OPTIMASI PEMILIHAN KARAKTER DALAM GAME MENGGUNAKAN METODE ANT COLONY OPTIMIZATION

SKRIPSI

Oleh:
ECOFAH RADITYA DANISWARA
NIM. 220605110155

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2026

OPTIMASI PEMILIHAN KARAKTER DALAM GAME MENGGUNAKAN METODE ANT COLONY OPTIMIZATION

SKRIPSI

Diajukan Kepada:
Universitas Islam Negeri Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
ECOFAH RADITYA DANISWARA
220605110155

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2026

HALAMAN PERSETUJUAN

OPTIMASI PEMILIHAN KARAKTER DALAM GAME MENGGUNAKAN METODE ANT COLONY OPTIMIZATION

SKRIPSI

Oleh :
ECOFAH RADITYA DANISWARA
NIM. 220605110155

Telah Diperiksa dan Disetujui untuk Diuji:
Tanggal: 29 Juni 2026

Pembimbing I,

Dr. Ir. Fresy Nugroho, M.T, IPM., ASEAN Eng
NIP. 19710722 201101 1 001

Pembimbing II,

Dr. Ir. Fachrul Kurniawan, M.MT., IPU
NIP. 19771020 200912 1 001

Mengetahui,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M.Kom
NIP. 19841010 201903 1 012

HALAMAN PENGESAHAN

OPTIMASI PEMILIHAN KARAKTER DALAM GAME MENGGUNAKAN METODE ANT COLONY OPTIMIZATION

SKRIPSI

Oleh :
ECOFAH RADITYA DANISWARA
NIM. 220605110155

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal: 29 Juni 2026

Susunan Dewan Penguji

Ketua Penguji	: Prof. Dr. H.Muhammad Faisal, S.Kom, M.T NIP. 19740510 200501 1 007	( )
Anggota Penguji I	: Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )
Anggota Penguji II	: Dr. Ir. Fresy Nugroho, M.T., I.P.M., ASEAN Eng. NIP. 19710722 201101 1 001	( )
Anggota Penguji III	: Dr. Ir. Fachrul Kurniawan, ST., M.MT., IPU NIP. 19771020 200912 1 001	( )

Mengetahui dan Mengesahkan,
Ketua Program Studi Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Supriyono, M.Kom
NIP. 19841010 201903 1 012

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama	: Ecofah Raditya Daniswara
NIM	: 220605110155
Fakultas / Program Studi	: Sains dan Teknologi / Teknik Informatika
Judul Skripsi	: Optimasi Pemilihan Karakter Dalam Game Menggunakan Metode Ant Colony Optimization

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini merupakan hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 29 Juni 2026
Yang membuat pernyataan,

Ecofah Raditya Daniswara
NIM. 220605110155

MOTTO

"All we can do is do our best to relish this remarkable ride."

-Tim Lake-

HALAMAN PERSEMBAHAN

Alhamdulillahi rabbil 'alamin puji syukur kepada Allah SWT karena berkat rahmat dan petunjuk-Nya penulis dapat menyelesaikan skripsi ini. Shalawat serta salam selalu tercurahkan kepada Rasulullah SAW yang telah membawa kita dari zaman jahiliyah menuju addinul Islam.

Penulis mempersembahkan karya ini kepada diri orang tua, diri sendiri, para dosen, teman-teman, serta orang-orang yang telah membantu, mendoakan, serta menyemangati penulis dalam menuntaskan skripsi ini.

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabarakatuh

Segala puji dan rasa syukur penulis panjatkan ke hadiarat Allah *Subhanahu wa Ta'ala* atas limpahan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Shalawat dan salam semoga senantiasa tercurah kepada Nabi Muhammad SAW, semoga kita semua kelak memperoleh syafaat beliau di hari kiamat. Aamiin.

Penulis ucapkan terima kasih yang sebesar-besarnya kepada seluruh pihak yang memberikan dukungan dan motivasi kepada penulis sehingga dapat menyelesaikan skripsi ini. Ucapan terima kasih ini penulis disampaikan kepada:

1. Prof. Dr. Hj. Ilfi Nur Diana, M.Si, selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. H. Agus Mulyono, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Supriyono, M.Kom, selaku Ketua Program Studi Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Dr. Ir. Fresy Nugroho, ST., MT, IPM, selaku dosen pembimbing 1 serta mentor yang telah sangat sabar memberikan bimbingan, saran, dan dukungan yang sangat berarti bagi penulis selama proses penelitian.
5. Dr. Ir. Fachrul Kurniawan, ST., M.MT., IPU, selaku dosen pembimbing 2, atas arahan dan pencerahan yang sangat membantu dalam menyempurnakan karya ini.

6. Prof. Dr. H.Muhammad Faisal, S.Kom, M.T, selaku dosen Ketua Penguji dan Hani Nurhayati, M.T, selaku dosen Penguji I yang telah memberikan kritik dan saran yang membangun dalam penyusunan skripsi ini.
7. Seluruh staf dan dosen Program Studi Teknik Informatika yang telah memberikan ilmu, dukungan, dan fasilitas kepada penulis selama masa studi.
8. Ayahanda Rudi Cahyono dan Ibunda Ermin Afdlila yang memiliki peran besar dalam perjalanan penulis selama menempuh pendidikan perkuliahan. Terima kasih atas segala perjuangan, dukungan, perhatian, semangat, pengorbanan, serta doa yang senantiasa diberikan kepada penulis dalam setiap proses penyelesaian skripsi ini. Semoga Allah SWT senantiasa melimpahkan kesehatan, keberkahan, kebahagiaan, serta perlindungan kepada beliau berdua, dan membalas seluruh kebaikan dengan pahala yang berlipat ganda.
9. Zhul Azizah, yang senantiasa memberikan dukungan, semangat, motivasi, serta menjadi tempat berbagi cerita dan bertukar pikiran selama proses perkuliahan hingga penyelesaian skripsi ini. Terima kasih atas perhatian, kesabaran, doa, dan dukungan yang diberikan selama proses penyusunan penelitian ini. Kehadiran dan dukungan yang diberikan menjadi salah satu dorongan bagi penulis untuk tetap berusaha menyelesaikan studi ini dengan sebaik-baiknya.
10. Qiqi, Rofiq, Deshinta, Rizal, dan Ivanz yang senantiasa menjadi rekan berdiskusi dan bertukar pemikiran selama masa perkuliahan hingga proses

penyelesaian skripsi ini. Terima kasih atas dukungan, semangat, motivasi, serta berbagai pandangan dan pengalaman yang telah dibagikan, baik dalam bidang akademik maupun kehidupan sehari-hari, sehingga memberikan banyak pelajaran dan dorongan bagi penulis dalam menyelesaikan studi ini.

11. Teman-teman yang selalu hadir dan menemani penulis selama masa perkuliahan, yaitu serta teman-teman lainnya yang tidak dapat disebutkan satu per satu. Terima kasih atas kebersamaan, dukungan, semangat, serta berbagai proses suka dan duka yang telah dilalui bersama selama masa perkuliahan. Semoga segala kebaikan dan pengalaman yang telah dilalui menjadi langkah menuju kesuksesan bagi kita semua.
12. Seluruh pihak yang telah terlibat, baik secara langsung maupun tidak langsung dari awal perkuliahan hingga akhir penulisan skripsi ini

Penulis berharap semoga skripsi ini dapat memberikan manfaat di masa yang akan datang. Penulis juga menyadari bahwa karya ini masih memiliki kekurangan. Oleh karena itu, penulis sangat terbuka terhadap segala bentuk masukan dan saran yang membangun untuk pengembangan lebih lanjut.

Wassalamu'alaikum warahmatullahi wabarakatuh.

Malang, 29 Juni 2026

Penulis

DAFTAR ISI

HALAMAN PERSETUJUAN ..... **iii**
HALAMAN PENGESAHAN ..... **iv**
PERNYATAAN KEASLIAN TULISAN ..... **v**
MOTTO ..... **vi**
HALAMAN PERSEMBAHAN ..... **vii**
KATA PENGANTAR ..... **viii**
DAFTAR ISI ..... **xi**
DAFTAR GAMBAR ..... **xiii**
DAFTAR TABEL ..... **xiv**
ABSTRAK ..... **xv**
ABSTRACT ..... **xvi**
مستخلص البحث ..... **xvii**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Identifikasi Masalah 5
1.3 Batasan Penelitian 5
1.4 Tujuan Penelitian 6
1.5 Manfaat Penelitian 6
BAB II STUDI PUSTAKA **6**
2.1 Penelitian Terkait 6
2.2 Game Adaptif 12
2.2.1 Pengertian Game Adaptif 13
2.2.2 Tujuan Game Adaptif 14
2.2.3 Sistem Adaptif Hybrid dalam Game 15
2.2.4 Elemen dalam Game Adaptif 16
2.3 Ant Colony Optimization (ACO) 17
2.3.1 Pengertian Ant Colony Optimization 17
2.3.2 Rumus Ant Colony Optimization 18
2.4 *Rule-Based Adaptive System* 21
2.4.1 Pengertian Rule-Based Statis 21
2.4.2 Model Formal Production Line 22
2.4.3 Komponen Sistem *Rule-Based Statis* 23
2.4.4 Forward Chaining 25
2.4.5 Rule Matching 26
BAB III DESAIN DAN IMPLEMENTASI **15**
3.1 Desain Penelitian 15
3.1.1 Gambaran Sistem 15
3.1.2 Pembuatan Game The Night Dreamers 16
3.1.3 Rancangan Finate State Machine 17
3.1.4 Rancangan User Interface 18
3.1.5 Definisi dan Deskripsi Karakter 22
3.1.6 Pengumpulan Data Game 24
3.1.7 Rule-Based Adaptive System 25
3.1.8 Alur Sistem Pemilihan Karakter Adaptif 26
3.2 Implementasi Perhitungan Ant Colony Optimization 28
3.2.1 Alternatif 29

3.2.2 Kriteria 30
3.2.3 Perhitungan Ant Colony Optimization 37
3.2.4 Pengujian Usability (System Usability Scale) 47
BAB IV HASIL DAN PEMBAHASAN 47
4.1 Hasil Implementasi Game 47
4.1.1 Tampilan Game 47
4.2 Hasil Pengujian Sistem 51
4.2.1 Pengujian Performa pemain 52
4.2.2 Pengujian Rule-Based Adaptive System 54
4.3 Pengujian Rekomendasi Karakter ACO 57
4.3.1 Implementasi ACO 58
4.3.2 Hasil Rekomendasi Karakter 63
4.4 Analisis Pemilhan Karakter oleh Pemain 65
4.4.1 Data Performa Pemain per Permainan 66
4.4.2 Pilihan Karakter per Pemain 69
4.4.3 Rata-rata Clear Time per Pemain 71
4.5 Pengujian System Usability Scale 72
4.6 Integrasi dengan Islam 77
4.6.1 Hablum Minallah 78
4.6.2 Hablum Minannas 79
4.6.3 Hablum Minal'Alam 79
BAB V KESIMPULAN DAN SARAN 83
5.1 Kesimpulan 83
5.2 Saran 84
DAFTAR PUSTAKA

DAFTAR GAMBAR

Gambar 2. 1 Bentuk Umum Rule-Based (Sasikumar et all, 2007)..... 23
Gambar 2. 2 Komponen Rule-Based System (Sasikumar et all, 2007)..... 24
Gambar 3. 1Blok Diagram Sistem..... 16
Gambar 3. 2 Diagram Finite State Machine enemy 18
Gambar 3. 3 Sketsa MainMenu 19
Gambar 3. 4 Sketsa Pemilihan Karakter..... 20
Gambar 3. 5 Sketsa GamePlay 21
Gambar 3. 6 Sketsa GameOver 22
Gambar 3. 7 Flowchart Alur Sistem 27
Gambar 4. 1 Tampilan Antarmuka Menu Utama (Main Menu) 48
Gambar 4. 2 Tampilan Menu Pemilihan Karakter (Character Selection)..... 49
Gambar 4. 3 Tampilan Antarmuka Sesi Permainan (Gameplay)..... 50
Gambar 4. 4 Tampilan Panel Game Over..... 51
Gambar 4. 5 Tampilan Rekaman Data Performa Pemain..... 52
Gambar 4. 6 Grafik Rata-Rata Clear Time dan Hit Taken 53
Gambar 4. 7 Tampilan Aktivasi Aturan R-01 55
Gambar 4. 8 Tampilan Aktivasi Aturan R-02 55
Gambar 4. 9 Tampilan Aktivasi Aturan R-03 56
Gambar 4. 10 Grafik Perbandingan Hasil Aturan DDA 57
Gambar 4. 11 Tampilan Rekomendasi ACO Dalam Game 63
Gambar 4. 12 Frekuensi Rekomendasi Karakter dari ACO 65
Gambar 4. 13 Grafik Pemilihan Karakter oleh Pemain..... 70
Gambar 4. 14 Grafik Rata-Rata Clear Time Para Pemain..... 71

DAFTAR TABEL

Tabel 2. 1 Penelitian Terkait 9
Tabel 3. 1Profil, Kepribadian, dan Gaya Bermain Tujuh Karakter Game 23
Tabel 3. 2 Aturan Penyesuaian Tingkat Kesulitan 25
Tabel 3. 3 Nilai Statistik Atribut Dasar dan Tipe Gaya Bermain 29
Tabel 3. 4 Kriteria Penilaian Atribut Karakter dan Bobot Dasar 30
Tabel 3. 5 Skala Penilaian Kriteria Attack 31
Tabel 3. 6 Skala Penilaian Kriteria Speed 32
Tabel 3. 7 Skala Penilaian Kriteria Defense 33
Tabel 3. 8 Skala Penilaian Kriteria Stability 34
Tabel 3. 9 Skala Penilaian Kriteria Accuracy 35
Tabel 3. 10 Skala Penilaian Kriteria Recovery 36
Tabel 3. 11 Rekapitulasi Hasil Konversi Nilai Atribut 37
Tabel 3. 12 Hasil Perhitungan Nilai Heuristik Dasar 39
Tabel 3. 13 Bobot Kriteria dan Faktor Boost pada Kondisi Normal 40
Tabel 3. 14 Penyesuaian Bobot Kriteria dengan Faktor Boost 41
Tabel 3. 15 Rekapitulasi Nilai Heuristik Penyesuaian 42
Tabel 3. 16 Probabilitas Pemilihan Karakter pada Iterasi Pertama Stage 1. 43
Tabel 3. 17 Probabilitas Pemilihan Karakter pada Iterasi Pertama Stage 2. 44
Tabel 3. 18 Perkembangan Nilai Pheromone Karakter Iterasi 0-20 pada Stage 1. 46
Tabel 3. 19 Perkembangan Nilai Pheromone Karakter Iterasi 0-20 pada Stage 2. 47
Tabel 3. 20 Daftar Pernyataan Kuesioner Pengujian SUS. 48
Tabel 4. 1 Rata-Rata Clear Time dan Hit Taken Pemain Berdasarkan Stage 53
Tabel 4. 2 Hasil DDA Rule-Based 56
Tabel 4. 3 Frekuensi dan Persentase Aktivasi Aturan DDA Rule-Based. 64
Tabel 4. 4 Data Pemain dalam Game 66
Tabel 4. 5 Penjelasan Setiap Pertayaan SUS 72
Tabel 4. 6 Nilai SUS 73
Tabel 4. 7 Skor Responden 74
Tabel 4. 8 Hasil Perhitungan Skor SUS 75

ABSTRAK

Daniswara, Ecofah Raditya. 2026. **Optimasi Pemilihan Karakter Dalam Game Menggunakan Metode Ant Colony Optimization**. Skripsi. Program Studi Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Dr. Ir. Fresy Nugroho, M.T, IPM. (II) Dr. Ir. Fachrul Kurniawan, M.MT., IPU.

Kata Kunci: *Ant Colony Optimization, Dynamic Difficulty Adjustment,* Optimasi Pemilihan Karakter, Game.

Pemilihan karakter yang tepat dalam sebuah permainan digital (*game*) memegang peranan krusial dalam menentukan keberhasilan pemain menyelesaikan tantangan. Namun, ketidakseimbangan antara kemampuan pemain dan tingkat kesulitan *game* sering kali menyebabkan kebosanan (*boredom*) atau frustrasi (*frustration*). Penelitian ini bertujuan untuk mengoptimalkan pemilihan karakter menggunakan algoritma *Ant Colony Optimization* (ACO) serta mengintegrasikan sistem *Dynamic Difficulty Adjustment* (DDA) guna menyesuaikan tingkat kesulitan permainan secara dinamis. Algoritma ACO digunakan untuk memberikan rekomendasi karakter terbaik berdasarkan parameter performa pemain, sementara DDA mendistribusikan penyesuaian status musuh secara adaptif. Pengujian sistem dilakukan melalui analisis performa *gameplay* pada dua tingkatan (*Stage 1* dan *Stage 2*). Berdasarkan analisis data permainan, rekomendasi karakter berbasis ACO berhasil diikuti oleh pemain sebanyak 60,58% (103 kali) dengan rata-rata nilai *eta* rekomendasi sebesar 0,64 pada *Stage 1* dan 0,63 pada *Stage 2*. Sistem DDA secara efektif menyesuaikan permainan dengan rincian 53,5% pemain menerima penurunan kesulitan (Rule R-01), 40,5% mempertahankan kesulitan (Rule R-02), dan 5,8% mengalami peningkatan kesulitan (Rule R-03). Selain itu, terdapat peningkatan tantangan yang dinamis dengan rata-rata waktu penyelesaian (*clear time*) dari 56,33 detik pada *Stage 1* menjadi 72,69 detik pada *Stage 2*. Penelitian ini menyimpulkan bahwa kombinasi ACO dan DDA berhasil mengoptimalkan pemilihan karakter dan menghadirkan pengalaman bermain yang lebih adaptif dan proporsional bagi pemain.

ABSTRACT

Daniswara, Ecofah Raditya. 2026. **Optimizing In-Game Character Selection Using the Ant Colony Optimization Method.** Thesis. Informatics Engineering Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Advisors: (I) Dr. Ir. Fresy Nugroho, M.T, IPM. (II) Dr. Ir. Fachrul Kurniawan, M.MT., IPU.

Keywords: *Ant Colony Optimization, Dynamic Difficulty Adjustment, Optimization, Character Selection*

Proper character selection in a digital game plays a crucial role in determining a player's success in completing challenges. However, the imbalance between player skills and game difficulty often leads to boredom or frustration. This study aims to optimize character selection using the Ant Colony Optimization (ACO) algorithm and integrate a Dynamic Difficulty Adjustment (DDA) system to dynamically scale the game's difficulty level. The ACO algorithm is utilized to provide the best character recommendations based on player performance parameters, while the DDA adaptively modulates enemy stats. System testing was conducted through gameplay performance analysis across two levels (Stage 1 and Stage 2). Based on gameplay data analysis, ACO-based character recommendations were followed by players 60.58% of the time (103 times), with an average recommendation eta value of 0.64 in Stage 1 and 0.63 in Stage 2. The DDA system effectively adjusted the game environment, where 53.5% of players received a difficulty decrease (Rule R-01), 40.5% maintained difficulty (Rule R-02), and 5.8% experienced a difficulty increase (Rule R-03). Furthermore, a dynamic progression of challenges was observed, with the average clear time increasing from 56.33 seconds in Stage 1 to 72.69 seconds in Stage 2. This study concludes that the combination of ACO and DDA successfully optimizes character selection and delivers a more adaptive and balanced gaming experience for players.

مستخلص البحث

دانيسوارا، إيكوفا راديتيا. 2026 .تحسين اختيار الشخصيات في الألعاب باستخدام طريقة تحسين مستعمرة النمل (*Ant Colony Optimization*). بحث تخرج (أطروحة). قسم هندسة المعلوماتية، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية مالانج .المشرفون: (1) الدكتور المهندس فريسي نوغروهو، الماجستير في الهندسة، IPM. (2) الدكتور المهندس فخرول كورنياوان، الماجستير في إدارة التكنولوجيا، .IPU

الكلمات المفتاحية :تحسين مستعمرة النمل(*Ant Colony Optimization*) ، التعديل الديناميكي للصعوبة، **تحسين اختيار الشخصيات، اللعبة.**

يلعب الاختيار الصحيح للشخصيات في الألعاب الرقمية (*game*) دوراً حاسماً في تحديد نجاح اللاعب في إكمال التحديات. ومع ذلك، فإن عدم التوازن بين قدرة اللاعب ومستوى صعوبة اللعبة غالباً ما يؤدي إلى الشعور بالملل (*boredom*) أو الإحباط .(*frustration*) يهدف هذا البحث إلى تحسين عملية اختيار الشخصيات باستخدام خوارزمية تحسين مستعمرة النمل (ACO)بالإضافة إلى دمج نظام التعديل الديناميكي للصعوبة (DDA) لضبط مستوى صعوبة اللعبة بشكل ديناميكي. تُستخدم خوارزمية ACO لتقديم أفضل التوصيات للشخصيات بناءً على معايير أداء اللاعب، بينما يقوم نظام DDA بتوزيع التعديلات على حالة الأعداء بشكل تكيفي. تم إجراء اختبار النظام من خلال تحليل أداء أسلوب اللعب على مستويين (المرحلة 1 والمرحلة 2). وبناءً على تحليل بيانات اللعبة، تم إتباع التوصيات الخاصة بالشخصيات القائمة على ACO بنجاح من قبل اللاعبين بنسبة 60.58% (103 مرات) بمتوسط قيمة "إيتا (eta) "للتوصية يبلغ 0.64 في المرحلة الأولى و 0.63 في المرحلة الثانية. قام نظام DDA بتعديل اللعبة بشكل فعال حيث تلقى 53.5% من اللاعبين انخفاضاً في مستوى الصعوبة) القاعدة(R-01 ، وحافظ 40.5% على نفس مستوى الصعوبة) القاعدة(R-02 ، وشهد 5.8% زيادة في الصعوبة) القاعدة .(R-03 بالإضافة إلى ذلك، كانت هناك زيادة ديناميكية في التحدي مع ارتفاع متوسط وقت الإكمال (*clear time*) من 56.33 ثانية في المرحلة الأولى إلى 72.69 ثانية في المرحلة الثانية. يخلص هذا البحث إلى أن الجمع بين ACO و DDA قد نجح في تحسين عملية اختيار الشخصيات وتقديم تجربة لعب أكثر تكيفاً وتناسباً للاعبين.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan pada industri game saat ini tidak hanya berfungsi sebagai media hiburan, tetapi bisa sebagai media interaktif yang mampu adaptasi Terhadap pemain untuk meningkatkan pengalaman bermain. Pendekatan pemain mendorong para pengembang untuk membuat sistem permainan yang lebih cerdas, dinamis dan sesuai dengan gaya bermain para pemain.

Seiring dengan kebutuhan akan pengalaman bermain yang lebih personal, muncul istilah game adaptif, yaitu permainan yang mampu menyesuaikan elemen di dalamnya melalui kondisi atau performa pemain. Sistem adapatif ini memungkinkan game untuk mengontrol tingkat kesulitan, gaya bermain, dan pemilihan karakter secara otomatis supaya sesuai dengan kemampuan pemain. Pendekatan ini terbukti bisa meningkatkan keterlibatan (engagement), fokus dan juga kepuasan para pemain. Namun, kondisi sekarang ini beberapa game masih menggunakan sistem statis yang dimana pemain memilih karakter dan tingkat kesulitan secara manual tanpa ada adaptasi dinamis. Hal ini membuat pengalaman bermain menjadi kurang menantang bagi pemain yang memiliki gaya bermain berbeda.

