Dynamics Difficulty Adjusment Metode Evolutionary MCTS with Flexible Search Horizon pada Multi-Action Adversarial Games untuk Penyesuaian Tingkat Permainan

Andhika Evantia Irawan, Liliana, Hans Juwiantho
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: andhikaevantia@gmail.com, lilian@petra.ac.id, Hans.juwiantho@petra.ac.id

ABSTRAK

Dynamic Difficulty Adjustment (DDA) merupakan metode yang memodifikasi perilaku AI agar menyesuaikan kemampuan pemain. Sejauh ini penelitian mengenai DDA pada *Monte Carlo Tree Search* mampu memberikan tingkat tantangan yang sesuai. Namun keunggulan MCTS dalam mencari solusi pada strategi jangka panjang belum diimplentasikan dengan maksimal karena sejauh ini hanya digunakan pada jenis permainan 2D *real-time fighting* yang merupakan salah satu jenis game strategi jangka pendek.

Penelitian ini menggabungkan DDA dengan *evolutionary monte carlo tree search with flexible horizon* (FH-EMCTS). FH-EMCTS merupakan penggabungan vanilla MCTS dengan *Evolutionary algorithm.* Metode ini meningkatkan ruang pencarian menjadi lebih panjang dalam batasan tertentu. Pemberian DDA pada FH-EMCTS dilakukan dengan merubah cara pemilihan tindakan dan penilaian pada tiap *node.*

Hasil dari penelitian ini adalah agen AI yang menggunakan FH-EMCTS dengan DDA dapat diimplementasikan kedalam *multi-action adversarial game* dan dapat memberikan tingkat kesulitan yang seimbang terhadap agen AI lainnya maupun manusia. Berdasarkan hasil survei percobaan agen AI melawan manusia menunjukkan bahwa agen AI yang paling menyenangkan dan realistis bukanlah agen AI yang memiliki kemampuan yang paling baik dalam hal persentase kemenangan tetapi agen AI yang memiliki tingkat kemenangan sekitar 50%.

Kata Kunci: *Dynamics Difficulty Adjustment, Multi-Action Adversarial games, Penyesuaian tingkat permainan, MCTS with Flexible Search Horizon*

ABSTRACT

Dynamic Difficulty Adjustment (DDA) is a method that modifies AI behavior to suit the player's abilities. So far, research on DDA in Monte Carlo Tree Search has been able to provide an appropriate level of challenge. However, the advantages of MCTS in finding solutions to long-term strategies have not been maximally implemented because so far it is only used in 2D real-time fighting games, which are short-term strategy game.

This study combines DDA with evolutionary monte carlo tree search with flexible horizon (FH-EMCTS). FH-EMCTS is combination of vanilla MCTS with an Evolutionary algorithm. This method increases the length of the search space to certain extent. Giving DDA to FH-EMCTS is done by changing the way of selecting actions and assessing each node.

The result of this research is that AI agents that use FH-EMCTS with DDA can be implemented into multi-action adversarial game and can provide balanced level of difficulty to other AI agents and humans. Based on the results of survey of AI agents against humans, it shows that the most fun and realistic AI agents are not the AI agents who have the best ability of winning percentage but AI agents who have win rate of around 50%.

Keywords: *Dynamics Difficulty Adjustment, Multi-Action Adversarial games, Game level adjustment, MCTS with Flexible Search Horizon*

1. PENDAHULUAN

Membuat *Artificial Intelligence* menantang dan seru adalah salah satu bagian penting dalam pembuatan *game.* Untuk memastikan pemain tetap imersif dalam sebuah *game,* tingkat tantangan yang diberikan kepada pemain perlu disesuaikan berdasarkan kemampuan pemain saat itu [11]. Jika sebuah *game* terlalu mudah atau terlalu susah, pemain bisa saja frustasi dan meninggalkan permainan tersebut [8].

S. Demediuk et al menjelaskan pada awalnya untuk tingkatan tantangan diatur dengan cara memilih tingkat kesulitan yang diinginkan oleh pemain pada *game* tersebut dimulai dari "easy","medium","hard" atau bisa juga dengan peningkatan tantangan berdasarkan *progress level* dari pemain saat itu [13]. Namun hal ini membatasi *game developer* untuk mengembangkan variasi tantangan yang lainnya. Permasalahan lainnya dengan menggunakan sistem tersebut adalah *game developer* harus memiliki asumsi yang tepat dalam menentukan perbedaan pada tiap tingkatan tantangan [12]. Jika asumsi yang dibuat tidak tepat akan membuat kepuasan pemain berkurang.

Untuk mengatasi masalah itu akhirnya diperkenalkanlah *Dynamic Difficulty Adjustment* oleh Robin Hunicke dan Vernell Chapman pada tahun 2004 [17]. *Dynamic Difficulty Adjustment adalah* metode yang memungkinkan tingkat tantangan yang diberikan permainan kepada pemain berada pada tingkat yang tepat bagi mereka. Penelitian yang dilakukan oleh S. Demediuk.et al menggunakan DDA kedalam vanilla Monte Carlo Tree Search [12, 13]. Penelitian ini menghasilkan 5 jenis agent AI dimana 2 berdasarkan atas strategi pemilihan tindakan (*Reactive Outcome Sensitive Action Selection* (ROSAS) / *Proactive Outcome Sensitive Action Selection* (POSAS) dan 3 berdasarkan strategi evaluasi bermain heuristik (*True* ROSAS/POSAS/*Adaptive True* ROSAS) serta dapat bekerja dengan baik untuk game bergenre 2D *real-time fighting.*

Hingga tahun 2019, *Dynamic Difficulty Adjustment* masih menggunakan *vanilla MCTS*. Namun akhir-akhir ini MCTS berkembang pesat dikarenakan kesuksesan dalam berbagai *game* yang kompleks. Salah satu penelitian dari perkembangan MCTS ini adalah penelitian yang dilakukan oleh Baier dan Cowling yang menggabungkan vanilla MCTS dengan *Evolutionary algorithm* serta peningkatan pada ruang pencarian yang lebih panjang dalam batasan tertentu [7]. Metode ini menghasilkan pencarian yang lebih

efektif dan lebih dalam dan cocok untuk digunakan sebagai agen AI pada *multi-action adversarial games*.

