TUGAS AKHIR – TE 141599

SKENARIO DINAMIS MENGGUNAKAN DCA (*DYNAMIC CHALLENGING LEVEL ADAPTER*) PADA PERMAINAN 2D BERGENRE *REAL TIME STRATEGY TOWER DEFENSE*

Ramadhany Candra Arif Putra
NRP 2211100044

Dosen Pembimbing
Dr. Supeno Mardi Susiki N., S.T., M.T.
Christyowidiasmoro, S.T., M.T.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015

TUGAS AKHIR – TE 141599

SKENARIO DINAMIS MENGGUNAKAN DCA (*DYNAMIC CHALLENGING LEVEL ADAPTER*) PADA PERMAINAN 2D BERGENRE *REAL TIME STRATEGY TOWER DEFENSE*

Ramadhany Candra Arif Putra
NRP 2211100044

Dosen Pembimbing
Dr. Supeno Mardi Susiki N., S.T., M.T.
Christyowidiasmoro, S.T., M.T.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – TE 141599

DYNAMIC SCENARIO WITH DCA (DYNAMIC CHALLENGING LEVEL ADAPTER) ON 2D REAL TIME STRATEGY TOWER DEFENSE GAME

Ramadhany Candra Arif Putra
NRP 2211100044

Advisors
Dr. Supeno Mardi Susiki N., S.T., M.T.
Christyowidiasmoro, S.T., M.T.

Departement of Electrical Engineering
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

SKENARIO DINAMIS MENGGUNAKAN DCA (*DYNAMIC CHALLENGING LEVEL ADAPTER*) PADA PERMAINAN 2D BERGENRE *REAL TIME STRATEGY TOWER DEFENSE*

TUGAS AKHIR

Diajukan untuk Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Komputer dan Telematika
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Supeno Mardi S. N., S.T., M.T.	Christyowidiasmoro, S.T., M.T.
NIP. 197003131995121001	NIP. 198301272009121004

SURABAYA
JULI, 2015

ABSTRAK

Nama	:	Ramadhany Candra Arif Putra
Judul	:	Skenario Dinamis Menggunakan DCA (*Dynamic Challenging Level Adapter*) pada Permainan 2D Bergenre *Real Time Strategy Tower Defense*
Pembimbing	:	1. Dr. Supeno Mardi S. N., S.T., M.T. 2. Christyowidiasmoro, S.T., M.T.

Permainan *Real-time Strategy* (RTS) merupakan permainan yang bersifat kompetitif, antara pemain melawan pemain ataupun melawan *non-player character* (NPC). Banyak pengembang permainan menggunakan kemampuan NPC lawan yang terstruktur secara statis. Kemampuan tersebut tidak selalu dapat memberikan tingkat tantangan (*challenging rate* (CR)) yang sesuai ke berbagai tipe karakter pemain. Oleh karena itu dibutuhkan suatu kecerdasan buatan (AI) yang dapat mengatur kemampuannya terhadap berbagai tipe karakter pemain, sehingga dihasilkan pola permainan yang dinamis serta tingkat tantangan yang sesuai. Pada tugas akhir ini akan diimplementasikan *dynamic challenging level adapter* (DCA) sebagai mekanisme untuk beradaptasi terhadap unsur permainan dan menentukan pemilihan keputusan. Pemilihan keputusan akan menggunakan bantuan *decision tree* dan penyelesaian *knapsack problem* digunakan untuk memilih kombinasi pasukan yang tepat dalam mengatasi keadaan. Untuk mengukur performa sistem AI dengan DCA, dibuat AI yang didasari tingkat kesulitan yang umum digunakan yaitu mudah, sedang, dan sukar untuk menjadi lawan tanding. Hasil pengujian AI dengan DCA didapatkan rata-rata dari rata-rata selisih nilai CR tiap waktu untuk hasil pertarungan dengan berbagai tipe tingkat kesulitan adalah 34,02, dibandingkan dengan rata-rata untuk hasil pertarungan dengan tingkat kesulitan yang setara adalah 33,65 sehingga hanya terpaut sebesar 0,37. Sedangkan dengan tingkat kesulitan tidak setara didapatkan rata-rata terpaut sebesar 55,98 dari rata-rata tingkat kesulitan setara. Dengan hasil tersebut dapat disimpulkan penggunaan DCA dapat menjadi pengganti tingkat kesulitan yang statis namun tetap dengan tingkat tantangan yang sesuai bagi tiap tipe karakter pemain.

Kata Kunci: DCA, Kecerdasan Buatan, RTS, *Decision Tree*, *Challenging Rate*, *Knapsack Problem*.

ABSTRACT

Name	:	Ramadhany Candra Arif Putra
Title	:	*Dynamic Scenario Using DCA (Dynamic Challenging Level Adapter) on 2D Real Time Strategy Tower Defense Game*
Advisors	:	1. Dr. Supeno Mardi S. N., S.T., M.T. 2. Christyowidiasmoro, S.T., M.T.

Real-time Strategy (RTS) game is a game with competitive environment, that can be played by player versus player or non-player character (NPC). Many game developer implement enemy NPC with static pattern. Ability of static NPC not always give challenge to any kind of model player. Because of that, it needed an artificial intelligence (AI) that can maintain it's ability to any kind of model player, so it's produce dynamic flow of gameplay and balance in challenging level. In this final project will be implemented dynamic challenging level adapter (DCA) as mechanism for adapting of element in game environment and choose decision. Decision selection will be handled with decision tree to decide outcome for all condition, and solution of knapsack problem will be used for choosing unit combination to handle the condition. To see what AI NPC with DCA is capable of, another AI is created that implement simple mechanism with default difficulty level like easy, medium, and hard. This new AI will be enemy for AI with DCA. The result of implementation NPC with DCA give the average of average difference in CR each time was 34,02 as the result of battle NPC with DCA and without DCA with different difficulty level. Comparing the result of NPC without DCA battle with the same difficulty level, the average of average difference in CR each time was 33,65, adrift at 0.37. While with different difficulty level got the average adrift at 55,98 from the same difficulty level. With these result we can conclude the use of DCA can be a substitute for static difficulty level but still with the same challenging level to every type of model player.

Keywords: Artificial Intelligence, Challenging Rate, DCA, Decision Tree, Knapsack Problem, RTS.

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT atas segala limpahan berkah, rahmat, serta hidayah-Nya, penulis dapat menyelesaikan penelitian ini dengan judul : **Skenario Dinamis Menggunakan DCA (*Dynamic Challenging Level Adapter* pada Permainan 2D Bergenre *Real Time Strategy Tower Defense*)**.

Penelitian ini disusun dalam rangka pemenuhan bidang riset di Jurusan Teknik Elektro ITS, Bidang Studi Teknik Komputer dan Telematika, serta digunakan sebagai persyaratan menyelesaikan pendidikan S1. Penelitian ini dapat terselesaikan tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Keluarga, Ibu dan Bapak tercinta yang telah memberikan dorongan spiritual dan material dalam penyelesaian buku penelitian ini.
2. Bapak Dr. Tri Arief Sardjono, S.T., M.T. selaku Ketua Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember.
3. Secara khusus penulis mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Dr. Supeno Mardi S. N. S.T., M.T. dan Bapak Christyowidiasmoro S.T., M.T. atas bimbingan selama mengerjakan penelitian.
4. Bapak-ibu dosen pengajar Bidang Studi Teknik Komputer dan Telematika, atas pengajaran, bimbingan, serta perhatian yang diberikan kepada penulis selama ini.
5. Seluruh teman-teman angkatan e-51 serta teman-teman *B201-crew* Laboratorium Bidang Studi Teknik Komputer dan Telematika.

Kesempurnaan hanya milik Allah SWT, untuk itu penulis memohon segenap kritik dan saran yang membangun. Semoga penelitian ini dapat memberikan manfaat bagi kita semua. Amin.

Surabaya, Juni 2015

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

Penelitian ini dilatar belakangi oleh berbagai kondisi yang menjadi acuan. Selain itu juga terdapat beberapa permasalahan yang akan dijawab sebagai luaran dari penelitian.

1.1 Latar belakang

Permainan *Real-time Strategy* (RTS) *Tower Defense* (TD) merupakan permainan yang memiliki komponen strategi dengan unsur membangun dan atau mengatur sumber daya dengan pola permainan yang bergerak secara *real-time* [1]. Permainan RTS TD memiliki tujuan untuk menghancurkan bangunan utama lawan dan juga mempertahankan bangunan utama pemain. Untuk meraih tujuan tersebut pemain akan ditantang dengan adanya pasukan lawan yang memiliki tujuan yang sama. Tantangan tersebut memiliki beberapa macam tingkat kesulitan yang dapat dipilih oleh pemain dengan menyesuaikan kemampuan pemain.

Tingkat kesulitan yang ada pada sebuah permainan RTS TD akan mempengaruhi tingkat kemampuan *non-player character* (NPC) sebagai musuh untuk dapat melawan pemain. Untuk mendapatkan kemampuan itu NPC diberi kecerdasan buatan (*Artificial Intelligence* atau AI). Untuk menyesuaikan tantangan terhadap berbagai karakter pemain, terdapat beberapa tingkat kesulitan dasar yang diberikan dalam permainan RTS TD. Tingkat kesulitan dasar pada umumnya dibagi menjadi tiga yaitu, *easy* (mudah), *medium* (sedang), dan *hard* (sukar). Namun dengan tingkat kesulitan dasar, pemain akan merasa permainan terlalu mudah atau terlalu susah [2]. Untuk menyelesaikan permasalahan tersebut banyak penelitian yang mengaplikasikan berbagai bentuk kecerdasan buatan. Salah satunya adalah dengan mengadopsi strategi yang sama dengan pemain. Namun dengan menggunakan kecerdasan buatan seperti itu, pola permainan akan selalu sama setiap pertandingan, sehingga hasil permainan akan selalu sama. Kecerdasan buatan seperti itu dianggap memiliki permasalahan permainan yang statis.

Dengan adanya permasalahan pola permainan yang sama atau statis, dikembangkan kembali kecerdasan buatan menggunakan me-

kanisme *Dynamic Challenging Level Adapter* (DCA). DCA merupakan mekanisme yang akan menentukan tindakan sesuai dengan aksi dan keadaan yang terjadi. Dengan penggunaan DCA, pola serta hasil permainan setiap pertandingan akan dinamis dan juga menyesuaikan tingkat tantangan (*Challenging Rate* atau CR) pemain. Pada penelitian [3] dicoba penerapan DCA pada permainan RTS Star Craft 2 (SC2) yang menganalisa lingkungan permainan dengan melihat bangunan yang dibangun serta ras pasukan yang ada didalam permainan. Pada penelitian tersebut dilakukan pengukuran dengan mengadu kemampuan kecerdasan buatan dengan DCA dan tanpa DCA. Kesimpulan yang didapat adalah tingkat tantangan permainan dengan penerapan DCA akan semakin seimbang, serta akan melipat gandakan waktu permainan dan juga pemenang akan ditentukan saat mendekati akhir permainan.

Dalam tugas akhir ini akan digunakan kecerdasan buatan dengan mengimplementasi DCA yang akan menganalisa unsur umum permainan serta parameter dalam lingkungan permainan. Hasil analisa akan digunakan untuk mengambil keputusan kapan dan pasukan apa yang harus dikeluarkan dalam suatu skenario serta mengontrol pengumpulan sumber daya (*resource*). Diharapkan dengan menggunakan DCA akan memberikan performa yang lebih baik serta tetap menjaga kedinamisan tingkat kesulitan.

1.2 Permasalahan

Penggunaan kecerdasan buatan dalam permainan RTS masih memiliki pola permainan statis. Hal tersebut dapat membuat pemain akan mudah menghafal atau memprediksi jalannya permainan sehingga tingkat tantangan akan menjadi dirasa mudah atau permainan akan dirasa terlalu susah karena pola permainan yang tidak sesuai dengan pengalaman pemain.

1.3 Tujuan

Penerapan kecerdasan buatan dengan mekanisme DCA dalam permainan RTS diharapkan akan membuat permainan menjadi lebih dinamis serta menyeimbangkan tingkat tantangan (CR) dan kesulitan dalam permainan. Pola dinamis yang ingin dicapai dalam tugas akhir ini adalah keadaan yang selalu berusaha mengimbangi pola permainan pemain sehingga jarak CR tidak berpaut terlalu ja-

uh. Hal tersebut akan membuat segala jenis pemain dapat bermain dengan tantangan dan kesulitan yang setara. Selain itu penggunaan algoritma ini dapat dimanfaatkan untuk penelitian lanjutan mengenai Artificial Inteligence serta optimasi dalam permainan RTS.

1.4 Batasan masalah

Untuk memfokuskan permasalahan yang akan diangkat maka dilakukan pembatasan masalah. Batasan-batasan masalah tersebut diantaranya adalah:

1. DCA akan mengontrol dalam skenario waktu dan tipe pasukan yang akan dimunculkan.
2. *Challenging Rate* (CR) atau tingkat tantangan akan mengacu pada persamaan 2.1.
3. Permainan akan berbasis pada prototipe permainan RTS TD.
4. Permainan akan menggunakan Unity3D sebagai *Game Engine*.

1.5 Sistematika Penulisan

Laporan penelitian tugas akhir ini tersusun dalam sistematika dan terstruktur sehingga lebih mudah dipahami dan dipelajari oleh pembaca maupun seseorang yang hendak melanjutkan penelitian ini. Sistematika laporan penelitian ini didasarkan dengan alur sebagai berikut.

BAB I PENDAHULUAN

Bab ini berisi uraian tentang latar belakang, permasalahan, tujuan penelitian, batasan masalah dan sistematika laporan.

BAB II TEORI PENUNJANG

Pada bab ini berisi tentang uraian secara sistematis teori-teori yang berhubungan dengan permasalahan yang dibahas pada pengerjaan tugas akhir ini. Teori-teori ini digunakan sebagai dasar dalam pengerjaan, yaitu informasi terkait, teori mengenai model permainan yang digunakan, algoritma yang akan digunakan, teori mengenai Unity3D sebagai *Game Engine*, dan teori-teori penunjang lainya.

BAB III DESAIN DAN IMPLEMENTASI

Bab ini berisi tentang penjelasan-penjelasan terkait sistem yang dibuat. Guna mendukung itu, digunakanlah blok diagram agar sistem yang akan dibuat dapat terlihat dan mudah dibaca untuk diim-

plentasikan pada pembuatan permainan. Selain itu digunakan juga gambar-gambar yang merepresentasikan bentuk permainan serta antar muka permainan yang dibuat.

BAB IV PENGUJIAN DAN ANALISA

Bab ini menjelaskan tentang pengujian yang dilakukan terhadap sistem hasil dari tugas akhir ini dan menganalisa sistem tersebut. Spesifikasi perangkat keras dan perangkat lunak yang digunakan juga disebutkan dalam bab ini. Sehingga ketika akan dikembangkan lebih jauh, spesifikasi perlengkapannya bisa dipenuhi tanpa harus melakukan uji coba perangkat lunak maupun perangkat keras lagi.

BAB V PENUTUP

Bab ini merupakan penutup yang berisi kesimpulan dari sistem perangkat lunak yang telah di buat dari hasil pengujian yang telah dilakukan. Saran dan kritik yang membangun untuk pengembangkan lebih lanjut juga dituliskan pada bab ini.