Aspek penting dalam game adaptif berbasis perilaku pemain adalah pemilihan karakter. Pemilihan karakter merupakan bagian yang sangat penting Terhadap performa dan kepuasan pemain. Karakter adalah bagian dalam game biasanya memiliki atribut, kemampuan dan gaya bermain yang berbeda. Jika

karakter yang dipilih tidak sesuai dengan gaya bermain atau kondisi pemain dalam game, hal ini dapat mempengaruhi kenyamanan dan keterlibatan pemain selama bermain. Sebaliknya, jika sistem bisa merekomendasikan atau menyesuaikan karakter pemain sesuai dengan score, kondisi dan gaya pemain, pengalaman bermain dapat menjadi lebih optimal dan menyenangkan.

Permasalah utama adalah bagaimana sistem dapat menentukan karakter yang sesuai dengan performa pemain tepat. Pemilihan karakter melibatkan banyak unsur, yaitu data pemain, atribut karakter dan situasi permainan. Beberapa penelitian sebelumnya telah mencoba menggunakan algoritma kecerdasan buatan seperti genetic algorithm dan particle swarm optimization untuk menyelesaikan masalah optimasi dalam game. Akan tetapi, penggunaan metode tersebut masih mempunyai keterbasatan dalam hal konvergensi dan akurasi hasil optimasi. Untuk itu diperlukan metode yang mampu melakukan proses pencarian solusi terbaik secara adaptif dengan melihat banyak kemungkinan dalam waktu yang cepat.

Menjawab permasalah tersebut, penelitian ini memakai salah satu metode yaitu *Ant Colony Optimization* (ACO). ACO merupakan algoritma yang terinspirasi dari perilaku koloni semut dalam mencari jalan tercepat menuju sumber makanan. Prinsip kerja ACO yaitu bagaimana “semut-semut” buatan melacak berbagai jalur dan meninggalkan jejak feromon untuk menandai solusi yang optimal. Metode ini memiliki kemampuan untuk melihat solusi optimal melalui proses adaptif, sehingga cocok diterapkan dalam pengambilan keputusan, termasuk dalam pemilihan karakter game.

Untuk membuat sistem ini, digunakan metode Ant Colony Optimization (ACO). Sistem membutuhkan algoritma yang optimal untuk mencari solusi dari kumpulan alternatif karakter dengan menghitung berbagai parameter permainan seperti skor, waktu bermain, dan juga tingkat keberhasilan misi yang diperoleh pemain. Pendekatan ini tidak hanya meningkatkan kemampuan sistem dalam pemilihan karakter dengan performa pemain, tetapi mendorong terciptanya pengalaman bermain yang lebih personal, adaptif dan dinamis melalui data peforma pemain. Pendekatan ini memungkinkan proses pemilihan karakter secara otomatis, sehingga permainan dapat menyesuaikan gaya pemain dan kemampuan pemain saat bermain. Dalam penerapannya, teknologi ini memungkinkan digunakan pada tipe game adaptif untuk menghadirkan interaksi yang lebih menari, interaktif dan tantangan yang sesuai dengan kemampuan pemain.

Dalam pandangan islam, sifat baik dan buruknya suatu manusia bisa dilihat dari karakteristiknya.

اِنَّ الْاِنْسَانَ خُلِقَ هَلُوْعًا ۙ ١٩ اِذَا مَسَّهُ الشَّرُّ جَزُوْعًا ۙ ٢٠ وَاِذَا مَسَّهُ الْخَيْرُ مَنُوْعًا ۙ ٢١

"Sesungguhnya manusia diciptakan bersifat keluh kesah lagi kikir. Apabila ia ditimpa kesusahan ia berkeluh kesah, dan apabila ia mendapat kebaikan (harta) ia amat kikir." (Q.S. Al-Ma'arij: 19-21).

Menurut Quraish Shihab dalam *Tafsir Al-Mishbah*, Surah Al-Ma'arij ayat 19–21 menjelaskan bahwa manusia memiliki sifat dasar yang cenderung lemah dalam menghadapi kondisi tertentu. Ketika manusia ditimpa kesulitan, ia mudah berkeluh kesah dan kehilangan ketenangan, sedangkan ketika memperoleh kenikmatan atau keuntungan, manusia cenderung mempertahankannya secara berlebihan dan enggan berbagi. Namun, sifat tersebut bukanlah sifat mutlak yang

tidak dapat diubah, melainkan kecenderungan dasar manusia yang dapat dikendalikan melalui keimanan, pengendalian diri, dan pembentukan akhlak yang baik. Tafsir ini menunjukkan bahwa perilaku dan respons manusia dapat berbeda-beda tergantung kondisi yang dihadapi serta karakter yang dimiliki oleh masing-masing individu.

Meskipun manusia memiliki kecenderungan sifat tertentu sebagaimana dijelaskan dalam Al-Qur'an, Islam tidak menjadikan sifat tersebut sebagai sesuatu yang tidak dapat diperbaiki. Menurut M. Quraish Shihab dalam *Tafsir Al-Mishbah*, sifat dasar manusia dapat dikendalikan melalui pembentukan akhlak, pengendalian diri, dan pendidikan moral. Konsep tersebut diperkuat oleh hadits Rasulullah SAW yang diriwayatkan oleh Imam Malik:

إِنَّمَا بُعِثْتُ لِأُتَمِّمَ مَكَارِمَ الْأَخْلَاقِ

"Sesungguhnya aku diutus untuk menyempurnakan akhlak yang mulia." (HR. Malik).

Menurut Imam Malik dalam kitab *Al-Muwatta'*, hadits tersebut menjelaskan bahwa Rasulullah SAW diutus untuk membentuk dan menyempurnakan akhlak manusia agar menjadi lebih baik. Hadits ini menunjukkan bahwa sifat dan karakter manusia bukan merupakan sesuatu yang bersifat mutlak, melainkan dapat dibentuk melalui pembiasaan, pendidikan moral, dan pengendalian diri. Penjelasan tersebut sejalan dengan tafsir M. Quraish Shihab dalam *Tafsir Al-Mishbah* terhadap Surah Al-Ma'arij ayat 19–21 yang menjelaskan bahwa manusia memiliki kecenderungan sifat tertentu dalam menghadapi suatu keadaan, seperti mudah berkeluh kesah ketika mengalami kesulitan dan cenderung mempertahankan kesenangan yang

dimilikinya. Namun, sifat tersebut masih dapat dikendalikan dan diperbaiki melalui pembentukan akhlak yang baik.

1.2 Identifikasi Masalah

Berdasarkan latar belakang yang sudah dijelaskan, masalah utama dalam penelitian ini adalah bagaimana membuat game adaptif dalam optimasi pemilihan karakter menggunakan metode *Ant Colony Optimization* berdasarkan data pencapaian pemain yang diperoleh.

1.3 Batasan Penelitian

Adapun batasan masalah ada penelitian ini adalah:

a. Penelitian ini berfokus pada proses optimasi pemilihan karakter menggunakan metode *Ant Colony Optimization* (ACO), di mana sistem akan menyesuaikan karakter yang dipilih berdasarkan variabel yang ditetapkan.

b. Kriteria yang digunakan dalam proses optimasi pemilihan karakter berdasarkan parameter kondisi dan perilaku pemain, seperti kecepatan reaksi, keberhasilan misi, dan durasi permainan. Kriteria yang ada akan menjadi nilai *benefit* dan *cost* sesuai pengaruhnya Terhadap hasil pemilihan karakter adaptif.

c. Variabel yang digunakan dalam proses optimasi pemilihan karakter terdiri dari beberapa atribut yang berbeda, seperti kekuatan, kecepatan, dan pertahanan.

1.4 Tujuan Penelitian

Tujuan penelitian ini untuk mengembangkan sebuah game yang mampu melakukan optimasi pemilihan karakter menggunakan metode *Ant Colony Optimization* (ACO) berdasarkan data pencapaian pemain yang didapatkan selama permainan.

1.5 Manfaat Penelitian

Peneltian ini diharapkan dapat memberikan manfaat dalam penerapan metode *Ant Colony Optimization* (ACO) untuk pemilihan karakter berdasarkan data pencapaian pemain dalam sistem game adaptif. Sistem ini diharapkan meningkatkan tingkat interaktif dan kenyamanan pemain melalui pemilihan karakter yang sesuai dengan performa dan pencapaian pemain. Penerapan metode *Ant Colony Optimization* diharapkan mampu melakukan proses optimasi secara efisien dan akurat dalam menentukan karakter terbaik. Peneliti berharap hasil dari penelitian ini dapat menjadi referensi bagi pengembang game lain, khususnya dalam penerapan algoritma optimasi untuk sistem pengambilan keputusan adaptif untuk menciptakan game yang lebih interaktif dan inovatif.

BAB II

STUDI PUSTAKA

2.1 Penelitian Terkait

Daftar penelitian yang berhubungan dengan penelitian ini bisa dilihat pada table 2.1. Beberapa penelitian sebelumnya pernah menerapkan metode *Ant Colony Optimization* (ACO) dalam pengembangan game dan sistem yang adaptif. Penelitian dari Kickmeier-Rust dan Holzinger (2019) menggunakan ACO untuk membantu sistem adaptasi pada *serious game* yang berfokus pada pembelajaran. Kemudian, Yoqsa (2023) memakai ACO sebagai cara baru untuk pengambilan keputusan dalam game strategi yang hasilnya lebih cepat dan efisien disbanding metode lainnya. Van de Put (2020) juga menerapkan ACO untuk menyesuaikan tingkat kesulitan pemain sesuai kemampuan pemain. Lalu, Inmaculada Rodríguez (2022) membuat sistem gamifikasi adaptif yang menyesuaikan pengalaman bermain berdasarkan profil pemain yang terus berubah. Di lain sisi, Hojoon Lee (2022) mengembangkan sistem rekomendasi otomatis untuk pemilihan karakter di game MOBA dengan menggunakan data peforma pemain. Dari beberapa penelitian sebelumnya bisa disimpulkan metode *Ant Colont Optimization* (ACO) berpotensi besar dalam membuat game jadi lebih cerdas dan adaptif. Akan tetapi, sampai sekarang masih belum banyak penelitian yang fokus pada menggunakan metode ACO untuk mengoptimalkan pemilihan karakter dalam game.

Penelitian oleh (Kickmeier-Rust & Holzinger, 2019) menerapkan algoritma *Ant Colony Optimization* (ACO) dalam pengembangan serious game yang bersifat

edukatif. Tujuannya menciptakan sistem pembelajaran yagn adpatif dengan menyesuaikan tingkat kesulitan dan konten pembelajaran sesuai kemampuan pemain. ACO digunakan sebagai mekanisme optimasi dalam memilih jalur dan tantangan yang paling sesuai, sehingga pemain dapat belajar dengan ritme dan kesulitan yang proporsional. Persamaan dengan penelitian ini menerapkan ACO sebagai mekanisme penyesuaian adaptif dalam game, sedangkan penelitian yang ditulis penulis mengarah pada penerapan ACO dalam keputusan pemilihan karakter secara adaptif berdasarkan performa pemain.

Penelitian oleh (Angeles-Garcia et al., 2023) membandingkan performa algoritma *Ant Colony Optimization* (ACO) dengan minimax dalam pengambilan keputusan strategi dalam game. Tujuannya menilai efektifitas kedua algoritma dalam menghasilkan keputusan optimal dengan waktu komputasi yang efisien pada kondisi permainan yang dinamis. Persamaan dengan penelitian ini terdapat pada penggunaan ACO sebagai pendekatan optimasi untuk meningkatkan kecerdasan sistem dalam permainan. Namun, Perbedaan dari penelitian penulis yaitu pada arah penerapan yang dimana penelitian sebelumnya membahas perbandingan dua algoritma untuk menentukan strategi terbaik, sedangkan penelitian penulis hanya menggunakan ACO untuk mengoptimalkan pemilihan karakter dalam game berbasis performa pemain.

Penelitian oleh (Put et al., 2020) mengembangkan model pembelajaran adaptif berbasis dengan menerapkan algoritma *Ant Colony Optimization* (ACO) sebagai metode utama dalam proses optimasi. Tujuannya untuk menciptakan sistem permainan yang mampu menyesuaikan tingkat kesuilatan berdasarkan performa

pemain supaya pembelajaran lebih efektif dan menarik. Persamaan dengan peneltian penulis tedapat pada pemanfaatan ACO sebagai mekanisme optimasi adaptif dalam sistem permainan, sementara perbedaannya terlihat pada penyesuaian tingkat kesulitan pembelajaran. Perbedaan dari penelitian sebelumnya lebih fokus pada penyesuaian tingkat kesulitan pembelajaran, sedangkan penelitian penulis fokus pada pengembangan game adapting yang mengoptimalkan proses pemilihan karakter berdasarkan performa pemain.

Penelitian oleh (Rodríguez & Puig, 2022) mengusulkan pendekatan adaptive gamification yang menyesuaikan elemen permainan dengan karakteristik dan motivasi pemain melalui penggunaan profil pemain dinamis. Tujuannya untuk meningkatkan kontribusi dan motivasi pemain dengan mengubah tingkat tantangan, hadiah, dan interaksi berdasarkan perilaku pengguna. Persamaan dengan penelitian penulis terlihat pada fokus adaptasi game terhadap perilaku pemain supaya pengalaman permainan lebih personal dan menarik. Namun, perbedaannya tidak menggunakan algoritma optimasi ACO, melainkan hanya menggunakan analisis profil pemain sebagai dasar adaptasi tanpa proses perhitungan optimasi.

Penelitian oleh (Lee et al., n.d.) mengembangkan sistem rekomendasi karakter dalam game Multi*player* Online Battle Arena (MOBA) menggunakan machine learning yang dinamakan Draftrec. Sistemnya dirancang untuk membantu pemain dalam proses pemilihan karakter secara strategis dengan mempertimbangkan gaya bermain, kerjasama antar karakter dan juga Riwayat kemenangan tim. Model dikembangkan dengan menganalisis lebih dari 280.000 data pertandingan menggunakan pembelajaran mendalam untuk memprediksi

kombinasi karakter terbaik. Hasil penelitian menunjukan bahwa sistem rekomendasi ini mampu meningkatkan tingkat kemenangan tim dan mengurangi kesalahan pada pemilihan karakter, serta memberikan pengalaman bermain yang lebih strategis dan efisien. Persamaan dengan penelitian penulis terdapat pada pemilihan karakter berbasis data performa pemain untuk meningkatkan pengalaman bermain. Perbedaan dari penelitian penulis terlihat pada metode yang digunakan, di mana penelitian DratRec menerapkan machine learning dan analisis statistic berdasarkan data historis.

Tabel 2. 1 Penelitian Terkait

No	Penelitian	Judul	Persamaan	Perbedaan
1	Kickmeier-Rust, M. D.& Holzinger, A. (2019)	*Interactive Ant Colony Optimization to Support Adaptation in Serious Games*.	Mengintegrasikan ACO dalam desain game adaptif untuk meningkatkan pengalaman pemain.	Menerapkan ACO pada penyesuaian pembelajaran dalam serious game untuk adaptasi pada lingkungan belajar.
2	Angeles-Garcia et al., (2023)	*Optimizing Strategy Games: Ant Colony Optimization vs. Minimax Algorithm*.	Membahas penggunaan ACO dalam pengambilan keputusan untuk meningkatkan peforma sistem game.	Fokus pada membandingkan ACO dengan Minimax pada peforma algoritma dalam strategi game.
3	Put, E.; et al. (2020)	*Adaptive Game-Based Learning Model Using Ant Colony Optimization*.	Menggunakan metode *Ant Colony Optimization* (ACO) untuk menyesuaikan sistem dalam game supaya adaptif terhadap peforma pemain	Lebih fokus pada adaptasi tingkat kesulitan pembelajaran untuk peningkatan efektivitas belajar.
4	Rodríguez, A.; et al. (2022)	*Towards Adaptive Gamification: A Method Using Dynamic Player Profile and a Case Study*.	Mengembangkan game adaptif yang menyesuaikan elemen permainan dengan karakter pemain.	Mengimplementasikan profil **dinamis** untuk adaptasi tantangan dan motivasi pemain dengan data kualitatif berdasarkan perilaku pemain.
5	Lee, K.; Hwang, D.; Kim, H.; Lee, B.;	*DraftRec: Personalized Draft Recommendation for Winning in Multi-Player*	Sama-sama bertujuan untuk meningkatkan pengalaman pemain melalui pemilihan karakter yang	Menggunakan pendekatan machine learning dan analisis data historis dalam sistem rekomendasi karakter dan berfokus pada peningkatan strategi

No	Penelitian	Judul	Persamaan	Perbedaan
	Choo, J. (2022)	*Online Battle Arena Games*.	optimal berdasarkan data pemain.	kemenangan dalam game MOBA kompetitif

Penelitian oleh (Kickmeier-Rust & Holzinger, 2019) menerapkan algoritma *Ant Colony Optimization* (ACO) dalam pengembangan serious game yang bersifat edukatif. Tujuannya menciptakan sistem pembelajaran yagn adpatif dengan menyesuaikan tingkat kesulitan dan konten pembelajaran sesuai kemampuan pemain. ACO digunakan sebagai mekanisme optimasi dalam memilih jalur dan tantangan yang paling sesuai, sehingga pemain dapat belajar dengan ritme dan kesulitan yang proporsional. Persamaan dengan penelitian ini menerapkan ACO sebagai mekanisme penyesuaian adaptif dalam game, sedangkan penelitian yang ditulis penulis mengarah pada penerapan ACO dalam keputusan pemilihan karakter secara adaptif berdasarkan performa pemain.

Penelitian oleh (Angeles-Garcia et al., 2023) membandingkan performa algoritma *Ant Colony Optimization* (ACO) dengan minimax dalam pengambilan keputusan strategi dalam game. Tujuannya menilai efektifitas kedua algoritma dalam menghasilkan keputusan optimal dengan waktu komputasi yang efisien pada kondisi permainan yang dinamis. Persamaan dengan penelitian ini terdapat pada penggunaan ACO sebagai pendekatan optimasi untuk meningkatkan kecerdasan sistem dalam permainan. Namun, Perbedaan dari penelitian penulis yaitu pada arah penerapan yang dimana penelitian sebelumnya membahas perbandingan dua algoritma untuk menentukan strategi terbaik, sedangkan penelitian penulis hanya menggunakan ACO untuk mengoptimalkan pemilihan karakter dalam game berbasis performa pemain.

Penelitian oleh (Put et al., 2020) mengembangkan model pembelajaran adaptif berbasis dengan menerapkan algoritma *Ant Colony Optimization* (ACO) sebagai metode utama dalam proses optimasi. Tujuannya untuk menciptakan sistem permainan yang mampu menyesuaikan tingkat kesuilatan berdasarkan performa pemain supaya pembelajaran lebih efektif dan menarik. Persamaan dengan peneltian penulis tedapat pada pemanfaatan ACO sebagai mekanisme optimasi adaptif dalam sistem permainan, sementara perbedaannya terlihat pada penyesuaian tingkat kesulitan pembelajaran. Perbedaan dari penelitian sebelumnya lebih fokus pada penyesuaian tingkat kesulitan pembelajaran, sedangkan penelitian penulis fokus pada pengembangan game adapting yang mengoptimalkan proses pemilihan karakter berdasarkan performa pemain.

Penelitian oleh (Rodríguez & Puig, 2022) mengusulkan pendekatan adaptive gamification yang menyesuaikan elemen permainan dengan karakteristik dan motivasi pemain melalui penggunaan profil pemain dinamis. Tujuannya untuk meningkatkan kontribusi dan motivasi pemain dengan mengubah tingkat tantangan, hadiah, dan interaksi berdasarkan perilaku pengguna. Persamaan dengan penelitian penulis terlihat pada fokus adaptasi game terhadap perilaku pemain supaya pengalaman permainan lebih personal dan menarik. Namun, perbedaannya tidak menggunakan algoritma optimasi ACO, melainkan hanya menggunakan analisis profil pemain sebagai dasar adaptasi tanpa proses perhitungan optimasi.

Penelitian oleh (Lee et al., n.d.) mengembangkan sistem rekomendasi karakter dalam game Multi*player* Online Battle Arena (MOBA) menggunakan machine learning yang dinamakan Draftrec. Sistemnya dirancang untuk membantu

pemain dalam proses pemilihan karakter secara strategis dengan mempertimbangkan gaya bermain, kerjasama antar karakter dan juga Riwayat kemenangan tim. Model dikembangkan dengan menganalisis lebih dari 280.000 data pertandingan menggunakan pembelajaran mendalam untuk memprediksi kombinasi karakter terbaik. Hasil penelitian menunjukan bahwa sistem rekomendasi ini mampu meningkatkan tingkat kemenangan tim dan mengurangi kesalahan pada pemilihan karakter, serta memberikan pengalaman bermain yang lebih strategis dan efisien. Persamaan dengan penelitian penulis terdapat pada pemilihan karakter berbasis data performa pemain untuk meningkatkan pengalaman bermain. Perbedaan dari penelitian penulis terlihat pada metode yang digunakan, di mana penelitian DratRec menerapkan machine learning dan analisis statistic berdasarkan data historis.

2.2 Game Adaptif

Perkembangkan teknologi dan kecerdasan buatan (*AI*) di industri game sudah membawa perubahan besar terhadap cara pemain berinteraksi dengan sistem permainan. Sekarang ini game tidak hanya sebagai media hiburan saja, melainkan mampu beradaptasi dengan kemampuan dan preferensi pemain secara dinamis. Penerapan adaptif terciptanya kesiambangan antara tantangan dan kemampuan pemain dengan menyesuaikan scenario permainan secara otomatis. Pendekatan adaptif membuat pengalaman bermain menjadi lebih konstekstual, menarik dan unik bagi setiap individu. Dengan adanya kecerdasan buatan, game adaptif mampu menciptakan pengalaman bermain yang lebih realistis, interaktif dan mendalam, sehingga membuat pemain tertantang tanpa merasa bosan selama bermain.

2.2.1 Pengertian Game Adaptif

Game adaptif merupakan permainan yang menyesuaikan elemen-elemen di dalamnya berdasarkan kemampuan, perilaku dan kondisi pemain secara dinamis. Sistem bekerja dengan mengumpulkan data dari aktifitas pemain dari tingkat keberhasilan, waktu penyelesaian dan gaya bermain, lalu dilakukan penyesuaian otomatis terhadap tingkat kesulian, karakter bahkan scenario permainan. Penyesuaian dilakukan secara otomatis melalui analisis data pemain menggunakan machine learning dan data analytics yang memungkinkan sistem membaca dan pemprediksi perilaku pemain secara real time (Rodríguez & Puig, 2022). Dengan adanya sistem adaptif, pemain dapat merasakan pengamalam bermain yang lebih dinamis, menantang dan sesuai dengan karakteristik pemain.

Perkembangan konsep game adaptif juga sejalan dengan meningkatnya peran teknologi dalam kehidupan, setelah masa pandemi COVID-19 telah merubah cara masyarakat berinteraksi dengan teknologi, termasuk dalam bermain game. Game modern tidak hanya dianggap sebagai sarana hiburan semata, tetapi juga menjadi media pembelajaran, komunikasi sosial, dan pengembangan diri di berbagai kalangan. Berdasarkan laporan Global Games Market Report tahun 2021, jumlah pemain game di seluruh dunia telah mencapai lebih dari 3 miliar pengguna, menunjukkan betapa luasnya jangkauan dan pengaruh industri ini dalam kehidupan sehari-hari (Mekni et al., 2022). Perubahan perilaku pemain yang semakin menuntut pengalaman bermain yang realistis dan personal mendorong munculnya konsep game adaptif, di mana sistem permainan dirancang untuk menyesuaikan

tingkat kesulitan, karakter, dan lingkungan berdasarkan interaksi pemain secara langsung (Ruipérez-valiente et al., 2021).