Oleh karena itu penelitian ini menggabungkan DDA MCTS dengan algoritma *Evolutionary MCTS with Flexible Search Horizon*. Penelitian ini dilakukan pada salah satu jenis permainan strategi jangka panjang (*multi-action adversarial*) yaitu *Hero AIcademy*. Penelitian ini menggabungkan DDA kedalam algoritma FH-EMCTS. Metode yang digunakan dalam penelitian ini juga akan diukur dalam kualitatif mengenai pengaruh DDA MCTS dalam AI terhadap pemain.

2. TINJAUAN STUDI

Penelitian yang dilakukan oleh Demediuk et al. pada tahun 2017 dan 2019, menggunakan *vanilla MCTS* sebagai dasar dari DDA pada agen AI dalam permainan *fighting*[12, 13]. Permainan bergenre *fighting* sendiri hanya membutuhkan strategi jangka pendek. Hal ini menyebabkan performa dari MCTS yang memiliki kemampuan metode yang efektif dalam mengembangkan agen AI dalam sebuah permainan yang membutuhkan strategi jangka panjang dan ruang pencarian yang lebih luas kurang dapat termanfaatkan dalam penelitian tersebut.

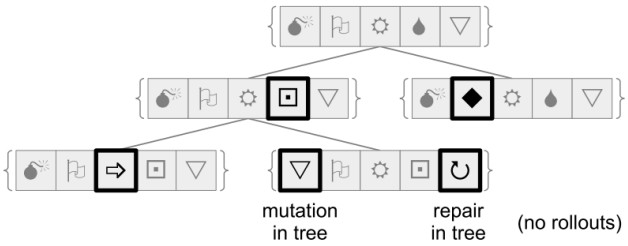
Penelitian yang dilakukan oleh Baier dan Cowling pada tahun 2018 berhasil membuat *variant MCTS* yang baru yakni *Evolutionary MCTS with Flexible Search Horizon* yang telah disesuaikan dan dapat digunakan dalam permainan *multi-action adversarial* [6, 7]. Namun demikian hingga saat ini masih belum ada penelitian yang menggunakan DDA terhadap jenis *variant MCTS* yang dibuat oleh Baier dan Cowling ini. Sehingga dari penelitian-penelitian yang sudah ada sebelumnya, diperlukan penelitian untuk menggabungkan DDA Demediuk et.al dengan FH-EMCTS kedalam permainan strategi jangka panjang.

3. METODOLOGI PENELITIAN

3.1 *Evolutionary* MCTS

Evolutionary MCTS (EMCTS) adalah pendekatan menggunakan MCTS untuk memainkan multi-action turn based adversarial games. EMCTS menggabungkan tree search dari MCTS dengan pendekatan genome dari evolutionary algorithms [6]. Evolutionary algorithms sendiri merupakan pengoptimalan algoritma yang terinspirasi oleh seleksi alam yang telah digunakan secara luas untuk mengembangkan dan melatih agen AI untuk berbagai macam permainan [19, 20].

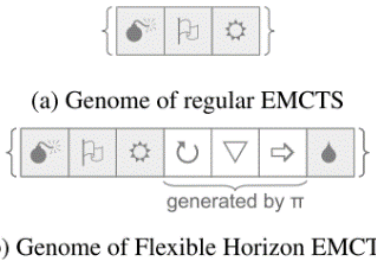
Tidak seperti *vanilla MCTS* yakni membuat *root* dari *empty action*, EMCTS membangun root seperti pada gambar 1 yang terdiri dari *complete sequence* dari tindakan-tindakan yang dapat dilakukan seperti layaknya *genomes* [7]. Pada pemilihan *genomes* nya sendiri dipilih dari populasi *genomes* yang ada dengan menggunakan *greedy action* untuk mencari yang terbaik untuk dijadikan sebuah *root*. Perbedaan selanjutnya dari *vanilla MCTS* terletak pada pembuatan *nodes* dan *edges* baru, dimana pembuatan *nodes* baru dilakukan dengan cara memutasi (menggantinya dengan tindakan yang lain yang dapat dilakukan) salah satu tindakan dari *complete sequence* dari *parent nodes* dan posisi dari tindakan yang dimutasi direpresentasikan sebagai sebuah *edges*. EMCTS juga tidak menggunakan *simulation* layaknya *vanilla MCTS*, melainkan menggunakan *evaluates* pada *leaf nodes* tersebut secara langsung.