BAB 2
TINJAUAN PUSTAKA

Bab 2, akan dibahas mengenai teori penunjang dan perangkat lunak yang digunakan sebagai bahan acuan dan referensi agar tugas akhir ini menjadi lebih terarah

2.1 Permainan *Real-time Strategy*(RTS)

Real-time Strategy (RTS) merupakan kategori permainan strategi yang umumnya terfokus dalam pertarungan militer atau perang. Permainan RTS seperti Warcraft^{TM} atau Line Ranger^{TM} memerlukan pemain untuk mengontrol pasukan yang terdiri dari berbagai macam tipe dan mengalahkan pasukan lawan. Pertarungan tersebut terjadi dalam medan perang virtual yang bergerak secara *real-time*. Umumnya untuk meraih kemenangan dalam permainan RTS, pemain akan bergantung pada efisiensi mengumpulkan dan mengatur sumber daya atau *resource*, serta menggunakan dengan baik sumber daya tersebut untuk berbagai elemen aksi dalam permainan. Elemen aksi dalam permainan dapat berupa memunculkan pasukan, membuat bangunan untuk mengumpulkan sumber daya kembali, dan meningkatkan level pasukan dan atau bangunan [4].

Ciri-ciri game dengan genre *Real Time Strategy* adalah sebagai berikut :

1. Game RTS berjalan secara simultan, dimana lebih dari satu pemain dapat melakukan aksi dalam waktu yang bersamaan. Aksi tersebut terjadi secara duratif, yang artinya memerlukan beberapa waktu untuk menyelesaikannya.
2. Game RTS dimainkan secara *real-time*, artinya pemain memiliki waktu yang terbatas untuk menentukan pergerakan selanjutnya.
3. Sebagian besar game RTS ditampakkan secara parsial. Pemain hanya dapat melihat bagian peta yang telah dijelajahinya. Teknik ini disebut *fog-of-war*.
4. Sebagian besar game RTS bersifat non-deterministik. Tiap aksi mempunyai kesempatan untuk berhasil.
5. Game RTS mempunyai kompleksitas yang sangat tinggi baik dari segi ruang (*space*) maupun jumlah aksi yang tersedia pada

tiap siklus keputusan [5].

Dalam permainan RTS pasukan lawan menjadi tantangan bagi pemain untuk menyelesaikan misinya atau meraih kemenangan. Berhasil atau tidak misi pemain dilihat dari keberhasilan untuk menghancurkan bangunan utama musuh dan mempertahankan bangunan utama pemain dari serangan musuh. Lawan dalam permainan RTS dapat berupa sesama manusia atau melawan kecerdasan buatan.

2.2 *Dynamic Difficulty Adjustment* (DDA)

Dynamic difficulty adjustment (DDA) atau yang dapat dikenal dengan *dynamic game balancing* (DGB) merupakan suatu teknik untuk melakukan otomasi perubahan tingkat tantangan (*difficulty*) pada permainan digital yang *real-time*, berbasis pada kemampuan pemain atau usaha yang dilakukan pemain saat bermain untuk memberikan pemain pengalaman yang optimal atau sering disebut sebagai *flow* atau alur permainan [6].

Berbagai model DDA dikembangkan untuk masing-masing tipe permainan digital. Salah satunya adalah *dynamic challenging level adapter* (DCA) yang diimplementasikan pada permainan RTS. DCA merupakan mekanisme komputer untuk menentukan tindakan secara otomatis terhadap tingkah laku pemain yang beragam [3]. Pada dasarnya penggunaan DCA akan menyesuaikan tingkah laku komputer dengan memperhatikan tingkah laku pemain didalam permainan. Ide utama penggunaan mekanisme DCA ini adalah menganalisa unsur dasar dari permainan misalnya, tipe pasukan, jumlah pasukan, jumlah sumber daya, keadaan sumber daya, keadaan bangunan, dan unsur umum lain dalam permainan RTS. Proses mekanisme DCA diawali dengan mengekstraksi *event* yang pemain lakukan, lalu dilakukan pengolahan dan mengevaluasi apa tindakan yang seharusnya dilakukan. Dengan proses pertimbangan maka mekanisme DCA akan memilih skenario terbaik yang akan dilakukan dalam kurun waktu tersebut. Pada umumnya dalam permainan RTS terdapat dua unsur umum yang dapat dilakukan pertama adalah mengumpulkan sumber daya dan kedua adalah mengeluarkan pasukan. Kedua unsur ini yang akan menjadi roda putar dinamis dalam pengaplikasian DCA.

2.3 *Challenging Rate*(CR)

Challenging Rate (CR) atau tingkat tantangan merupakan suatu nilai yang digunakan untuk menentukan keseimbangan antara kemampauan AI dan kemampuan lawannya. Hal tersebut digunakan untuk mendapatkan nilai adaptif dan nilai keefektifan dari penggunaan DCA terhadap tingkat kemampuan lawan [3]. CR akan mengacu pada persamaan (2.1):

$$CR = \frac{A}{a} + \frac{H}{b} + \frac{P}{c} + \frac{C}{d} \tag{2.1}$$

Dalam persamaan (2.1), parameter A (Attack) merupakan nilai jumlah seluruh kekuatan serang dari pasukan, parameter H (Health Point) merupakan nilai jumlah seluruh life point pasukan dan bangunan yang ada, parameter P (Population) merupakan nilai jumlah seluruh populasi didalam permainan, dan parameter C (Cost) merupakan nilai pengeluaran keuangan atau resource yang digunakan. Parameter a, b, c, dan d merupakan nilai konstan untuk normalisasi persamaan (2.1) yang nilainya akan mengacu pada nilai normal dari status yang dituju.

2.4 *Artificial Intelligence* (AI)

Artificial Intelligence (AI) adalah kecerdasan yang ada pada sebuah mesin atau perangkat lunak yang dibuat dengan memiliki tujuan-tujuan tertentu. Dalam permainan digital, AI bertugas sebagai pemroses tingkah laku dan pengambil keputusan dari lawan dalam permainan atau dikenal dengan *nonplayer character* (NPC) [7] . Seiring dengan berkembangnya teknologi komputer dan permainan digital, telah banyak ditawarkan riset dan ide yang berhubungan dengan AI. Dalam genre permainan umum seperti *action games*, *role-playing games*, dan *strategy games*, tingkah laku dari AI biasanya diimplementasikan sebagai variasi dari aturan dasar yang telah ditentukan didalam permainan. Tetapi dengan adanya *machine-learning technique*, AI dapat berkembang dengan sendirinya. Contohnya dapat meningkatkan performa dengan mempelajari dari kesalahan dan kesuksesannya dan juga dapat beradaptasi de-

ngan kekuatan dan kelemahan dari pemain atau mempelajari lawan dengan meniru strategi permainan yang digunakan.

2.5 *Decision Tree*

Decision tree atau pohon keputusan merupakan suatu cara menentukan keputusan dengan menghasilkan dua kemungkinan data yaitu ya atau tidak. Untuk menentukan keputusan, data yang digunakan sebagai masukan akan dibuat membentuk pohon keputusan dan aturan keputusan [8]. Pembentukan pohon keputusan yang dibuat didasari pada klasifikasi data yang digunakan sebagai masukan, sehingga pemilihan keputusan akan didasari oleh aturan yang disesuaikan dengan klasifikasinya [9]. Data masukan dibuat sebagai sekumpulan data yang telah disesuaikan dengan klasifikasinya masing-masing sehingga membentuk sebuah tabel. Nilai *entropy* didapatkan dengan menggunakan kumpulan data masukan yang akan digunakan untuk membentuk pohon keputusan. *Entropi* merupakan jumlah bit yang dibutuhkan untuk mengekstrak suatu kelas dari sejumlah data acak dalam suatu ruang sampel.

Dalam *decision tree* untuk menghasilkan hasil keputusan yang optimal dibutuhkan beberapa proses antara lain membuat tabel terdistribusi terpadu dengan menyatakan semua nilai kejadian pada setiap rule, menghitung tingkat independensi antara kriteria pada suatu rule, yaitu antara atribut dan target atribut dan mengeliminasi kriteria yang tidak perlu, yaitu yang tingkat independensinya tinggi [8]. Atribut dalam *decision tree* merupakan parameter yang dibuat sebagai kriteria dalam pembentukan pohon, sedangkan target atribut merupakan atribut yang menyatakan data solusi per data atau penentu keputusannya.

2.6 *Knapsack Problem*

Knapsack problem adalah permasalahan dalam menentukan opmitasi nilai kombinasi. Sebagai contoh permasalahannya adalah terdapat sekumpulan barang yang memiliki suatu nilai dan bobot tertentu, bagaimana kombinasi barang yang dapat dimasukan pada suatu tempat yang memiliki keterbatasan total bobot yang dapat ditampung, namun memiliki total nilai barang yang maksimal. Banyak model permasalahan lainnya untuk permasalahan ini namun intinya adalah terdapat suatu limit tertentu untuk dapat menam-

pung suatu data namun ingin memiliki hasil dengan nilai terbesar.

Permasalahan ini dibagi menjadi dua macam [10] yaitu:

1. 0-1 *Knapsack Problem*
   Pada permasalahan ini *item* atau barang yang dapat dipilih merupakan barang yang tidak dapat dipisahkan sehingga dapat diambil satu barang atau tidak. Permasalahan ini dapat diselesaikan dengan menggunakan pendekatan *dynamic programming*.
2. *Fractional Knapsack Problem*
   Pada permasalahan ini barang yang digunakan merupakan barang yang dapat dipisahkan atau dipecah sehingga dapat mengambil sebagian dari suatu barang. Permasalahan ini dapat diselesaikan menggunakan *greedy algorithm*.

Pendekatan penyelesaian yang paling umum digunakan adalah *dynamic programming*. Karena meskipun tingkat kerumitan perhitungan yang harus dilakukan untuk menyelesaikan *knapsack problem* teknik *dynamic programming* mampu melakukannya dengan tingkat kecepatan yang tinggi.

BAB 3
DESAIN DAN IMPLEMENTASI SISTEM

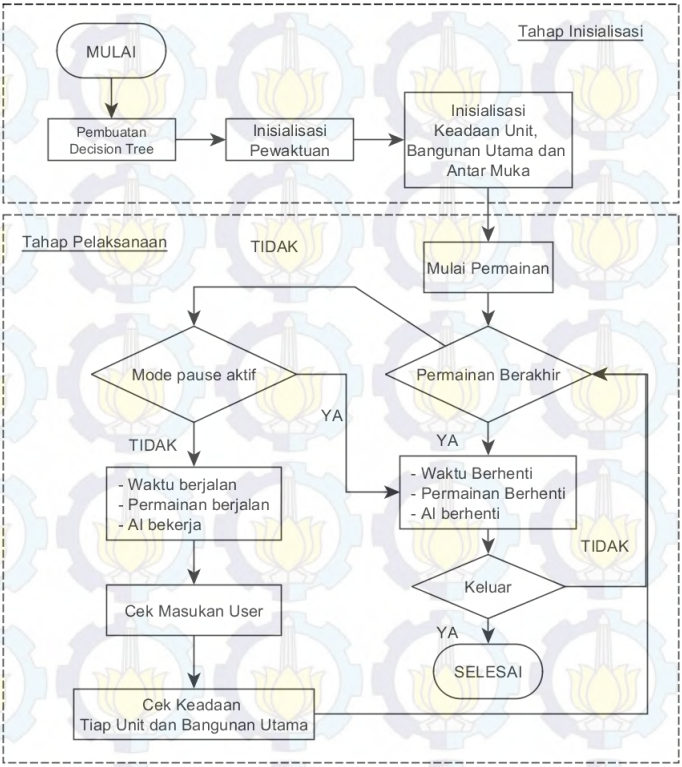
3.1 Desain Sistem

Desain sistem dalam tugas akhir ini dibagi menjadi tiga sistem besar yaitu desain sistem permainan *line defense*, desain sistem *non-player character* (NPC) lawan yang kedepannya akan disebut dengan kecerdasan buatan (AI) tanpa DCA, dan desain sistem NPC AI dengan DCA.

3.1.1 Desain Sistem Permainan *Line Defense*

Sistem permainan yang didesain merupakan sebuah permainan dengan genre *line defense*. Dalam sistem ini terdapat dua bagian utama yaitu tahap inisialisasi, dan tahap pelaksanaan seperti terlihat pada Gambar 3.1. Dalam tahap inisialisasi proses dimulai dengan membentuk atau mengolah *decision tree* sehingga dapat dihasilkan rumus untuk menentukan keputusan didalam sistem kecerdasan buatan. Selain itu dalam tahap inisialisasi ini disiapkan juga berbagai komponen sistem yang digunakan seperti inisialisasi pewaktuan, inisialisasi keadaan unit dan bangunan utama, dan inisialisasi antar muka. Tahap selanjutnya adalah pelaksanaan, yaitu tahap inti dari permainan ini bekerja. Pada tahap ini terjadi perputaran sistem yang akan terus mengecek hal-hal yang menjadi inti permainan antara lain: perintah masukan pemain, kalkulasi pewaktu, proses kecerdasan buatan, dan kejadian-kejadian yang meliputi unit pasukan dan bangunan utama yang menjadi unsur dalam permainan. Pada tahap pelaksanaan terdapat proses interupsi yang dapat digunakan untuk menghentikan permainan sejenak dan atau menghentikan permainan.

Proses pengecekan masukan pemain dibagi menjadi dua bagian, yang pertama masukan untuk mengatur tipe unit yang diinginkan dan yang kedua digunakan untuk mengatur kejadian didalam antar muka. Untuk pengaturan tipe unit dan macam tipe unit akan dijelaskan lebih detail pada sub Bab Unsur Permainan. Kejadian didalam antar muka terdapat dua macam yang digunakan yaitu yang pertama untuk mengatur wilayah pandang terhadap arena permainan dan yang kedua untuk melakukan penghentian sementara sistem atau dikenal dengan *pause* permainan. Proses pengecekan keadaan



Gambar 3.1: Desain sistem permainan

unit dan bangunan utama digunakan untuk menentukan kejadian yang terjadi didalam permainan, contohnya adalah penentuan apakah sedang dalam kondisi pertarungan atau tidak, dan pengecekan apakah permainan telah selesai atau belum. Penjelasan lebih detail mengenai kondisi permainan akan dijelaskan dalam sub bab tersendiri.

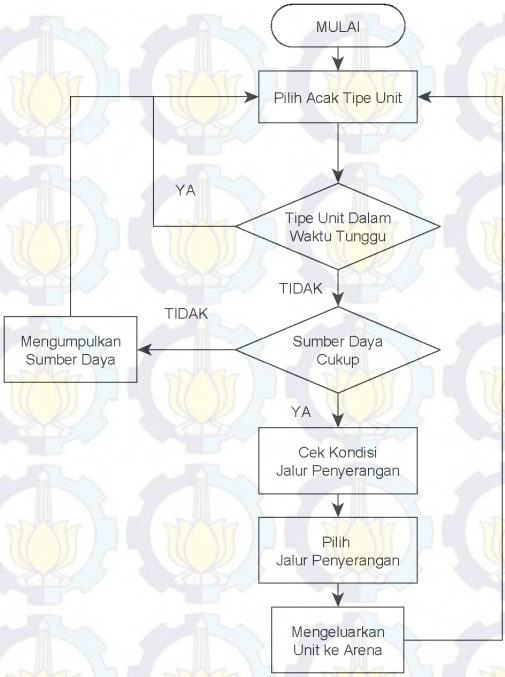
3.1.2 Desain Sistem NPC AI tanpa DCA

Non-player character (NPC) bertindak sebagai lawan didalam permainan yang digunakan dalam tugas akhir ini. Untuk dapat bekerja sesuai dengan fungsinya sebagai lawan, dibutuhkan kecerdasan

buatan (AI) yang akan berperan sebagai otak dari NPC tersebut. Dalam tugas akhir ini diterapkan proses AI seperti pada Gambar 3.2. AI dalam model ini akan bertindak sebagai NPC umum yang akan mengimplementasi pola acak bersyarat. Proses AI ini akan selalu dilaksanakan dalam interval 2 detik. Proses awal dimulai dari memilih tipe unit secara acak, selanjutnya mengecek apakah unit tersebut dalam waktu tunggu. Jika unit yang terpilih dalam waktu tunggu maka akan kembali memilih unit secara acak hingga mendapatkan unit yang tidak dalam waktu tunggu. Proses berikutnya akan mengecek apakah harga unit tersebut cukup dengan sumber daya yang dimiliki oleh NPC. Jika sumber daya kurang maka NPC akan memilih mengumpulkan sumber daya dahulu, jika tidak kurang maka akan dilakukan kalkulasi untuk menentukan jalur penyerangan mana yang akan dipilih. Proses pemilihan jalur penyerangan ini akan menggunakan syarat berupa probabilitas dari kepadatan suatu jalur penyerangan. Setelah itu maka akan dikeluarkan unit yang telah dipilih pada jalur penyerangan yang dipilih. Proses ini akan berjalan terus hingga pada sistem permainan (Gambar 3.1) menetapkan proses berhenti berjalan.