Selain digunakan dalam hiburan, konsep game adaptif juga telah diterapkan di sektor pendidikan dan pelatihan profesional karena kemampuannya menyesuaikan proses pembelajaran dengan kemampuan, kecepatan, dan gaya belajar masing-masing pengguna. Sistem ini memberikan kesempatan bagi pemain atau peserta pelatihan untuk memperoleh pengalaman belajar yang lebih menarik, interaktif, dan tidak membosankan (Ruipérez-valiente et al., 2021). Dengan demikian, game adaptif kini tidak hanya berfungsi sebagai media hiburan semata, tetapi juga berkembang menjadi sarana edukatif dan simulatif yang membantu pengguna beradaptasi terhadap berbagai situasi dan tantangan secara lebih efektif, dinamis, dan fleksibel.

2.2.2 Tujuan Game Adaptif

Secara umum, tujuan dari pengembangan game adaptif untuk menghadirkan pengalaman bermain yang lebih personal, responsif, dan relevan bagi setiap pemain. Sistem ini dirancang agar mampu menyesuaikan berbagai aspek permainan seperti tingkat kesulitan, karakter, serta jalannya cerita berdasarkan kemampuan, gaya bermain, dan preferensi pengguna. Dengan kemampuan adaptasi tersebut, permainan dapat memberikan tingkat tantangan yang seimbang sehingga pemain tetap tertarik dan termotivasi untuk melanjutkan permainan tanpa merasa bosan atau frustasi.

Game adaptif juga bertujuan untuk meningkatkan keterlibatan pemain melalui interaksi yang lebih dinamis dan realistis (Shabadurai et al., 2024). Sistem adaptif memungkinkan permainan bereaksi terhadap tindakan pemain secara langsung, menciptakan pengalaman yang terasa hidup dan unik bagi setiap individu. Secara lebih luas, penerapan game adaptif tidak hanya terbatas pada dunia hiburan, tetapi juga mulai dimanfaatkan dalam bidang pendidikan, pelatihan, dan simulasi (Troussas et al., 2025). Melalui kemampuan menyesuaikan konten dengan kebutuhan pengguna, game adaptif dapat membantu proses pembelajaran dan pengembangan keterampilan dengan cara yang lebih efektif, kontekstual, serta menyenangkan.

2.2.3 Sistem Adaptif Hybrid dalam Game

Elemen adaptasi yang dikembangkan dalam penelitian ini adalah sistem hybrid yang merujuk pada penggabungan dua atau lebih pendekatan untuk menciptakan sistem yang lebih komprehensif. Proses adaptasi ini memanfaatkan data interaksi pemain seperti tingkat keberhasilan, waktu reaksi, serta pola pengambilan keputusan selama permainan berlangsung. Data tersebut kemudian dianalisis menggunakan pendekatan hybrid yang menggabungkan *Ant Colony Optimization* (ACO) untuk optimasi pemilihan karakter dan *Rule-Based* Adaptive System untuk penyesuaian tingkat kesulitan *enemy* secara real-time. Pendekatan ini membuat sistem pemilihan karakter dan *enemy* menjadi lebih cerdas dan dinamis dibandingkan metode konvensional yang hanya mengandalkan pilihan manual atau parameter statis (Dorigo & Stützle, 2020).

Implementasi metode ACO dalam konteks ini bekerja dengan cara meniru perilaku semut dalam mencari rute paling efisien untuk mencapai sumber makanan. Dalam game, setiap "semut" merepresentasikan kemungkinan pilihan karakter, dan algoritma secara iteratif memperkuat jalur terbaik berdasarkan evaluasi performa pemain. Hasilnya, sistem dapat secara otomatis menyesuaikan rekomendasi karakter sesuai dengan kemampuan pengguna tanpa memerlukan input tambahan. Pendekatan ini tidak hanya meningkatkan efisiensi sistem adaptasi, tetapi juga memberikan pengalaman bermain yang lebih personal, menantang, dan relevan bagi setiap individu (Lee et al., n.d.).

Selain pemilihan karakter, integrasi sistem analisis performa berbasis data real-time juga menjadi elemen penting dalam pengembangan game adaptif ini. Data yang dikumpulkan dari gameplay, seperti tingkat keberhasilan misi, waktu penyelesaian, dan kesalahan yang dilakukan, digunakan untuk menyesuaikan tingkat kesulitan serta perilaku karakter musuh dalam permainan. Dengan cara ini, sistem tidak hanya menyesuaikan karakter pemain, tetapi juga dapat menciptakan keseimbangan antara tantangan dan kemampuan pemain yang terus berubah (Rodríguez & Puig, 2022).

2.2.4 Elemen dalam Game Adaptif

Sumber data utama dalam penelitian ini berasal dari aktivitas pemain selama bermain game, yang secara otomatis direkam oleh sistem melalui fitur event tracking dan real-time logging pada engine game. Data ini diambil dari indikator performa seperti jumlah skor, durasi waktu penyelesaian misi, jumlah kekalahan

atau kehilangan nyawa, dan tingkat keberhasilan dalam menyelesaikan tantangan (Lee et al., n.d.). Sistem ini juga merekam pola perilaku pemain, seperti bermain secara agresif, defensif, atau seimbang, untuk digunakan sebagai dasar penyesuaian karakter berikutnya (Rodríguez & Puig, 2022).

2.3 Ant Colony Optimization (ACO)

Algoritma *Ant Colony Optimization* (ACO) merupakan salah satu metode *metaheuristik* yang digunakan untuk menyelesaikan berbagai permasalahan optimasi yang kompleks, termasuk pada pengembangan sistem adaptif dalam game. Algoritma ini terinspirasi dari perilaku koloni semut dalam menemukan jalur terpendek menuju sumber makanan dengan meninggalkan jejak kimia yang disebut *pheromone* sebagai penanda jalur terbaik. Dalam penelitian ini, ACO dimanfaatkan untuk membantu proses pemilihan karakter yang optimal berdasarkan performa pemain selama bermain. Melalui proses iteratif yang menyerupai perilaku semut, algoritma ini mengevaluasi dan memperkuat solusi terbaik secara bertahap. Dengan ini, sistem dapat menyesuaikan rekomendasi karakter secara dinamis sesuai dengan data performa pemain (Blum, 2024).

2.3.1 Pengertian Ant Colony Optimization

Dalam bidang komputasi, algoritma *Ant Colony Optimization* **(ACO)** bekerja dengan merepresentasikan semut sebagai agen cerdas yang menjelajahi berbagai kemungkinan solusi dari suatu permasalahan. Setiap agen atau semut akan memilih jalur berdasarkan dua faktor utama, yaitu nilai *pheromone* yang

menandakan seberapa baik jalur tersebut digunakan sebelumnya, serta nilai *heuristik* yang menggambarkan informasi lokal atau pengetahuan tentang kualitas solusi yang sedang dicari. Melalui proses pembaruan *pheromone* yang dilakukan secara berulang, algoritma ini mampu menemukan solusi yang paling optimal secara bertahap.

Seiring dengan perkembangan teknologi dan meningkatnya permasalahan optimasi, *Algoritma Ant Colony Optimization* (ACO) terus mengalami perkembangan. Awalnya diterapkan pada *Travelling Salesman Problem (TSP)*, ACO kini banyak digunakan di berbagai bidang seperti penjadwalan, perencanaan rute, hingga sistem cerdas berbasis data (Dorigo & Stützle, 2020). Dalam konteks modern, algoritma ini juga mulai dimanfaatkan dalam pengembangan *game adaptif* untuk menyesuaikan sistem permainan berdasarkan perilaku dan performa pemain secara real-time (Science, 2022).

2.3.2 Rumus Ant Colony Optimization

Algoritma *Ant Colony Optimization* (ACO) bekerja berdasarkan dua proses utama, yaitu pemilihan jalur solusi dan pembaruan pheromone. Proses ini didasarkan pada interaksi antarsemut buatan yang secara kolektif mencari solusi terbaik melalui mekanisme umpan balik positif (*positive feedback*). Setiap semut akan memilih jalur atau solusi berdasarkan probabilitas tertentu yang dipengaruhi oleh intensitas pheromone (τ) dan nilai *heuristik* (η).

2.3.2.1 Inisialisasi Semut dan Parameter

Pada tahap awal, ditentukan jumlah semut (m) dan jumlah karakter (n) yang akan menjadi kandidat solusi. Setiap semut merepresentasikan proses pencarian kombinasi karakter terbaik berdasarkan data performa pemain. Nilai pheromone awal (τ_0) diberikan secara acak atau seragam untuk setiap karakter. Parameter utamanya adalah pengaruh intensitas pheromone α (alpha), pengaruh informasi *heuristik* β (beta), tingkat penguapan pheromone ρ (rho), jumlah iterasi maksimum t_{max}.

Rumus inisialisasi pheromone:

$$\tau_{ij}(0) = \tau_0 \tag{2.1}$$

Nilai awal τ_0 biasanya diatur kecil, misalnya 0.1–1.0, untuk memberikan peluang eksplorasi di awal iterasi (Dorigo & Stützle, 2020).

2.3.2.2 Fungsi Heuristik

Nilai *heuristik* (η_{ij}) digunakan untuk menilai kualitas atau kelayakan karakter berdasarkan kriteria performa pemain. konteks penelitian ini, *heuristik* dihitung dari beberapa parameter. Nilai *heuristik* bisa dirumuskan seperti ini:

$$\eta_{ij} = \frac{D_i + S_i + A_i}{T_i} \tag{2.2}$$

Semakin tinggi nilai η_{ij}, semakin baik performa karakter tersebut, sehingga peluangnya untuk dipilih meningkat.

2.3.2.3 Probabilitas Pemilihan Karakter

Setiap semut akan memilih karakter berikutnya berdasarkan probabilitas kombinasi antara pheromone dan nilai *heuristik*. Rumus probabilitasnya:

$$P_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \in \text{allowed}} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} \tag{2.3}$$

Semakin tinggi nilai pheromone dan *heuristik* suatu karakter, semakin besar kemungkinan karakter tersebut dipilih untuk iterasi berikutnya. Dalam penelitian ini, sistem pemilihan ini berfungsi untuk menentukan karakter adaptif yang paling sesuai dengan gaya bermain pemain (Angeles-Garcia et al., 2023).

2.3.2.4 Pembaruan Pheromone

Setelah seluruh semut menyelesaikan proses pemilihan karakter, nilai pheromone diperbarui berdasarkan hasil evaluasi. Pheromone pada karakter terbaik akan diperkuat, sedangkan pada karakter kurang optimal akan diuapkan sebagian. Pembaruan pheromone dilakukan menggunakan rumus.

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij} \tag{2.4}$$

Dengan rumus dibawah ini:

$$\Delta\tau_{ij} = \begin{cases} \frac{Q}{L_k}, & \text{jika jalur } k \text{ digunakan} \\ 0, & \text{lainnya} \end{cases} \tag{2.5}$$

Hal ini Q adalah konstanta, dan L_k merepresentasikan hasil performa karakter yang dipilih semut ke-k. Proses ini memastikan bahwa karakter dengan

performa baik akan mendapat penguatan pheromone lebih tinggi (Put & Elbert, 2020).

2.3.2.5 Iterasi dan Konvergasi

Tahap terakhir pada proses ACO melakukan iterasi secara berulang sampai nilai pheromone yang terbentuk antar karakter mencapai kondisi stabil atau tidak mengalami perubahan. Proses ini akan berhenti jika jumlah iterasi maksimum telah tercapai atau hasil pemilihan karakter tidak menunjukkan perbedaan signifikan. Pada tahap ini dihasilkan karakter yang dianggap paling optimal dan sesuai dengan gaya bermain pemain, sehingga sistem dalam game dapat memberikan pengalaman bermain yang lebih adaptif dan personal bagi setiap pengguna.

2.4 *Rule-Based Adaptive System*

Rule-Based Adaptive System adalah sistem yang menggunakan aturan (rules) yang sudah didefinisikan sebelumnya untuk membuat keputusan adaptif secara otomatis berdasarkan kondisi yang terjadi. Sistem bekerja dengan pola *IF-THEN* jika kondisi terpenuhi. Dalam game adaptif, *Rule-Based* System digunakan untuk menyesuaikan parameter *enemy* secara *real-time* berdasarkan evaluasi peforma pemain (Fisher & Kulshreshth, 2024).

2.4.1 Pengertian Rule-Based Statis

Rule-Based Statis merupakan pendekatan sistem berbasis aturan di mana seluruh kondisi *IF-THEN* didefinisikan secara tetap oleh pengembang sebelum sistem dijalankan dan tidak berubah selama runtime berlangsung. Setiap aturan

dalam sistem ini bersifat *deterministik*, artinya untuk input yang sama akan selalu menghasilkan output yang sama sehingga keputusan sistem dapat diprediksi dan diverifikasi secara langsung tanpa memerlukan proses pelatihan data.

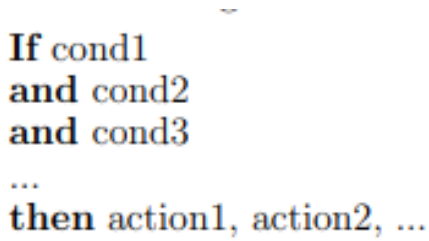
Pendekatan *Rule-Based* Statis berbeda dengan pendekatan berbasis data (*machine learning*) yang memerlukan dataset besar untuk pelatihan. Dalam *Rule-Based* Statis, semua pengetahuan tentang aturan keputusan sudah diketahui di awal dan dinyatakan secara eksplisit oleh domain expert. Hal ini menjadikan pendekatan *Rule-Based* Statis sangat cocok untuk sistem yang memiliki ambang batas (threshold) yang jelas dan logika keputusan yang transparan (Sasikumar et al., n.d.).

Pendekatan ini banyak diterapkan dalam pengembangan game adaptif karena transparansinya yang tinggi dan kemudahannya dalam proses debugging serta validasi sistem. Setiap keputusan yang diambil sistem dapat dijelaskan dengan mudah kepada pengguna melalui penjelasan dari aturan mana yang diaktifkan (Mortazavi et al., 2024).

2.4.2 Model Formal Production Line

Dalam sistem pakar, setiap pengetahuan yang dimiliki oleh domain expert direpresentasikan secara eksplisit dalam bentuk *production rule* sebuah format standar yang telah digunakan secara luas di berbagai sistem berbasis aturan. Sasikumar et al. (2007) menjelaskan bahwa sebuah *production rule* selalu disusun dalam struktur baku yang terdiri atas dua bagian utama, yaitu bagian *IF* yang disebut sebagai *antecedent* (premis atau kondisi), dan bagian *THEN* yang disebut

sebagai *consequent* (kesimpulan atau aksi). Secara formal, struktur tersebut dituliskan sebagai:



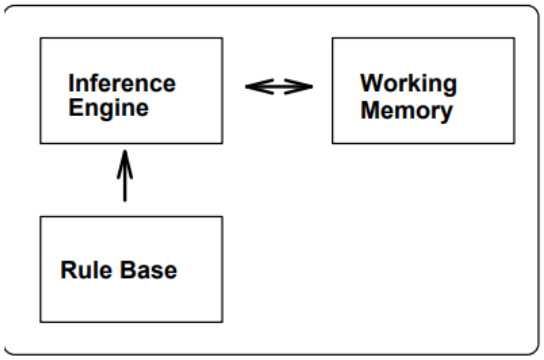
Gambar 2. 1 Bentuk Umum *Rule-Based* (Sasikumar et all, 2007)

Bagian *antecedent* berisi satu atau lebih kondisi yang harus dipenuhi berdasarkan fakta-fakta yang tersimpan di dalam *Working Memory*. Kondisi-kondisi ini dapat berupa perbandingan numerik, status tertentu, atau relasi logis lain yang dapat dievaluasi kebenarannya. Apabila terdapat lebih dari satu kondisi dalam antecedent, maka kondisi-kondisi tersebut dapat dirangkai menggunakan operator logika *AND* (semua kondisi harus benar) atau *OR* (cukup salah satu kondisi yang benar). Sebagai contoh, aturan R-01 pada sistem penelitian ini memiliki antecedent yang terdiri dari tiga kondisi yang dihubungkan dengan operator *OR*, sehingga aturan akan terpicu jika minimal satu dari ketiga kondisi tersebut terpenuhi (Makolo et al., 2023). Penggunaan operator ini memberikan fleksibilitas pada perancang untuk menentukan seberapa ketat atau longgarnya persyaratan pemicu sebuah aturan.

2.4.3 Komponen Sistem *Rule-Based Statis*

Rule-Based System dibangun di atas tiga komponen utama yang bekerja secara terintegrasi. Komponen pertama adalah *Rule-Based* yang menyimpan

seluruh aturan dalam format IF-THEN, komponen kedua adalah *Working Memory* yang menampung data aktual hasil rekaman secara real-time, dan komponen ketiga adalah *Inference Engine* yang mencocokkan fakta di *Working Memory* dengan aturan di *Rule-Based* untuk menentukan tindakan yang akan dijalankan. Ketiga komponen ini tidak berdiri sendiri-sendiri, melainkan saling berinteraksi dalam satu siklus komputasi berkelanjutan yang memungkinkan sistem merespon perubahan data secara langsung dan otomatis tanpa campur tangan pengembang saat runtime (Lucas & Gaag, 1991; Sasikumar et al., n.d.).



Gambar 2. 2 Komponen *Rule-Based System* (Sasikumar et all, 2007)

Komponen pertama yaitu *Rule-Based*, *Rule-Based* berfungsi sebagai repositori seluruh aturan *IF-THEN* yang telah didefinisikan oleh perancang sebelum sistem dijalankan. Aturan-aturan ini merepresentasikan logika keputusan yang harus diambil berdasarkan kondisi tertentu yang mungkin terjadi selama permainan. Dalam penelitian ini, *Rule-Based* menampung sejumlah aturan statis yang menetapkan hubungan antara data performa pemain dan aksi penyesuaian parameter permainan. Seluruh aturan di dalam *Rule-Based* bersifat tetap dan tidak mengalami perubahan selama runtime. Modifikasi hanya dapat dilakukan oleh

pengembang pada tahap desain, sehingga konsistensi dan prediktabilitas aturan tetap terjaga.

Kedua *Working Memory*, area penyimpanan sementara yang menampung fakta-fakta aktual hasil rekaman sistem secara real-time. Fakta-fakta ini mencerminkan kondisi terkini dari lingkungan permainan dan aktivitas pemain. Data di dalam *Working Memory* terus diperbarui sepanjang permainan berlangsung, sehingga selalu menyajikan informasi terbaru yang siap digunakan oleh mesin inferensi. Dengan demikian, *Working Memory* menjadi sumber data primer yang menggambarkan dinamika permainan secara langsung tanpa memerlukan proses pengolahan tambahan.

Terakhir *Inference Engine* berperan sebagai otak sistem yang menjembatani *Rule-Based* dan *Working Memory*. Tugas utamanya adalah mencocokkan fakta di *Working Memory* dengan kondisi pada aturan di *Rule-Based* menggunakan teknik forward chaining. Proses pencocokan dilakukan secara berurutan, mengevaluasi setiap aturan hingga ditemukan yang memenuhi syarat. Setelah aturan yang sesuai teridentifikasi, *Inference Engine* mengeksekusi aksi yang telah ditentukan tanpa intervensi tambahan. Dengan mekanisme ini, *Inference Engine* menentukan langkah adaptif yang tepat berdasarkan data aktual, sehingga sistem dapat mengambil keputusan secara mandiri.

2.4.4 Forward Chaining

Strategi inferensi yang digunakan dalam penelitian ini adalah *forward chaining*, yaitu pendekatan berbasis data (*data-driven*). Dalam forward chaining,

sistem memulai proses dari fakta-fakta yang telah tersedia di *Working Memory* dan secara progresif menerapkan aturan-aturan yang relevan hingga sebuah kesimpulan tercapai atau tidak ada lagi aturan yang dapat dieksekusi (Sasikumar et al., n.d.). Alur kerjanya berlangsung secara siklik: pertama, *Inference Engine* membaca data terkini dari *Working Memory*. Selanjutnya, engine mengevaluasi antecedent setiap aturan di *Rule-Based* secara berurutan, dimulai dari R-01. Apabila seluruh kondisi pada antecedent suatu aturan terpenuhi, aturan tersebut langsung dieksekusi dan consequent-nya dijalankan. Jika antecedent tidak terpenuhi, evaluasi berlanjut ke aturan berikutnya (R-02, lalu R-03) hingga ditemukan aturan yang cocok. Keunggulan utama pendekatan ini adalah sistem mampu merespons perubahan data pemain secara langsung tanpa perlu menunggu tercapainya goal tertentu, sehingga penyesuaian difficulty dapat terjadi secara real-time setiap kali pemain menyelesaikan atau mengulang sebuah stage

2.4.5 Rule Matching

Rule matching adalah proses menyeluruh untuk menentukan aturan mana di dalam *Rule-Based* yang dapat dieksekusi berdasarkan fakta-fakta yang ada di *Working Memory*. Proses ini terdiri atas empat tahapan utama. Tahap pertama adalah *pattern matching*, yaitu memeriksa apakah kondisi-kondisi yang tertulis pada antecedent suatu aturan cocok dengan data yang tersimpan di *Working Memory*. Selanjutnya, *variable binding* dilakukan dengan menetapkan nilai konkret dari fakta ke variabel-variabel yang terdapat dalam aturan, sehingga kondisi dapat dievaluasi lebih lanjut. Setelah variabel terikat, tahap *condition*

evaluation mengecek apakah setiap kondisi bernilai benar (*true*) atau salah (*false*). Apabila seluruh kondisi bernilai benar, aturan tersebut dimasukkan ke dalam *conflict set* sebagai kandidat eksekusi

BAB III

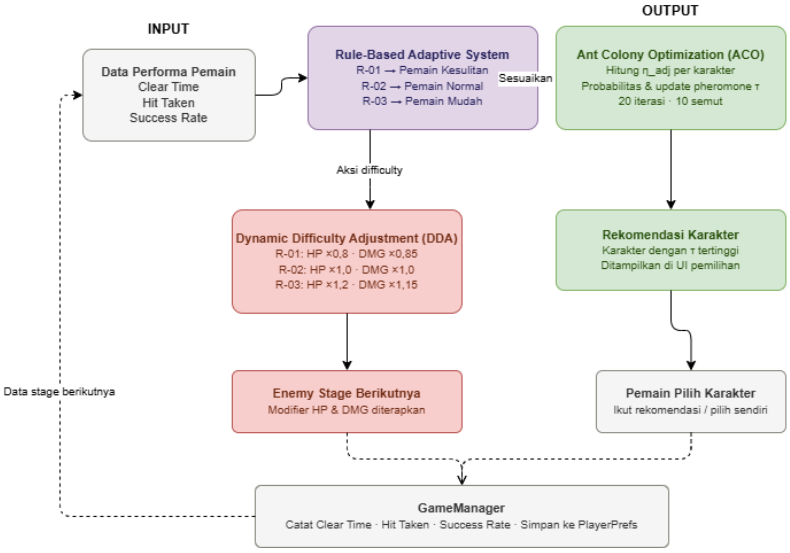
DESAIN DAN IMPLEMENTASI

3.1 Desain Penelitian

Rancangan penelitian mencakup perancangan sistem, struktur data yang digunakan, perancangan algoritma ACO, Perancangan algoritma Rule Based, rancangan implementasi, serta rancangan pengujian yang akan dilakukan untuk mengevaluasi hasil penelitian. Seluruh aspek ini disusun untuk memberikan gambaran komprehensif mengenai bagaimana penelitian dilaksanakan secara terstruktur dan sistematis.

3.1.1 Gambaran Sistem

Sistem pemilihan karakter adaptif pada game *The Night Dreamers* dibangun dengan mengintegrasikan dua komponen utama, yaitu *Rule-Based Adaptive System* dan *Ant Colony Optimization* (ACO). Kedua komponen ini bekerja secara sinergis dalam kerangka *Decision Support System* (DSS) untuk menghasilkan rekomendasi karakter yang adaptif berdasarkan performa pemain. Blok diagram sistem secara keseluruhan ditunjukkan pada Gambar 3.1.