Gambar 1 Struktur *Tree* dari *Evolutionary* MCTS. *Nodes* merepresentasikan dari kumpulan tindakan (*genomes*), *Edges* merepresentasikan satu tindakan yang dimutasi. *Repair* terjadi ketika mutasi menghasilkan kumpulan tindakan yang tidak bisa dijalankan pada *turn* tersebut[6]**.**

3.2 E-MCTS *with Flexible Search Horizon*

Evolutionry MCTS with Flexible Search Horizon (FH-EMCTS) adalah peingkatan dari EMCTS dimana pada FH-EMCTS memungkinkan untuk memperluan pencarian EMCTS ke kedalaman yang diinginkan di luar state saat ini. Terdapat 2 perbedaan utama antara EMCTS dengan FH-EMCTS yakni pada FH-EMCTS mengijinkan untuk pencarian yang lebih dalam dan terdiri dari model perilaku yang lebih cepat yang membuat deep searching menjadi lebih efektif dan lebih singkat[7]. *Genomes* dalam EMCTS mengkodekan hanya tindakan dari pergantian saat ini yang sedang dicari, sedangkan panjang *genomes* / pencarian h dalam FH-EMCTS adalah parameter yang dapat diatur ke nilai yang lebih tinggi seperti pada gambar 2. π menggunakan Greedy Action AI. Pada FH-EMCTS untuk mengurangi waktu pencarian dan untuk meningkatkan efektivitas, digunakan pembatasan untuk setiap pilihan dari Greedy Action AI menjadi 10 aksi yang menjanjikan dengan cara diurutkan dari yang paling tinggi serta ditambahkan lazy generation dari mutation.



Gambar 2 Contoh genomes dari regular EMCTS dan FH-EMCTS. Dimana Action points = 3, untuk search horizon yang dipilih untuk FH-EMCTS adalah 7 [7]

3.3 *Dynamic Difficulty Adjustment*

Alexander et al. melakukan penelitian untuk menemukan tingkat kesulitan yang paling menyenangkan untuk semua jenis pemain dengan cara mengatur tingkat kesulitan yang berbeda secara berurutan [2]. Hasil penelitian tersebut menunjukkan pemain kasual lebih menyukai permainan yang memiliki tingkat kesulitan yang lebih rendah atau tidak sesuai dengan kemampuan mereka. Namun pemain yang berpengalaman lebih memilih tingkat kesulitan yang sesuai dengan kemampuan mereka atau lebih tinggi [10].

Dynamic Difficulty Adjustment (DDA) adalah sistem yang dapat membantu menangani preferensi ini, sebagai mereka menyesuaikan kurva kesulitan untuk setiap jenis pemain [5][16] dan dalam beberapa kasus, menurut pengaruhnya [1]. Dari

penelitian Colwell et al. menunjukkan bahwa baik pemain yang tidak terampil maupun terampil menyukai permainan ketika permainan tersebut dapat beradaptasi sesuai dengan performa mereka [9]. Ang & Mitchell melakukan penelitian terakit dampak dari DDA pada *nine dimensions of the flow experience*. Hasil menunjukkan bahwa sistem DDA memberikan dampak yang positif terhadap pengalaman bermain dari pemain, dimana DDA dapat mengijinkan pemain untuk "*enter a state of flow much faster and for a longer periods of time"*[4].

Tingkatan kesulitan dari DDA dapat berganti secara real time bergantung pada kondisi atau profile dari pemain[14]. Tujuan dari DDA adalah menjaga keseimbangan dari tingkatan kesulitan didalam permainan sesuai dengan kemampuan pemain[3]. Teknik DDA dapat menggunakan fungsi penilaian untuk menentukan tingkat kesulitan. Fungsi ini disebut *Challenge Function*. Namun untuk game tertentu harus diatur oleh parameter yang relevan agar dapat mempengaruhi kinerja pemain.

3.4 DDA pada MCTS

MCTS pernah digunakan untuk membuat *DDA AI agents* sebelumnya oleh Hao et.al, namun pendekatan yang mereka lakukan memiliki beberapa batasan hal ini disebabkan oleh rumus yang digunakan tidak bergantung pada *starting state* sehingga tidak dapat membedakan mana yang menguntungkan mana yang tidak [15]. Hingga akhirnya Demediuk et. Al melakukan pendakatan MCTS dengan menggunakan *Upper Confidence Bound 1 applied to Tress* (UCT) untuk *DDA AI Agents*. Teknik DDA ini ditambahkan oleh agen untuk menghitung nilai dari tiap tindakan dalam kondisi tertentu untuk menghasilkan tingkat tantangan yang sesuai untuk pemain. Terdapat beberapa jenis *AI Agent* yang dihasilkan dari metode ini yakni [12, 13]:

1. *Reactive Outcome Sensitive Action Selection*

   ROSAS membangun search tree menggunakan MCTS, dimana permainan dievaluasi dengan tujuan untuk mempromosikan tindakan yang kuat(sama dengan MCTS Normal). Rumus (1) [12, 13] untuk memilih tindakan:

   $$action = arg\max_{a} - |r[a].score| \quad (1)$$

   Dimana *r* adalah *root node* dari *tree* dan *r[a]* adalah *child* dari *r* sesuai dengan tindakan *a.*

2. *Proactive Outcome Sensitive Action Selection*

   POSAS mengambil pendekatan yang berbeda untuk pemilihan akhir, semua tindakan dalam interval yang ditentukan sekitar nol semuanya dianggap sama berharganya dan satu dipilih secara acak. Rumus (2) [12, 13] untuk memilih tindakan:

   $$action = arg\max_{a} - ([r[a].score] - I_h)^+ \quad (2)$$

   Dimana *r* adalah *root node* dari *tree*, *r[a]* adalah *child* dari *r* sesuai dengan tindakan *a*, I_h adalah setengah ukuran interval sekitar nol dimana semua tindakan dianggap sama-sama berharga, dan $(.)^+$ menunjukkan *ramp function* (fungsi berperilaku seperti fungsi identitas untuk angka positif dan mengembalikan 0 untuk angka negatif).