3.1.3 Desain Sistem NPC AI dengan DCA

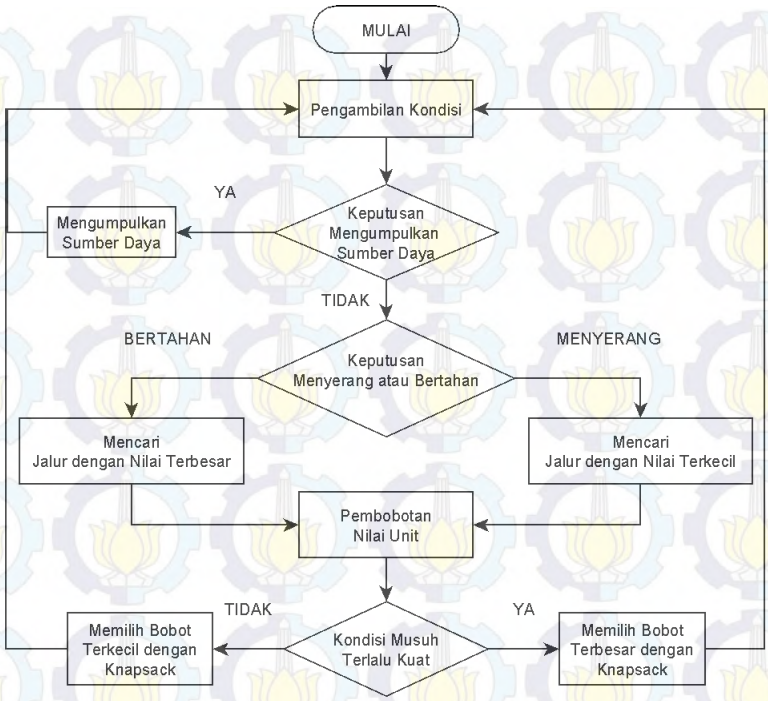
Proses pada pengontrol *non-player character* (NPC) dengan kecerdasan buatan (AI) menggunakan *Dynamic Challenging Level Adapter* (DCA) (selanjutnya disebut dengan AI DCA) digunakan untuk menghasilkan alur permainan yang menyeimbangkan keadaan lawannya. Proses dimulai dengan pengambilan data kondisi permainan yang akan dilakukan setiap 2 detik sekali. Kondisi permainan yang diambil meliputi kondisi *Challenging Rate* (CR) yang menerapkan persamaan (2.1), kondisi sumber daya yang dimiliki AI DCA, kondisi keadaan unit apakah terdapat unit yang sedang dalam waktu tunggu, serta kondisi nilai perbedaan antara CR yang dimiliki pasukan DCA dengan CR lawannya. Setelah pengambilan kondisi permainan data akan diolah untuk menentukan perintah apa yang akan dipilih antara menunggu untuk mengumpulkan sumber daya atau mengeluarkan pasukan. Pemilihan kondisi ini menggunakan algoritma *Decision Tree* dengan mengambil parameter dari data kondisi permainan. Jika kondisi yang dipilih adalah menunggu untuk mengumpulkan sumber daya maka kondisi akan selesai, sedangkan



Gambar 3.2: Desain sistem tanpa DCA

jika kondisi mengeluarkan unit yang dipilih maka akan dilanjutkan untuk pemilihan kondisi berikutnya.

Proses selanjutnya memilih antara mengelurkan unit dengan tujuan untuk bertahan(*defense*) atau menyerang(*attack*). Pemilihan ini juga menggunakan *decision tree* guna menentukan kondisi apa yang dipilih dengan parameter yang diambil adalah kondisi CR, jarak perbedaan nilai CR yang dimiliki pasukan DCA dengan lawannya, serta kondisi keadaan nilai kondisi pada jalur(*line*) dalam permainan. Setelah pemilihan maka tiap-tiap kondisi memiliki aksi yang berbeda, yaitu jika memilih menyerang maka akan dipilih jalur dengan nilai terkecil dan akan membobot pasukan yang ada dengan bobot yang mengutamakan kekuatan serang(*attack power*) dari tiap tipe pasukan, dan jika memilih bertahan maka akan dipilih jalur dengan nilai terbesar dan akan membobot pasukan yang ada



Gambar 3.3: Desain sistem DCA

dengan bobot yang mengutamakan ketahanan (*hit point*) dari tiap tipe pasukan. Setelah proses pembobotan maka akan dilakukan pengecekan apakah kondisi saat ini lawan berada pada kondisi nilai yang terlalu jauh dengan kondisi pasukan DCA, jika memang demikian maka akan dipilih pembobotan dengan nilai terkecil menjadi yang terbesar dan sebaliknya sedangkan jika tidak dalam kondisi tersebut maka pembobotan akan menggunakan nilai awal. Selanjutnya adalah pemilihan kombinasi tipe pasukan apa saja yang akan dikeluarkan menggunakan algoritma penyelesaian *knapsack problem* dengan parameter jumlah tipe pasukan, harga untuk mengeluarkan masing-masing pasukan, bobot masing-masing pasukan serta keadaan sumber daya saat ini. Proses tersebut merupakan proses terakhir sehingga akan kembali pada pengambilan kondisi permainan seper-

ti pada tahap awal. Desain sistem pada DCA diilustrasikan seperti pada Gambar 3.3.

3.2 Alur Kerja

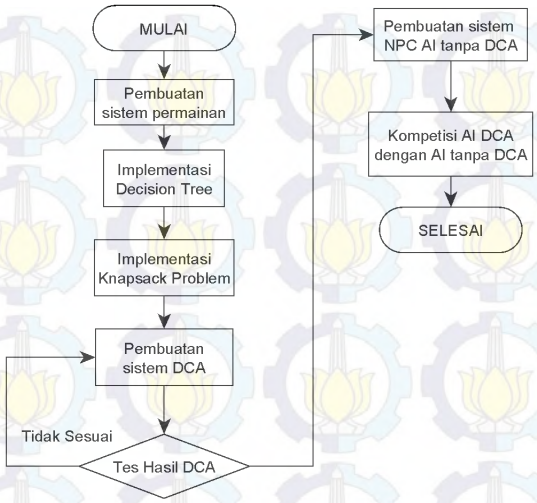
Alur kerja dalam pengerjaan tugas akhir ini terbagi oleh enam tahapan proses. Dalam masing-masing proses memiliki luaran yang dihasilkan dan dijadikan inputan untuk proses pada sistem permainan yang dibuat. Tahapan proses dari tugas akhir ini adalah sebagai berikut:

1. Pembuatan sistem permainan.
   Pembuatan sistem ini meliputi, unsur permainan, kondisi permainan, dan antar muka permainan.
2. Implementasi *decision tree* pada permainan.
   Implementasi *decision tree* digunakan untuk menentukan suatu luaran dari banyak inputan sekaligus.
3. Implementasi algoritma penyelesaian *knapsack problem* pada permainan.
   Implementasi *knapsack problem* digunakan untuk memilih kombinasi terbaik dari unit yang dibutuhkan pada keadaan tertentu.
4. Pembuatan sistem DCA.
   Dilakukan pembuatan sistem sesuai dengan desain sistem pada bab 3.1.
5. Tes *decision tree* dan algoritma penyelesaian *knapsack problem* pada sistem DCA.
   Dilakukan pengetesan pada hasil *decision tree* serta *knapsack problem* untuk menunjukan kesesuaian dengan hasil yang diharapkan.
6. Pembuatan sistem kecerdasan buatan NPC pasukan lawan.
   Pembuatan sistem kecerdasan buatan pasukan NPC digunakan untuk mencoba apakah sistem DCA telah sesuai dengan yang diharapkan.

Keseluruhan alur kerja dari tugas akhir ini dapat dilihat pada Gambar 3.4.

3.3 Pembuatan Sistem Permainan

Pembuatan sistem permainan dibagi menjadi tiga tahap yang akan membentuk suatu permainan secara utuh, tahap-tahap terse-



Gambar 3.4: Alur kerja

but meliputi unsur permainan, kondisi permainan, dan antar muka permainan.

3.3.1 Unsur Permainan

Pada model permainan yang dibuat unsur yang digunakan dalam permainan ini dibagi menjadi 4 unsur besar yaitu tipe pasukan, bangunan utama, jalur penyerangan, dan sumber daya. Masing-masing unsur dapat dijelaskan sebagai berikut:

1. Tipe Pasukan.
   Dalam model permainan yang digunakan ini terdapat 4 macam tipe pasukan masing-masing yaitu *Swordman*, *Archer*, *Assassin*, dan *Crusader* sesuai dengan Gambar 3.5. Masing-masing tipe pasukan tersebut memiliki status yang berbeda-beda sesuai dengan Tabel 3.1. Dari data Tabel 3.1 dapat dibagi menjadi dua poin penting yang akan digunakan untuk penentuan unit mana yang dibutuhkan yaitu kekuatan serang(*attack power*) dan ketahanan(*hit point*). Dua poin tersebut masing-masing merepresentasikan suatu keadaan yaitu, pada kekuatan serang merepresentasikan tingkat bahaya su-

atu unit terhadap lawannya, sedangkan nilai ketahanan merepresentasikan tingkat durabilitas dari suatu unit terhadap lawannya. Status lainnya merepresentasikan seberapa berharganya suatu unit yang dapat dilihat dari harga unit serta lamanya waktu tunggunya (*cooldown time*). Tiap tipe pasukan akan memiliki 4 buah *state* animasi yang menunjukan kondisi mereka saat ini yaitu keadaan *idle* atau tidak melakukan apa-apa, keadaan berjalan, keadaan menyerang, dan keadaan mati seperti pada Gambar 3.6.

Tabel 3.1: Status masing-masing tipe unit

Tipe Unit	HP (Hit Point)	ATK (Attack Power)	ASPD (ATK Speed)	ARNG (ATK Range)	COOLDOWN	COST
Swordman	750	110	1.3	1.3 s	1.5 s	150
Archer	500	110	2	2 s	2.3 s	200
Assassin	750	275	1.2	0.8 s	3 s	300
Crusader	1250	165	1.2	1.5 s	4 s	450

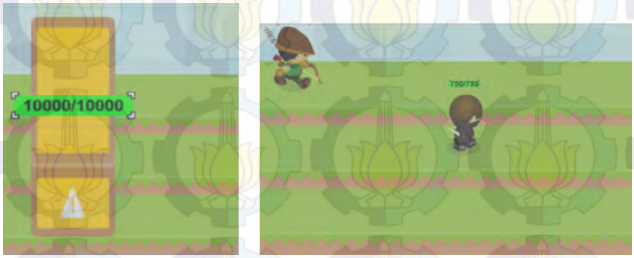
Gambar 3.5: Penggambaran masing-masing tipe unit dari kiri ke kanan: *Swordman*, *Archer*, *Assassin*, dan *Crusader*

Gambar 3.6: *State* animasi unit dari kiri ke kanan: *idle*, jalan, serang, mati

2. Bangunan Utama.
   Bangunan utama merupakan representatif dari tujuan permainan yaitu menghancurkan bangunan utama lawan dan atau

mempertahankan bangunan utama pemain. Dalam permainan ini bangunan utama akan diletakan pada tiap-tiap ujung jalur, dan juga sebagai posisi kemunculan suatu pasukan. Keadaan dari durabilitas bangunan utama akan diperlihatkan dengan nilai ketahanan yang ditampilkan langsung pada layar. Nilai ketahanan dari bangunan utama ini juga masuk hitungan dalam kondisi CR sesuai dengan formula 2.1.

3. Jalur Penyerangan.
   Untuk menyambungkan antara bangunan utama satu dengan yang lainnya dan untuk menentukan lajur perjalanan pasukan, dalam permainan ini diberikan 3 macam jalur penyerangan. Dengan menggunakan 3 macam jalur penyerangan maka dapat dipilih strategi penyerangan atau pertahanan yang digunakan saat permainan berlangsung. Jalur penyerangan ini juga menjadi salah satu parameter dalam penentuan keputusan didalam sistem DCA.



Gambar 3.7: Model bangunan utama(kiri), serta jalur penyerangan(kanan)

4. Sumber Daya.
   Dalam permainan dibutuhkan suatu perputaran alur, dalam permainan ini digunakan sumber daya sebagai salah satu parameter nilai yang menjadi bagian perputaran alur tersebut. Sumber daya dalam permainan ini akan bertambah seiring dengan waktu berjalan, dan akan digunakan untuk membayar saat ingin mengeluarkan pasukan. Sumber daya dalam permainan ini sangat berperan penting dalam perputaran alur karena untuk mencapai tujuan permainan pemain harus menghancurkan bangunan utama lawannya atau mempertahankan

bangunannya agar tidak hancur duluan dengan cara mengeluarkan pasukan untuk bertahan ataupun menyerang.

3.3.2 Kondisi Permainan

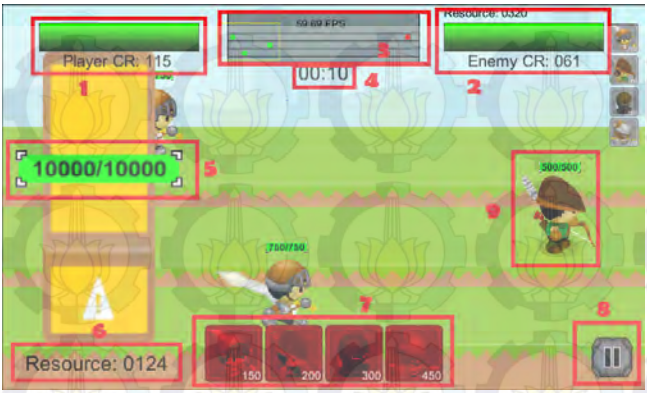
Dalam permainan ini terdapat dua macam kondisi yang merepresentasikan keadaan tujuan dalam permainan yaitu:

1. Kondisi Pertarungan.
   Merupakan kondisi pada saat suatu unit pemain berada pada jangkauan serang unit lawan yang berada pada jalur yang sama atau sebaliknya. Kondisi ini akan memicu *state* animasi serang pada unit yang telah memasuki jangkauan serangnya sehingga dapat melakukan pengurangan ketahanan kondisi lawannya. Kondisi ini dapat diperlihatkan pada *state* animasi seperti pada Gambar 3.6.
2. Kondisi Selesai.
   Kondisi ini dibagi menjadi dua yaitu kondisi kalah dan kondisi menang. Kondisi menang merupakan kondisi pada saat bangunan utama lawan berada pada *hit point* 0 dan bangunan utama pemain memiliki *hit point* lebih dari 0, sehingga dinyatakan pemain berhasil melakukan tujuan atau objektifnya. Sedangkan kondisi kalah merupakan kondisi yang berkebalikan dari sebelumnya yaitu bangunan utama pemain berada pada *hit point* 0 dan bangunan utama lawan memiliki *hit point* lebih dari 0, sehingga dinyatakan pemain gagal dalam melakukan objektifnya.