Gambar 3. 1Blok Diagram Sistem

Berdasarkan Gambar 3.1, alur kerja sistem dimulai dari pengambilan data performa pemain (*Clear Time, Hit Taken, Success Rate*) setelah setiap stage selesai. Data tersebut diproses oleh *Rule-Based Adaptive System* untuk dua tujuan sekaligus: menyesuaikan parameter enemy pada stage berikutnya melalui mekanisme *Dynamic Difficulty Adjustment* (DDA), dan memodifikasi bobot heuristik karakter sebagai masukan bagi ACO. Selanjutnya ACO melakukan optimasi probabilistik untuk menentukan karakter dengan kecocokan tertinggi terhadap kondisi pemain, yang kemudian direkomendasikan melalui antarmuka pemilihan karakter.

3.1.2 Pembuatan Game The Night Dreamers

Pembuatan *The Night Dreamers* dilakukan dengan menggunakan Unity Engine 6. Unity dipilih karena menyediakan ekosistem pengembangan yang

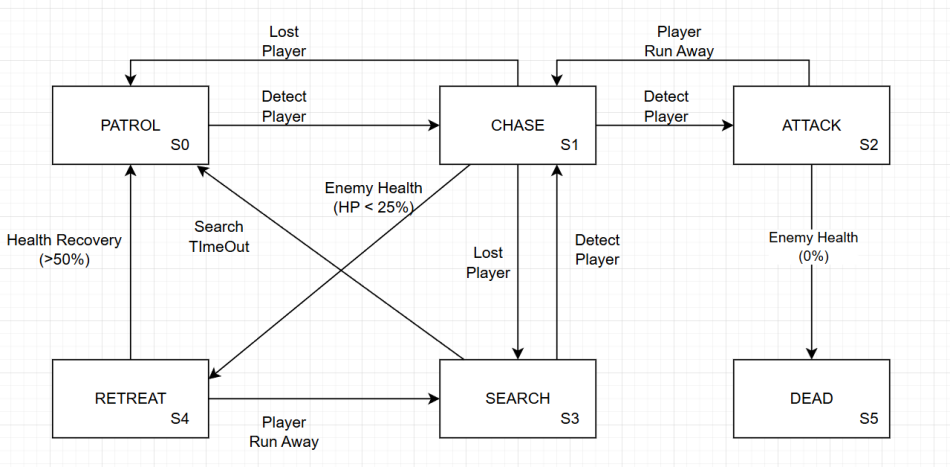
lengkap untuk game 3D berbasis PC, termasuk dukungan untuk integrasi skrip C# yang digunakan dalam implementasi algoritma ACO dan *Rule-Based Adaptive System*. Dalam pengembangannya, digunakan pendekatan *Finite State Machine* (FSM) untuk mengelola status atau state dari setiap karakter yang dikomputasi, menciptakan struktur game yang lebih teratur dan dinamis.

Gameplay untuk Game "*The Night Dreamers*" diawali pada saat player menekan icon game sehingga sistem permainan dapat dimulai. Setelah permainan berjalan, scene akan berpindah menuju cuplikan opening cutscene yang berisi cerita prolog asal muasal player berada di dunia mimpinya. Setelah cuplikan selesai, scene akan berganti untuk menampilkan *Main Menu* yang berisikan opsi *Start, Continue, Option*, dan *Exit*.

Setelah pemain memilih opsi Start dan permainan berlangsung, pemain akan masuk pada tahap awal permainan dengan kondisi state Player Idle. Apabila pemain menekan tombol WASD, state akan berubah menjadi Player Move. Pemain dapat menyerang enemy, serta menyelesaikan stage untuk melanjutkan ke level berikutnya.

3.1.3 Rancangan Finate State Machine

Selain gameplay pemain, *The Night Dreamers* juga menggunakan *Finite State Machine* (FSM) pada enemy untuk mengatur perilaku musuh terhadap kondisi pemain. FSM pada enemy memiliki enam state utama, yaitu Patrol, Chase, Attack, Search, Retreat, dan Dead. Diagram FSM enemy dapat dilihat pada Gambar 3.3



Gambar 3. 2 Diagram *Finite State Machine enemy*

Pada kondisi awal, enemy berada dalam state Patrol (S0), di mana musuh bergerak mengikuti rute patroli yang telah ditentukan. Apabila enemy mendeteksi keberadaan pemain dalam jarak tertentu, state berpindah menjadi Chase (S1). Jika jarak enemy dan pemain berdekatan, state berpindah menjadi Attack (S2). Apabila pemain berhasil melarikan diri, state berpindah ke Search (S3) selama 8 detik sebelum kembali ke Patrol. State Retreat (S4) aktif ketika HP enemy di bawah 25%, dan state Dead (S5) aktif ketika HP mencapai nol.

3.1.4 Rancangan User Interface

Untuk meningkatkan interaksi antara pemain dengan permainan, diperlukan user interface (UI) yang mudah dimengerti. Pada game The Night Dreamers terdapat beberapa komponen UI utama, yaitu Main Menu, Choose Character, dan HUD Player.

a. Main Menu

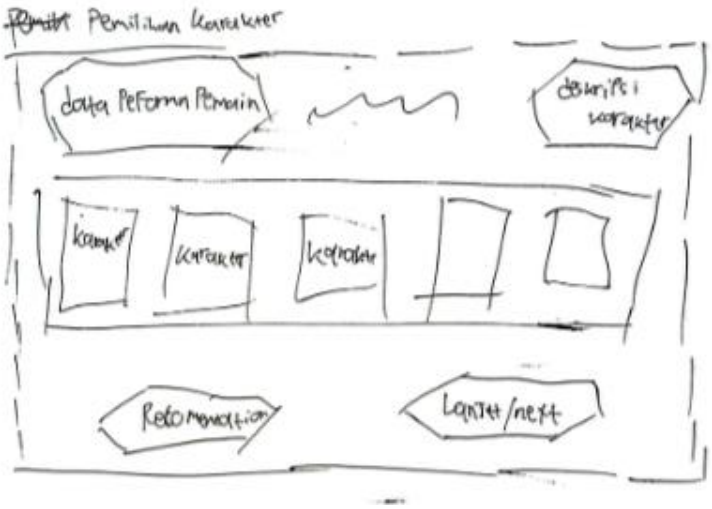
Halaman Main Menu dirancang sebagai antarmuka awal yang menjembatani pemain sebelum memasuki sirkuit permainan utama. Komponen Start berfungsi untuk menginisialisasi variabel state baru dan memulai permainan dari stage pertama dilihat pada gambar 3.3.

Gambar 3. 3 Sketsa *MainMenu*

Pada gambar Tombol Load terintegrasi dengan sistem penyimpanan lokal (save file) untuk memuat progres permainan terakhir yang disimpan oleh pemain. Sementara itu, menu Setting menyediakan opsi konfigurasi performa visual dan audio untuk kenyamanan bermain, dan tombol Quit digunakan untuk mengeksekusi fungsi keluar dari aplikasi permainan secara aman.

b. Choose Character

Halaman pemilihan karakter tidak hanya menampilkan visualisasi aset tujuh karakter, melainkan juga mengintegrasikan kalkulasi sistem cerdas secara langsung.

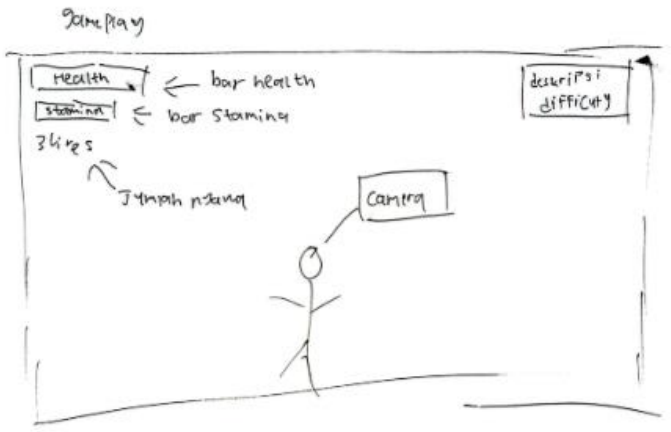


Gambar 3. 4 Sketsa Pemilihan Karakter

Setiap karakter memiliki representasi atribut dasar yang unik (seperti Speed, Defense, atau Attack). Di samping atribut tersebut, sistem menampilkan nilai heuristik yang bersifat dinamis. Nilai ini merupakan representasi numerik hasil komputasi algoritma Ant Colony Optimization (ACO) yang melacak dan menganalisis rekam jejak performa serta kecenderungan gaya bermain pemain pada stage sebelumnya. Karakter dengan nilai heuristik tertinggi menunjukkan rekomendasi optimal yang disesuaikan oleh sistem untuk menyeimbangkan atau mengadaptasi tingkat kesulitan permainan berikutnya.

c. Gameplay

Implementasi antarmuka saat permainan berlangsung (HUD - Heads-Up Display) difokuskan pada penyampaian informasi vital pemain secara real-time.

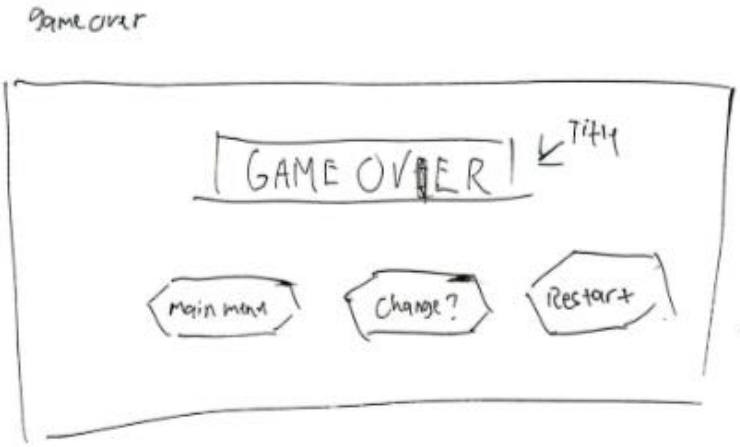


Gambar 3. 5 Sketsa *GamePlay*

Informasi Health (HP) ditampilkan dalam bentuk bar visual untuk menunjukkan sisa daya tahan karakter terhadap serangan musuh atau rintangan. Indikator Stamina merepresentasikan kapasitas energi yang terkuras saat karakter melakukan aksi tertentu (seperti berlari cepat atau menghindar) dan akan terisi kembali secara otomatis saat karakter dalam posisi statis. Sedangkan indikator Live menunjukkan sisa kesempatan atau nyawa cadangan yang dimiliki pemain sebelum memicu kondisi Game Over. Seluruh data pada HUD ini menjadi parameter input yang terus dipantau oleh mesin aturan (Rule-Based DDA) untuk menyesuaikan intensitas permainan secara adaptif.

d. GameOver

GameOver menggambarkan kondisi akhir permainan (state transition) ketika variabel Health atau Live pemain telah mencapai nilai nol sebelum berhasil menyelesaikan objektif stage.



Gambar 3. 6 Sketsa *GameOver*

Halaman ini berfungsi sebagai interupsi alur permainan sekaligus memberikan evaluasi akhir bagi pemain. Antarmuka ini menyediakan dua pilihan kendali utama: tombol Retry untuk memicu fungsi reset yang mengembalikan pemain ke titik awal stage saat ini dengan kondisi atribut yang telah dipulihkan, dan tombol Main Menu untuk memutuskan sesi permainan aktif dan mengalihkan navigasi kembali ke halaman utama game.

3.1.5 Definisi dan Deskripsi Karakter

The Night Dreamers terdapat tujuh karakter yang dapat dipilih oleh pemain. Setiap karakter memiliki kepribadian, gaya bermain, dan nilai atribut yang berbeda-beda. Karakter ini dirancang untuk memenuhi berbagai gaya bermain, mulai dari pemain yang agresif hingga pemain yang hati-hati. Sistem ACO akan merekomendasikan karakter yang paling sesuai berdasarkan data performa pemain selama stage berlangsung. Definisi selengkapnya disajikan pada Tabel 3.1.

Tabel 3. 1Profil, Kepribadian, dan Gaya Bermain Tujuh Karakter Game

No	Nama	Kepribadian	Gaya Bermain
K1	Pemberani (Brave)	Berani, agresif, maju terus tanpa takut risiko	Menyerang langsung, tidak bertahan
K2	Penakut (Coward)	Mudah panik, menghindari konfrontasi	Lari dan menghindar dari musuh
K3	Pemalu (Shy)	Diam-diam kuat, menyerang dari jauh	Serangan jarak jauh, presisi tinggi
K4	Penjaga (Guardian)	Setia, protektif, mengutamakan pertahanan	Bertahan dan menerima serangan
K5	Seimbang (Balanced)	Tenang, fleksibel, beradaptasi cepat	All-rounder, cocok berbagai situasi
K6	Petualang (Explorer)	Penasaran, energik, selalu bergerak	Menjelajah cepat, hindari musuh
K7	Bijaksana (Wise)	Hati-hati, strategis, perhitungan matang	Strategi perlahan, serangan terukur

Pada Tabel 3.1 bisa dilihat, karakter K1 (Pemberani) memiliki sifat agresif dengan Attack tertinggi (85), namun Defense rendah (40) sehingga optimal untuk pemain dengan Clear Time singkat dan Hit Taken kecil. K2 (Penakut) mengandalkan Speed maksimal (90) untuk menghindari serangan langsung, cocok untuk pemain dengan Hit Taken tinggi. K3 (Pemalu) unggul di Accuracy (90) dengan pendekatan serangan jarak jauh berpresisi tinggi. K4 (Penjaga) menonjol dengan Defense (90) dan Recovery (80) tertinggi, ideal untuk pemain yang sering menerima serangan. K5 (Seimbang) merupakan all-rounder dengan semua atribut bernilai 65. K6 (Petualang) mengandalkan Speed (85) dan Stability (70) untuk mobilitas eksplorasi. K7 (Bijaksana) memiliki kombinasi Accuracy (80) dan Defense (75) yang baik untuk strategi terukur.

3.1.6 Pengumpulan Data Game

Pengumpulan data performa pemain adalah tahap awal dalam sistem pemilihan karakter adaptif. Game secara otomatis mencatat aktivitas pemain selama bermain menggunakan mekanisme *event tracking* dan *realtime logging* melalui komponen DataLogger yang terintegrasi di dalam Unity Engine. Tiga parameter utama yang dikumpulkan adalah sebagai berikut:

a. Clear Time

Clear Time merupakan waktu yang dibutuhkan pemain untuk menyelesaikan satu stage permainan. Parameter ini digunakan untuk mengechek dan melihat apakah pemain bermain dengan strategi cepat atau bermain aman. Pemain dengan Clear Time yang cepat cenderung cocok dengan karakter bertipe ofensif atau cepat.

b. Hit Taken

Hit Taken merupakan jumlah serangan enemy yang mengenai pemain selama satu stage berlangsung. Nilai Hit Taken yang tinggi mengindikasikan pemain mengalami kesulitan dalam menghindari serangan, sehingga sistem akan mempertimbangkan karakter dengan atribut Defense dan Recovery yang lebih tinggi.

c. Succes Rate

Success Rate merupakan persentase keberhasilan pemain dalam menyelesaikan objektif misi di setiap stage. Nilai *Success Rate* rendah menunjukkan pemain sering gagal menyelesaikan misi, sehingga sistem akan merekomendasikan karakter yang lebih mudah digunakan atau lebih defensive.

Seluruh data dicatat dan disimpan otomatis setiap pemain menyelesaikan stage. Data yang terkumpul akan digunakan sebagai input pada proses klasifikasi *Rule-Based* dan perhitungan heuristik pada algoritma ACO.

3.1.7 Rule-Based Adaptive System

Rule-Based Adaptive System membaca data performa pemain (*Clear Time, Hit Taken, Success Rate*) setelah setiap stage selesai dan menentukan penyesuaian tingkat kesulitan enemy untuk stage berikutnya. Proses ini memastikan bahwa tantangan yang dihadapi pemain selalu seimbang dengan kemampuannya. Sistem menggunakan tiga aturan utama seperti yang ditunjukkan pada Tabel 3.2.

Tabel 3. 2 Aturan Penyesuaian Tingkat Kesulitan

Rule	Kondisi IF	Clear Time	Hit Taken	Success Rate	THEN Difficulty
R-01	Pemain Kesulitan	> 90 detik	≥ 8 kali	< 60%	**TURUN**
R-02	Pemain Normal	60–90 detik	3–7 kali	60%–80%	**TETAP**
R-03	Pemain Mudah	< 60 detik	≤ 2 kali	> 80%	**NAIK**

R-01 (Pemain Kesulitan) aktif ketika *Clear Time* > 90 detik atau *Hit Taken* ≥ 8 kali atau *Success Rate* < 60%. Kondisi ini menandakan pemain mengalami kesulitan signifikan. Penetapan ambang 90 detik dipilih karena melebihi 1,5 kali durasi *normal* stage (60 detik), mengindikasikan pemain tidak mampu bergerak efisien. Ketika kondisi ini terpenuhi, HP *enemy* diturunkan 20%, damage enemy diturunkan 15%, dan jumlah *enemy* dikurangi 1.

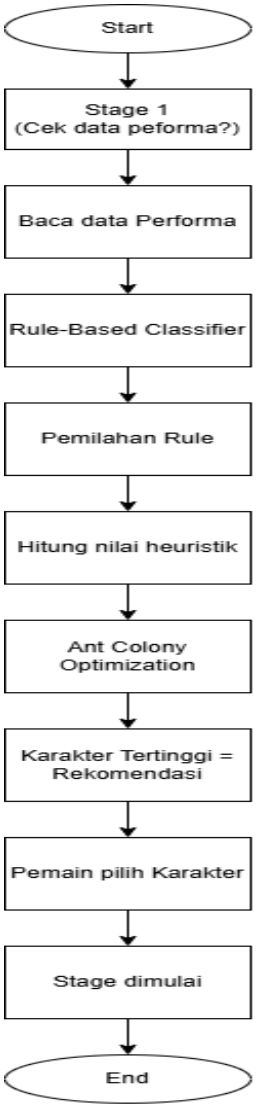
R-02 (Pemain Normal) aktif ketika *Clear Time* berada di rentang 60–90 detik dan *Hit Taken* antara 3–7 kali dan *Success Rate* antara 60%–80%. Rentang ini

mencerminkan zona keseimbangan optimal di mana pemain cukup tertantang tetapi tidak frustasi. Seluruh parameter enemy tidak mengalami perubahan.

R-03 (Pemain Mudah) aktif ketika *Clear Time* < 60 detik, *Hit Taken* ≤ 2 kali, dan *Success Rate* > 80%. Kondisi ini mengindikasikan pemain sangat mahir dan berisiko mengalami kebosanan. HP *enemy* dinaikkan 20%, *damage* dinaikkan 15%, dan jumlah *enemy* bertambah 1 unit.

3.1.8 Alur Sistem Pemilihan Karakter Adaptif

Alur sistem pemilihan karakter adaptif menggambarkan urutan proses dari awal permainan hingga dihasilkannya rekomendasi karakter kepada pemain. Alur ini mencakup seluruh tahapan mulai dari pengambilan data performa, inisialisasi parameter ACO, normalisasi data, perhitungan heuristik, hingga iterasi optimasi dan penentuan karakter terbaik. Flowchart alur sistem ditunjukkan pada Gambar 3.8.



Gambar 3. 7 Flowchart Alur Sistem

Pada gambar 3.3, pemain memulai permainan untuk pertama kalinya, sistem memeriksa keberadaan data performa dari stage sebelumnya yang tersimpan di dalam *PlayerPrefs*. Apabila data belum tersedia, pemain dapat memilih karakter secara bebas tanpa rekomendasi dari algoritma ACO.

Setelah stage diselesaikan, GameManager mencatat tiga parameter performa secara otomatis, yaitu *Clear Time*, *Hit Taken*, dan *Success Rate*, yang

kemudian disimpan melalui fungsi penyimpanan data stage sebelum perpindahan *scene* berlangsung.

Pada *scene* pemilihan karakter, sistem membaca data tersebut dan menjalankan klasifikasi berbasis aturan untuk menentukan aturan aktif berdasarkan evaluasi terhadap nilai ambang batas yang telah ditetapkan. Aturan aktif yang terpilih kemudian digunakan oleh mekanisme DDA untuk menetapkan modifier HP dan DMG musuh pada stage berikutnya, sekaligus menyesuaikan bobot heuristic aturan pertama meningkatkan bobot DEF dan STA, aturan kedua mempertahankan bobot default, sedangkan aturan ketiga meningkatkan bobot ATK dan ACC.

Algoritma ACO selanjutnya dijalankan sebanyak 20 iterasi dengan 10 agen semut untuk menghasilkan karakter dengan nilai feromon tertinggi sebagai rekomendasi. Pemain bebas mengikuti rekomendasi tersebut atau memilih karakter lain sesuai preferensi. Stage berikutnya kemudian dimulai dengan penerapan modifier DDA pada seluruh musuh, dan sistem memulai pencatatan sesi permainan baru secara otomatis.

3.2 Implementasi Perhitungan Ant Colony Optimization

Perhitungan ACO dilakukan setelah sistem mendapatkan data performa pemain dari setiap stage yang diselesaikan. Komponen ACO terdiri dari tiga elemen utama: *alternatif* (karakter yang dapat direkomendasikan), *kriteria* (parameter penilaian karakter), dan *perhitungan optimasi* (proses iteratif ACO hingga konvergensi). Ketiga elemen ini bekerja secara terintegrasi untuk menghasilkan rekomendasi karakter yang paling sesuai dengan kondisi performa pemain.

3.2.1 Alternatif

Alternatif merupakan pilihan-pilihan solusi yang menjadi bahan evaluasi dalam proses pengambilan keputusan. Dalam penelitian ini, alternatif yang digunakan adalah tujuh karakter yang dapat direkomendasikan kepada pemain pada game *The Night Dreamers*. Setiap karakter memiliki atribut yang berbeda sehingga menghasilkan tingkat kecocokan yang bervariasi bagi tiap pemain. Nilai atribut lengkap seluruh karakter disajikan pada Tabel 3.3.

Tabel 3. 3 Nilai Statistik Atribut Dasar dan Tipe Gaya Bermain

No	Karakter	ATK	SPD	DEF	STA	ACC	REC	Total	Tipe
K1	Pemberani	85	65	40	55	70	45	360	Offense
K2	Penakut	35	90	55	60	65	70	375	Speed
K3	Pemalu	60	50	55	65	90	55	375	Precision
K4	Penjaga	45	45	90	80	50	80	390	Defense
K5	Seimbang	65	65	65	65	65	65	390	All-Round
K6	Petualang	55	85	50	70	60	60	380	Mobility
K7	Bijaksana	55	50	75	70	80	70	400	Strategy

Karakter K7 Bijaksana memiliki total atribut tertinggi (400) dengan distribusi yang paling merata antara Accuracy, Defense, Stability, dan Recovery. K4 Penjaga dan K5 Seimbang masing-masing mencapai total 390, namun dengan distribusi yang berbeda K4 fokus pada Defense dan Recovery, sedangkan K5 memiliki semua atribut bernilai sama (65). Perbedaan distribusi atribut ini yang akan dioptimasi oleh ACO untuk mencocokkan karakter dengan kondisi performa pemain.

3.2.2 Kriteria

Kriteria merupakan parameter yang digunakan untuk menilai dan membandingkan setiap alternatif karakter. Setiap kriteria diberi bobot yang mencerminkan tingkat kepentingannya terhadap keberhasilan pemain. Seluruh kriteria bersifat benefit, artinya semakin tinggi nilai atribut semakin baik karakter tersebut untuk direkomendasikan. Daftar kriteria dan bobot disajikan pada Tabel 3.4.