3. *True-ROSAS and True-POSAS*

   Kekuatan utama MCTS adalah mengeksplorasi pencarian tree asymmetrically. Meskipun ini berguna untuk membuat keputusan yang paling tepat, ROSAS dan POSAS tidak dapat mengambil keuntungan dari ini; pada kenyataannya, yang terjadi adalah sebaliknya: wilayah tree yang paling mereka minati, yang dengan hasil seimbang, kurang dieksplorasi. Hal ini disebabkan oleh strategi evaluasi bermain, yang lebih menghargai hasil yang menguntungkan, mengarah pada eksplorasi yang lebih dalam di daerah-daerah tersebut. Untuk mengatasi batasan ini, kami mengusulkan dua DDA tambahan Agen AI, *True ROSAS* dan *True POSAS*. Kedua agen ini mengubah strategi heuristik MCTS sehingga pohon diperluas dengan cara yang selaras dengan tujuan masing-masing. Rumus *True ROSAS* (3) dan Rumus *True POSAS* (4) [12, 13]:

   $$node.score = -|h_s| \quad (3)$$

   $$node.score = -(|h_s| - I_h)^+ \quad (4)$$

   Dimana h_s adalah strategi heuristik yang digunakan, dimana pada *2D real-time fighting game* menggunakan perbedaan poin kesehatan. Ketika *tree* sepenuhnya terbuat, *True-ROSAS* akan memilih tindakan dengan kunjungan terbanyak (atau yang dengan skor tertinggi). Ini memastikan perluasa pohon sejalan dengan kebijakan pemilihan tindakan algoritma ROSAS, memastikan area yang paling relevan dieksplorasi lebih dalam.

4. *Adaptive True-ROSAS*

   AI Agent AT-ROSAS memanfaatkan kekuatan asymmetric search tree dengan mengubah area yang dieksplorasi lebih banyak, dengan mengubah metrik penilaian dalam evaluasi node MCTS. Dimana pada penelitian yang bersangkutan, peneliti menggunakan variabel perbedaan kesehatan sebelum tindakan dan setelah tindakan dilakukan dan memasukkan ini ke dalam skor simpul sesuai dengan Rumus (5) [12, 13]:

   $$node.score = \begin{cases} \frac{y-x}{y} \; if \; 0 \leq HP\,diff \leq |orig.HP\,diff.| \\ \frac{y+x}{y} \; if |orig.HP\,diff.| \leq HP\,diff < 0, \\ 0 \; otherwise \end{cases} \quad (5)$$

   di mana x adalah perbedaan HP setelah tindakan dan y adalah perbedaan HP sebelum tindakan.

3.5 *Hero AIcademy*

Hero AIcademy adalah game kloningan java pada gambar 3, milik Justesen Niels dari sebuah permainan bernama *Hero Academy*. Permainan ini merupakan permainan *turn-based strategy game* antar 2 orang dimana permainan ini sendiri tidak berlangsung secara *real-time*. Permainan dari *Hero Academy* sendiri terinspirasi dari catur, dimana pada pertandingannya sendiri mirip seperti *Heroes of Might & Magic series*(Justesen et al.,2017). Dalam permainan ini, setiap pemain memiliki *deck* yang terdiri dari *combat units* dan *spells* yang dapat mereka keluarkan dalam papan permainan yang menggunakan 9 x 5 petak.

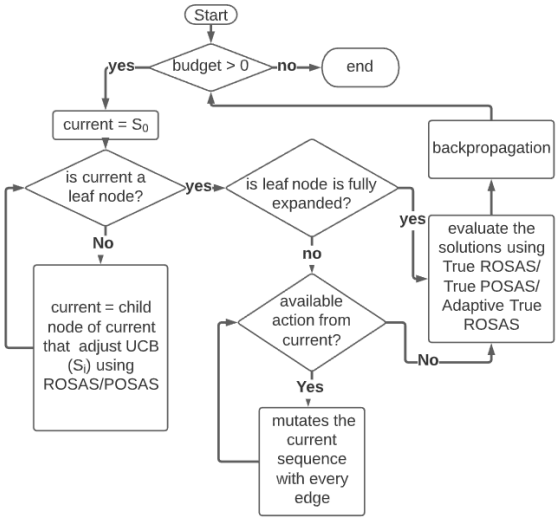
Tujuan dari permainan ini adalah menghancurkan *crystal* miliki lawan atau ketika salah satu dari pemain kehabisan *units* untuk dimainkan. Tiap *turn* pemain akan diberikan 6 *item/unit* yang keluar secara *random* antara *pawn* atau *spell* yang dapat digunakan oleh pemain untuk menyerang pihak lawan dalam *turn* tersebut. Tiap *turn* pemain memiliki 5 *action points* (AP), dimana tiap tindakan yang dilakukan mulai dari menaruh unit, menyerang pihak lawan, mengaktifkan *spell* membutuh 1 AP. Tiap 1 *turn* permainan terdiri dari rata-rata 30 hingga 60 *action* yang ada dalam 1 *game state*, sehingga perkiraan dari *branching factor* dalam 1 *turn* berkisar antara $30^5 \approx 2.4 \times 10^6$ hingga $60^5 \approx 7.8 \times 10^8$ [6].