3.3.3 Antar Muka Permainan

Untuk dapat menampilkan informasi dan interaksi kepada pemain dibuatlah tampilan antar muka dalam permainan ini. Tampilan antar muka seperti pada Gambar 3.8, memiliki sembilan macam unsur informasi yang ditampilkan dijelaskan sesuai dengan nomor yang ada, yaitu:

1. Informasi *Hit Point* Bangunan Utama Pemain.
   Dalam informasi ini ditampilkan data kondisi bangunan utama pemain sehingga jika pemain tidak melihat secara langsung kondisi *hit point* pada bar bangunan utamanya ia dapat langsung melihat dari bagian informasi ini.
2. Informasi *Hit Point* Bangunan Utama Lawan.



Gambar 3.8: Tampilan antar muka yang digunakan

Dalam informasi ini ditampilkan data kondisi bangunan utama lawan sehingga pemain dapat memantaunya dari jauh tanpa harus melihat langsung bar *hit point* dari bangunan lawan.

3. Peta Kecil (*Mini-Map*).
   Informasi ini digunakan untuk menampilkan letak posisi dari unit yang telah berada pada arena permainan, dengan identifikasi warna hijau merupakan unit milik pemain sedangkan warna merah merupakan unit milik lawan. Selain itu juga ditampilkan di posisi jalur mana unit tersebut berada.
4. Pewaktu (*Timer*).
   Pewaktu digunakan untuk menampilkan informasi berapa lama suatu permainan telah berlangsung.
5. Bar *Hit Point* Bangunan Utama.
   Bar ini memberikan informasi keadaan *hit point* bangunan utama secara langsung, namun hanya ditampilkan tepat diatas bangunan utama yang bersangkutan sehingga jika tidak dapat melihat secara langsung *hit point* suatu bangunan utama dapat menggunakan informasi yang berada pada kiri atau kanan atas.
6. Keadaan Sumber Daya.
   Informasi ini menampilkan kondisi sumber daya saat ini yang berupa nilai dari total sumber daya. Informasi ini akan berubah-ubah seiring dengan waktu dan saat pemain mengeluarkan bangunan.

7. Tombol Unit.
   Informasi ini memberikan tampilan kepada pemain keadaan dari tiap unit berupa, tipe unit, harga untuk mengeluarkan, keadaan waktu tunggu, serta keadaan kecukupan sumber daya. Tipe unit ditampilkan dengan perbedaan bentuk dari tiap unit yang ada, harga ditampilkan langsung dengan nilai angkanya, sedangkan waktu tunggu dan kondisi kecukupan sumber daya ditunjukan dengan warna, merah untuk keadaan kurang cukup sumber daya, dan warna agak gelap menunjukan unit sedang dalam waktu tunggu. Informasi ini juga digunakan untuk memilih unit apa yang ingin dikeluarkan dengan meng-klik langsung gambar.
8. Tombol Pause.
   Antar muka ini ditujukan untuk melakukan penghentian permainan dan sehingga dapat keluar dari permainan.
9. Representatif Unit.
   Tampilan ini menunjukan kondisi unit yang ada di permainan, bar pada atas unit menunjukan keadaan *hit point* yang dimiliki unit tersebut. Sedangkan tampilan unit itu sendiri akan memberikan informasi mengenai kondisi atau *state* apa yang sedang dilakukan.

Seluruh antar muka ini bertujuan untuk dapat memberikan informasi yang jelas kepada pemain sehingga dapat mengatur strategi saat bermain.

3.4 Implementasi *Decision Tree*

Proses implementasi *decision tree* dimulai dengan penentuan kondisi yang akan menjadi parameter masukan. Pada sistem DCA ini digunakan dua buah *decision tree* yang pertama mengatur pemilihan kondisi apakah harus mengumpulkan sumber daya atau tidak, dan yang kedua untuk mengatur pemilihan kondisi apakah memilih kondisi menyerang atau bertahan.

3.4.1 *Decision Tree* Pemilihan Mengumpulkan Sumber Daya

Pada proses *decision tree* yang pertama, parameter yang digunakan sebagai masukan adalah kondisi-kondisi sebagai berikut:

1. Kondisi *Challenging Rate*(CR).
   Parameter ini akan melihat apakah kondisi CR pasukan NPC AI DCA berada pada nilai yang lebih dari atau kurang dari CR pasukan lawannya. Nilai ini menunjukan apakah lawan berada pada tingkat keaktifan tingi atau tidak dilihat dari seberapa banyak unit yang sudah dikeluarkan. Karena jika semakin aktif maka nilai CR akan terus meningkat.
2. Perbedaan nilai *Challenging Rate*(CR).
   Parameter ini akan menentukan seberapa jauh perbedaan nilai antara CR pasukan AI DCA dengan CR pasukan AI tanpa DCA. Perbedaan ini akan menggunakan nilai-nilai ambang yang ditentukan sesuai pada Tabel 3.2.

Tabel 3.2: Keterangan *decision tree* mengumpulkan sumber daya

Keterangan	1	2	3
DCR = Beda CR	Kecil (<50)	Sedang (>= 50 && <80	Besar (>= 80)
CR = Kondisi CR	Kurang dari Lawan	Lebih dari Lawan	
MON = Kondisi Sumber Daya	Kurang (< unit termurah)	Cukup (>= unit termurah)	
COOL = Kondisi waktu tunggu Unit	Ada	Tidak	
DEC = Keputusan	Mengumpulkan Sumber Daya	Mengelurakan Unit	

3. Kondisi waktu tunggu Unit.
   Parameter ini akan menentukan apakah terdapat unit yang berada pada waktu tunggu atau tidak. Kondisi ini dipilih karena jika terdapat unit dalam waktu tunggu maka unit yang ingin dikeluarkan ada kemungkinan sedang berada pada waktu tunggu. Sehingga unit tersebut tidak dapat digunakan untuk melakukan penyerangan atau pertahanan.
4. Kondisi sumber daya AI DCA.
   Parameter ini akan mengecek jumlah sumber daya yang dimiliki AI DCA apakah berada pada nilai ambang tertentu yang ditentukan sesuai pada Tabel 3.2. Pemilihan nilai ambang tersebut dikarenakan untuk dapat mengeluarkan sebuah

unit maka dibutuhkan minimal sumber daya yang cukup untuk membeli unit termurah.

Parameter masukan yang telah ditentukan selanjutnya akan digabung menjadi kumpulan data yang akan menghasilkan keputusan sesuai dengan hasil parameter masukan. Untuk kumpulan data pada *decision tree* yang pertama dapat dilihat pada Tabel 3.3. Kode angka pada tabel tersebut berasal dari Tabel 3.2.

Tabel 3.3: Hasil keputusan *decision tree* mengumpulkan sumber daya

DCR	CR	MON	COOL	DEC
1	1	1	1	1
1	1	1	2	1
1	1	2	1	1
1	1	2	2	2
1	2	1	1	1
1	2	1	2	1
1	2	2	1	1
1	2	2	2	2
2	1	1	1	1
2	1	1	2	1
2	1	2	1	2
2	1	2	2	2
2	2	1	1	1
2	2	1	2	1
2	2	2	1	1
2	2	2	2	2
3	1	1	1	1
3	1	1	2	1
3	1	2	1	2
3	1	2	2	2
3	2	1	1	1
3	2	1	2	1
3	2	2	1	1
3	2	2	2	1

3.4.2 *Decision Tree* Pemilihan Penyerangan atau Pertahanan

Pada proses *decision tree* yang kedua, parameter yang digunakan sebagai masukan adalah kondisi-kondisi sebagai berikut:

1. Kondisi *Challenging Rate*(CR).
   Parameter yang digunakan sama dengan *decision tree* pertama, yang akan melihat apakah kondisi CR pasukan AI DCA berada pada nilai yang lebih dari atau kurang dari CR pasukan AI tanpa DCA.
2. Perbedaan nilai *Challenging Rate*(CR).
   Parameter yang digunakan sama dengan *decision tree* pertama, yang akan menentukan seberapa jauh perbedaan nilai antara CR pasukan AI DCA dengan CR pasukan AI tanpa DCA. Perbedaan ini akan menggunakan nilai-nilai ambang yang ditentukan sesuai pada Tabel 3.5.
3. Kondisi Jalur.
   Parameter ini akan mengecek nilai dari sebuah jalur apakah sudah melebihi suatu ambang tertentu yang merupakan penanda bahwa jalur tersebut memiliki tingkat bahaya yang tinggi. Nilai ambang yang digunakan seperti pada Tabel 3.5. Nilai ambang tersebut dipilih karena dengan sejumlah pasukan tersebut sebuah jalur sudah memiliki tingkat ancaman yang tinggi.

Tabel 3.4: Keterangan *decision tree* menyerang atau bertahan

Keterangan	1	2	3
DCR = Beda CR	small (< 50)	Sedang (>= 50 && < 80)	Besar (> 80)
CR = Kondisi CR	Kurang Dari Lawan	Lebih Dari Lawan	
LINE = Kondisi Jalur (Total Unit in Line > 3)	Sedikit (1)	Sedang (2)	Banyak (3)
DEC = Keputusan	Menyerang	Bertahan	

Seperti pada *decision tree* pertama, akan dibuat kumpulan data yang akan digunakan untuk membentuk tabel untuk memberikan

hasil keputusan sesuai dengan yang masukannya. Untuk kumpulan data dari *decision tree* yang kedua dapat dilihat pada Tabel 3.5 dengan keterangan kode angka pada Tabel 3.4.

Tabel 3.5: Hasil keputusan *decision tree* menyerang atau bertahan

DCR	CR	LINE	DEC
1	1	1	1
1	1	2	2
1	1	3	2
1	2	1	1
1	2	2	1
1	2	3	1
2	1	1	1
2	1	2	1
2	1	3	2
2	2	1	1
2	2	2	1
2	2	3	1
3	1	1	1
3	1	2	2
3	1	3	2
3	2	1	1
3	2	2	1
3	2	3	1

Hasil dari pengambilan keputusan dari tiap *decision tree* akan digunakan sebagai masukan untuk sistem DCA yang dibuat.

3.5 Implementasi *Spawning Unit* Menggunakan Algoritma Penyelesaian *Knapsack Problem*

Proses pemilihan secara otomatis suatu kombinasi data dibutuhkan dalam penerapan DCA pada model permainaan ini karena digunakan untuk menentukan kombinasi unit apa yang tepat dengan keterbatasan serta keadaan saat ini. Untuk itu digunakan model algoritma penyelesaian *knapsack problem*. Proses tersebut akan menentukan dari 4 macam unit dengan bobot yang berbeda

dan dengan keadaan sumber daya yang ada, berupa kombinasi unit yang pas dalam keadaan tersebut. Sebagai contoh dengan keadaan menyerang unit dengan bobot kekuatan serang yang tinggi akan didahulukan dibandingkan dengan unit yang memiliki bobot kekuatan serang yang sedikit. Begitu juga dengan keadaan bertahan sistem akan memilih unit dengan bobot ketahanan yang lebih untuk dikeluarkan dalam keadaan bertahan.

Dalam penyelesaian *knapsack problem* terdapat empat parameter inputan yaitu *value* atau nilai objek, *weight* atau bobot objek, macam objek atau benda, dan kapasitas dari kantong (*knapsack*). Dengan empat parameter tersebut untuk implementasi kedalam sistem DCA yang dibuat maka empat parameter itu masing-masing adalah

1. Nilai objek.
   Untuk nilai objek yang menjadi masukan adalah harga untuk mengeluarkan masing-masing tipe unit.
2. Bobot objek.
   Untuk bobot objek yang menjadi masukan adalah bobot tiap tipe unit yang telah diolah pada proses setelah pemilihan kondisi menyerang atau bertahan.
3. Macam objek.
   Merupakan nilai banyaknya tipe unit yang ada yaitu sejumlah empat.
4. Kapasitas.
   Kapasitas yang digunakan dan menjadi batasannya adalah jumlah sumber daya yang sedang dimiliki pada saat tersebut.

Setelah tiap masukan berhasil diolah maka hasil yang didapatkan adalah sebuah kombinasi tipe unit yang dibutuhkan sesuai dengan kondisi pada saat itu.

3.6 Implementasi NPC AI Tanpa DCA sebagai AI Tandingan

Dalam tugas akhir ini ingin dicapai suatu sistem AI yang dapat menyesuaikan kemampuannya terhadap berbagai kemampuan pemain. Oleh karena itu dibutuhkan AI lainnya yang akan di adu kemampuannya sehingga dapat diukur kemampuan AI dengan DCA apakah sudah dapat mengatur kemampuannya.

Untuk AI yang menjadi lawan akan disebut dengan AI tan-

dingan. AI tandingan ini akan mengikuti sistem dari NPC AI tanpa DCA, namun akan memiliki tingkat kesulitan yang dibedakan. Pada permainan pada umumnya tingkat kesulitan digunakan untuk memberikan tantangan kepada pemain. Pada AI ini untuk memberikan tingkat tantangan yang berbeda akan dibedakan menjadi tiga tingkat kesulitan yaitu mudah, sedang dan sukar.

Perbedaan dari tiap tingkat kesulitan tersebut terletak pada kecepatan sumber daya untuk bertambah. Pada kondisi normal kecepatan bertambahnya sumber daya yang telah didesain dalam sistem adalah 10 per 0.2 detik. Untuk memberikan perbedaan tingkat tantangan maka diubah jumlah penambahan sumber daya tersebut, masing-masing seperti berikut:

1. Tingkat Kesulitan Mudah: 8 sumber daya per 0.2 detik (40 sumber daya per detik).
2. Tingkat Kesulitan Sedang: 10 sumber daya per 0.2 detik (50 sumber daya per detik).
3. Tingkat Kesulitan Sukar: 12 sumber daya per 0.2 detik (60 sumber daya per detik).

Alasan memilih peningkatan sumber daya sebagai penentu tingkat kesulitan dikarenakan sumber daya menjadi bagian penting yang akan meningkatkan nilai ancaman. Nilai ancaman yang dimaksud adalah populasi unit yang dapat dimunculkan dalam kurun waktu tertentu. Karena untuk mengeluarkan sebuah unit dibutuhkan sumber daya maka jika penambahan sumber daya lebih banyak dari normal akan membuat lebih cepat untuk mengeluarkan unit baru atau lebih cepat mengeluarkan unit termahal. Dengan adanya unit didalam arena maka akan meningkatkan nilai ancaman terhadap lawannya karena unit tersebut pasti akan menyerang kearah bangunan utama lawannya.

Sebagai perbandingan secara matematis, jika permainan berlangsung dalam jangka waktu 3 menit (180 detik) dan tidak terjadi peperangan di arena dan tipe unit yang dikeluarkan adalah unit termurah (150 sumber daya) tanpa memikirkan waktu tunggu, maka dapat dirumuskan total jumlah populasi unit yang dapat dibuat dalam jangka waktu tersebut adalah:

1. Tingkat Kesulitan Mudah (40 sumber daya per detik)
   $TotalPasukan = (180 * 40)/150 = 48 unit$
2. Tingkat Kesulitan Sedang (50 sumber daya per detik)

$TotalPasukan = (180 * 50)/150 = 60unit$

3. Tingkat Kesulitan Sukar (60 sumber daya per detik)
   $TotalPasukan = (180 * 60)/150 = 72unit$

Sehingga dapat dilihat bahwa dengan tingkat kesulitan sukar maka jumlah unit yang dapat dihasilkan akan lebih banyak dibandingkan dengan tingkat kesulitan dibawahnya, begitu juga sebaliknya tingkat kesulitan mudah akan menghasilkan unit tersedikit sehingga ancamannya lebih rendah dari tingkat kesulitan lainnya.

BAB 4
PENGUJIAN DAN ANALISA

Bab ini akan dipaparkan hasil pengujian dan analisa dari desain sistem dan implementasi yang telah dibahas pada bab 3. Pengujian akan ditujukan untuk memperlihatkan apakah sistem permainan telah berjalan dengan baik serta implementasi kecerdasan buatan menggunakan DCA telah memberikan hasil yang optimal. Untuk dapat memperlihatkan hasil kemampuan kecerdasan buatan akan dilakukan pengukuran dengan memperlihatkan hasil grafik dan juga hasil kalkulasi rata-rata nilai CR tiap waktu dari grafik yang ditampilkan. Pengujian dilakukan menggunakan editor pada perangkat lunak Unity versi 5.0 pada sistem operasi Windows 8. Bentuk pengujian yang dilakukan penulis untuk menguji kemampuan AI dengan DCA adalah sebagai berikut.

