in Natural and Artificial Systems.

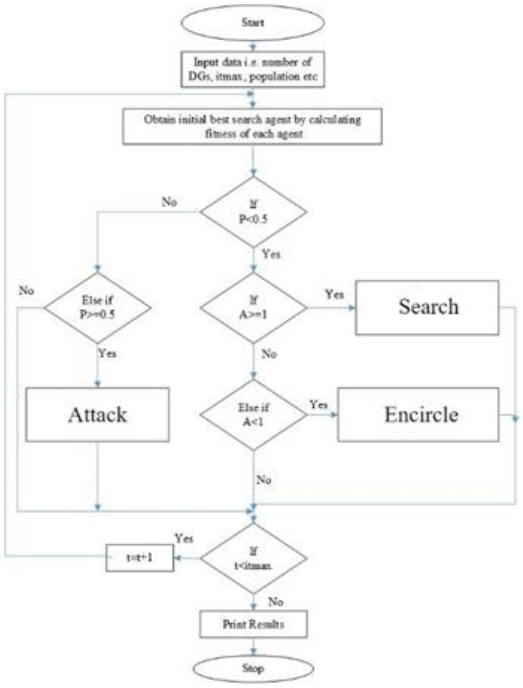


Gambar 5. Algoritma Genetik

Pada algoritma genetic terdapat 3 komponen penting yaitu Selection, Crossover dan MUTATION. Selection adalah metode untuk mengambil top performing individual dalam suatu populasi yang akan di jadikan sebagai parent untuk proses crossover. Crossover adalah metode pembuatan populasi untuk generasi baru dengan menggunakan parent dari proses selection di mana prosesnya adalah dengan menggabungkan gen dari kedua parent untuk membuat individu baru. Mutation adalah metode untuk menyimulasikan mutasi yang dapat terjadi pada perubahan tiap generasi di mana setiap individual baru dari hasil crossover akan memiliki peluang terkena mutasi.

2.4. Algoritma WOA

Algoritma WOA adalah algoritma optimasi yang pertama kali diusulkan oleh Mirjalili dan Andrew Lewis pada tahun 2016(Mirjalili & Lewis, 2016). Inspirasi algoritma ini diperoleh dari Paus Bungkuk yang memiliki ciri khusus dalam mendapatkan mangsanya. Ciri tersebut adalah bubble net spiral (Watkins & Schevill, 1979) dimana Paus akan mengeluarkan gelembung-gelembung udara dari bawah air secara melingkar di area mangsa dan semakin lama area tersebut akan semakin mengecil.



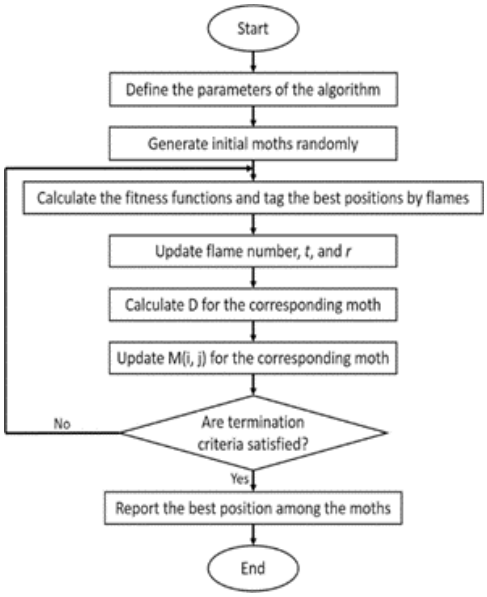
Gambar 6. Algoritma WOA

Pada algoritma WOA terdapat 2 macam kondisi, jika P < 0.5 maka algoritma akan mengitari mangsa dengan 2 jenis metode yaitu metode SEARCH dan metode ENCIRCLE. Akan tetapi jika P >= 0.5, maka algoritma akan masuk ke kondisi bubble-net attack dengan metode ATTACK.

2.5. Algoritma MFO

Algoritma MFO adalah algoritma optimasi yang pertama kali diusulkan oleh Mirjalili pada tahun 2015(Mirjalili, 2015). Inspirasi algoritma ini berasal dari Moth Flame atau ngengat yang memiliki keunikan pada pergerakannya. Keunikan tersebut terjadi saat malam hari, dimana ngengat menggunakan cahaya dari bulan sebagai panduan untuk dapat terbang lurus (Gaston, Bennie, Davies, & Hopkins, 2013). Keunikan ini lah yang menyebabkan pergerakan ngengat saat berada dekat dengan area sumber cahaya. Ngengat akan selalu terbang pada sudut lampu tersebut sehingga pergerakan ngengat akan memutari lampu itu saja dan menghasilkan lingkaran spiral (Frank, 2006).