Tabel 3. 4 Kriteria Penilaian Atribut Karakter dan Bobot Dasar

No	Kriteria	Nama	Bobot	Jenis	Deskripsi	Simbol
C1	Attack	Kekuatan Serangan	0,20	Benefit	Besarnya damage yang dapat diberikan kepada enemy	w_1
C2	Speed	Kecepatan Gerak	0,15	Benefit	Kecepatan bergerak dan melakukan aksi dalam game	w_2
C3	Defense	Pertahanan	0,20	Benefit	Kemampuan menerima dan menyerap serangan enemy	w_3
C4	Stability	Kestabilan	0,15	Benefit	Keseimbangan gerakan dan kemudahan kontrol karakter	w_4
C5	Accuracy	Ketepatan	0,15	Benefit	Ketepatan serangan mengenai target enemy	w_5
C6	Recovery	Pemulihan	0,15	Benefit	Kemampuan memulihkan HP dan stamina setelah serangan	w_6

Attack (w_1=0,20) dan Defense (w_3=0,20) ditetapkan sebagai bobot tertinggi karena kedua atribut ini secara langsung menentukan kemampuan karakter

mengalahkan enemy dan bertahan hidup — dua aspek fundamental keberhasilan misi. Speed, Stability, Accuracy, dan Recovery masing-masing mendapat bobot 0,15 karena berperan sebagai atribut pendukung yang seimbang. Total seluruh bobot = 1,00

Skala nilai setiap kriteria ditetapkan secara linear dari 1 hingga 5, di mana nilai 1 menunjukkan kondisi paling rendah dan nilai 5 menunjukkan kondisi paling tinggi. Penetapan skala linear ini konsisten dengan asumsi algoritma ACO bahwa semakin tinggi nilai heuristik semakin baik karakter tersebut untuk dipilih. Rentang skala yang sama diterapkan untuk semua kriteria sebagai berikut.

A. *Attack*

Kriteria Attack menilai kemampuan ofensif karakter dalam memberikan damage kepada enemy. Semakin tinggi nilai Attack, semakin besar damage yang dihasilkan sehingga durasi pertempuran lebih singkat.

Tabel 3. 5 Skala Penilaian Kriteria Attack

No	Rentang Nilai	Skala	Keterangan
1	< 40	1	Sangat Lemah
2	40–54	2	Lemah
3	55–69	3	Sedang
4	70–84	4	Kuat
5	≥ 85	5	Sangat Kuat

Skala Attack pada Tabel 3.5 disusun untuk mengukur sejauh mana kemampuan ofensif karakter dapat memperpendek durasi pertempuran dan meningkatkan peluang keberhasilan stage. Nilai di bawah 50 digolongkan Sangat Lemah karena pada rentang ini karakter memerlukan jumlah serangan yang jauh

lebih banyak untuk mengalahkan enemy, sehingga risiko terkena serangan balik meningkat signifikan. Rentang 50–59 masuk kategori Lemah, masih di bawah rata-rata kemampuan ofensif standar yang diharapkan dalam desain game. Attack antara 60 dan 69 dianggap Sedang, cukup untuk menyelesaikan stage dalam batas waktu normal tanpa terlalu bergantung pada atribut pendukung. Nilai 70–79 dikategorikan Kuat karena telah memberikan keunggulan ofensif yang nyata, memungkinkan pemain mengeliminasi enemy lebih cepat. Adapun nilai di atas 80 ditetapkan sebagai Sangat Kuat

B. *Speed*

Kriteria Speed menggambarkan kecepatan bergerak karakter. Semakin tinggi Speed, semakin lincah karakter dalam menghindari serangan dan mempersingkat waktu perpindahan posisi.

Tabel 3. 6 Skala Penilaian Kriteria Speed

No	Rentang Nilai	Skala	Keterangan
1	< 40	1	Sangat Lambat
2	40–54	2	Lambat
3	55–69	3	Sedang
4	70–84	4	Cepat
5	≥ 85	5	Sangat Cepat

Skala speed pada Tabel 3.6 menggolongkan Speed di bawah 40 sebagai Sangat Lambat karena karakter sulit merespons perubahan posisi enemy secara efektif. Rentang 40–54 tergolong Lambat dan masih membatasi mobilitas pemain dalam eksplorasi maupun pertempuran. Speed antara 55 dan 65 dianggap Seimbang, memberikan mobilitas yang cukup tanpa mengorbankan kontrol. Kategori Cepat (66–75) mulai memberikan keuntungan taktis berupa

pergerakan lincah, sedangkan nilai di atas 75 masuk kategori Sangat Cepat yang menjadi ciri khas karakter seperti Penakut dan Petualang; mobilitas tinggi ini sengaja dihadirkan sebagai kompensasi atas kelemahan di atribut lainnya.

C. *Defense*

Kriteria Defense menentukan kemampuan karakter menyerap serangan enemy. Semakin tinggi Defense, semakin lama karakter bertahan dalam pertempuran.

Tabel 3. 7 Skala Penilaian Kriteria Defense

No	Rentang Nilai	Skala	Keterangan
1	< 40	1	Sangat Lemah
2	40–54	2	Lemah
3	55–69	3	Sedang
4	70–84	4	Kuat
5	≥ 85	5	Sangat Kuat

Kriteria Defense pada Tabel 3.7 digunakan untuk menilai daya tahan karakter dalam menyerap damage dari *enemy*. Defense di bawah 50 dikategorikan Sangat Lemah karena karakter akan kehilangan HP secara drastis hanya dalam beberapa serangan. Rentang 50–59 dianggap Lemah, masih sangat bergantung pada keterampilan pemain untuk menghindar. Defense 60–69 masuk kategori Seimbang dan menjadi standar minimal agar karakter dapat bertahan cukup lama dalam pertempuran normal. Nilai 70–79 dikategorikan Kuat, memberikan toleransi lebih besar terhadap kesalahan pemain. Adapun Defense di atas 80, yang hanya dijumpai pada karakter Penjaga, ditetapkan sebagai Sangat Kuat dan secara desain diperuntukkan bagi pemain dengan Hit Taken tinggi serta pemula yang membutuhkan proteksi ekstra.

D. *Stability*

Kriteria Stability menilai keseimbangan gerakan dan kemudahan kontrol karakter. Semakin tinggi Stability, semakin mudah pemain mengarahkan karakter dengan presisi.

Tabel 3. 8 Skala Penilaian Kriteria Stability

No	Rentang Nilai	Skala	Keterangan
1	< 40	1	Sangat Tidak Stabil
2	40–54	2	Tidak Stabil
3	55–69	3	Stabil
4	70–84	4	Sangat Stabil
5	≥ 85	5	Lebih Stabil

Stability pada Tabel 3.8 menunjukkan tingkat keseimbangan gerakan dan kemudahan kontrol karakter selama permainan. Nilai di bawah 40 dianggap Tidak Stabil karena menyebabkan pergerakan karakter terasa licin dan sulit diarahkan, meningkatkan risiko kegagalan misi akibat kesalahan kontrol. Rentang 40–54 masuk Kurang Stabil, masih dapat mengganggu kenyamanan terutama pada pertarungan intens. Stability 55–65 ditetapkan sebagai Stabil, memberikan pengalaman kendali yang wajar dan dapat diandalkan di berbagai situasi. Kategori Sangat Stabil (66–75) mulai menawarkan presisi yang tinggi, membantu manuver yang lebih rumit. Di atas 75 disebut Lebih Stabil dan hanya ditempatkan pada karakter yang mengutamakan kelincahan terkendali seperti Petualang, sehingga efektivitas eksplorasi dan respons gerak dapat tetap optimal.

E. *Accuracy*

Accuracy menggambarkan ketepatan serangan karakter dalam mengenai musuh. Nilai Accuracy yang ideal berada di tingkat menengah hingga tinggi karena dapat meningkatkan efektivitas serangan.

Tabel 3. 9 Skala Penilaian Kriteria Accuracy

No	Rentang Nilai	Skala	Keterangan
1	< 40	1	Tidak Akurat
2	40–54	2	Kurang Akurat
3	55–69	3	Akurat
4	70–84	4	Sangat Akurat
5	≥ 85	5	Lebih Akurat

Accuracy pada Tabel 3.9 mencerminkan probabilitas keberhasilan serangan mengenai target. Accuracy di bawah 40 dikategorikan Tidak Akurat karena tingginya tingkat kegagalan serangan membuat durasi pertempuran menjadi sangat tidak efisien. Rentang 40–54 dianggap Kurang Akurat, masih terlalu banyak serangan yang meleset. Accuracy 55–70 termasuk Akurat, sudah memadai untuk sebagian besar pertarungan. Pada rentang 71–85, Accuracy dianggap Sangat Akurat dan mulai memberikan keunggulan kompetitif. Di atas 85, kategori Lebih Akurat diterapkan, yang secara eksklusif dimiliki oleh karakter Pemalu untuk mendukung gaya bermain presisi tinggi, di mana setiap serangan jarak jauh hampir selalu mencapai target.

F. *Recovery*

Recovery menunjukkan kemampuan karakter untuk memulihkan diri setelah menerima serangan, seperti regenerasi HP atau pemulihan stamina. Nilai

Recovery yang ideal adalah di rentang menengah, di mana karakter cukup kuat namun tetap memberi tantangan pada pemain.

Tabel 3. 10 Skala Penilaian Kriteria Recovery

No	Rentang Nilai	Skala	Keterangan
1	< 40	1	Sangat Tidak Stabil
2	40–54	2	Tidak Stabil
3	55–69	3	Stabil
4	70–84	4	Sangat Stabil
5	≥ 85	5	Lebih Stabil

Recovery pada Tabel 3.10 menentukan kecepatan pemulihan HP dan stamina setelah menerima serangan. Recovery di bawah 40 dinilai Sangat Lambat karena jeda regenerasi yang panjang membuat karakter sangat rentan pada pertempuran beruntun. Rentang 40–55 digolongkan Lambat, sehingga pemain harus sangat berhati-hati dalam mengelola jeda regenerasi. Recovery 56–70 dianggap Normal, cukup untuk menjaga ketahanan tanpa mengganggu ritme permainan. Kategori Cepat pada 71–85 secara signifikan meningkatkan ketahanan dalam pertempuran panjang. Nilai di atas 85 masuk Sangat Cepat, yang hanya diberikan pada karakter Penjaga, sehingga kombinasi dengan Defense tinggi menghasilkan daya tahan maksimal yang cocok untuk pemain yang sering menerima serangan.

Hasil konversi nilai atribut seluruh karakter ke nilai skala 1–5 berdasarkan tabel-tabel di atas, beserta nilai heuristik (η) dasar yang diperoleh, disajikan secara rekap pada Tabel 3.11.

Tabel 3. 11 Rekapitulasi Hasil Konversi Nilai Atribut

No	Karakter	ATK	SPD	DEF	STA	ACC	REC	Nilai Skala
K1	Pemberani	85**(5)**	65**(3)**	40**(2)**	55**(3)**	70**(4)**	45**(2)**	**η=0.6400**
K2	Penakut	35**(1)**	90**(5)**	55**(3)**	60**(3)**	65**(3)**	70**(4)**	**η=0.6100**
K3	Pemalu	60**(3)**	50**(2)**	55**(3)**	65**(3)**	90**(5)**	55**(3)**	**η=0.6300**
K4	Penjaga	45**(2)**	45**(2)**	90**(5)**	80**(4)**	50**(2)**	80**(4)**	**η=0.6400**
K5	Seimbang	65**(3)**	65**(3)**	65**(3)**	65**(3)**	65**(3)**	65**(3)**	**η=0.6000**
K6	Petualang	55**(3)**	85**(5)**	50**(2)**	70**(4)**	60**(3)**	60**(3)**	**η=0.6500**
K7	Bijaksana	55**(3)**	50**(2)**	75**(4)**	70**(4)**	80**(4)**	70**(4)**	**η=0.7000**

Berdasarkan Tabel 3.12, K7 Bijaksana memperoleh nilai heuristik dasar tertinggi (η = 0,7000) karena distribusi atributnya paling merata dengan bobot tinggi pada Defense dan Accuracy. K5 Seimbang dengan semua atribut bernilai sama (sk=3) menghasilkan η = 0,6000 yang merupakan nilai terbawah, namun paling stabil dan konsisten. Nilai heuristik dasar ini selanjutnya akan disesuaikan berdasarkan kondisi performa pemain melalui mekanisme penyesuaian bobot.

3.2.3 Perhitungan Ant Colony Optimization

Proses perhitungan ACO dilakukan dalam empat langkah utama yang berurutan: (1) perhitungan nilai heuristik (η), (2) penyesuaian heuristik berdasarkan kondisi pemain, (3) perhitungan probabilitas pemilihan karakter, dan (4) pembaruan pheromone per iterasi. Parameter ACO yang digunakan: α=1, β=2, ρ=0,3, jumlah semut=10, iterasi maksimum=20, τ_0=0,1.

3.2.3.1 Perhitungan Nilai Heuristik (η)

Nilai heuristik (η) mengukur tingkat kecocokan awal setiap karakter berdasarkan atribut yang dimilikinya. Dihitung menggunakan jumlah tertimbang nilai skala dari enam atribut, dinormalisasi ke rentang 0–1. Nilai *heuristik* berperan penting dalam menentukan probabilitas pemilihan karakter oleh setiap semut, di mana karakter dengan nilai *heuristik* yang lebih tinggi akan memiliki peluang lebih besar untuk dipilih.

$$\eta\_i = (w\_ATK \times sk\_ATK + w\_SPD \times sk\_SPD + w\_DEF \times sk\_DEF + w\_STA \times sk\_STA + w\_ACC \times sk\_ACC + w\_REC \times sk\_REC) \,/\, (\Sigma \text{ bobot} \times skala\_max)$$

Keterangan:
η_i = nilai heuristik karakter ke-i
w_k = bobot kriteria ke-k
sk_k = nilai skala kriteria ke-k (skala 1–5)
skala_maks = 5 (nilai skala tertinggi)

Untuk perhitungan untuk K7 Bijaksana (ATK=55→sk3, SPD=50→sk2, DEF=75→sk4, STA=70→sk4, ACC=80→sk4, REC=70→sk4):

$$\begin{aligned}\eta\_K7 &= (0{,}20\times3 + 0{,}15\times2 + 0{,}20\times4 + 0{,}15\times4 + 0{,}15\times4 + 0{,}15\times4) \,/\, 5 \\ &= (0{,}60 + 0{,}30 + 0{,}80 + 0{,}60 + 0{,}60 + 0{,}60) \,/\, 5 \\ &= 3{,}50 \,/\, 5 = 0{,}7000\end{aligned}$$

Hasil konversi nilai skala dan perhitungan nilai heuristik dasar seluruh karakter disajikan pada Tabel 3.12. Angka dalam kurung menunjukkan nilai skala (1–5) hasil konversi dari nilai atribut.

Tabel 3. 12 Hasil Perhitungan Nilai Heuristik Dasar

No	Karakter	ATK	SPD	DEF	STA	ACC	REC	η Dasar
K1	Pemberani	85 (5)	65 (3)	40 (2)	55 (3)	70 (4)	45 (2)	0,6400
K2	Penakut	35 (1)	90 (5)	55 (3)	60 (3)	65 (3)	70 (4)	0,6100
K3	Pemalu	60 (3)	50 (2)	55 (3)	65 (3)	90 (5)	55 (3)	0,6300
K4	Penjaga	45 (2)	45 (2)	90 (5)	80 (4)	50 (2)	80 (4)	0,6400
K5	Seimbang	65 (3)	65 (3)	65 (3)	65 (3)	65 (3)	65 (3)	0,6000
K6	Petualang	55 (3)	85 (5)	50 (2)	70 (4)	60 (3)	60 (3)	0,6500
K7	**Bijaksana**	**55 (3)**	**50 (2)**	**75 (4)**	**70 (4)**	**80 (4)**	**70 (4)**	**0,7000**

Berdasarkan Tabel 3.12, K7 Bijaksana memperoleh nilai heuristik dasar tertinggi ($\eta = 0{,}7000$) karena distribusi atributnya paling merata dengan nilai skala tinggi pada DEF (4), STA (4), ACC (4), dan REC (4). K5 Seimbang menghasilkan $\eta = 0{,}6000$ yang merupakan nilai terendah karena semua atributnya bernilai skala 3 (sedang). Nilai heuristik dasar ini selanjutnya disesuaikan berdasarkan kondisi performa pemain.

3.2.3.2 Penyesuaian Heuristik

Setelah Rule-Based mengklasifikasikan kondisi pemain, bobot kriteria disesuaikan sebelum ACO dijalankan. Rumus penyesuaian bobot adalah sebagai berikut.

$$w'_k = w_k \times \text{faktor\_boost}_k$$
$$\eta_{adj} = (\Sigma\, w'_k \times sk_k) \,/\, (\Sigma w' \times 5)$$

A. Stage 1 (R-02 (Kondisi Normal))

Data performa yang digunakan: Clear Time = 57,1 detik, Hit Taken = 6, Success Rate = 100%. Kondisi ini memenuhi kriteria R-02 (60–90 detik, 3–7 kali, 60%–80%). Pada R-02 tidak ada penyesuaian bobot, sehingga η_adj = η dasar untuk semua karakter. Detail bobot pada kondisi ini disajikan pada Tabel 3.13.

Tabel 3. 13 Bobot Kriteria dan Faktor Boost pada Kondisi Normal

Kriteria	Bobot (w)	Faktor Boost	Bobot Baru (w')	Keterangan
ATK	0,20	1,0	0,20	Tidak berubah
SPD	0,15	1,0	0,15	Tidak berubah
DEF	0,20	1,0	0,20	Tidak berubah
STA	0,15	1,0	0,15	Tidak berubah
ACC	0,15	1,0	0,15	Tidak berubah
REC	0,15	1,0	0,15	Tidak berubah
Total	**1,00**	—	**1,00**	R-02: tidak ada penyesuaian

Berdasarkan Tabel 3.13, pada kondisi R-02 (Normal) seluruh bobot kriteria tidak mengalami perubahan karena pemain dinilai berada dalam kondisi normal tidak terlalu kesulitan maupun terlalu mudah. Faktor boost untuk semua kriteria bernilai 1,0 sehingga bobot baru (w') identik dengan bobot dasar (w) dengan total bobot tetap 1,00. Kondisi ini berarti algoritma ACO menjalankan perhitungan menggunakan nilai heuristik (η) dasar tanpa penyesuaian, sehingga rekomendasi karakter yang dihasilkan murni berdasarkan nilai atribut masing-masing karakter tanpa mempertimbangkan kebutuhan khusus pemain.

B. Stage 2 (R-01 (Kondisi Kesulitan))

Kemudian di stage 2 data performa yang digunakan: Clear Time = 86,5 detik, Hit Taken = 10, Success Rate = 100%. Kondisi ini memenuhi kriteria R-01 karena Hit Taken ≥ 8. Sistem melakukan boost pada bobot DEF (×1,3) untuk memprioritaskan karakter bertahan, dan STA (×1,2) untuk memprioritaskan karakter dengan kestabilan tinggi. Detail perubahan bobot disajikan pada Tabel 3.14.

Tabel 3. 14 Penyesuaian Bobot Kriteria dengan Faktor Boost

Kriteria	Bobot (w)	Faktor Boost	Bobot Baru (w')	Keterangan
ATK	0,20	1,0	0,20	Tidak berubah
SPD	0,15	1,0	0,15	Tidak berubah
DEF	**0,20**	**× 1,3**	**0,26**	BOOST — pemain sering kena hit
STA	**0,15**	**× 1,2**	**0,18**	BOOST — pemain perlu stabilitas
ACC	0,15	1,0	0,15	Tidak berubah
REC	0,15	1,0	0,15	Tidak berubah
Σw'	**1,00**	—	**1,09**	Total bobot setelah penyesuaian

perhitungan η_adj untuk K7 Bijaksana (DEF=sk4, STA=sk4) dengan bobot baru (Σw' = 1,09):

$$\eta\_K7\_adj = (0{,}20\times3 + 0{,}15\times2 + 0{,}26\times4 + 0{,}18\times4 + 0{,}15\times4 + 0{,}15\times4) \,/\, (1{,}09 \times 5)$$

$$= (0{,}60 + 0{,}30 + 1{,}04 + 0{,}72 + 0{,}60 + 0{,}60) \,/\, 5{,}45$$

$$= 3{,}86 \,/\, 5{,}45 = 0{,}7083$$

Rekap nilai η_adj seluruh karakter beserta kontribusi bobot DEF dan STA setelah penyesuaian disajikan pada Tabel 3.15.

Tabel 3. 15 Rekapitulasi Nilai Heuristik Penyesuaian

No	Karakter	Σ(w'×sk)	Σw'×5	Kontribusi DEF & STA	η_adj
K1	Pemberani	3,4100	5,4500	0,26×2 + 0,18×3	0,6257
K2	Penakut	3,3200	5,4500	0,26×3 + 0,18×3	0,6092
K3	Pemalu	3,4200	5,4500	0,26×3 + 0,18×3	0,6275
K4	Penjaga	3,6200	5,4500	**0,26×5 + 0,18×4**	0,6642
K5	Seimbang	3,2700	5,4500	0,26×3 + 0,18×3	0,6000
K6	Petualang	3,4900	5,4500	0,26×2 + 0,18×4	0,6404
K7	**Bijaksana**	**3,8600**	5,4500	**0,26×4 + 0,18×4**	**0,7083**

K7 Bijaksana memperoleh η_adj tertinggi (0,7083) karena memiliki kombinasi DEF=sk4 dan STA=sk4 yang mendapat boost. K4 Penjaga juga mengalami kenaikan menjadi 0,6642 karena memiliki DEF=sk5 dan STA=sk4. Sebaliknya, K1 Pemberani dan K6 Petualang mengalami penurunan karena nilai DEF-nya rendah (sk2) sehingga kontribusi boost justru kecil secara absolut

3.2.3.3 Perhitungan Probabilitas

Pada setiap iterasi, probabilitas semut memilih karakter ke-i dihitung menggunakan rumus standar ACO yang mempertimbangkan nilai pheromone (τ) dan heuristik (η):

$$p_i = (\tau_i \hat{} \alpha \times \eta_i \hat{} \beta) \; / \; \Sigma_j \, (\tau_j \hat{} \alpha \times \eta_j \hat{} \beta)$$

Keterangan:
p_i = probabilitas pemilihan karakter ke-i
τ_i = nilai pheromone karakter ke-i
η_i = nilai heuristik (atau η_adj setelah penyesuaian)
$\alpha = 1$ (bobot pheromone), $\beta = 2$ (bobot heuristik)

A. Probabilitas Iterasi Pertama (Stage 1)

Pada iterasi pertama, seluruh $\tau_0 = 0,1$ sehingga probabilitas hanya ditentukan oleh nilai η. Perhitungan untuk K7 Bijaksana:

$$\text{Numerator K7} = 0,1\text{^}1 \times 0,7000\text{^}2 = 0,1 \times 0,4900 = 0,049000$$
$$\Sigma \text{ numerator } = 0,040960 + 0,037210 + 0,039690 + 0,040960 + 0,036000 + 0,042250 + 0,049000 = 0,286070$$
$$\text{p\_K7} = 0,049000 / 0,286070 = 0,1713 \ (17,13\%)$$

Probabilitas seluruh karakter pada iterasi pertama Stage 1 disajikan pada Tabel 3.16.

Tabel 3. 16 Probabilitas Pemilihan Karakter pada Iterasi Pertama Stage 1.