4.1 Pengujian Kesesuaian Sistem Permainan

Pada pengujian kesesuaian sistem permainan, hal yang diamati adalah apakah permainan ini telah bekerja sesuai dengan desain sistem yang diinginkan. Pengujian ini dilakukan dengen mencoba seluruh proses dalam permainan dan mencoba fungsi-fungsi yang ada apakah telah berjalan dengan baik atau tidak. Hasil dari pengujian seluruh fungsi berjalan dengan sesuai seperti pada desain sistem. Hasil yang didapatkan dari pengujian kesesuaian sistem permainan ini terdapat pada Tabel 4.1.

Dari tabel dapat dilihat bahwa keseluruhan sistem permainan telah dapat berjalan dengan baik sehingga permainan dapat menjadi representatif dari permainan RTS *Tower Defense*.

4.2 Pengujian Sistem Kecerdasan Buatan

Pengujian sistem kecerdasan buatan dilakukan untuk memberikan data apakah kecerdasan buatan dengan DCA telah berhasil membuat kedinamisan tingkat kesulitan. Untuk dapat memperlihatkan hasil tersebut perlu dilakukan pengujian terlebih dahulu terhadap kecerdasan buatan tanpa DCA. Oleh karena itu pada pengujian sistem kecerdasan buatan akan dibagi menjadi dua bagian yaitu pengujian kemampuan AI tanpa DCA melawan AI tanpa DCA

Tabel 4.1: Hasil pengujian fungsi

No	Nama Jenis Fungsi	Fungsi dapat berjalan
1	Tombol Mengeluarkan Unit	Ya
2	Unit Idle	Ya
3	Unit Berjalan	Ya
4	Unit Menyerang	Ya
5	Unit Mati	Ya
6	Pewaktu	Ya
7	Tombol Pause	Ya
8	Kondisi Selesai	Ya

dan yang kedua adalah pengujian kemampuan AI dengan DCA melawan AI tanpa DCA. Untuk dapat mengukur apakah nilai yang dihasilkan telah seimbang akan dilakukan penghitungan rata-rata selisih nilai CR tiap waktu dari hasil tiap pertandingan. Penghitungan tersebut dilakukan dengan menselisihkan nilai CR antara pasukan kubu kiri dengan kubu yang kanan.

4.2.1 AI tanpa DCA dengan Tingkat Kesulitan Setara

Pengujian kemampuan tingkat kesulitan akan didasari dengan tiga macam tingkat kesulitan dasar yang digunakan yaitu mudah (*easy*), sedang (*medium*), dan sukar (*hard*). Dalam pengujian ini ditampilkan data berupa lama suatu pertandingan dan nilai dari *challenging rate* (CR). Pada pengujian ini diuji kondisi permainan antara dua kubu pasukan yang memiliki tingkat kesulitan yang sama. Pengujian ini bertujuan untuk memperlihatkan bagaimana kondisi permainan jika tingkat kemampuan yang digunakan setara, sehingga dapat digunakan untuk menyimpulkan apakah kedinamisan kecerdasan buatan menggunakan DCA sudah dapat menggantikan tingkat kesulitan yang statis. Proses pengujian tingkat kesulitan setara dibagi menjadi tiga macam percobaan sebagai berikut:

1. AI tanpa DCA *Easy* melawan AI tanpa DCA *Easy*.

   Pengujian dengan mengadu antara pasukan dengan tingkat

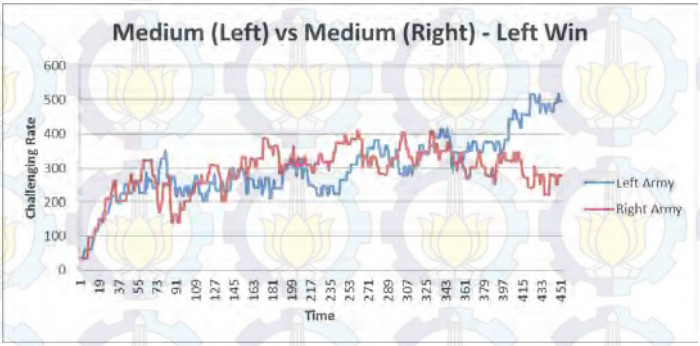
kesulitan *easy* melawan *easy*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.1. Dalam beberapa kali percobaan diambil sepuluh sampel data yang ditunjukan pada Tabel 4.2. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 290,4 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 27,46.

Gambar 4.1: Salah satu hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Easy*

Tabel 4.2: Hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Easy*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	222	12,91	Easy Left
2	251	27,96	Easy Right
3	255	34,37	Easy Right
4	475	23,94	Easy Right
5	308	33,90	Easy Left
6	318	30,48	Easy Right
7	232	35,02	Easy Left
8	181	16,16	Easy Left
9	223	35,58	Easy Left
10	439	24,26	Easy Right
Mean	290,4	27,46	

2. AI tanpa DCA *Medium* melawan AI tanpa DCA *Medium*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *medium* melawan *medium*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.2. Dalam beberapa kali percobaan diambil sepuluh sampel data yang ditunjukan pada Tabel 4.3. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 321 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 32,66.



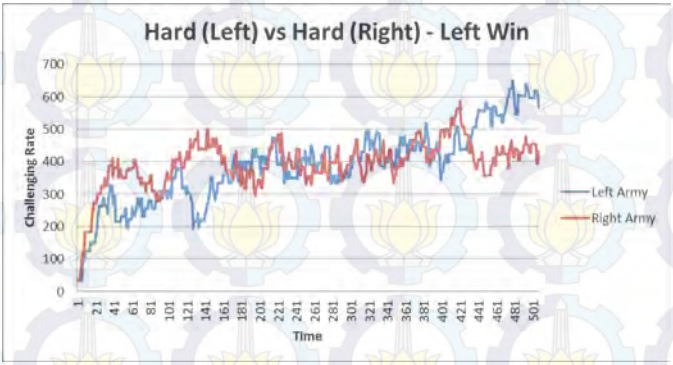
Gambar 4.2: Salah satu hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Medium*

3. AI tanpa DCA *Hard* melawan AI tanpa DCA *Hard*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *hard* melawan *hard*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.3. Dalam beberapa kali percobaan diambil sepuluh sampel data yang ditunjukan pada Tabel 4.4. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 321,8 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 40,84.

Dari pengujian tingkat kesulitan setara dapat dirata-rata nilai rata-rata selisih nilai CR tiap waktu adalah sebesar 33,65 seperti pada Tabel 4.5. Nilai tersebut terjadi karena nilai tingkat tantangan yang dihasilkan selalu setara, karena keadaan yang dimiliki antar kubu tidak memiliki perbedaan parameter yang berbeda, sehingga

Tabel 4.3: Hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Medium*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	450	19,14	Medium Left
2	254	26,35	Medium Left
3	404	41,53	Medium Right
4	526	25,49	Medium Left
5	128	35,23	Medium Left
6	192	30,67	Medium Right
7	271	49,12	Medium Right
8	445	35,99	Medium Left
9	202	32,60	Medium Left
10	338	30,50	Medium Left
Mean	321	32,66	



Gambar 4.3: Salah satu hasil pengujian AI tanpa DCA *Hard* melawan AI tanpa DCA *Hard*

proses ini dapat menjadi contoh model hasil pertandingan antara pemain yang memiliki kemampuan yang sama dengan sistem yang dibangun dalam suatu permainan.

4.2.2 AI tanpa DCA dengan Tingkat Kesulitan Berbeda

Untuk memberikan data mengenai ketidakseimbangan dalam permainan, maka dilakukan pengujian dengan mengadu NPC de-

Tabel 4.4: Hasil pengujian AI tanpa DCA *Hard* melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	506	37,13	Hard Left
2	384	44,38	Hard Right
3	179	49,30	Hard Right
4	200	46,88	Hard Left
5	303	47,71	Hard Right
6	358	39,95	Hard Right
7	327	43,34	Hard Left
8	380	25,56	Hard Left
9	367	41,17	Hard Left
10	214	32,95	Hard Left
Mean	321,8	40,84	

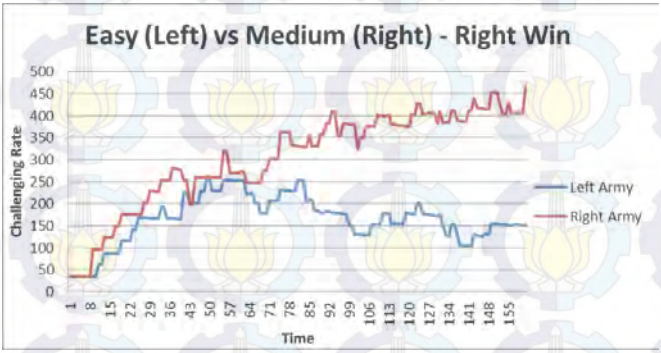
Tabel 4.5: Hasil pengujian AI tanpa DCA tingkat kesulitan setara

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Keterangan
1	290,4	27,46	Easy vs Easy
2	321	32,66	Medium vs Medium
3	321,8	40,84	Hard vs Hard
Mean	311,07	33,65	

ngan tingkat kesulitan yang berbeda. Hal ini perlu dilakukan karena dibutuhkan suatu ambang pembanding antara nilai yang dianggap sudah cukup seimbang dan belum seimbang. Dengan mengadu NPC dengan tingkat kesulitan berbeda akan didapatkan nilai ambang untuk data yang belum seimbang. Pengujian ini akan dilakukan dengan membandingkan tiga kombinasi tingkat kesulitan dasar yaitu:

1. AI tanpa DCA *Easy* melawan AI tanpa DCA *Medium*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *easy* melawan *medium*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.4. Sama seperti pada pengujian sebelumnya diambil sepuluh sampel data yang ditunjukan pada Tabel 4.6. Dalam tabel ter-

sebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 159,6 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 74,00.



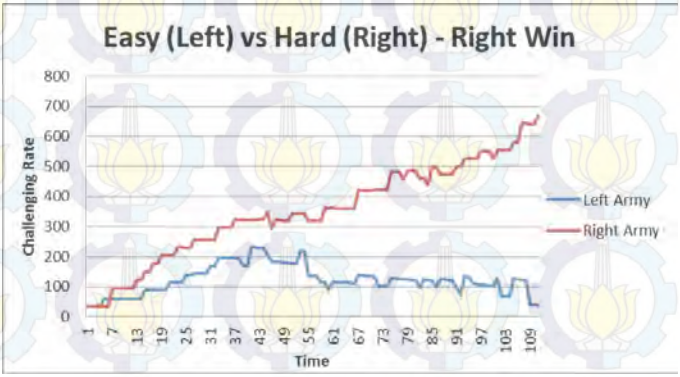
Gambar 4.4: Salah satu hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Medium*

Tabel 4.6: Hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Medium*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	160	70,17	Medium
2	231	62,35	Medium
3	160	89,12	Medium
4	120	82,90	Medium
5	127	88,90	Medium
6	138	77,41	Medium
7	149	60,43	Medium
8	160	58,45	Medium
9	111	92,68	Medium
10	240	57,55	Medium
Mean	159,6	74,00	

2. AI tanpa DCA *Easy* melawan AI tanpa DCA *Hard*. Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *easy* melawan *hard*. Dalam pengujian ini didapatk-

an salah satu hasil grafik yang dapat dilihat pada Gambar 4.5. Pengujian dilakukan berulang kali, dengan sepuluh sampel data memberikan hasil data yang ditunjukan pada Tabel 4.7. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 120,40 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 104,09.

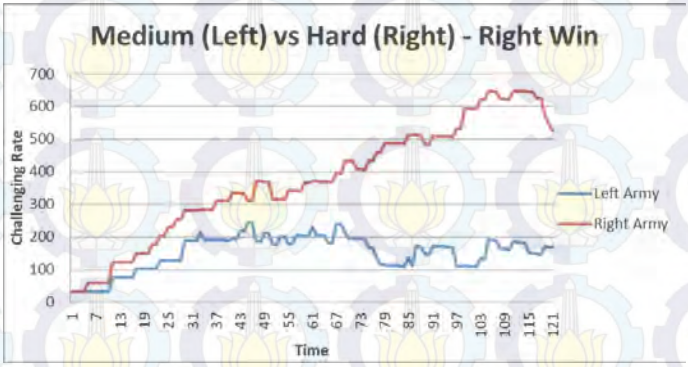


Gambar 4.5: Salah satu hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Hard*

Tabel 4.7: Hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	109	115,09	Hard
2	114	103,99	Hard
3	121	99,53	Hard
4	159	90,93	Hard
5	112	97,16	Hard
6	114	108,85	Hard
7	134	109,37	Hard
8	112	103,67	Hard
9	84	101,28	Hard
10	145	111,06	Hard
Mean	120,40	104,09	

3. AI tanpa DCA *Medium* melawan AI tanpa DCA *Hard*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *medium* melawan *hard*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.6. Pengujian dilakukan berulang kali, dengan mengambil sepuluh sampel data dihasilkan data yang ditunjukan pada Tabel 4.8. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 143,9 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 90,82.



Gambar 4.6: Salah satu hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Hard*

Dari hasil pengujian ini dapat disimpulkan bahwa dengan mengadu AI dengan tingkat kesulitan yang berbeda, maka hasil yang didapatkan adalah NPC dengan tingkat kesulitan yang lebih tinggi akan selalu mendominasi permainan. Hal ini dapat dilihat dari hasil grafik yang ada dan rata-rata dari rata-rata selisih nilai CR tiap waktu sebesar 89,63 yang ditampilkan pada Tabel 4.9. Data CR dari NPC yang menggunakan AI dengan tingkat kesulitan lebih tinggi akan memiliki CR yang meningkat secara cepat sehingga nilai selisih yang dihasilkan besar, sedangkan dengan tingkat kesulitan yang sama NPC akan memiliki nilai CR yang stabil atau nilai selisih yang kecil. Untuk parameter waktu permainan, dengan mengadu tingkat kesulitan yang berbeda, lama waktu bermain akan lebih cepat di-

Tabel 4.8: Hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	119	101,56	Hard
2	223	78,26	Hard
3	129	108,13	Hard
4	118	110,35	Hard
5	138	82,24	Hard
6	112	90,23	Hard
7	145	77,07	Hard
8	182	72,10	Hard
9	114	103,84	Hard
10	159	84,40	Hard
Mean	143,9	90,82	

bandingkan dengan mengadu tingkat kesulitan yang setara. Sesuai

Tabel 4.9: Hasil pengujian AI tanpa DCA tingkat kesulitan tidak setara

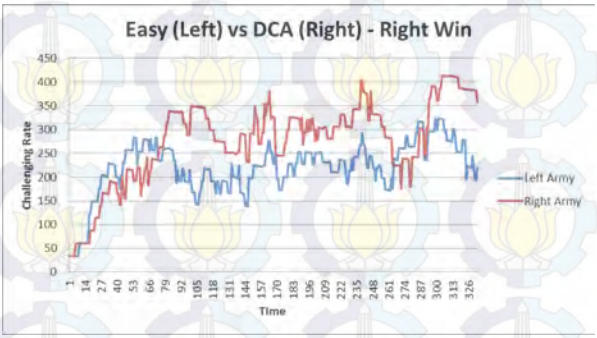
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Keterangan
1	159,6	74,00	Easy vs Medium
2	120,40	104,09	Easy vs Hard
3	143,9	90,82	Medium vs Hard
Mean	141,3	89,63	

dengan persamaan 2.1 yang digunakan untuk mengukur tingkat tantangan (CR), parameter populasi sangat menentukan kenaikan dari nilai CR karena dengan penambahan populasi maka akan otomatis meningkatkan nilai serangan dan durabilitas. Penambahan populasi yang sangat tinggi ini diakibatkan karena model perbedaan tingkat kesulitan yang digunakan mengakibatkan perbedaan kecepatan suatu kubu untuk mengeluarkan unit untuk menyerang atau bertahan.