Pada algoritma MFO, sebuah ngengat adalah individu atau calon solusi dari masalah yang ada. Dimana solusi terbaik dapat diandaikan sebagai cahaya yang diputari oleh ngengat tersebut atau yang disebut sebagai Flame. Algoritma ini cukup sederhana apabila dibandingkan dengan algoritma baru lainnya. Hal ini dikarenakan tidak adanya pengecekan kondisi dan hanya memiliki 1 metode untuk melakukan perubahan nilai yaitu pada tahap update position moth. Di tahapan tersebut, tiap individu pada populasi akan dilakukan perubahan nilai posisi berdasarkan referensi individu lainnya.



Gambar 6. Algoritma WOA

2.6. Fungsi Fitnes Permainan 2048 pada Penelitian ini

Fungsi fitnes adalah nilai untuk mengetahui seberapa tinggi performa dari sebuah individu dalam menyelesaikan masalah. Pada penelitian ini, dikarenakan tujuan dari permainan 2048 adalah mencapai 2048 maka fitness function akan difokuskan untuk mencapai 2048.

$$fitnes = ((\text{Highest Tile} / 2048) + (\text{Score} / \text{High Score})) / 2$$

Rumus fitnes diatas adalah rumus yang digunakan pada penelitian ini, dimana:

- Highest Tile adalah nilai tile tertinggi pada map sekarang
- Score adalah score sekarang
- High score adalah score tertinggi yang pernah dicapai selama permainan berjalan,

Contoh perhitungan: jika highest tile = 1024, score = 11260, dan high score = 12064 maka fitness yang didapat adalah ((1024/2048) + 11260/12064) / 2 = 0.716.

3. Hasil dan Pembahasan

Tabel 1. Hasil Uji Coba Jumlah Generasi

Map Size	Genetic		WOA		MFO	
	NN	Tree	NN	Tree	NN	Tree
4x4	-	-	-	-	-	-
5x5	2	-	3	4	2	4
6x6	1	6	1	2	1	2

Tabel 1 menampilkan jumlah generasi yang dibutuhkan untuk menyelesaikan permainan 2048. Berdasarkan hasil uji coba tersebut:

- Ukuran map 4x4 tidak ada yang berhasil menyelesaikan permainan,
- Ukuran map 5x5 menunjukkan penggunaan NN mendapatkan hasil yang lebih bagus dengan selisih 1 atau 2 generasi dibandingkan Tree. Sedangkan secara algoritma, MFO dan genetic dapat menyelesaikan map 5x5 lebih cepat dibandingkan WOA.
- Ukuran map 6x6, algoritma MFO, WOA dan Genetic dapat menyelesaikan map 6x6 hanya dengan 1 generasi saja. Akan tetapi saat menggunakan arsitektur Tree, hanya algoritma MFO dan WOA menyelesaikan map 6x6 dengan cepat.

Tabel 2. Hasil Uji Coba Waktu Bermain

Map Size	Genetic		WOA		MFO	
	NN	Tree	NN	Tree	NN	Tree
4x4	-	-	-	-	-	-
5x5	41	-	46	48	41	43
6x6	46	65	44	75	45	65

Tabel 2 menampilkan waktu yang dibutuhkan untuk menyelesaikan permainan 2048 dalam satuan detik. Berdasarkan hasil uji coba tersebut:

- Ukuran map 4x4 tidak ada yang berhasil menyelesaikan permainan
- Ukuran map 5x5 menunjukkan penggunaan NN mendapatkan hasil yang lebih cepat dengan 41-46 detik dibandingkan dengan Tree yang membutuhkan 43-48 detik. Dari sisi algoritma, MFO dan Genetic dapat menyelesaikan map 5x5 dengan lebih cepat dibandingkan WOA.
- Ukuran map 6x6 dari sisi arsitektur menunjukan hasil yang serupa dengan ukuran map 5x5 akan tetapi dari sisi algoritma waktu yang digunakan tidak terpaut jauh.