Karakter	τ_i	η dasar	η^2	$\tau_i \times \eta_i^2$	p_i	%
K1 Pemberani	0,1000	0,6400	0,4096	0,040960	0,1432	14,32%
K2 Penakut	0,1000	0,6100	0,3721	0,037210	0,1301	13,01%
K3 Pemalu	0,1000	0,6300	0,3969	0,039690	0,1387	13,87%
K4 Penjaga	0,1000	0,6400	0,4096	0,040960	0,1432	14,32%
K5 Seimbang	0,1000	0,6000	0,3600	0,036000	0,1258	12,58%
K6 Petualang	0,1000	0,6500	0,4225	0,042250	0,1477	14,77%
K7 Bijaksana	**0,1000**	**0,7000**	**0,4900**	**0,049000**	**0,1713**	**17,13%**
Σ	—	—	—	**0,286070**	**1,0000**	**100%**

Berdasarkan Tabel 4.x, pada iterasi pertama seluruh karakter memiliki nilai pheromone awal yang sama ($\tau_0 = 0,1$) sehingga probabilitas pemilihan sepenuhnya ditentukan oleh nilai heuristik (η) masing-masing karakter. K7 Bijaksana

memperoleh probabilitas tertinggi sebesar **17,13%** karena memiliki η dasar tertinggi (0,7000), diikuti K6 Petualang (14,77%) dan K1 Pemberani serta K4 Penjaga yang berbagi probabilitas yang sama (14,32%) karena keduanya memiliki η yang identik (0,6400). K5 Seimbang memperoleh probabilitas terendah (12,58%) karena seluruh atributnya bernilai skala sedang (sk=3). Hasil probabilitas ini menjadi dasar pemilihan semut pada iterasi pertama sebelum pheromone diperbarui pada iterasi-iterasi berikutnya.

B. Probabilitas Iterasi Pertama (Stage 2)

Pada Stage 2 nilai η diganti dengan η_adj hasil penyesuaian R-01. Berikut perhitungan untuk K7 Bijaksana:

$$\text{Numerator K7} = 0{,}1\text{^}1 \times 0{,}7083\text{^}2 = 0{,}1 \times 0{,}5016 = 0{,}050163$$
$$\Sigma \text{ numerator} = 0{,}039149 + 0{,}037109 + 0{,}039379 + 0{,}044119 + 0{,}036000 + 0{,}041007 + 0{,}050163 = 0{,}286925$$
$$\text{p\_K7} = 0{,}050163 \,/\, 0{,}286925 = 0{,}1748\ (17{,}48\%)$$

Probabilitas seluruh karakter pada iterasi pertama Stage 2 disajikan pada Tabel 3.17.

Tabel 3. 17 Probabilitas Pemilihan Karakter pada Iterasi Pertama Stage 2.

Karakter	τ_i	η_adj	η^2	$\tau_i \times \eta_i^2$	p_i	%
K1 Pemberani	0,1000	0,6257	0,3915	0,039149	0,1364	13,64%
K2 Penakut	0,1000	0,6092	0,3711	0,037109	0,1293	12,93%
K3 Pemalu	0,1000	0,6275	0,3938	0,039379	0,1372	13,72%
K4 Penjaga	0,1000	0,6642	0,4412	0,044119	0,1538	15,38%
K5 Seimbang	0,1000	0,6000	0,3600	0,036000	0,1255	12,55%

Karakter	τ_i	η_adj	η^2	$\tau_i \times \eta_i^2$	p_i	%
K6 Petualang	0,1000	0,6404	0,4101	0,041007	0,1429	14,29%
K7 Bijaksana	**0,1000**	**0,7083**	**0,5016**	**0,050163**	**0,1748**	**17,48%**
Σ	—	—	—	**0,286925**	**1,0000**	**100%**

K7 Bijaksana memiliki probabilitas tertinggi pada kedua skenario (17,13% dan 17,48%). Pada Stage 2, probabilitas K4 Penjaga meningkat dari 14,32% menjadi 15,38% akibat boost DEF dan STA, menunjukkan bahwa penyesuaian R-01 berhasil meningkatkan peluang karakter bertahan untuk dipilih oleh semut.

3.2.3.4 Pembaruan Pheromone

Setelah seluruh semut menyelesaikan pemilihan pada satu iterasi, pheromone diperbarui melalui dua proses: evaporasi dan deposit. Proses ini diulang hingga iterasi ke-20.

$$\text{Evaporasi: } \tau_i \leftarrow (1 - \rho) \times \tau_i$$
$$\text{Deposit: } \tau_i \leftarrow \tau_i + \eta_i \times \text{kunjungan}_i$$
$$[\rho = 0{,}3]$$

Keterangan:
τ_i = nilai pheromone karakter ke-i
ρ = koefisien evaporasi = 0,3
kunjungan_i = jumlah semut yang memilih karakter ke-i pada iterasi

A. Pembaruan Pheromone Stage 1 (R-02)

Pada iterasi pertama Stage 1, misalkan 2 semut memilih K7 Bijaksana dan 0 semut memilih K4 Penjaga. Perhitungan update pheromone K7:

$$\text{Evaporasi K7} : \tau\_K7 = (1 - 0{,}3) \times 0{,}1 = 0{,}0700$$
$$\text{Deposit K7} : \tau\_K7 = 0{,}0700 + 0{,}7000 \times 2 = 0{,}0700 + 1{,}4000 = 1{,}4700$$

Perkembangan nilai pheromone seluruh karakter dari iterasi 0 hingga 20 pada Stage 1 disajikan pada Tabel .

Tabel 3. 18 Perkembangan Nilai Pheromone Karakter Iterasi 0-20 pada Stage 1.

Iterasi	K1	K2	K3	K4	K5	K6	K7*
0 (awal)	0,1000	0,1000	0,1000	0,1000	0,1000	0,1000	0,1000
1	0,7100	1,2900	0,7000	0,0700	1,2700	1,3700	1,4700
3	1,4359	3,7431	2,4850	0,0343	3,2623	0,6713	2,4003
5	2,8796	4,5181	2,2887	0,0168	4,2385	0,3289	3,3461
10	4,5391	3,5588	0,6007	0,0028	2,6822	0,0553	9,7880
15	3,8645	5,1385	0,1010	0,0005	1,4349	0,0093	11,3061
20	**2,9574**	**3,6003**	**0,0170**	**0,0001**	**1,9410**	**0,0016**	**13,6641**

Berdasarkan Tabel 3.18, K7 Bijaksana mulai mendominasi sejak iterasi ke-10 (τ = 9,7880) dan terus meningkat hingga mencapai τ = 13,6641 pada iterasi ke-20. K4 Penjaga hampir tidak pernah dipilih semut (τ = 0,0001) karena nilai η-nya tidak cukup tinggi untuk bersaing dalam kondisi normal.

B. Pembaruan Pheromone Stage 2 (R-01)

Pada Stage 2 proses deposit menggunakan η_adj sebagai pengganti η. Perhitungan untuk K7 Bijaksana dengan 2 semut memilihnya pada iterasi pertama.

$$\text{Evaporasi K7} : \tau\_K7 = (1 - 0{,}3) \times 0{,}1 = 0{,}0700$$
$$\text{Deposit K7} : \tau\_K7 = 0{,}0700 + 0{,}7083 \times 2 = 0{,}0700 + 1{,}4166 = 1{,}4865$$

Perkembangan nilai pheromone seluruh karakter dari iterasi 0 hingga 20 pada Stage 2 disajikan pada Tabel 3.19.

Tabel 3. 19 Perkembangan Nilai Pheromone Karakter Iterasi 0-20 pada Stage 2.

Iterasi	K1	K2	K3	K4	K5	K6	K7*
0 (awal)	0,1000	0,1000	0,1000	0,1000	0,1000	0,1000	0,1000
1	0,6957	1,2883	0,6975	0,0700	1,2700	1,3507	1,4865
3	1,4046	3,7381	2,4754	0,0343	3,2623	0,6619	2,4282
5	2,8156	4,5120	1,6522	0,0168	4,8385	0,3243	3,3854
10	4,4377	3,5540	0,4929	0,0028	2,1831	0,0545	10,6117
15	3,5634	2,1789	1,0179	0,0005	3,1510	0,0092	12,0542
20	**1,3347**	**1,8466**	**2,3318**	**0,0001**	**0,7354**	**0,0015**	**16,4309**

Pada Stage 2, K7 Bijaksana mencapai $\tau = 16,4309$ pada iterasi ke-20 lebih tinggi dibanding Stage 1 (13,6641). Hal ini terjadi karena η_adj K7 meningkat dari 0,7000 menjadi 0,7083 akibat penyesuaian R-01, sehingga setiap kunjungan semut memberikan deposit yang lebih besar. K4 Penjaga meskipun mendapat boost η_adj (0,6642), tidak berhasil konvergen karena semut lebih konsisten memilih K7 yang memiliki η_adj lebih tinggi sejak iterasi pertama.

3.2.4 Pengujian Usability (System Usability Scale)

Pengujian *usability* tujuannya untuk menilai tingkat kemudahan penggunaan, kenyamanan interaksi, dan sejauh mana pengguna dapat menerima game yang dikembangkan. Pengujian melibatkan 20 responden yang diberikan kesempatan untuk memainkan game dan kemudian mengisi kuesioner SUS. Kuesioner terdiri dari 10 pernyataan dengan skala penilaian 1 hingga 5, di mana angka 1 berarti *Sangat Tidak Setuju* dan angka 5 berarti *Sangat Setuju*. Setelah

datanya terkumpul, langkah selanjutnya adalah mengakumulasi sesuai rumus berikut:

Dengan:

S1, S3, S5, S7, S9 = Skor dari pertanyaan bernada positif.

S2, S4, S6, S8, S10 = Skor dari pertanyaan bernada negatif.

Setelah seluruh data terkumpul, dilakukan analisis menggunakan statistik deskriptif untuk mendapatkan gambaran umum mengenai pengalaman pengguna ketika memainkan game adaptif ini. Daftar pernyataan yang akan digunakan dalam metode SUS ini terhimpun dalam table .

Tabel 3. 20 Daftar Pernyataan Kuesioner Pengujian SUS.

No.	Pertanyaan	Pilihan Skala
1.	Saya berpikir game ini menarik untuk dimainkan.	1 = Sangat Tidak Setuju 5 = Sangat Setuju
2.	Saya merasa antarmuka game ini terlalu rumit.	
3.	Saya merasa game ini mudah digunakan.	
4.	Saya membutuhkan bantuan orang lain agar dapat memainkan game ini.	
5.	Fitur-fitur dalam game ini berjalan dengan baik dan logis.	
6.	Ada terlalu banyak inkonsistensi dalam game ini.	
7.	Saya yakin sebagian besar orang akan cepat mempelajari cara memainkan game ini.	
8.	Saya merasa game ini terlalu membingungkan.	
9.	Saya merasa percaya diri saat memainkan game.	

No.	Pertanyaan	Pilihan Skala
10.	Saya harus belajar banyak hal sebelum dapat memainkan game ini.	

Kuesioner diatas terdiri dari pernyataan bernada positif (nomor 1, 3, 5, 7, 9) dan pernyataan bernada negatif (nomor 2, 4, 6, 8, 10) yang disusun secara berselang-seling. Setiap pernyataan diukur menggunakan skala Likert 1 hingga 5, di mana angka 1 berarti "Sangat Tidak Setuju" dan angka 5 berarti "Sangat Setuju". Responden diminta memberikan penilaian berdasarkan pengalaman mereka setelah memainkan game The Night Dreamers. Skor dari seluruh pernyataan kemudian diolah menggunakan rumus SUS untuk memperoleh nilai akhir yang mencerminkan tingkat kemudahan penggunaan, kenyamanan, dan penerimaan pengguna terhadap game yang dikembangkan. Target skor SUS yang diharapkan adalah ≥ 70, yang termasuk dalam kategori "Good" berdasarkan standar interpretasi SUS.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil Implementasi Game

Hasil Implementasi game The Night Dreamers mencakup semua komponen yang telah di rancang sebelumnya. Game dibangun menggunakan Unity Engine 6 dengan fokus pada user experience yang optimal dan integrasi sistem adaptif. Berikut adalah hasil implementasi dari aspek-aspek game tersebut.

4.1.1 Tampilan Game

Tampilan game dirancang dengan mempertimbangkan aspek kenyamanan pengguna agar informasi dapat tersampaikan secara jelas tanpa mengganggu proses bermain. Struktur tampilan dibuat sederhana namun tetap informatif sehingga pemain dapat memahami fitur dan navigasi game dengan mudah.

a. MainMenu

Tampilan Main Menu merupakan halaman awal yang ditampilkan ketika pemain pertama kali menjalankan game The Night Dreamers. Menu ini berfungsi sebagai pusat navigasi utama sebelum pemain memasuki gameplay. Pada tampilan ini tersedia beberapa tombol utama seperti Start Game, Load Game, Settings, dan Quit yang memudahkan pemain dalam mengakses fitur permainan. Berikut tampilan MainMenu dalam game.

Gambar 4. 1 Tampilan Antarmuka Menu Utama (*Main Menu*)

Pada gambar 4.1 *Main Menu* berfungsi sebagai gerbang utama *player* untuk memulai game. Tampilan diatas menampilkan empat tombol utama, Pertama *Start* untuk memulai permainan baru, Kedua Continue untuk melanjutkan *save game*, ketiga *Option* untuk mengseting suara dalam game, resolution dan lainnya, Terakhir tombol *Quit* kegunaanya untuk menutup game.

b. Character Selection

Tampilan Character Selection digunakan sebagai media bagi pemain untuk memilih karakter yang akan digunakan dalam permainan. Pada halaman ini, sistem menampilkan beberapa karakter dengan atribut, kemampuan, dan tipe permainan yang berbeda-beda. Informasi karakter yang ditampilkan meliputi nama karakter, tipe permainan, atribut statistik, serta rekomendasi hasil perhitungan metode *Ant Colony Optimization*.

Gambar 4. 2 Tampilan Menu Pemilihan Karakter (*Character Selection*)

Pada Gambar 4.2 menampilkan implementasi sistem pemilihan karakter pada game *The Night Dreamers* yang menggunakan metode *Ant Colony Optimization*. Tampilan ini muncul setelah pemain menyelesaikan suatu stage permainan, di mana sistem terlebih dahulu mengevaluasi performa pemain berdasarkan beberapa parameter, yaitu *Clear Time*, *Hit Taken*, dan *Success Rate*. Data performa tersebut kemudian digunakan sebagai input dalam proses perhitungan sistem ACO untuk menentukan karakter yang paling sesuai digunakan pada stage berikutnya.

c. Gameplay

Hasil tampilan gameplay *The Night Dreamers* dapat dilihat pada Gambar 4.3. Tampilan menggunakan perspektif kamera orang ketiga (*third-person*) dengan lingkungan 3D bertemakan kota malam hari. Pemain mengendalikan karakter yang telah dipilih melalui sistem pemilihan karakter adaptif dan bergerak dalam area stage yang telah dirancang.

Gambar 4. 3 Tampilan Antarmuka Sesi Permainan (*Gameplay*)

Berdasarkan Gambar 4.3, tampilan gameplay terdiri dari dua area utama. Pertama, area gameplay yang menampilkan lingkungan perkotaan 3D dengan karakter pemain di tengah dan satu unit enemy di depannya yang dilengkapi *health bar* berwarna hijau di atas kepala. Kedua, area HUD (*Heads-Up Display*) yang menampilkan informasi status secara *real-time* di pojok kiri atas berupa *Health Bar*, *Stamina Bar*, dan *Lives*, serta di pojok kanan atas berupa *Enemy Counter* (0/1), DDA Rule, dan *Difficulty*.

d. GameOver

Panel *Gameover* pada gambar 4.4 menandakan pemain kalah dalam game. Panel ini dibuat untuk *player* memilih ingin melanjutkan gamenya, memilih karakter yang lain atau kembali *mainmenu.*

Gambar 4. 4 Tampilan Panel *Game Over*

Gambar 4.4 ada 3 tombol utama, pertama itu tombol *restart* kegunaanya untuk mengulang kembali permainan tanpa mengganti karakter. Kedua tombol *change* berfungsi untuk mengganti karakter dan mengulang kembali stage yang gagal. Terakhir tombol *mainmenu* untuk kembali ke bagian awal permainan.

4.2 Hasil Pengujian Sistem

Pengujian data dilakukan untuk mengevaluasi keseluruhan sistem pemilihan karakter adaptif yang mengintegrasikan *Rule-Based Adaptive System*, *Dynamic Difficulty Adjustment* (DDA), dan *Ant Colony Optimization* (ACO) pada game *The Night Dreamers*. Pengujian melibatkan sejumlah pemain yang memainkan game dari Stage 1 hingga Stage 2 secara berurutan sebanyak tiga kali percobaan. Selama pengujian berlangsung, sistem secara otomatis mencatat data performa setiap pemain melalui komponen GameManager, meliputi *Clear Time*, *Hit Taken*, dan *Success Rate*.

Data yang terkumpul kemudian dianalisis melalui empat aspek pengujian. Pertama, pengujian performa pemain untuk melihat kemampuan pemain

berdasarkan parameter yang tercatat. Kedua, pengujian kondisi pemain dalam *Rule-Based* untuk memverifikasi ketepatan klasifikasi rule (R-01/R-02/R-03) dan penerapan modifier DDA. Ketiga, pengujian pemilihan karakter untuk mengevaluasi hasil rekomendasi ACO berdasarkan kondisi pemain. Keempat, analisis pemilihan karakter oleh pemain untuk melihat sejauh mana pemain mengikuti rekomendasi sistem. Keseluruhan hasil pengujian disajikan pada sub-bab berikut.

4.2.1 Pengujian Performa pemain

Pengujian performa pemain dilakukan dengan mengumpulkan data selama pemain menyelesaikan setiap stage permainan. Data yang diperoleh digunakan untuk mengetahui tingkat kemampuan pemain berdasarkan waktu penyelesaian, interaksi pemain terhadap *enemy*, serta keberhasilan pemain dalam menyelesaikan permainan.

Gambar 4. 5 Tampilan Rekaman Data Performa Pemain

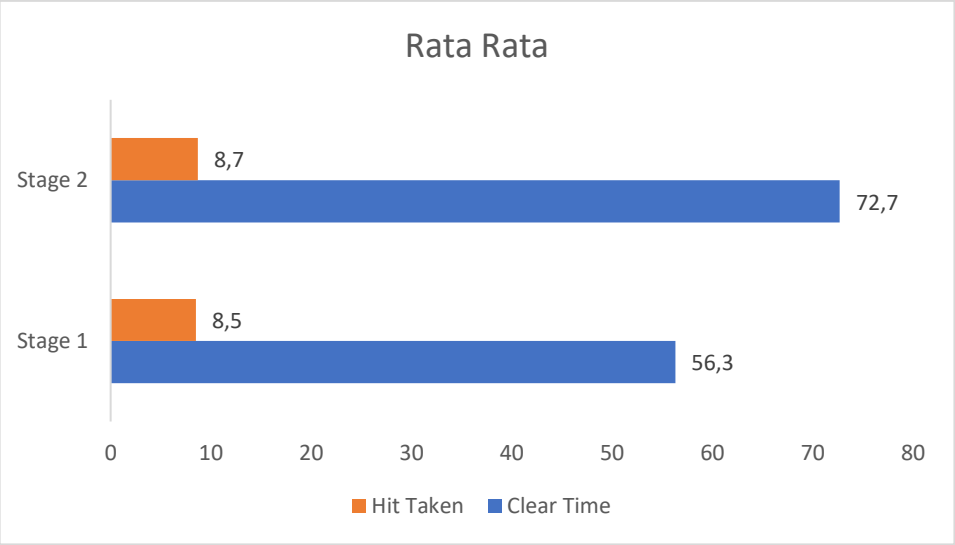
Gambar 4.5 menunjukan data performa pemain dalam stage tersebut. Terlihat ada *clear time, hit taken,* dan *success rate.* Setiap stage akan muncul

peforma pemain jadi pemain bisa meilhatnya sendiri. Hasil rata-rata performa pemain berdasarkan stage dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Rata-Rata *Clear Time* dan *Hit Taken* Pemain Berdasarkan *Stage*

Stage	Avg Clear Time (s)	Avg Hit Taken
Stage 1	56,33258427	8,519607843
Stage 2	72,69019608	8,725663717

Dari Tabel 4.1 diketahui bahwa terjadi peningkatan rata-rata waktu penyelesaian pada Stage 2 dibandingkan Stage 1. Hal ini menunjukkan bahwa perbedaan tingkat stage memberikan pengaruh terhadap tingkat kesulitan yang dirasakan pemain. Selain itu, nilai *hit taken* juga menunjukkan jumlah interaksi pemain dengan serangan *enemy* selama permainan berlangsung.



Gambar 4. 6 Grafik Rata-Rata *Clear Time* dan *Hit Taken*

Berdasarkan Gambar 4.6, dapat diketahui bahwa terdapat perbedaan performa pemain antara Stage 1 dan Stage 2 berdasarkan parameter *clear time* dan

hit taken. Pada Stage 1 diperoleh rata-rata waktu penyelesaian sebesar 56,33 detik dengan rata-rata terkena serangan *enemy* sebanyak 8,52 kali, sedangkan pada Stage 2 meningkat menjadi 72,69 detik dengan rata-rata *hit taken* sebesar 8,72 kali. Peningkatan nilai tersebut menunjukkan bahwa tingkat kesulitan yang dirasakan pemain mengalami perubahan seiring bertambahnya stage permainan, sehingga parameter *clear time* dan *hit taken* dapat digunakan sebagai indikator untuk menentukan kondisi pemain pada proses klasifikasi menggunakan metode *Rule-Based* sebelum dilakukan proses rekomendasi karakter menggunakan *Ant Colony Optimization.*

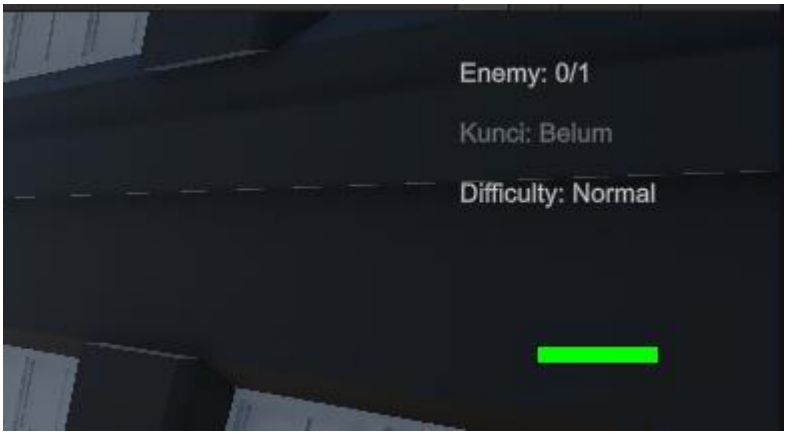
4.2.2 Pengujian Rule-Based Adaptive System

Data performa pemain yang telah diperoleh selanjutnya digunakan untuk menentukan kondisi pemain menggunakan metode *Rule-Based*. Proses identifikasi dilakukan berdasarkan parameter *clear time*, *hit taken*, dan *success rate* yang diperoleh selama pemain menyelesaikan permainan. Aturan yang digunakan bertujuan untuk mengetahui tingkat kemampuan pemain sehingga sistem dapat melakukan penyesuaian tingkat kesulitan *enemy* berdasarkan kondisi pemain.

Pada penelitian ini terdapat tiga kondisi pemain yang digunakan yaitu pemain sulit (R-01), pemain normal (R-02), dan pemain mudah (R-03). Pemain yang masuk kategori sulit akan diberikan penyesuaian tingkat kesulitan berupa penurunan tingkat kesulitan *enemy*, pemain normal mempertahankan kondisi permainan, sedangkan pemain mudah akan diberikan peningkatan tingkat kesulitan.

Gambar 4. 7 Tampilan Aktivasi Aturan R-01

Pada Gambar 4.7 diperlihatkan R-01 yang dimana *player* kesulitan di stage tersebut. Indikator R-01 bisa dilihat pada perubahan kesulitan yang menjadi easy.