4.2.3 Pengujian Kemampuan AI dengan DCA Melawan AI tanpa DCA

Pengujian ini dilakukan untuk memperlihatkan bagaimana kemampuan dari AI dengan DCA apakah dapat memberikan keseimbangan dalam tingkat kesulitan atau tidak. Pengujian ini dilakukan dengan mengadu AI menggunakan DCA dengan AI tanpa DCA yang memiliki tiga macam tingkat kesulitan. Proses pengujian sama seperti pada proses pengujian sebelumnya, dan dihitung juga hasil rata-rata selisih nilai CR tiap waktu agar dapat dilihat apakah kemampuan AI menggunakan DCA akan dapat memberikan keseimbangan terhadap tingkat kesulitan. Pengujian yang dilakukan adalah sebagai berikut:

1. Pengujian Kemampuan AI dengan DCA melawan AI tanpa DCA *Easy*.
   Salah satu hasil pengujian ditampilkan dalam grafik sesuai pada Gambar 4.7. Pengujian dengan mengadu AI dengan DCA melawan AI tanpa DCA dilakukan berkali-kali dengan mengambil sepuluh sampel yang dapat dilihat pada Tabel 4.10.



Gambar 4.7: Salah satu hasil pengujian AI dengan DCA melawan AI tanpa DCA *Easy*

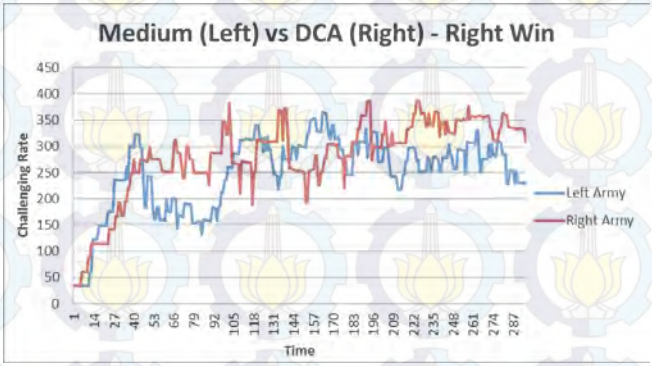
   Dari tabel tersebut didapatkan data rata-rata waktu permainan untuk melawan AI tanpa DCA *easy* adalah 252 detik, dan dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 37,49.
2. Pengujian Kemampuan AI dengan DCA melawan AI tanpa

Tabel 4.10: Hasil pengujian AI tanpa DCA *Easy* melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	331	39,12	DCA
2	138	39,12	DCA
3	250	34,17	DCA
4	241	30,20	Easy
5	193	43,23	DCA
6	242	34,11	DCA
7	391	44,92	DCA
8	246	41,31	Easy
9	242	27,95	Easy
10	250	40,79	DCA
Mean	252	37,49	

DCA *Medium*.

Salah satu hasil pengujian ditampilkan dalam bentuk grafik sesuai dengan Gambar 4.8. Pengujian dengan mengadu AI dengan DCA melawan AI tanpa DCA dilakukan berkali-kali dengan mengambil sepuluh sampel yang dapat dilihat pada Tabel 4.11.



Gambar 4.8: Salah satu hasil pengujian AI dengan DCA melawan AI tanpa DCA *Medium*

Dari tabel tersebut didapatkan data rata-rata waktu perma-

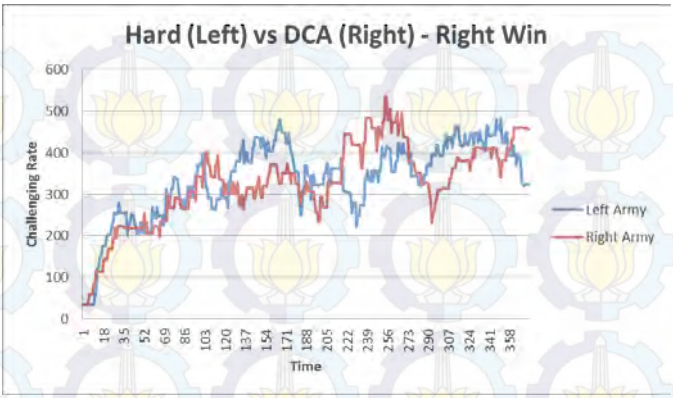
Tabel 4.11: Hasil pengujian AI tanpa DCA *Medium* melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	293	37,96	DCA
2	485	30,79	DCA
3	162	17,03	Medium
4	254	27,82	DCA
5	308	38,42	DCA
6	322	38,74	DCA
7	259	23,79	Medium
8	327	41,86	DCA
9	178	26,39	Medium
10	303	31,82	DCA
Mean	289,1	31,46	

inan untuk melawan AI tanpa DCA *medium* adalah 289,10 detik, dan dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 31,46.

3. Pengujian Kemampuan AI dengan DCA melawan AI tanpa DCA *Hard*.
   Salah satu hasil pengujian ditampilkan dalam bentuk grafik sesuai dengan Gambar 4.9. Pengujian dengan mengadu AI dengan DCA melawan AI tanpa DCA dilakukan berkali-kali dengan mengambil sepuluh sampel yang dapat dilihat pada Tabel 4.12.
   Dari tabel tersebut didapatkan data rata-rata waktu permainan untuk melawan AI tanpa DCA *Hard* adalah 299,7 detik, dan dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 35,66.

Pengujian dengan menggunakan sistem DCA memberikan hasil yang terlihat seimbang dengan menghasilkan rata-rata selisih nilai CR tiap waktu yang lebih kecil dibandingkan pada pengujian dengan tingkat kesulitan yang berbeda. Hasil rata-rata dari rata-rata selisih nilai CR tiap waktu dari ketiga pengujian tersebut sebesar 34,02 yang ditampilkan pada Tabel 4.13. Sedangkan dengan pengujian AI tanpa DCA yang setara didapatkan rata-rata sebesar 33,65



Gambar 4.9: Salah satu hasil pengujian AI dengan DCA melawan AI tanpa DCA *Hard*

Tabel 4.12: Hasil pengujian AI tanpa DCA *Hard* melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	372	29,72	DCA
2	265	40,78	DCA
3	172	29,32	DCA
4	262	41,33	DCA
5	321	33,19	Hard
6	377	32,25	Hard
7	169	43,67	DCA
8	224	41,35	DCA
9	515	30,80	DCA
10	320	34,23	DCA
Mean	299,7	35,66	

sedangkan AI tanpa DCA yang berbeda tingkat kesulitan memiliki rata-rata 89,63. Dari ketiga data tersebut dapat dilihat bahwa penggunaan DCA telah mampu menggantikan fungsi tingkat kesulitan statis karena hanya terpaut dalam 0,37 pada rata-rata selisih nilai CR tiap waktu.

Tabel 4.13: Hasil pengujian AI tanpa DCA melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Keterangan
1	252	37,49	DCA vs Easy
2	289,1	31,46	DCA vs Medium
3	299,7	35,66	DCA vs Hard
Mean	280,27	34,02	

Hal ini diakibatkan karena dengan menggunakan DCA, sistem akan mempertahankan kondisi nilai tingkat tantangan (CR) sesuai dengan syarat atau aturan pada *decision tree* yang diimplementasikan pada sistem DCA. Sesuai dengan persamaan 2.1 untuk mengukur CR, DCA akan mengatur jumlah populasi yang dikeluarkan dan juga memilih unit mana yang terbaik untuk dikeluarkan sehingga CR yang dihasilkan tidak akan terpaut terlalu jauh dengan lawannya. Dengan *decision tree* juga, DCA mengatur keadaan sehingga lawan tidak akan memenangkan pertandingan dengan mudah dilihat dari hasil pemenang yang tidak selalu sama. Dari pengujian dua macam model pengujian antara kecerdasan buatan tanpa DCA dan dengan DCA dapat dilihat bahwa hasil menggunakan DCA sudah mendekati hasil pada kecerdasan buatan dengan tingkat kesulitan yang sama. Dari data tingkat kemenangan DCA masih memiliki probabilitas kemenangan lebih tinggi yaitu 73.33% (8 kali kalah dalam 30 data sampel). Hal ini kemungkinan diakibatkan oleh kemampuan DCA masih terlalu tinggi untuk ditandingkan dengan AI tanpa DCA yang telah didesain untuk saat ini. Keterangan mengenai grafik dan data sampel yang lebih detail akan ditampilkan pada bagian lampiran dari buku tugas akhir ini.

BAB 5
PENUTUP

5.1 Kesimpulan

Dari hasil implementasi dan pengujian kemampuan kecerdasan buatan dengan DCA yang sudah dilakukan dapat ditarik beberapa kesimpulan:

1. Hasil pengujian antara AI tanpa DCA dengan tingkat tantangan yang setara akan memberikan hasil kemenangan acak antara satu kubu dengan yang lain. Karena keadaan tiap kubu yang seimbang maka penentuan kemenangan tidak memiliki faktor penentu yang pasti dan memberikan hasil yang acak. Waktu yang dibutuhkan untuk menentukan pemenang cukup lama yaitu rata-rata 311,07 detik atau sekitar 5 menit 11 detik.
2. Hasil pengujian antara AI tanpa DCA dengan tingkat tantangan yang berbeda akan selalu menghasilkan kemenangan pada kubu yang menggunakan tingkat kesulitan yang lebih tinggi. Sebagai contoh tingkat kesulitan *hard* akan selalu menang melawan tingkat kesulitan *medium* dan *easy*. Karena tingkat kesulitan yang lebih tinggi memiliki keunggulan dalam kemampuan membuat pasukan lebih cepat. Waktu yang dibutuhkan untuk menentukan kemenangan lebih cepat dari ketika melawan tingkat tantangan yang setara yaitu rata-rata 141,3 detik atau sekitar 2 menit 21 detik.
3. Hasil pengujian antara AI tanpa DCA dengan tingkat kesulitan setara memiliki rata-rata dari rata-rata selisih nilai CR tiap waktu sebesar 33,65 dan dengan AI tanpa DCA dengan tingkat kesulitan berbeda memiliki rata-rata sebesar 89,63. Dari nilai tersebut dapat disimpulkan dengan mengadu kemampuan yang setara (dalam hal ini tingkat kesulitan yang setara) rata-rata dari rata-rata selisih nilai CR tiap waktu yang dihasilkan akan lebih kecil dan terpaut sebesar 55,98 dari kemampuan yang tidak setara.
4. Dengan penggunaan AI menggunakan DCA melawan berbagai macam tingkat kesulitan memberikan rata-rata dari rata-rata selisih nilai CR tiap waktu sebesar 34,02. Dengan nilai tersebut dibandingkan dengan nilai rata-rata pada AI tanpa DCA

dengan tingkat kesulitan setara memiliki perbedaan nilai yang kecil yaitu 0,37. Sehingga dapat disimpulkan AI menggunakan DCA telah berhasil untuk memiliki kemampuan yang memberikan kesetaraan atau keseimbangan tingkat kesulitan, meskipun lawannya berbeda tingkat kemampuan. Waktu yang dibutuhkan untuk menentukan kemenangan rata-rata adalah 280,27 detik atau sekitar 4 menit 36 detik.

5. Kemampuan AI dengan DCA masih tinggi untuk mendapatkan kemenangan. Dengan 30 data sampel, AI dengan DCA hanya mengalami kekalahan sebanyak delapan kali sehingga tingkat kemenangannya masih tinggi yaitu sebesar 73,33%. Hal ini dapat diakibatkan oleh kemampuan pemilihan keputusan yang masih belum banyak sehingga terdapat keadaan yang masih belum terkondisi.

5.2 Saran

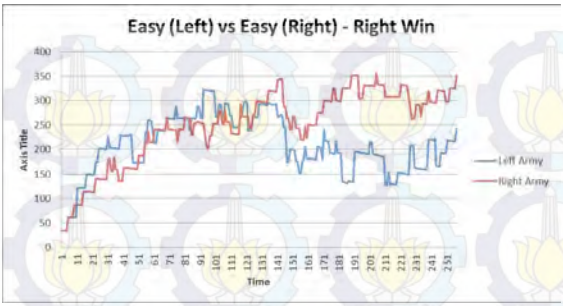
Untuk pengembangan lebih lanjut mengenai tugas akhir ini dan untuk memberikan hasil pengujian yang optimal, disarankan untuk melakukan beberapa langkah lanjutan:

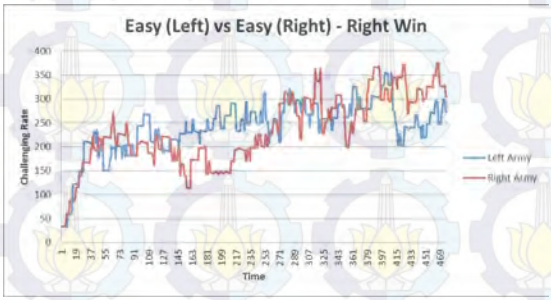
1. Pembuatan *decision tree* dengan aturan (*rule*) dan atribut yang lebih detail dan parameter masukan yang lebih banyak, agar keputusan yang dipilih lebih presisi.
2. Membuat model tipe unit yang lebih variatif, sehingga terdapat lebih banyak kombinasi unit pasukan yang dapat dibentuk.
3. Perombakan model alur kerja sistem DCA dengan mencoba mengganti pengambilan keputusan mengunakan *decision tree* menjadi *behaviour tree*.
4. Implementasi DCA pada model permainan yang lain seperti RTS *Tower Defense* yang berbasis *wave* atau gelombang serangan musuh.