Tabel 3. Hasil Uji Coba Score Tertinggi

Map Size	Gen	Genetic		WOA		MFO	
		NN	Tree	NN	Tree	NN	Tree
4x4	1	5.2k	1.7k	5.2k	1.7k	5.2k	1.7k
	5	5.5k	2.4k	5.4k	2.5k	6.3k	2.7k
	10	5.8k	2.8k	7.0k	3.0k	7.2k	7.1k
5x5	1	19.8k	6.7k	19.8k	6.7k	19.8k	6.7k
	5	21.1k	8.3k	25.5k	13.9k	26.0k	12.9k
	10	33.2k	8.6k	40.5k	16.2k	41.7k	27.9k
6x6	1	103.5k	19.7k	103.5k	19.7k	103.5k	19.7k
	5	186.5k	27.1k	129.0k	94.5k	106.7k	44.1k
	10	188.1k	28.0k	195.8k	112.3k	217.0k	121.5k

Tabel 3 menunjukan score tertinggi yang dapat diperoleh selama uji coba dilakukan. Berdasarkan hasil uji coba tersebut:

- Score tertinggi diperoleh saat algoritma evolutionary digabungkan dengan arsitektur NN. Baik untuk ukuran papan 4x4, 5x5, ataupun 6x6.

- Dari segi algoritma, WOA, dan MFO menghasilkan score yang lebih tinggi dibandingkan algoritma genetik. Sedangkan antara WOA dan MFO sendiri, performa kedua algoritma tidak jauh berbeda.

4. Simpulan

Berdasarkan ujicoba dan observasi performa yang dilakukan selama penelitian ini, maka dapat disimpulkan bahwa: Pada ukuran map 5x5, metode optimasi tercepat dalam menyelesaikan permainan 2048 didapatkan oleh algoritma MFO dan genetic yang dikombinasikan dengan arsitektur NN. Pada ukuran map 6x6, kombinasi dengan arsitektur NN dapat menyelesaikan permainan 2048 dengan lebih cepat dibandingkan arsitektur Tree. Pada semua ukuran, kombinasi dengan arsitektur NN dapat menghasilkan nilai score tertinggi dibandungkan arsitektur Tree. Optimasi dengan algoritma WOA dan MFO menghasilkan score yang lebih tinggi dibandingkan algoritma genetik.

Daftar Rujukan

Boumaza, A. (2009). On the evolution of artificial Tetris players. The IEEE Symposium on Computational Boumaza, A. (2009, September). On the evolution of artificial Tetris players. In *2009 IEEE Symposium on Computational Intelligence and Games* (pp. 387-393). IEEE.

Dickey, M. R. (2014). Puzzle Game 2048 Will Make You Forget Flappy Bird Ever Existed. Business Insider. Retrieved, 27

Ferri-Benedetti, F. (2014). The creator of 2048 tells us about the secret behind the game's success. Retrieved from https://en.softonic.com/articles/interview-with-the-creator-of-2048.

Fogel, D. B. (1998). Evolutionary Computation. The Fossil Record. Selected Readings on the History of Evolutionary Computation. *Classifier Systems*.

Fogel, D. B., Fogel, L. J., & Porto, V. W. (1990). Evolving neural networks. *Biological cybernetics*, *63*(6), 487-493.

Frank, K. D., Rich, C., & Longcore, T. (2006). Effects of artificial night lighting on moths. *Ecological consequences of artificial night lighting*, *13*, 305-344.

Gaston, K. J., Bennie, J., Davies, T. W., & Hopkins, J. (2013). The ecological impacts of nighttime light pollution: a mechanistic appraisal. *Biological reviews*, *88*(4), 912-927. https://doi.org/10.1111/brv.12036

Hintz, K. J., & Spofford, J. J. (1990, September). Evolving a neural network. In *Proceedings. 5th IEEE International Symposium on Intelligent Control 1990* (pp. 479-484). IEEE. https://doi.org/10.1109/IS-IC.1990.128500

Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, *89*, 228-249.https://doi.org/10.1016/j.knosys.2015.07.006

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in engineering software*, *95*, 51-67. /https://doi.org/10.1016/j.advengsoft.2016.01.008

Neller, T. W. (2015). Pedagogical possibilities for the 2048 puzzle game. *Journal of Computing Sciences in Colleges*, *30*(3).

Olson, R. (2014). Artificial Intelligence has crushed all human records in 2048. Here's how the AI pulled it off.

Turing, A. M. (2009). *Computing machinery and intelligence* (pp. 23-65). Springer Netherlands.

Watkins, W. A., & Schevill, W. E. (1979). Aerial observation of feeding behavior in four baleen whales: Eubalaena glacialis, Balaenoptera borealis, Megaptera novaeangliae, and Balaenoptera physalus. *Journal of Mammalogy*, *60*(1), 155-163. https://doi.org/10.2307/1379766.