Gambar 3. Game *Hero AIcademy*. [18]

3.6 DDA metode FH-EMCTS

Pada penelitian ini, *EMCTS with Flexible Search Horzion* menggunakan *search horizon (h)* berjumlah 5 ketika *action point* 5, berjumlah 2 ketika *action point* 10, dan berjumlah 0 ketika *action point* 15. Namun pada penelitian sebelumnya yakni Baier dan Cowling tidak dijelaskan dari h tersebut berapa banyak digunakan untuk *opponent action* dan berapa banyak digunakan untuk *action* yang digunakan untuk menanggapi *opponent action*. Sehingga pada penelitian ini menggunakan jumlah h-1 sebagai *action* yang digunakan untuk menanggapi *opponent action* dan sisanya sebagai *opponent action.*

Untuk penggunaan *Dynamics Difficulty Adjustment* pada *EMCTS with Flexible Search Horizon*, maka dilakukan penambahan 5 jenis DDA yakni *Reactive Outcome Sensitive Action Selection* (ROSAS), *Proactive Outcome Sensitive Action Selection* (POSAS), *True*-ROSAS, *True-POSAS, Adaptive True* ROSAS kedalam *vanilla* E-MCTS dengan *Flexible Search Horizon.* Perbedaan dari DDA MCTS yang dilakukan oleh Demediuk et al, dengan penelitian ini terletak pada parameter dalam proses *adjustmentnya*, dimana parameter yg digunakan dari 5 jenis DDA ini adalah perbedaan total *health* dari *tower* tiap pemain, sedangkan pada DDA *EMCTS with flexible search horizon* menggunakan *heuristic function* yang digunakan untuk mengetahui nilai dari setiap *state* pada *multi-action adversarial games.* Untuk ROSAS dan POSAS akan menentukan *action-action* yang dapat dipilih atau tersedia untuk nantinya akan dimutasi, sedangkan untuk *True*-ROSAS, *True-POSAS, Adaptive True* ROSAS akan menjadi tambahan dalam mengevaluasi tiap solusi yang ada setelah dimutasi.



Gambar 4. *Flowchart* agen AI DDA FH-EMCTS

4. PENGUJIAN DAN DISKUSI

Untuk permainan yang digunakan untuk menguji AI sendiri menggunakan game Hero AIcademy sebagai penguji dari AI. Hero AIcademy sendiri merupakan cloning dari sebuah game dengan genre multi-action adversarial games bernama Hero Academy. Untuk mengetahui apakah penggabungan DDA kedalam FH-EMCTS dapat dimanfaatkan secara maksimal pengujian dilakukan pada action point standard yakni 5 action point dan budget yang diberikan untuk komputasi juga menggunakan standard pada pengujian-pengujian sebelumnya yakni 1 detik, hal ini didasarkan pada penelitian sebelumnya yang dilakukan oleh Baier dan Cowling ketika melakukan pengujian terhadap EMCTS with flexible search horizon.

4.1 Desain AI Lawan

Dalam penelitian ini, digunakan 4 jenis *AI* yang dapat digunakan dalam permainan *multi-action adversarial games* khususnya pada *game hero AIcademy,* yang nantinya digunakan sebagai lawan dari agen *AI* yang menggunakan DDA metode E-MCTS dengan *Flexible Search Horizon* yakni:

1. *Greedy Action,* dijadikan sebagai salah satu AI penguji dikarenakan *Greedy Action* sendiri menggunakan konsep dari *greedy algorithm,* dimana algoritma ini mencari hasil yang terbaik dari semua kemungkinan yang dapat dilakukan pada saat itu.
2. *Non-exploring MCTS,* dijadikan sebagai salah satu AI penguji dikarenakan *Non-exploring MCTS* merupakan jenis *variant MCTS* yang sangat mirip dengan *vanilla MCTS* tanpa adanya perubahan yang signifikan, sehingga dapat dikatakan seperti *vanilla MCTS* sehingga *non-exploring MCTS* digunakan sebagai pengujian ketika agen AI yang dibuat ketika melawan dengan *vanilla MCTS.*
3. *Greedy Online Evolutionary Planning,* dijadikan sebagai salah satu AI penguji dikarenakan *GOEP* adalah salah satu *variant* dari *evolutionary algorithm* yang dapat digunakan dalam permainan *multi-action adversarial games* serta merupakan salah satu teori utama yang digunakan dalam membentuk *EMCTS with flexible search horizon*.
4. *Vanilla Flexible-Horizon Evolutionary* MCTS, dijadikan sebagai salah satu AI penguji untuk mengetahui perbandingan agen AI ketika tanpa DDA dan ketika menggunakan implementasi DDA MCTS yang dibuat oleh Demediuk et al.

4.2 Desain Survei Agen AI

Dalam survei ini akan dilakukan terhadap 15 peserta yang akan melawan 1 *agent AI FH-EMCTS* dan 5 *agent AI* FH-EMCTS dengan DDA yang berbeda. Setiap peserta memiliki kemampuan dan pemahaman yang berbeda-beda mengenai *game multi-action adversarial games* yang diujikan, terdiri dari yang tidak paham dan tidak mengetahui sama sekali hingga memiliki pehaman dan pengetahuan yang tinggi dalam *genre game* dalam penelitian ini. Sebelum melakukan pengujian setiap peserta akan mengisi survei terkait jenis kelamin, serta tingkat kemampuan dan pemahaman pemain dalam *game* yang akan dimainkan. Dalam pengujian ini setiap peserta akan melawan setiap *agent AI* sebanyak 3 kali. Jumlah pengujian sebanyak 3 kali didasarkan atas penelitian sebelumnya yang dilakukan oleh Demediuk et al. Kemudian setelah melakukan pertandingan maka setiap peserta akan diajukan beberapa pertanyaan berikut:

1. Dari skala 1 – 5 seberapa sulit lawan yang dihadapi? (1 menunjukkan sangat mudah dan 5 menunjukkan sangat susah):

Pertanyaan ini diberikan untuk mengetahui seberapa sulit AI yang sedang dilawan oleh pemain. Sehingga nilai yang ideal dalam tingkat tantangan AI yang diberi DDA seharusnya berapada pada nilai 3, yang mengindikasikan bahwa agen AI tidak terlalu mudah dan tidak terlalu susah. Pada penelitian ini, peneliti berekspektasi bahwa *vanilla FH-EMCTS* akan memiliki nilai yang paling tinggi dibanding dengan agen AI DDA.