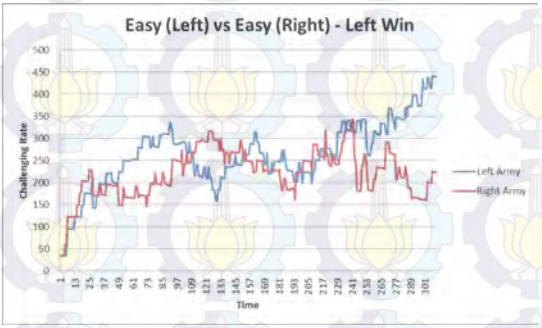
LAMPIRAN

1. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI tanpa DCA *Easy*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	222	12,91	Easy Left
2	251	27,96	Easy Right
3	255	34,37	Easy Right
4	475	23,94	Easy Right
5	308	33,90	Easy Left
6	318	30,48	Easy Right
7	232	35,02	Easy Left
8	181	16,16	Easy Left
9	223	35,58	Easy Left
10	439	24,26	Easy Right
Mean	290,4	27,46	


Easy (Left) vs Easy (Right) - Right Win
Axis Title
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

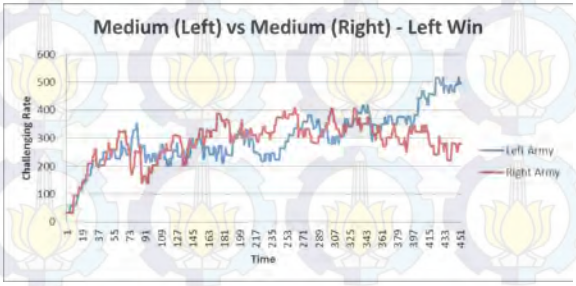

Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

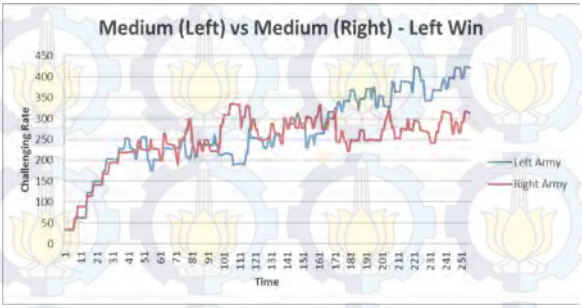

Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

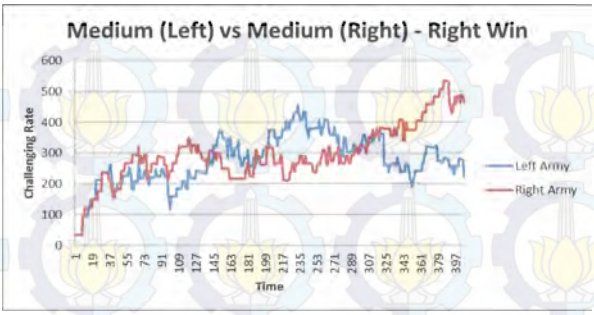

Easy (Left) vs Easy (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

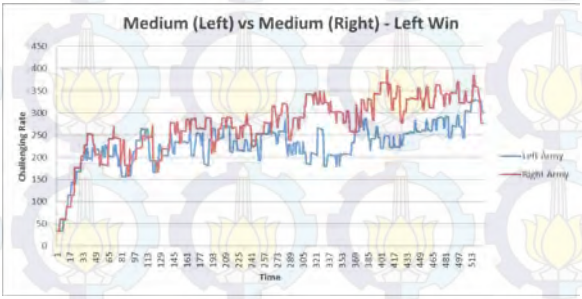
2. Hasil Pengujian AI tanpa DCA *Medium* Melawan AI tanpa DCA *Medium*

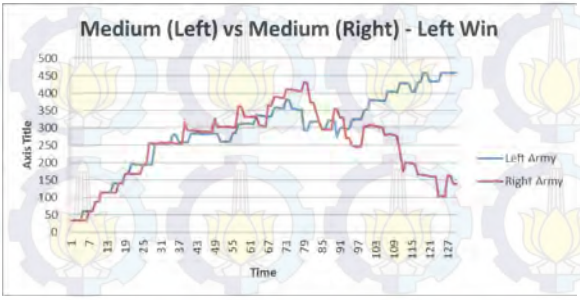
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	450	19,14	Medium Left
2	254	26,35	Medium Left
3	404	41,53	Medium Right
4	526	25,49	Medium Left
5	128	35,23	Medium Left
6	192	30,67	Medium Right
7	271	49,12	Medium Right
8	445	35,99	Medium Left
9	202	32,60	Medium Left
10	338	30,50	Medium Left
Mean	321	32,66	

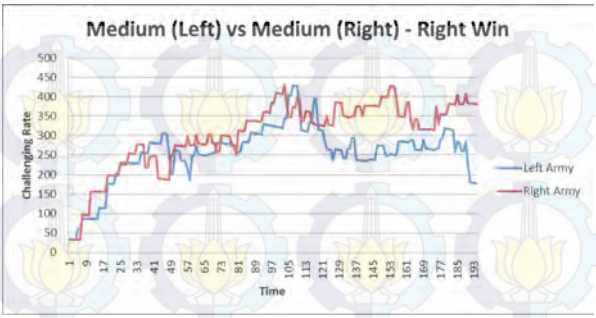


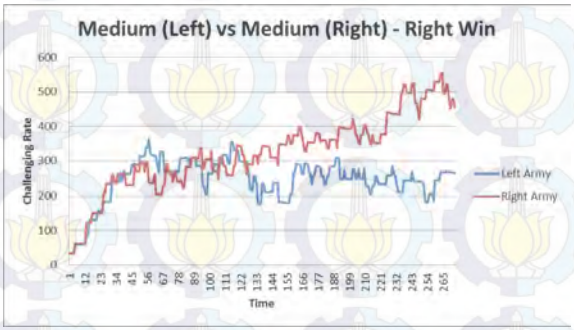


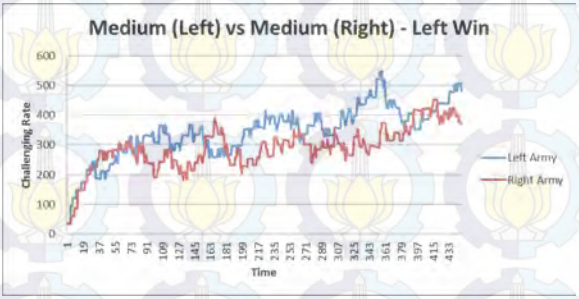

Medium (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

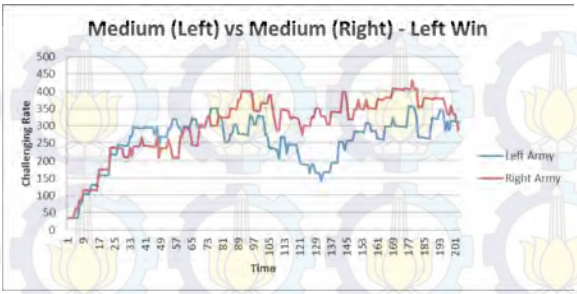

Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Left Win
Axis Title
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

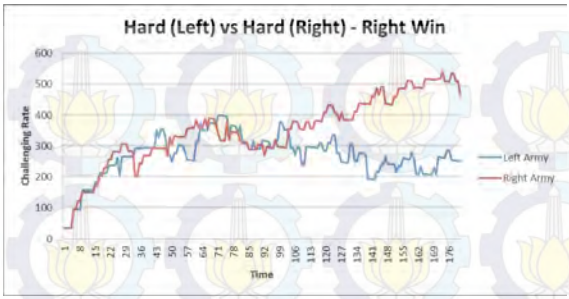

Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

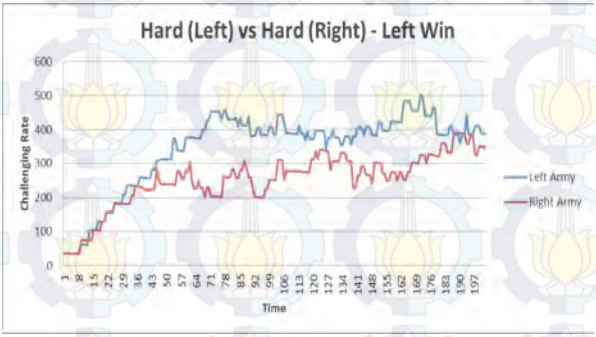

Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

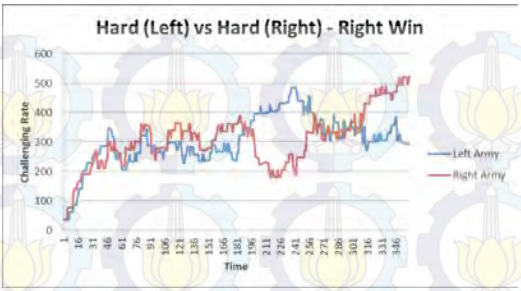
3. Hasil Pengujian AI tanpa DCA *Hard* Melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	506	37,13	Hard Left
2	384	44,38	Hard Right
3	179	49,30	Hard Right
4	200	46,88	Hard Left
5	303	47,71	Hard Right
6	358	39,95	Hard Right
7	327	43,34	Hard Left
8	380	25,56	Hard Left
9	367	41,17	Hard Left
10	214	32,95	Hard Left
Mean	321,8	40,84	

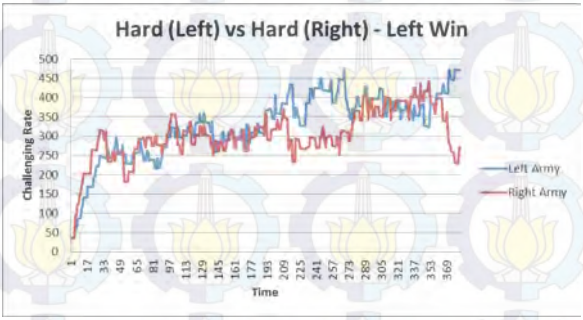

Hard (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

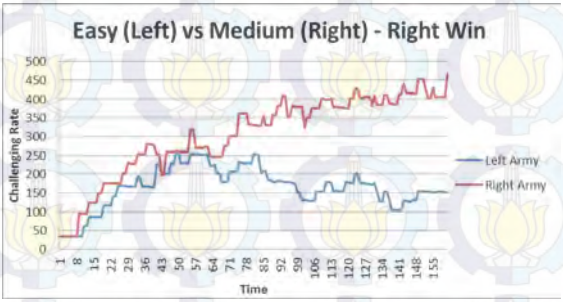

Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

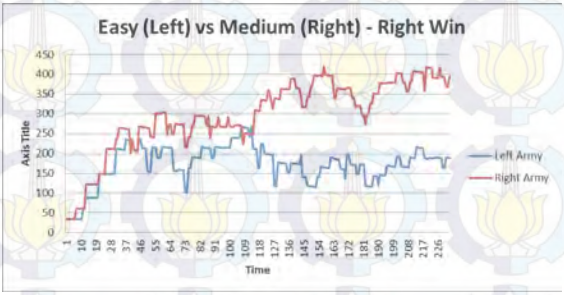

Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

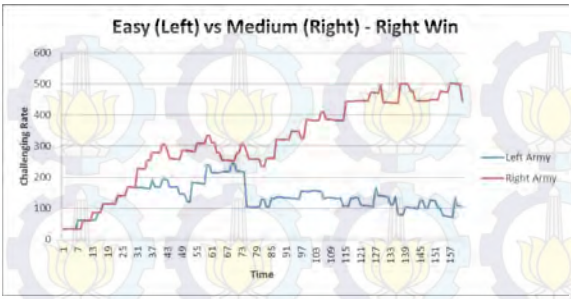

Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

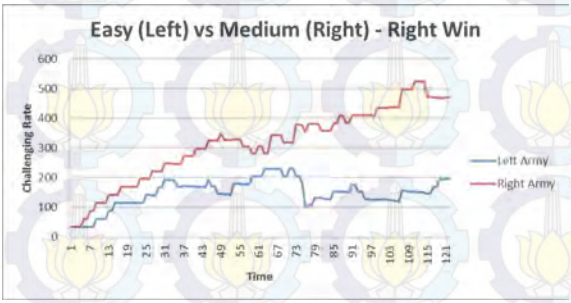
4. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI tanpa DCA *Medium*

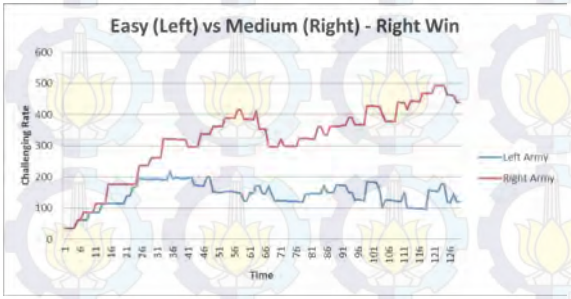
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	160	70,17	Medium
2	231	62,35	Medium
3	160	89,12	Medium
4	120	82,90	Medium
5	127	88,90	Medium
6	138	77,41	Medium
7	149	60,43	Medium
8	160	58,45	Medium
9	111	92,68	Medium
10	240	57,55	Medium
Mean	159,6	74,00	

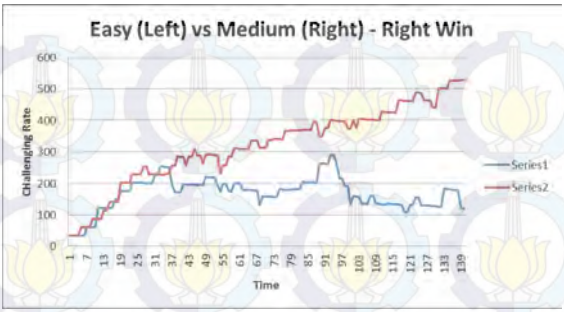


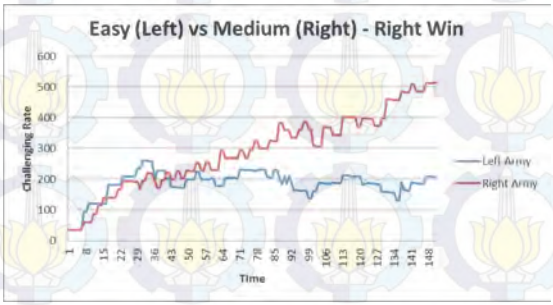


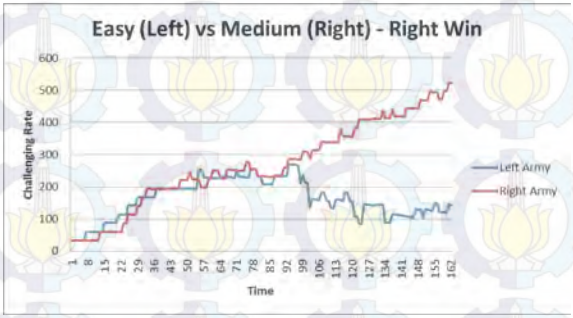

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

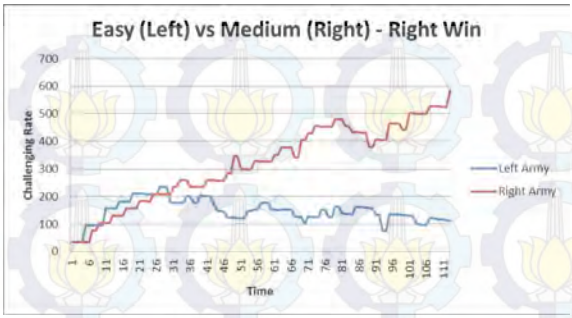

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

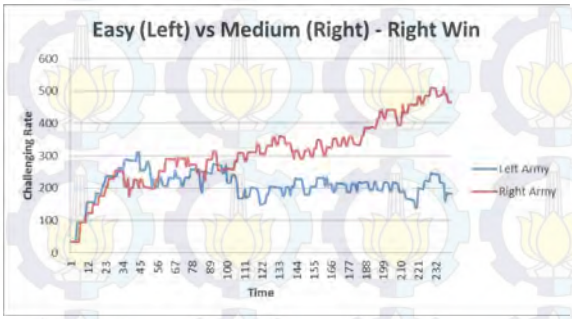

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Series1
Series2


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

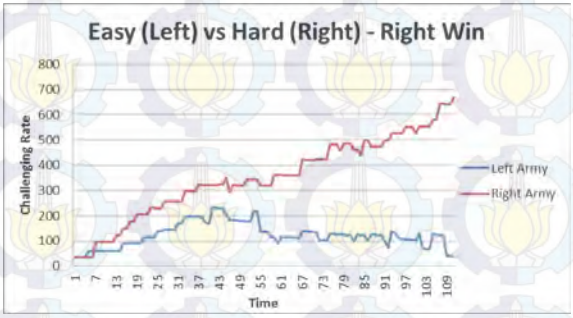

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

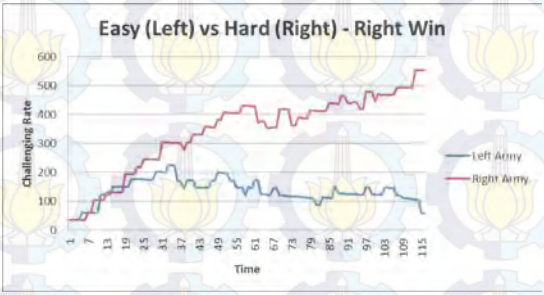

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

5. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI tanpa DCA *Hard*

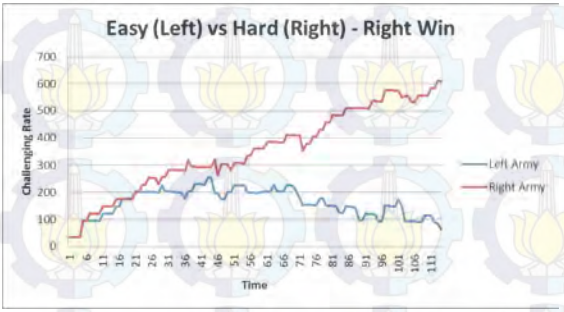
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	109	115,09	Hard
2	114	103,99	Hard
3	121	99,53	Hard
4	159	90,93	Hard
5	112	97,16	Hard
6	114	108,85	Hard
7	134	109,37	Hard
8	112	103,67	Hard
9	84	101,28	Hard
10	145	111,06	Hard
Mean	120,40	104,09	