Gambar 4. 8 Tampilan Aktivasi Aturan R-02

Pada Gambar 4.8 diperlihatkan R-02 yaitu *normal*. Setiap awal permainan kesulitan akan disetting menjadi normal. Stage selanjutnya akan di hitung dari peforma pemain apakah kesulitannya berkurang atau ditambah.

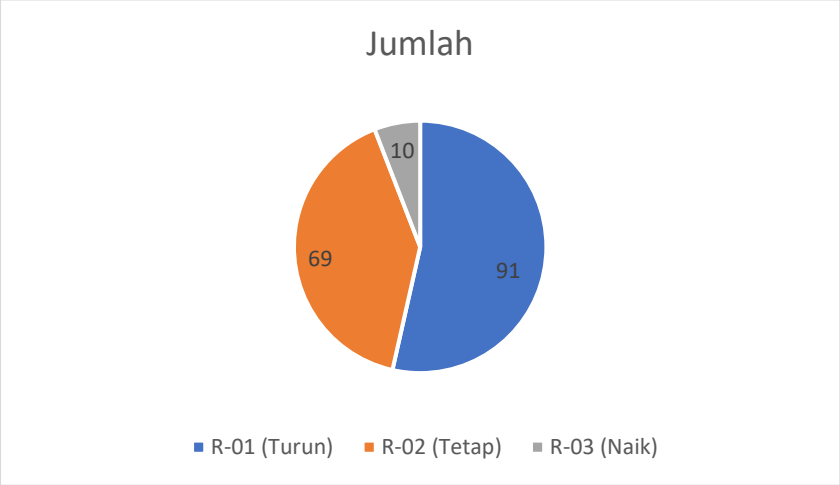
Gambar 4. 9 Tampilan Aktivasi Aturan R-03

Gambar 4.9 menunjukan kesulitan hard yang dimana menandakan pemain mudah menyelesaikan stage sebelumnya. Kesulitan hard akan tercapai pada saat pemain terlalu mudah untuk menyelesaikan stage sebelumnya. Selanjutnya total DDA *Rule-Based* akan dijelaskan pada tabel 4.3 dibawah.

Tabel 4. 2 Hasil DDA *Rule-Based*

DDA Rule	Jumlah	Persentase (%)
R-01 (Turun)	91	53,5%
R-02 (Tetap)	69	40,6%
R-03 (Naik)	10	5,9%

Berdasarkan Tabel 4.2, hasil pengujian kondisi pemain menggunakan metode *Rule-Based* menunjukkan bahwa kategori pemain sulit (R-01) memiliki jumlah data terbanyak yaitu 91 data, kemudian kategori pemain normal (R-02) sebanyak 69 data, dan kategori pemain mudah (R-03) sebanyak 10 data. Hasil tersebut menunjukkan bahwa sebagian besar pemain mengalami tingkat kesulitan yang lebih tinggi selama permainan berdasarkan parameter *clear time*, *hit taken*, dan *success rate.*



Gambar 4. 10 Grafik Perbandingan Hasil Aturan DDA

Berdasarkan Gambar 4.2, hasil pengujian kondisi pemain menunjukkan kategori pemain sulit (R-01) memiliki jumlah data paling tinggi yaitu sebanyak 91 data, kemudian kategori pemain normal (R-02) sebanyak 69 data, sedangkan kategori pemain mudah (R-03) sebanyak 10 data. Hasil tersebut menunjukkan bahwa sebagian besar pemain mengalami tingkat kesulitan yang cukup tinggi selama permainan berlangsung berdasarkan parameter waktu penyelesaian, jumlah serangan yang diterima, dan tingkat keberhasilan pemain.

4.3 Pengujian Rekomendasi Karakter ACO

Proses pengujian pemilihan karakter dilakukan menggunakan metode *Ant Colony Optimization* (ACO) berdasarkan kondisi pemain yang diperoleh dari proses *Rule-Based*. Metode ACO digunakan dalam sistem *Decision Support System* (DSS) untuk menentukan karakter yang memiliki tingkat kecocokan terbaik terhadap kemampuan pemain. Hasil optimasi berupa rekomendasi karakter yang

diberikan sistem berdasarkan proses pencarian solusi menggunakan nilai kecocokan karakter. Selanjutnya, hasil rekomendasi dianalisis berdasarkan frekuensi karakter yang dipilih oleh sistem dan kesesuaiannya dengan pilihan pemain.

4.3.1 Implementasi ACO

Setelah data peforma pemain didapatkan data tersebut akan diolah dalam sistem ACO yang sudah di masukan dalam game. Pertama data akan dimasukin ke perhitungan nilai *heuristik* atau Eta. Sebagaimana akan ditunjukan pada Kode 4.1.

Pseudocode 4. 1

```
float[] ComputeEta(int n)
    {
        float[] eta = new float[n];
        float wSum = 0f;
        foreach (float w in WEIGHTS) wSum += w;

        for (int i = 0; i < n; i++)
        {
            var c = characterDatabase.characters[i];

            // Normalisasi: asumsikan nilai stat 0-100
            // Rumus eta = weighted sum of normalized stats
            eta[i] = (WEIGHTS[0] * c.atk +
```

```
                        WEIGHTS[1] * c.spd +
                        WEIGHTS[2] * c.def +
                        WEIGHTS[3] * c.sta +
                        WEIGHTS[4] * c.acc +
                        WEIGHTS[5] * c.rec) / (wSum * 100f);
                // Clamp agar tidak 0
                eta[i] = Mathf.Max(eta[i], 0.001f);
                  Debug.Log($"[ACO] {c.charName}: atk={c.atk} spd={c.spd}
def={c.def} sta={c.sta} acc={c.acc} rec={c.rec} → η={eta[i]:F4}");
            }
            return eta;
    }
```

Setelah perhitungan *heuristik* terpenuhi selanjutnya di sesuaikan nilai eta atau *heuristik* sesuai dengan peforma pemain. Sebagaimana akan ditunjukan pada Kode 4.2.

Pseudocode 4. 2

```
        float[] AdjustEtaByPerformance(float[] baseEta)
            {
                float[] adjusted = (float[])baseEta.Clone();
                float[] weights = (float[])WEIGHTS.Clone();

                if (_hitTaken >= 8 || _successRate < 0.60f)
```

```
            {
                Debug.Log("[ACO] Player kesulitan → boost DEF & STA
weight");
                weights[2] *= 1.3f; // DEF
                weights[3] *= 1.2f; // STA
            }
            else if (_hitTaken <= 2 && _successRate > 0.80f)
            {
              Debug.Log("[ACO] Player bagus → boost ATK & ACC weight");
                weights[0] *= 1.3f; // ATK
                weights[4] *= 1.2f; // ACC
            }
            int n = characterDatabase.characters.Count;
            float wSum = 0f;
            foreach (float w in weights) wSum += w;
            for (int i = 0; i < n; i++)
            {
                var c = characterDatabase.characters[i];
                adjusted[i] = (weights[0] * c.atk +
                               weights[1] * c.spd +
                               weights[2] * c.def +
                               weights[3] * c.sta +
                               weights[4] * c.acc +
```

```
                        weights[5] * c.rec) / (wSum * 100f);
                adjusted[i] = Mathf.Max(adjusted[i], 0.001f);
            }
            return adjusted;
        }
```

Setelah di adjust dengan memasukan nilai peforma pemain, selanjutnya nilai eta akan masuk ke dalam sistem ACO untuk pemilihan karakter terbaik. Sebagaimana akan di tunjukan pada Kode 4.3.

Pseudocode 4. 3

```
        int n = characterDatabase.characters.Count;
            //Hitung eta (heuristik)
            float[] eta = ComputeEta(n);
            //Adjust eta
            float[] adjEta = AdjustEtaByPerformance(eta);
            _etaValues = adjEta;
            //Inisialisasi pheromone
            float[] tau = new float[n];
            for (int i = 0; i < n; i++) tau[i] = 0.1f;
            //Iterasi ACO
            for (int iter = 0; iter < iterations; iter++)
            {
                // Hitung probabilitas pemilihan
```

```
        float[] prob = ComputeProbability(tau, adjEta, n);
        // Setiap semut memilih karakter
        int[] visitCount = new int[n];
        for (int ant = 0; ant < totalAnts; ant++)
        {
            int chosen = RouletteWheelSelection(prob, n);
            visitCount[chosen]++;
        }
        // Update pheromone
        // Evaporasi dulu
        for (int i = 0; i < n; i++)
            tau[i] = (1f - evaporation) * tau[i];
        for (int i = 0; i < n; i++)
        {
            if (visitCount[i] > 0)
                tau[i] += Q * adjEta[i] * visitCount[i];
        }
        for (int i = 0; i < n; i++)
            tau[i] = Mathf.Clamp(tau[i], 0.01f, 10f);
    }
    // Pilih karakter
    _recommendedIndex = GetBestIndex(tau, n);
```

Setelah seluruh proses ACO selesai dijalankan selama sejumlah iterasi yang telah ditentukan, sistem akan menghasilkan nilai pheromone (τ) akhir untuk setiap karakter. Karakter dengan nilai pheromone tertinggi kemudian dipilih sebagai hasil akhir rekomendasi sistem. Pada tahap ini, nilai *heuristik* (η) yang telah disesuaikan (adjusted eta) berperan sebagai faktor utama yang mempengaruhi arah penguatan pheromone selama proses optimasi.

4.3.2 Hasil Rekomendasi Karakter

Hasil rekomendasi karakter diperoleh setelah proses optimasi menggunakan metode ACO dilakukan berdasarkan data performa pemain. Setiap karakter memiliki peluang untuk direkomendasikan oleh sistem berdasarkan tingkat kecocokan terhadap kondisi pemain. Karakter dengan nilai kecocokan lebih tinggi akan memiliki frekuensi rekomendasi yang lebih besar dibandingkan karakter lainnya.

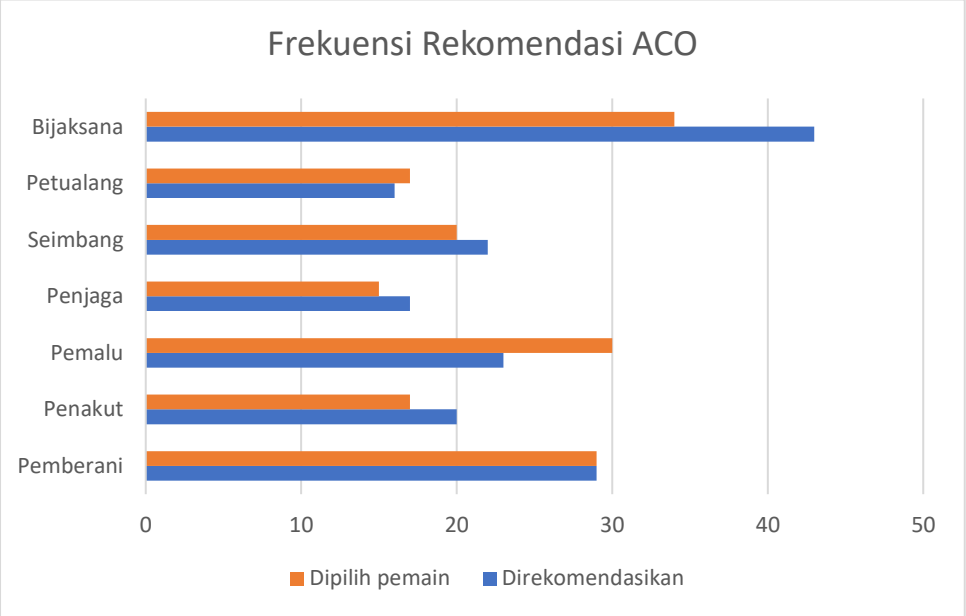
Gambar 4. 11 Tampilan Rekomendasi ACO Dalam Game

Berdasarkan hasil pengujian diperoleh jumlah rekomendasi karakter yang dihasilkan oleh sistem seperti pada Tabel 4.3.

Tabel 4. 3 Frekuensi dan Persentase Aktivasi Aturan DDA Rule-Based.

Karakter	Frek. Direkomendasikan	Frek. Dipilih
Pemberani	29	29
Penakut	20	17
Pemalu	23	30
Penjaga	17	15
Seimbang	22	20
Petualang	16	17
Bijaksana	43	34

Berdasarkan Tabel 4.3, terdapat perbedaan antara hasil rekomendasi sistem dengan keputusan pemain pada beberapa karakter. Karakter Pemberani memiliki nilai rekomendasi dan pemilihan yang sama yaitu sebanyak 29 kali, sedangkan karakter Pemalu memiliki jumlah pemilihan lebih tinggi dibandingkan rekomendasi yang diberikan sistem. Perbedaan tersebut menunjukkan bahwa pemain tidak selalu mengikuti rekomendasi ACO dan masih terdapat faktor keputusan pemain dalam menentukan karakter yang digunakan.



Gambar 4. 12 Frekuensi Rekomendasi Karakter dari ACO

Gambar 4.12 hasil rekomendasi ACO memiliki pola yang relatif mendekati dengan pilihan pemain pada sebagian besar karakter. Karakter Bijaksana menjadi karakter dengan rekomendasi tertinggi dari sistem, sedangkan karakter Pemalu lebih sering dipilih oleh pemain dibandingkan jumlah rekomendasi yang diberikan. Hal ini menunjukkan bahwa sistem ACO mampu memberikan rekomendasi berdasarkan perhitungan algoritma, namun keputusan akhir tetap dipengaruhi oleh pemain.

4.4 Analisis Pemilhan Karakter oleh Pemain

Pada analisis ini menampilkan pemilihan karakter oleh 32 pemain yang menguji game. Data mencakup semua pilihan karakter di setiap permainan dan stage, sehingga diperoleh gambaran tentang karakter mana yang paling disukai pemain.

4.4.1 Data Performa Pemain per Permainan

Pada Tabel 4.4 menampilkan data lengkap performa seluruh 32 pemain yang mengikuti pengujian. Setiap pemain menyelesaikan 3 run penuh (Run 1, Run 2, Run 3), di mana setiap run mencakup Stage 1 dan Stage 2. Kolom Kar. S1 dan Kar. S2 menunjukkan karakter yang dipilih, sedangkan Clear S1 dan Clear S2 mencatat waktu penyelesaian dalam detik.

Tabel 4. 4 Data Pemain dalam Game

Pemain	Run 1				Run 2				Run 3				Rata-rata	
	Kar. S1	Clear S1	Kar. S2	Clear S2	Kar. S1	Clear S1	Kar. S2	Clear S2	Kar. S1	Clear S1	Kar. S2	Clear S2	Avg Clear (s)	Avg Hit
P01	*Bijaksana*	43.6	*Seimbang*	86.0	*Bijaksana*	57.1	*Seimbang*	80.1	*Bijaksana*	86.5	*Bijaksana*	64.7	**69.7**	**7.0**
P02	*Penakut*	75.4	*Seimbang*	93.9	*Bijaksana*	75.8	*Pemberani*	40.8	*Penakut*	61.1	*Penjaga*	97.3	**74.1**	**5.5**
P03	*Pemberani*	84.3	*Pemberani*	59.3	*Penjaga*	81.5	*Penakut*	90.0	*Petualang*	71.8	*Pemalu*	55.2	**73.7**	**5.5**
P04	*Seimbang*	73.8	*Pemberani*	42.4	*Pemalu*	52.7	*Pemalu*	92.1	*Penjaga*	80.3	*Pemalu*	113.5	**75.8**	**5.8**
P05	*Bijaksana*	48.8	*Penjaga*	71.0	*Petualang*	74.8	*Pemalu*	48.8	*Bijaksana*	87.5	*Petualang*	66.5	**66.2**	**4.7**
P06	*Bijaksana*	55.0	*Petualang*	105.7	*Pemberani*	37.8	*Pemberani*	100.1	*Pemalu*	30.1	*Pemalu*	110.4	**73.2**	**5.0**
P07	*Penjaga*	44.1	*Pemalu*	83.5	*Pemberani*	49.2	*Bijaksana*	85.9	*Bijaksana*	69.7	*Seimbang*	71.6	**67.3**	**8.3**
P08	*Pemalu*	85.4	*Bijaksana*	62.1	*Penjaga*	79.8	*Pemalu*	41.6	*Petualang*	58.8	*Bijaksana*	60.5	**64.7**	**5.7**

Pemain	Run 1				Run 2				Run 3				Rata-rata	
	Kar. S1	Clear S1	Kar. S2	Clear S2	Kar. S1	Clear S1	Kar. S2	Clear S2	Kar. S1	Clear S1	Kar. S2	Clear S2	Avg Clear (s)	Avg Hit
P09	*Pemalu*	66.5	*Pemberani*	59.6	*Seimbang*	47.9	*Penakut*	55.6	*Penjaga*	54.2	*Bijaksana*	102.8	**64.4**	**7.2**
P10	*Penakut*	32.8	*Petualang*	106.6	*Bijaksana*	39.4	*Pemalu*	85.1	*Bijaksana*	52.8	*Penjaga*	48.6	**60.9**	**6.2**
P11	*Pemberani*	55.5	*Penakut*	71.8	*Pemberani*	49.3	*Bijaksana*	53.0	*Penjaga*	52.2	*Pemberani*	108.9	**65.1**	**6.3**
P12	*Penakut*	36.4	*Pemalu*	80.1	*Bijaksana*	33.7	*Penjaga*	62.6	*Penjaga*	87.8	*Pemberani*	94.3	**65.8**	**8.0**
P13	*Penakut*	65.6	*Pemberani*	85.7	*Seimbang*	51.0	*Seimbang*	42.4	*Pemberani*	57.7	*Pemalu*	81.8	**64.0**	**7.7**
P14	*Pemberani*	46.6	*Pemberani*	40.5	*Seimbang*	70.8	*Seimbang*	75.0	*Seimbang*	33.5	*Bijaksana*	57.3	**53.9**	**5.7**
P15	*Bijaksana*	55.1	*Petualang*	88.3	*Pemberani*	87.2	*Penakut*	86.1	*Petualang*	51.6	*Bijaksana*	81.1	**74.9**	**7.2**
P16	*Pemalu*	75.8	*Pemberani*	74.6	*Pemalu*	34.5	*Pemberani*	93.1	*Penakut*	87.0	*Penakut*	67.4	**72.1**	**7.2**
P17	*Penakut*	69.3	*Bijaksana*	113.3	*Bijaksana*	81.9	*Pemberani*	62.9	*Petualang*	42.8	*Pemalu*	50.5	**70.1**	**6.5**
P18	*Seimbang*	87.7	*Pemalu*	112.9	*Bijaksana*	55.1	*Pemberani*	98.4	*Pemberani*	47.9	*Bijaksana*	87.8	**81.6**	**5.5**
P19	*Bijaksana*	62.3	*Pemalu*	46.0	*Seimbang*	63.8	*Seimbang*	110.0	*Pemberani*	32.3	*Pemalu*	61.8	**62.7**	**6.7**
P20	*Pemberani*	87.3	*Seimbang*	49.4	*Penjaga*	75.8	*Bijaksana*	102.1	*Seimbang*	82.4	*Pemalu*	99.3	**82.7**	**6.2**
P21	*Bijaksana*	38.9	*Pemalu*	102.1	*Petualang*	40.4	*Bijaksana*	73.6	*Seimbang*	55.4	*Bijaksana*	87.0	**66.2**	**8.0**

Pemain	Run 1				Run 2				Run 3				Rata-rata	
	Kar. S1	Clear S1	Kar. S2	Clear S2	Kar. S1	Clear S1	Kar. S2	Clear S2	Kar. S1	Clear S1	Kar. S2	Clear S2	Avg Clear (s)	Avg Hit
P22	*Pemberani*	47.4	*Pemalu*	107.2	*Penakut*	66.7	*Penakut*	112.6	*Penakut*	73.8	*Pemberani*	42.5	**75.0**	**6.8**
P23	*Petualang*	34.8	*Penakut*	101.1	*Seimbang*	78.1	*Penjaga*	109.7	*Bijaksana*	36.2	*Petualang*	97.6	**76.2**	**5.0**
P24	*Pemalu*	30.3	*Petualang*	61.5	*Pemberani*	56.7	*Petualang*	78.2	*Penakut*	64.3	*Petualang*	73.6	**60.8**	**7.5**
P25	*Bijaksana*	48.9	*Pemalu*	106.3	*Petualang*	63.4	*Pemalu*	42.3	*Seimbang*	45.0	*Pemalu*	56.8	**60.4**	**7.2**
P26	*Pemalu*	32.0	-	-	-	-	*Pemalu*	46.7	*Unknown*	36.4	-	-	**38.4**	**5.3**
P27	-	-	*Seimbang*	40.8	*Unknown*	38.7	-	-	-	-	*Pemberani*	71.6	**50.4**	**2.7**
P28	*Unknown*	83.0	-	-	-	-	*Penjaga*	105.1	*Unknown*	34.9	-	-	**74.3**	**9.0**
P29	-	-	*Pemalu*	68.2	*Unknown*	40.1	-	-	-	-	*Bijaksana*	56.0	**54.8**	**5.7**
P30	*Unknown*	40.0	-	-	-	-	*Bijaksana*	69.1	*Unknown*	49.3	-	-	**52.8**	**8.3**
P31	-	-	*Pemberani*	79.2	*Unknown*	38.7	-	-	-	-	*Petualang*	110.2	**76.0**	**3.7**
P32	*Bijaksana*	32.6	-	-	-	-	*Penjaga*	70.9	-	-	-	-	**51.8**	**11.5**

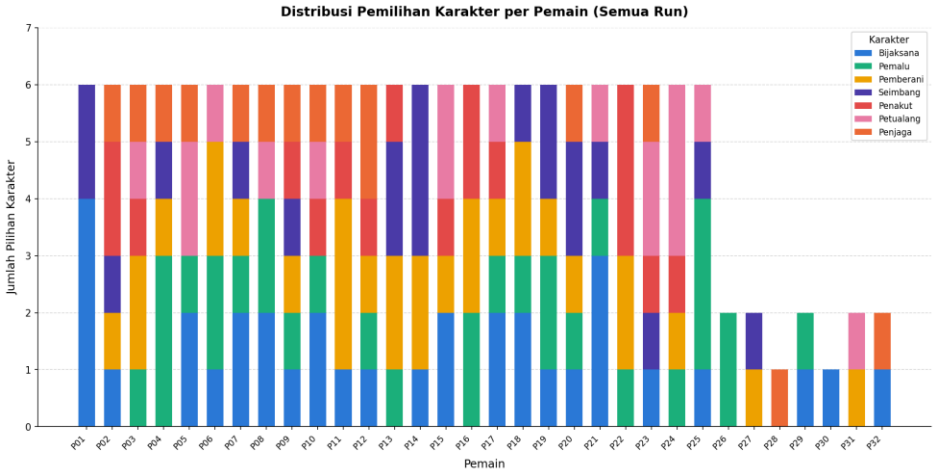
Berdasarkan Tabel 4.4, pengujian dilakukan terhadap 32 pemain (P01–P32) dengan masing-masing pemain memainkan game hingga tiga kali putaran (*run*) dari Stage 1 hingga Stage 2. Rata-rata waktu penyelesaian (*Clear Time*) seluruh pemain

berkisar antara 38,4 detik (P26) hingga 82,7 detik (P20), dengan rata-rata keseluruhan sebesar 67,3 detik. Rata-rata *Hit Taken* tertinggi dimiliki oleh P32 sebesar 11,5 kali dan terendah oleh P27 sebesar 2,7 kali.

Terdapat beberapa pemain yang tidak menyelesaikan seluruh tiga putaran secara lengkap, seperti P26, P27, P28, P29, P30, P31, dan P32, yang ditandai dengan nilai "-" pada kolom tertentu. Hal ini disebabkan keterbatasan waktu pengujian sehingga data yang diperoleh tidak selengkap pemain lainnya. Selain itu, beberapa entri menunjukkan karakter "Unknown" yang mengindikasikan data karakter tidak terekam dengan sempurna oleh sistem pada sesi tersebut, namun data *Clear Time* tetap berhasil dicatat sehingga masih dapat digunakan dalam analisis performa.