2. Dari skala 1 – 5 seberapa *enjoy* ketika berhadapan dengan lawan tersebut? (1 menunjukkan sangat tidak *enjoy* dan 5 menunjukkan sangat *enjoy*): Pertanyaan ini diberikan untuk mengetahui seberapa *enjoy* pemain dalam melawan agen AI yang sedang dihadapi. Hal ini digunakan untuk mengetahui dan memahami seberapa jauh tingkat *enjoy* dari tiap agen AI. Pada penelitian ini, peneliti berekspektasi bahwa agen AI DDA menjadi agen AI yang paling *enjoy* ketika menjadi musuh dalam sebuah permainan.
3. Dari skala 1 – 5 seberapa *realistic* lawan yang sedang dihadapi? (1 menunjukkan tidak realistis dan 5 menunjukkan sangat realistis): Pertanyaan ini diberikan untuk mengetahui seberapa realistis agen AI yang dilawan. Hal ini digunakan untuk mencari tau apakah jika agen AI sangat tidak realistis dalam artian sangat *artificial* maka akan menurunkan *enjoyment* dari pemain yang memainkannya.

4.3 Pengujian terhadap Agen AI

Tabel 1 menunjukkan aggregat kemenangan dari vanilla FH-EMCTS, ROSAS, POSAS, *True* ROSAS, *True* POSAS, *Adaptive True* ROSAS ketika melawan tiap agen AI yang diujikan dalam penelitian ini pada setiap Action Point yang diberikan dalam penelitian ini. Dari tabel 1 menunjukkan bahwa agen AI FH-EMCTS yang diberi DDA dapat memberikan *adjustment* sekitar 50% dari kemenangan *vanilla* FH-EMCTS. Tetapi seperti pada penelitian sebelumnya, pengujian terhadap agen AI lainnya pada penelitian ini tidak ada satupun jenis DDA kepada *variant* MCTS untuk *game multi-action adversarial games* yang bisa memberikan tingkat kemenangan yang tepat 50% sesuai dengan saran-saran yang telah disebutkan pada banyak penelitian sebelumnya terkait tingkat kesulitan yang tepat. Hal ini dikarenakan dari sifat algoritma DDA itu sendiri dimana memiliki kecenderungan untuk merespon dari tindakan pemain daripada mencari kemungkinan untuk menang, hal ini yang menyebabkan jenis-jenis algoritma DDA yang ada menjadi lebih banyak kalah dibandingkan menang.

Tabel 1. *Agregate Winrate* tiap *Agent AI*

	Winrate AP 5	*Winrate* AP 10	*Winrate* AP 15
FH-EMCTS	64%	54%	62%
ROSAS	35%	28%	38%
POSAS	38%	28%	30%
True ROSAS	32%	25%	32%
True POSAS	33%	24%	23%
Adaptive True ROSAS	37%	28%	28%

4.4 Pengujian terhadap manusia

Pengujian terhadap manusia ini dilakukan dengan melakukan pengujian terhadap 15 orang dimana tiap orang akan bertanding dengan 5 jenis *agent AI* dengan DDA dan 1 *agent AI* FH-EMCTS sebanyak 3 kali. *Agent AI* FH-EMCTS diambil untuk dapat membandingkan hasil *winrate* dari *vanilla* FH-EMCTS dengan jenis *agent AI* FH-EMCTS yang sudah diberikan *Dynamics Difficulty Adjustment*. Kemudian setelah mereka melakukan pertandingan akan diberikan kuisoner yang perlu diisi untuk mengetahui mengenai tingkat kesulitan, *enjoyment,* dan *realistic* dari tiap *agent AI* yang telah dibuat.

4.4.1 Hasil Survei Awal

Hasil survei awal merupakan hasil pengisian survei sebelum melakukan pertandingan dengan tujuan untuk mengetahui *familiarity* terhadap *game* yang akan dimainkan dan *skill* terhadap *game* dengan *genre* tersebut. Hasil dari survei untuk setiap peserta dapat dilihat pada tabel 2 dibawah ini.

Tabel 2. Hasil survei awal

Jenis Kelamin	Jumlah	*Familiarity*	Jumlah	*Skill*	Jumlah
Laki-Laki	13	5 (Sangat Familiar)	1	5 (Sangat Bagus)	2
Perempuan	2	4	5	4	5
		3	7	3	7
		2	2	2	2
		1 (Kurang Familiar)	0	1 (Kurang Bagus)	0

4.4.2 Hasil winrate pertandingan

Hasil *winrate* merupakan hasil pertandingan dari 15 peserta yang telah mengikuti pertandingan dimana setiap peserta melawan 1 jenis *agent AI* sebanyak 3 kali sehingga total dari pertandingan untuk setiap *Agent AI* berjumlah 45 pertandingan. Hasil dari *winrate* untuk setiap *Agent* AI dapat dilihat pada tabel dibawah ini dimana semakin tinggi persentase menunjukkan bahwa kemenangan dari *agent AI* tersebut tinggi.

Tabel 3. *Agregate Winrate* tiap *Agent AI* dengan jumlah *Action Point* 5

	Winrate
FH-EMCTS	86.67%
ROSAS	33.33%
POSAS	28.89%
True ROSAS	42.22%
True POSAS	35.56%
Adaptive True ROSAS	48.89%

Dari tabel 3 menunjukkan bahwa penggunaan DDA pada FH-EMCTS yang digunakan dalam *game multi-action adversarial* dapat memberikan *adjustment* yang sama seperti pada pengujian terhadap agen AI dimana dapat memberikan *adjustment* sekitar 50% dari kemenangan *vanilla* FH-EMCTS.

4.4.3 Difficulty tiap AI

Survei mengenai *Difficulty* untuk tiap *Agent AI* digunakan untuk menentukan dan melihat seberapa sulit *agent* AI yang dilawan oleh peserta. Survei ini memiliki rentang nilai 1 – 5, dimana 1 berarti sangat mudah dan 5 berarti sangat sulit. Hasil dari *difficulty* untuk setiap *Agent* AI dapat dilihat pada tabel dibawah ini.

Dari tabel 4 menunjukkan bahwa penggunaan DDA pada FH-EMCTS memberikan tingkat kesulitan yang hampir mendekati ideal yakni 3 sesuai dengan hipotesis yang telah dibuat pada bab III dan *vanilla* FH-EMCTS memiliki tingkat kesulitan yang paling tinggi bagi pemain yakni 4.4.

Tabel 4. *Difficulty* tiap *Agent AI* dengan jumlah *Action Point* 5

	MEAN
FH-EMCTS	4.4
ROSAS	2.3
POSAS	2.3
True ROSAS	2.7
True POSAS	2.2
Adaptive True ROSAS	3

4.4.4 Enjoyment tiap AI

Survei mengenai *Enjoyment* untuk tiap *Agent AI* digunakan untuk menentukan dan melihat seberapa menikmati peserta melawan *agent AI* tersebut. Survei ini memiliki rentang nilai 1 – 5, dimana 1 berarti sangat tidak menikmati dan 5 berarti sangat menikmati. Hasil dari *enjoyment* untuk setiap *Agent* AI dapat dilihat pada tabel dibawah ini.

Tabel 5. *Enjoyment* tiap *Agent AI* dengan jumlah *Action Point* 5

	MEAN
FH-EMCTS	2
ROSAS	3.8
POSAS	3.8
True ROSAS	3.5
True POSAS	3.3
Adaptive True ROSAS	3.3

Dari tabel 5 menunjukkan bahwa penggunaan DDA pada FH-EMCTS memberikan tingkat enjoyment yang lebih tinggi dibandingkan dengan *vanilla* FH-EMCTS.

4.4.5 Realistic tiap AI

Survei mengenai *Realistic* untuk tiap *Agent AI* digunakan untuk menentukan dan melihat seberapa realistis menurut peserta mengenai *agent AI* yang sedang dilawan. Survei ini memiliki rentang nilai 1 – 5, dimana 1 berarti sangat tidak realistis dan 5 berarti sangat realistis. Hasil dari *enjoyment* untuk setiap *Agent* AI dapat dilihat pada tabel dibawah ini.

Tabel 6. *Realistic* tiap *Agent AI* dengan jumlah *Action Point* 5

	MEAN
FH-EMCTS	1.8
ROSAS	3.6
POSAS	3.3
True ROSAS	3.4
True POSAS	3.2
Adaptive True ROSAS	3.8

Dari tabel 6 menunjukkan bahwa penggunaan DDA pada FH-EMCTS memberikan nilai tingkat *realistic* yang lebih tinggi dibandingkan dengan *vanilla* FH-EMCTS. Realistis dalam penelitian ini mengindikasikan pemain tidak merasakan sedang bermain dengan sebuah AI, melainkan pemain merasakan sedang bermain dengan pemain lainnya.

5. KESIMPULAN

Dari hasil pembuatan *agent AI Evolutionary Monte Carlo Tree Search with Flexible Search Horizon* dengan *Dynamics Difficulty Adjustment*, pengujian terhadap *agent* AI lainnya, dan survei mengenai *agent AI* yang telah dibuat, dapat disimpulkan bahwa.

- *Agent AI* yang menggunakan *Evolutionary Monte Carlo Tree Search with Flexible Search Horizon* dengan *Dynamics Difficulty Adjustment* dapat diimplementasikan kedalam *game multi-action adversarial games* dan dapat memberikan tingkat kesulitan yang seimbang terhadap *agent AI* lainnya maupun manusia.
- Berdasarkan hasil survei percobaan *agent AI* melawan manusia menunjukkan bahwa *agent AI* yang paling menyenangkan dan realistis bukanlah *agent AI* yang memiliki kemampuan yang paling baik dalam hal persentase kemenangan, melainkan *agent AI* yang seperti ROSAS dan POSAS yang dianggap oleh pemain lebih menyenangkan dan realistis. Selain itu juga tingkat kesulitan yang terlalu tinggi juga tidak membuat manusia yang memainkan *game* tersebut menjadi menyenangkan, survei menunjukkan bahwa tingkat kesulitan yang cukup tinggi berbanding terbalik dengan tingkat kepuasan bermain manusia yang memainkannya.

Saran yang dapat diberikan untuk penelitian lebih lanjut antara lain:

- Penelitian metode baru untuk *Dynamics Difficulty Adjustment* pada *Evolutionary Monte Carlo Tree Search with Flexible Search Horizon* untuk mendapatkan hasil yang lebih baik.