4.4.2 Pilihan Karakter per Pemain

Distribusi pemilihan karakter oleh setiap pemain pada seluruh run permainan disajikan pada Gambar 4.13. Grafik batang bertumpuk (stacked bar chart) berikut menampilkan jumlah total pemilihan setiap karakter masing- masing pemain (P01–P32) yang dihitung dari seluruh sesi Stage 1 dan Stage 2 pada setiap run yang diselesaikan.



Gambar 4. 13 Grafik Pemilihan Karakter oleh Pemain

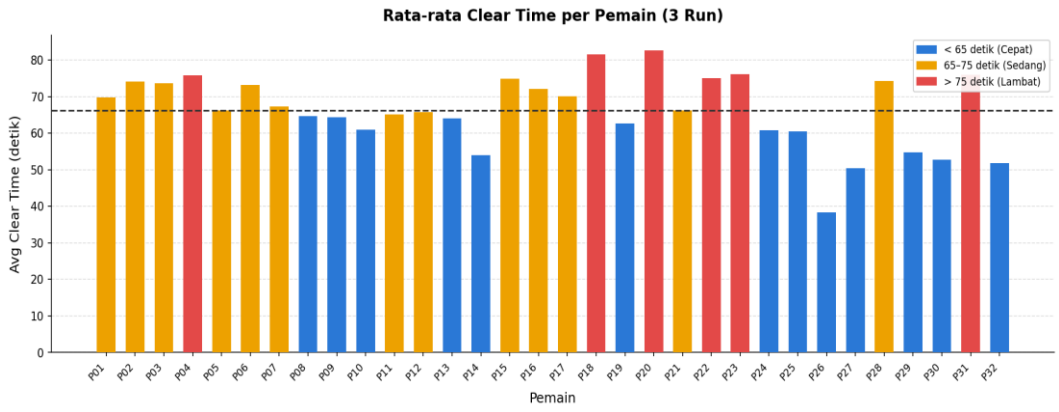
Pada Gambar 4.13, terlihat bahwa pemain P01 hingga P25 umumnya memiliki total 6 pilihan karakter karena menyelesaikan tiga *run* penuh (Stage 1 dan Stage 2 per *run*), sedangkan P26 hingga P32 memiliki total pilihan lebih sedikit (1–2 pilihan) karena tidak menyelesaikan seluruh *run* sebagaimana dijelaskan pada Tabel 4.4 sebelumnya.

Dari sisi distribusi karakter, Bijaksana (biru) dan Pemalu (hijau tua) merupakan dua karakter yang paling sering dipilih di hampir seluruh pemain. Hal ini konsisten dengan hasil perhitungan ACO yang menunjukkan bahwa Bijaksana memiliki nilai heuristik (η) tertinggi pada kondisi R-02 Normal sebesar 0,7000, sehingga sistem paling sering merekomendasikannya. Karakter Pemberani (hijau muda) juga cukup banyak dipilih terutama oleh pemain yang mendapat rekomendasi ACO pada kondisi R-03 (pemain mudah) di mana bobot ATK dan ACC ditingkatkan.

Sebaliknya, karakter Penjaga (oranye) dan Seimbang (ungu) relatif jarang dipilih secara konsisten oleh satu pemain, mengindikasikan bahwa kedua karakter ini hanya direkomendasikan ACO pada kondisi tertentu dan sebagian pemain memilih untuk tidak mengikuti rekomendasi tersebut. Variasi distribusi antar pemain juga menunjukkan adanya preferensi personal yang berbeda-beda, di mana beberapa pemain cenderung memilih karakter yang sama berulang kali meskipun sistem merekomendasikan karakter lain.

4.4.3 Rata-rata Clear Time per Pemain

Selain data karakter, rata-rata *Clear Time* setiap pemain dari seluruh *run* permainan disajikan pada Gambar 4.13. Grafik batang berikut menampilkan kategori kecepatan penyelesaian stage yang dibagi menjadi tiga kelompok: cepat (biru, < 65 detik), sedang (kuning, 65–75 detik), dan lambat (merah, > 75 detik). Garis putus-putus pada angka 67 detik menunjukkan rata-rata keseluruhan seluruh pemain.



Gambar 4. 14 Grafik Rata-Rata *Clear Time* Para Pemain

Berdasarkan Gambar 4.x, sebagian besar pemain berada pada kategori sedang (kuning) dengan rata-rata Clear Time 65–75 detik. Pemain dengan performa tercepat adalah P27 (38,4 detik) dan P26 (38,4 detik), sedangkan yang paling lambat adalah P20 (82,7 detik) dan P18 (81,6 detik). Pemain yang berada di atas garis rata-rata (> 67 detik) cenderung mendapatkan rule R-01 aktif sehingga sistem menurunkan difficulty dan merekomendasikan karakter defensif, sebaliknya pemain di bawah rata-rata berpotensi mendapatkan rule R-03 dengan rekomendasi karakter ofensif.

4.5 Pengujian System Usability Scale

Pengujian usability dilakukan untuk mengukur tingkat kemudahan penggunaan dan penerimaan pengguna terhadap game *The Night Dreamers* yang telah diimplementasikan. Metode yang digunakan adalah System Usability Scale (SUS). Kuesioner SUS terdiri dari 10 pernyataan dengan skala 1 hingga 5, di mana setiap pernyataan bersifat positif atau negatif secara bergantian. Pernyataan bernomor ganjil (Q1, Q3, Q5, Q7, Q9) bersifat positif, sedangkan pernyataan bernomor genap (Q2, Q4, Q6, Q8, Q10) bersifat negatif. Hal ini bertujuan untuk mengurangi bias responden dalam menjawab kuesioner. Daftar pernyataan kuesioner SUS disajikan pada Tabel 4.6.

Tabel 4. 5 Penjelasan Setiap Pertayaan SUS

No.	Pertanyaan	Jenis
Q1.	Saya berpikir game ini menarik untuk dimainkan.	Positif
Q2.	Saya merasa antarmuka game ini terlalu rumit.	Negatif
Q3.	Saya merasa game ini mudah digunakan.	Positif

No.	Pertanyaan	Jenis
Q4.	Saya membutuhkan bantuan orang lain agar dapat memainkan game ini.	Negatif
Q5.	Fitur-fitur dalam game ini berjalan dengan baik dan logis.	Positif
Q6.	Ada terlalu banyak inkonsistensi dalam game ini.	Negatif
Q7.	Saya yakin sebagian besar orang akan cepat mempelajari cara memainkan game ini.	Positif
Q8.	Saya merasa game ini terlalu membingungkan.	Negatif
Q9.	Saya merasa percaya diri saat memainkan game.	Positif
Q10.	Saya harus belajar banyak hal sebelum dapat memainkan game ini.	Negatif

Data menggunakan skala likert dengan rentang 1 sampai 5 mulai dari sangat tidak setuju sampai sangan setuju.

Tabel 4. 6 Nilai SUS

No.	Jawaban	Nilai
1	Sangat Tidak Setuju	1
2	Tidak Setuju	2
3	Netral	3
4	Setuju	4
5	Sangat Setuju	5

Hasil pengisian kuesioner oleh responden berupa skor setiap pertanyaan bisa dilihat pada tabel 4.8. Data ini menunjukan jawaban responden terhadap masing-masing pertanyaan yang diberikan.

Tabel 4. 7 Skor Responden

Responden	Skor Asli Responden									
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
R01	5	2	3	3	4	1	4	2	5	1
R02	4	2	4	1	3	2	3	3	5	1
R03	5	1	5	1	4	1	3	3	5	2
R04	5	2	5	3	4	2	5	3	4	2
R05	5	1	3	1	4	1	4	2	3	1
R06	4	2	4	1	4	2	5	1	5	1
R07	5	2	5	1	4	1	5	1	5	1
R08	4	1	5	3	5	2	4	1	5	1
R09	4	1	4	3	4	2	5	1	4	1
R10	3	1	5	3	5	2	4	3	3	3
R11	3	2	5	1	5	2	4	2	4	3
R12	5	3	4	1	4	2	3	2	5	1
R13	5	1	4	1	5	3	4	3	3	3
R14	4	2	3	1	5	2	5	2	3	1
R15	4	1	4	2	4	2	5	1	4	1
R16	5	1	5	1	4	2	4	1	3	1
R17	3	1	5	1	4	2	5	1	5	1
R18	5	1	4	1	4	2	5	2	5	2
R19	5	1	5	3	5	1	3	1	4	2
R20	4	3	4	2	4	1	5	3	4	3
R21	5	1	5	2	5	1	3	1	4	3
R22	4	3	5	2	4	2	4	1	5	1

Responden	Skor Asli Responden									
	Q1	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**	**Q10**
R23	5	2	4	2	4	3	5	1	3	3
R24	3	1	3	3	4	1	5	2	4	2
R25	4	2	4	1	4	1	5	2	4	1
R26	4	3	5	1	3	1	4	1	5	3
R27	5	2	5	2	4	3	3	2	5	1
R28	5	1	4	1	5	2	5	1	3	2
R29	5	3	4	3	5	1	4	2	4	1
R30	3	2	5	2	5	1	4	1	5	3
R31	5	1	4	1	5	3	4	2	3	3
R32	5	3	4	2	4	2	3	1	5	3

Hasil dari skor kuesioner SUS ini selanjutnya proses konversi menggunakan rumus SUS untuk setiap responden guna memperoleh nilai akhir SUS dalam skala 0 hingga 100. Hasil perhitungan skor akhir SUS untuk seluruh responden disajikan pada tabel 4.9 Hasil Perhitungan SUS.

Tabel 4. 8 Hasil Perhitungan Skor SUS

No	Responden	Jumlah	Nilai (Jumlah x 2,5)
1	**R01**	32	80.0
2	**R02**	30	75.0
3	**R03**	34	85.0
4	**R04**	31	77.5
5	**R05**	33	82.5
6	**R06**	35	87.5
7	**R07**	38	95.0
8	**R08**	35	87.5

No	Responden	Jumlah	Nilai (Jumlah x 2,5)
9	**R09**	33	82.5
10	**R10**	28	70.0
11	**R11**	31	77.5
12	**R12**	32	80.0
13	**R13**	30	75.0
14	**R14**	32	80.0
15	**R15**	34	85.0
16	**R16**	35	87.5
17	**R17**	36	90.0
18	**R18**	35	87.5
19	**R19**	34	85.0
20	**R20**	29	72.5
21	**R21**	34	85.0
22	**R22**	33	82.5
23	**R23**	30	75.0
24	**R24**	30	75.0
25	**R25**	34	85.0
26	**R26**	32	80.0
27	**R27**	32	80.0
28	**R28**	35	87.5
29	**R29**	32	80.0
30	**R30**	33	82.5
31	**R31**	31	77.5
32	**R32**	30	75.0
	Total Skor	**1043**	**2607.5**
	Skor Rata-rata (Hasil Akhir)		**81.5**

Berdasarkan hasil perhitungan pada tabel 4.9, diperoleh total skor sebesar 2607.5 dengan nilai rata-rata System Usability Scale (SUS) sebesar 81.5. Skor tertinggi diperoleh oleh responden R07 dengan nilai 95.0, sedangkan skor terendah diperoleh oleh responden R10 dengan nilai 70.0.

Nilai rata-rata SUS sebesar 81.5 yang diperoleh dari 32 responden berada pada rentang skor di atas 80,3, yang mengindikasikan bahwa sistem berada pada kategori B. Hal ini menunjukkan bahwa game The Night Dreamers beserta sistem rekomendasi karakter berbasis ACO dan mekanisme Dynamic Difficulty Adjustment (DDA) yang diimplementasikan memiliki tingkat usability yang tinggi dan dapat diterima dengan baik oleh pengguna.

4.6 Integrasi dengan Islam

Dalam pandangan Islam, karakter dan perilaku manusia merupakan aspek penting yang menunjukkan kualitas kepribadian seseorang. Al-Qur'an menjelaskan bahwa manusia memiliki kecenderungan sifat dasar yang dapat muncul dalam berbagai kondisi, baik positif maupun negatif.

"Manusia itu ibarat logam (seperti emas dan perak). Orang-orang yang terbaik di antara mereka pada masa Jahiliyah adalah yang terbaik pula pada masa Islam, jika mereka paham." (HR. Bukhari No. 3383 dan Muslim No. 2526).

Dalam kitab *Syarah Shahih Muslim* oleh Imam An-Nawawi, penggalan hadits tersebut mengisyaratkan bahwa setiap manusia terlahir dengan membawa modalitas bakat atau "bahan dasar" kepribadian yang tetap. Esensi risalah Islam adalah mengarahkan dan memaksimalkan pemahaman (*fiqh*) manusia tersebut agar potensi dasarnya dapat berdaya guna secara optimal dalam menghadapi segala kondisi.

4.6.1 Hablum Minallah

Allah menciptakan manusia dengan *syakilah*, yaitu corak kepribadian dan kecenderungan sifat yang bervariasi. Dalam konteks perancangan *game*, variasi tujuh karakter (seperti *Pemberani, Penakut, Bijaksana, Pemalu, Penjaga, Petualang,* dan *Seimbang*) merupakan representasi dari *syakilah* manusia ciptaan Allah. Implementasi *Dynamic Difficulty Adjustment* (DDA) dalam permainan ini berperan untuk menyeimbangkan ujian rintangan agar sesuai dengan porsi kemampuan masing-masing pembawaan pemain tersebut. Ketika pemain mampu mengendalikan sifat-sifatnya untuk menyelesaikan permainan yang bertemakan Shalat Tahajud ini, secara tidak langsung terbangun esensi *Hablum minallah* berupa latihan kesabaran (*Shabr*) dan keteguhan hati (*Istiqomah*) dalam beribadah kepada Allah SWT.

"Amalan yang paling dicintai oleh Allah adalah amalan yang dilakukan secara terus-menerus walaupun sedikit." (HR. Bukhari dan Muslim).

Hadis tersebut menunjukkan bahwa konsistensi dan keteguhan dalam melakukan suatu kebaikan merupakan hal yang dicintai Allah SWT. Dalam penelitian ini, proses pemain menghadapi tantangan permainan secara bertahap melalui sistem adaptasi menggambarkan latihan kesabaran dan kesinambungan usaha, sehingga dapat menjadi refleksi nilai istiqomah dalam kehidupan manusia.

4.6.2 Hablum Minannas

Setiap sifat dasar yang dimiliki manusia pada hakikatnya adalah potensi yang baik jika ditempatkan pada kondisi yang tepat. Rasulullah SAW bersabda, *"Manusia itu ibarat logam (emas dan perak). Orang-orang yang terbaik di antara mereka pada masa Jahiliyah adalah yang terbaik pula pada masa Islam, jika mereka paham."* (HR. Bukhari dan Muslim).

Hadits ini menunjukkan bahwa interaksi antarmanusia (*Hablum minannas*) memerlukan pemahaman mendalam terhadap karakter individu agar potensi terbaik mereka dapat diarahkan demi kemaslahatan bersama. Di dalam sistem *game*, prinsip ini diwujudkan melalui algoritma *Ant Colony Optimization* (ACO) yang bertindak sebagai pengarah keputusan (*Decision Support System*). Saat performa pemain menurun (misalnya akibat *Hit Taken* yang tinggi), sistem ACO mengevaluasi kecenderungan tersebut dan merekomendasikan sifat karakter yang paling adaptif untuk menutupi kelemahan pemain (misalnya merekomendasikan karakter 'Penjaga' yang unggul dalam pertahanan). Proses rekomendasi ini menjadi simulasi digital dari konsep *fiqh* karakter dalam Islam, di mana manusia diarahkan untuk memilih sikap (*akhlak*) yang paling tepat dan bijaksana dalam menghadapi setiap situasi sosial maupun tantangan hidup.

4.6.3 Hablum Minal'Alam

Dimensi *Hablum minal 'alam* tercermin dari landasan utama pembuatan algoritma kecerdasan buatan dalam penelitian ini. Algoritma ACO yang digunakan

untuk mencari karakter terbaik bagi pemain diadopsi langsung dari perilaku alamiah koloni semut saat mencari jalur terpendek menuju sumber makanan. Semut merupakan salah satu makhluk hidup yang perilakunya direkam secara khusus di dalam Al-Qur'an melalui Surah An-Naml. Melalui pendekatan biomimikri ini, logika berpikir dari makhluk alam diintegrasikan ke dalam sistem komputasi untuk membantu mengoptimalkan performa sifat-sifat manusia di dalam *game*.

"Jika seorang muslim menanam sebuah pohon atau menanam tanaman, lalu tanaman itu dimakan oleh burung, manusia, atau hewan, maka itu menjadi sedekah baginya." (HR. Bukhari dan Muslim).

Proses pemanfaatan ilmu pengetahuan yang terinspirasi dari alam semesta ini merupakan wujud nyata dari aktivitas *Tadabbur Alam* (merenungi ciptaan Allah). Hal ini membuktikan bahwa interaksi positif dengan alam dapat menghasilkan inovasi teknologi yang bermanfaat untuk mendukung aktivitas dan keputusan manusia.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian dan analisis terhadap optimasi pemilihan karakter serta penyesuaian tingkat kesulitan dalam *game* "The Night Dreamers", dapat disimpulkan bahwa integrasi metode *Ant Colony Optimization* (ACO) dan *Rule-Based System* telah berhasil diimplementasikan untuk membentuk sistem adaptif yang responsif. Sistem ini terbukti mampu merekam data performa pemain secara *real-time* meliputi parameter *Clear Time*, *Hit Taken*, dan *Success Rate* untuk memproses rekomendasi karakter sekaligus menyesuaikan tingkat kesulitan (DDA) secara dinamis. Kinerja algoritma ACO efektif dalam memandu keputusan pemilihan karakter melalui perhitungan nilai heuristik (η) dan pembaruan jejak *pheromone*. Algoritma tersebut secara matematis mampu mengadaptasi bobot atribut karakter ketika pemain berada dalam kondisi kesulitan (Rule R-01), sehingga karakter dengan spesifikasi daya tahan tinggi lebih diprioritaskan. Tingkat akseptabilitas sistem ini didukung oleh data yang menunjukkan bahwa pemain mengikuti rekomendasi karakter berbasis ACO sebanyak 60,58% (103 kali). Selain itu, penerapan logika *Rule-Based* sukses menyeimbangkan tantangan permainan dengan distribusi penyesuaian sebesar 53,5% memicu penurunan kesulitan (R-01), 40,6% mempertahankan tingkat kesulitan (R-02), dan 5,9% memicu peningkatan kesulitan (R-03). Secara keseluruhan, sistem adaptif ini berhasil menghadirkan kurva tantangan yang proporsional, yang terindikasi dari peningkatan rata-rata

waktu penyelesaian *stage* dari 56,33 detik pada Stage 1 menjadi 72,69 detik pada Stage 2.

5.2 Saran

Penyempurnaan dan pengembangan sistem pada penelitian selanjutnya, diusulkan beberapa saran sebagai berikut:

1. Penambahan Parameter Evaluasi Pemain, parameter input dapat diperluas untuk tidak sekadar bergantung pada *Clear Time*, *Hit Taken*, dan *Success Rate*. Pengembang selanjutnya dapat menambahkan variabel pengamatan pola pergerakan *player*, tingkat akurasi pemakaian proyektil, atau durasi *idle* untuk menghasilkan kalkulasi heuristik ACO yang lebih presisi dan terpersonalisasi.
2. Hibridisasi Algoritma dan Pendekatan Dinamis, nilai bobot adaptasi (*boosting*) pada *Rule-Based System* saat ini masih bersifat statis dan ditetapkan di awal oleh perancang. Penelitian berikutnya dapat mengintegrasikan pendekatan *Machine Learning* (seperti *Fuzzy Logic* atau *Neural Network*) untuk menyesuaikan rentang kondisi dan bobot penyesuaian probabilitas ACO secara mandiri seiring bertambahnya *dataset* permainan.
3. Komparasi Metode Metaheuristik, rekomendasi pemilihan karakter pada sistem ini dioptimasi murni melalui pendekatan ACO. Disarankan untuk melakukan analisis komparasi komputasi dengan metode optimasi metaheuristik lain, seperti *Particle Swarm Optimization* (PSO) atau *Genetic Algorithm* (GA), guna menilai dan membandingkan aspek kecepatan

konvergensi serta akurasi rute pemecahan solusi dalam ekosistem *game engine*.

DAFTAR PUSTAKA

Angeles-Garcia, Y., Legaria-Santiago, V. K., Azueto-Roos, A., & Calvo, H. (2023). Optimizing Strategy Games: Ant Colony Optimization vs. Minimax Algorithm. *2023 IEEE Symposium Series on Computational Intelligence, SSCI 2023*, *January*, 1449–1454. https://doi.org/10.1109/SSCI52147.2023.10371917

Blum, C. (2024). Physics of Life Reviews Ant colony optimization : A bibliometric review. *Physics of Life Reviews*, *51*(September), 87–95. https://doi.org/10.1016/j.plrev.2024.09.014

Fisher, N., & Kulshreshth, A. K. (2024). *Exploring Dynamic Difficulty Adjustment Methods for Video Games*. 230–255.

Kickmeier-Rust, M. D., & Holzinger, A. (2019). Interactive Ant Colony Optimization to Support Adaptation in Serious Games. *International Journal of Serious Games*, *6*(3), 37–50. https://doi.org/10.17083/ijsg.v6i3.308

Lee, H., Kim, H., & Lee, B. (n.d.). DraftRec : Personalized Draft Recommendation for Winning in Multi-Player Online Battle Arena Games. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25â•fi29, 2022, Virtual Event, Lyon, France* (Vol. 1, Issue 1). Association for Computing Machinery. https://doi.org/10.1145/3485447.3512278

Lucas, P. J. F., & Gaag, L. C. Van Der. (1991). *Principles of Expert Systems*.

Makolo, D., Sunday, Y., Riches, O. K., Eneji, S., & Oladipupo, O. K. (2023). *RULE-BASED SYSTEM*. *868*(08), 951–967.

Mekni, M., Jayaramireddy, C. S., Veera, S., Sai, V., & Naraharisetti, S. (2022). *Reinforcement Learning Toolkits for Gaming : A Comparative Qualitative Analysis*. 417–435. https://doi.org/10.4236/jsea.2022.1512024

Put, V. De, & Elbert, J. (2020). *Ant Colony Optimization for Model Checking Ant Colony Optimization for E . J . van de Put*.

Rodríguez, I., & Puig, A. (2022). *applied sciences Towards Adaptive Gamification : A Method Using Dynamic Player Profile and a Case Study*.

Ruipérez-valiente, J. A., Member, S., Gomez, M. J., Martínez, P. A., & Kim, Y. J. (2021). *Ideating and Developing a Visualization Dashboard to Support Teachers Using Educational Games in the Classroom*. https://doi.org/10.1109/ACCESS.2021.3086703

Sasikumar, M., Ramani, S., Raman, S. M., & Chandrasekar, R. (n.d.). *A Practical Introduction to Rule Based Expert Systems*.

Science, C. (2022). *Improving Ant Colony Optimization Efficiency for Solving Large TSP Instances*. 1–28. https://doi.org/10.1016/j.asoc.2022.108653

Shabadurai, Y., Chua, F., & Lim, T. (2024). *Dynamic Adaptive Gamification*

Framework to Improve User Gamification Experience for Online Training. *14*(1). https://doi.org/10.18178/ijiet.2024.14.1.2022

Troussas, C., Krouska, A., Mylonas, P., & Sgouropoulou, C. (2025). Personalized Instructional Strategy Adaptation Using TOPSIS: A Multi-Criteria Decision-Making Approach for Adaptive Learning Systems. *Information (Switzerland)*, *16*(5). https://doi.org/10.3390/info16050409