6. DAFTAR PUSTAKA

[1] Afergan, D., Peck, E.M., Solovey, E.T., Jenkins, A., Hincks, S.W., Brown, E.T., Chang, R. and Jacob, R.J.K. 2014. Dynamic difficulty using brain metrics of workload. *Conference on Human Factors in Computing Systems - Proceedings*. (2014), 3797–3806. DOI:https://doi.org/10.1145/2556288.2557230.

[2] Alexander, J.T., Sear, J. and Oikonomou, A. 2013. An investigation of the effects of game difficulty on player enjoyment. *Entertainment Computing*. 4, 1 (2013), 53–62. DOI:https://doi.org/10.1016/j.entcom.2012.09.001.

[3] Andrade, K.D.O., Pasqual, T.B., Caurin, G.A.P. and Crocomo, M.K. 2016. Dynamic difficulty adjustment with Evolutionary Algorithm in games for rehabilitation robotics. *2016 IEEE International Conference on Serious Games and Applications for Health, SeGAH 2016*. (2016). DOI:https://doi.org/10.1109/SeGAH.2016.7586277.

[4] Ang, D. and Mitchell, A. 2017. Comparing effects of dynamic difficulty adjustment systems on video game experience. *CHI PLAY 2017 - Proceedings of the Annual Symposium on Computer-Human Interaction in Play*. (2017), 317–327. DOI:https://doi.org/10.1145/3116595.3116623.

[5] Aponte, M.-V., Levieux, G. and Natkin, S. 2011. Difficulty in videogames. (2011), 1. DOI:https://doi.org/10.1145/2071423.2071484.

[6] Baier, H. and Cowling, P.I. 2018. Evolutionary MCTS for Multi-Action Adversarial Games. *IEEE Conference on Computatonal Intelligence and Games, CIG*. 2018-Augus, (2018), 1–8. DOI:https://doi.org/10.1109/CIG.2018.8490403.

[7] Baier, H. and Cowling, P.I. 2018. Evolutionary MCTS with flexible search horizon. *Proceedings of the 14th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2018*. Aiide (2018), 2–8.

[8] Chen, J. 2007. Flow in games (and everything else). *Communications of the ACM*. 50, 4 (2007), 31–34. DOI:https://doi.org/10.1145/1232743.1232769.

[9] Colwell, A.M. and Glavin, F.G. 2017. Colwell's castle defence: A custom game using dynamic difficulty adjustment to increase player enjoyment. *CEUR Workshop Proceedings*. 2086, (2017), 275–282.

[10] Constant, T. and Levieux, G. 2019. Dynamic difficulty adjustment impact on players' confidence. *Conference on Human Factors in Computing Systems - Proceedings*. (2019), 1–12. DOI:https://doi.org/10.1145/3290605.3300693.

[11] Cowley, B., Charles, D., Black, M. and Hickey, R. 2008. Toward an understanding of flow in video games. *Computers in Entertainment*. 6, 2 (2008), 1–27. DOI:https://doi.org/10.1145/1371216.1371223.

[12] Demediuk, S., Tamassia, M., Li, X. and Raffe, W.L. 2019. Challenging AI: Evaluating the Effect of MCTS-Driven Dynamic Difficulty Adjustment on Player Enjoyment. *ACM International Conference Proceeding Series*. (2019). DOI:https://doi.org/10.1145/3290688.3290748.

[13] Demediuk, S., Tamassia, M., Raffe, W.L., Zambetta, F., Li, X. and Mueller, F. 2017. Monte Carlo tree search based algorithms for dynamic difficulty adjustment. *2017 IEEE Conference on Computational Intelligence and Games, CIG 2017*. (2017), 53–59. DOI:https://doi.org/10.1109/CIG.2017.8080415.

[14] Gerling, K.M., Miller, M., Mandryk, R.L., Birk, M. and Smeddinck, J. 2014. Effects of skill balancing for physical abilities on player performance, experience and self-esteemin exergames. *Conference on Human Factors in Computing Systems - Proceedings*. (2014), 2201–2210. DOI:https://doi.org/10.1145/2556288.2556963.

[15] Hao, Y., He, S., Wang, J., Liu, X., Yang, J. and Huang, W. 2010. Dynamic difficulty adjustment of game AI by MCTS for the game Pac-Man. *Proceedings - 2010 6th International Conference on Natural Computation, ICNC 2010*. 8, Icnc (2010), 3918–3922. DOI:https://doi.org/10.1109/ICNC.2010.5584761.

[16] Hunicke, R. 2005. The case for dynamic difficulty adjustment in games. *ACM International Conference Proceeding Series*. 265, (2005), 429–433. DOI:https://doi.org/10.1145/1178477.1178573.

[17] Hunicke, R. and Chapman, V. 2004. AI for dynamic difficulty adjustment in games. *AAAI Workshop - Technical Report*. WS-04-04, January 2004 (2004), 91–96.

[18] Justesen, N., Mahlmann, T., Risi, S. and Togelius, J. 2017. IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES Playing Multi-Action Adversarial Games: Online Evolutionary Planning versus Tree Search. (2017), 1–10.

[19] Lucas, S.M. and Kendall, G. 2006. Evolutionary computation and games. *IEEE Computational Intelligence Magazine*. 1, 1 (2006), 10–18. DOI:https://doi.org/10.1109/MCI.2006.1597057.

[20] Risi, S. and Togelius, J. 2015. Neuroevolution in games: State of the art and open challenges. *IEEE Transactions on Computational Intelligence and AI in Games*. PP, 99 (2015), 1–19. DOI:https://doi.org/10.1109/TCIAIG.2015.2494596.