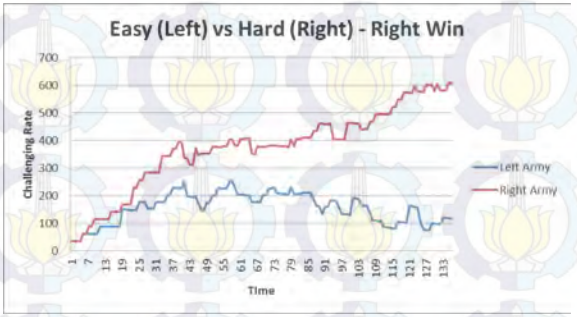


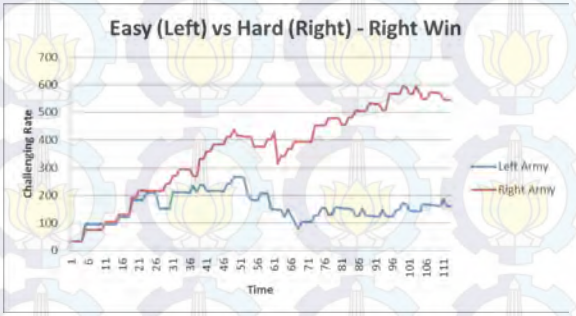

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

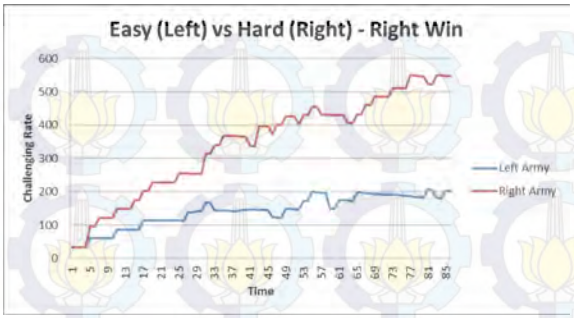

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

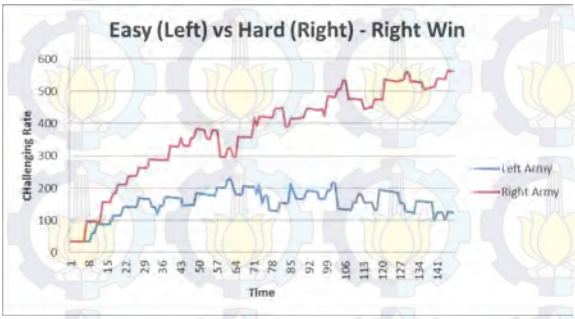

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

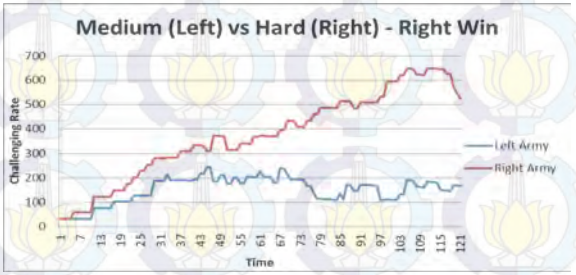

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

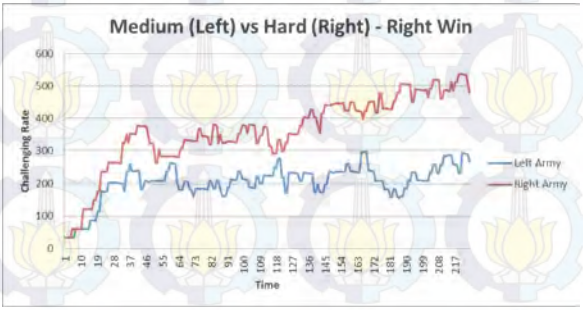

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
600
500
400
300
200
100
0
1
5
9
13
17
21
25
29
33
37
41
45
49
53
57
61
65
69
73
77
81
85
Time
Left Army
Right Army

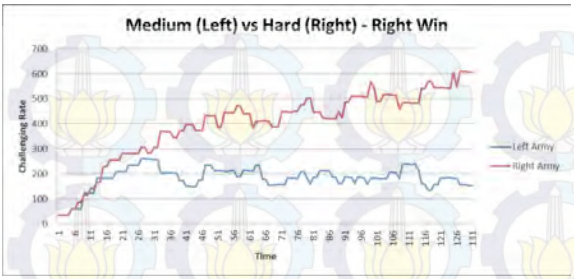

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
600
500
400
300
200
100
0
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
106
113
120
127
134
141
Time
Left Army
Right Army

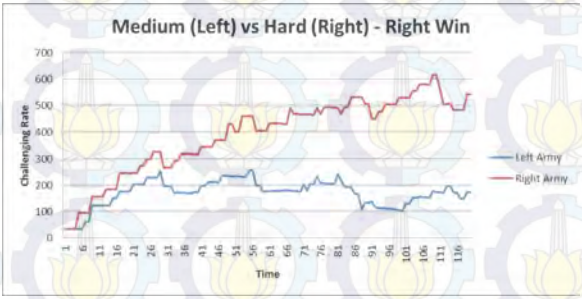
6. Hasil Pengujian AI tanpa DCA *Medium* Melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	119	101,56	Hard
2	223	78,26	Hard
3	129	108,13	Hard
4	118	110,35	Hard
5	138	82,24	Hard
6	112	90,23	Hard
7	145	77,07	Hard
8	182	72,10	Hard
9	114	103,84	Hard
10	159	84,40	Hard
Mean	143,9	90,82	



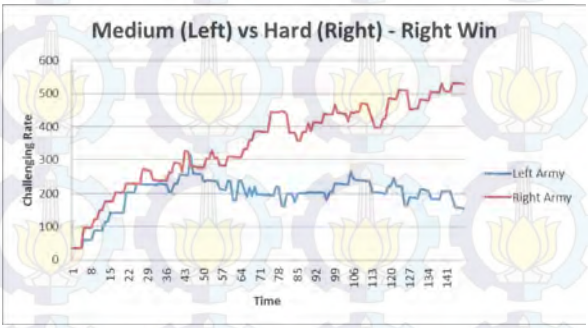


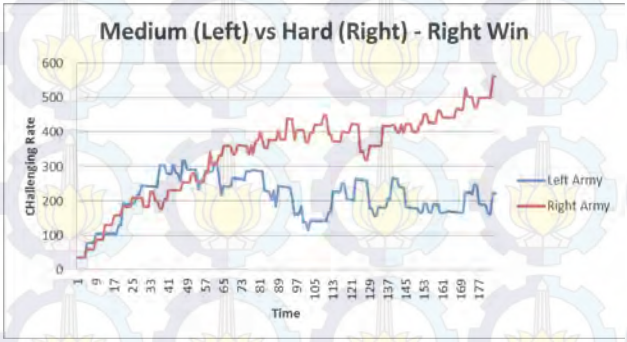

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

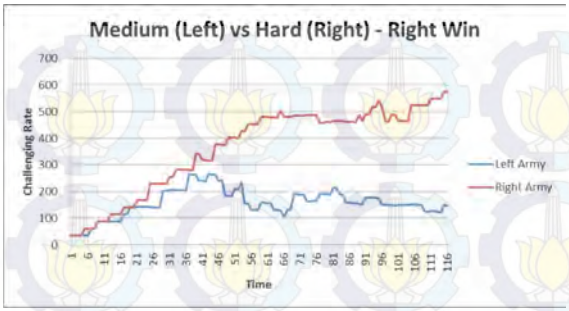

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

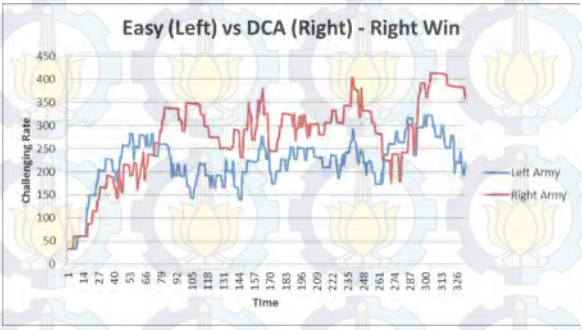

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

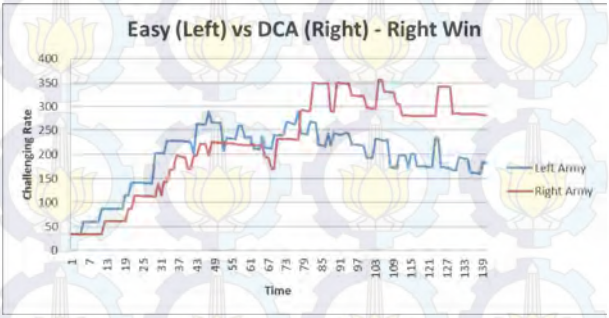

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

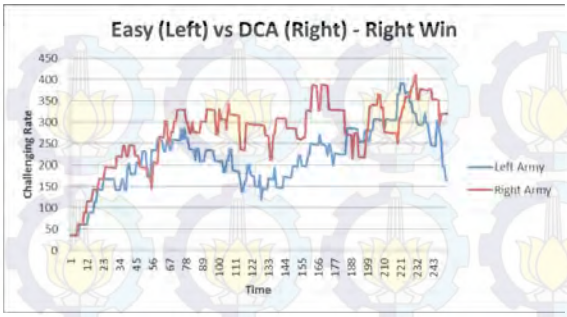

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

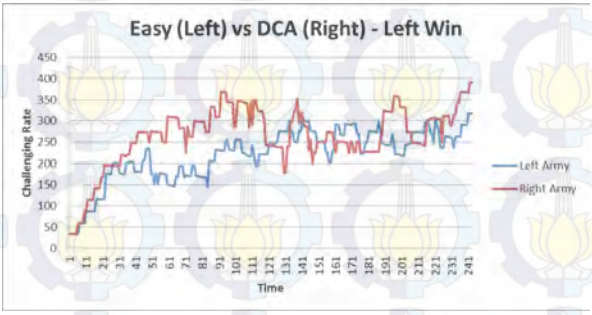
7. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI dengan DCA

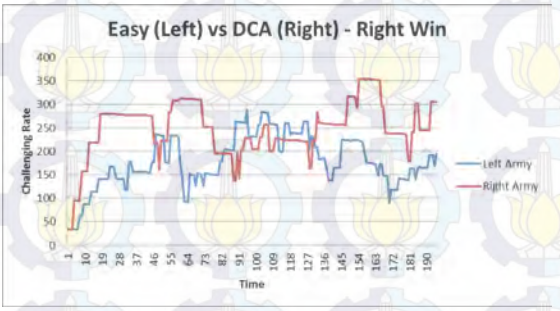
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	331	39,12	DCA
2	138	39,12	DCA
3	250	34,17	DCA
4	241	30,20	Easy
5	193	43,23	DCA
6	242	34,11	DCA
7	391	44,92	DCA
8	246	41,31	Easy
9	242	27,95	Easy
10	250	40,79	DCA
Mean	252	37,49	

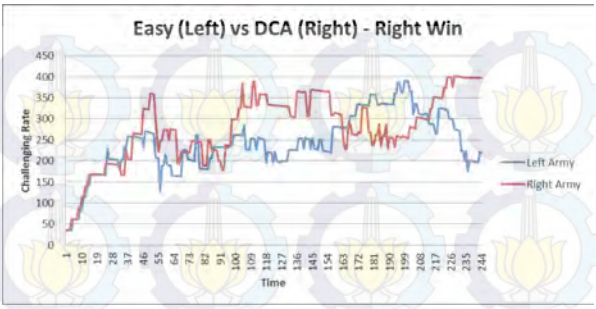


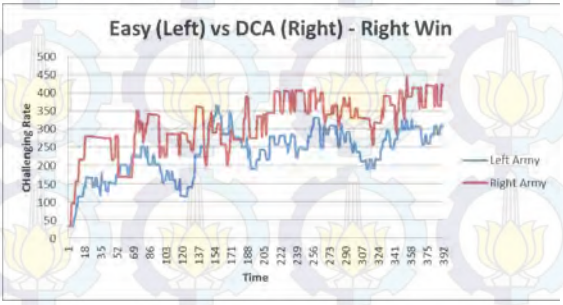


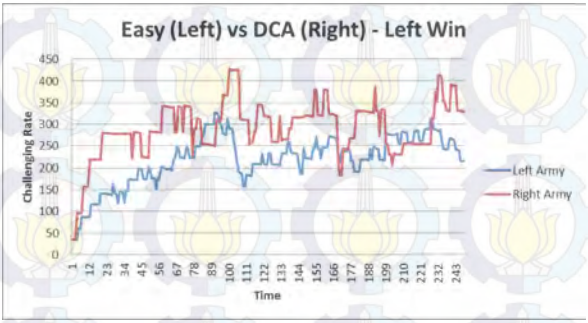

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

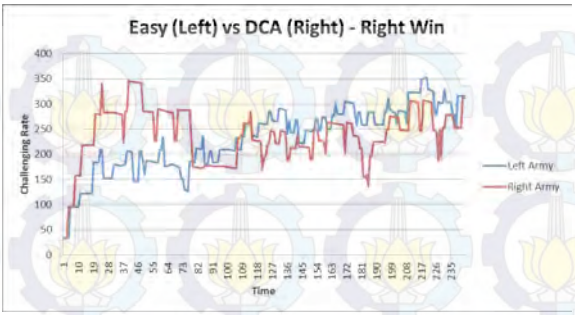

Easy (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

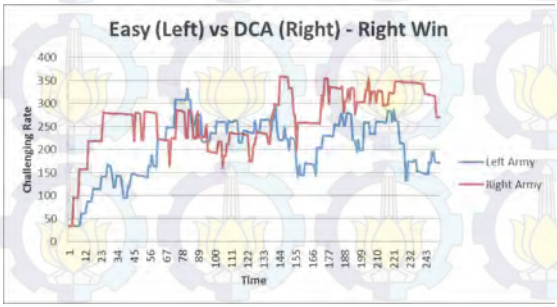

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

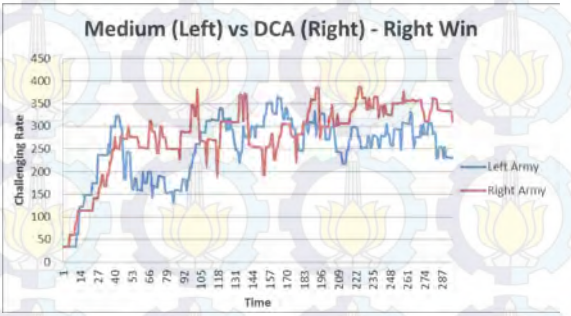

Easy (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

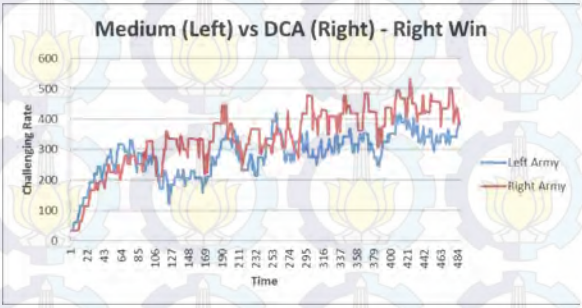

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

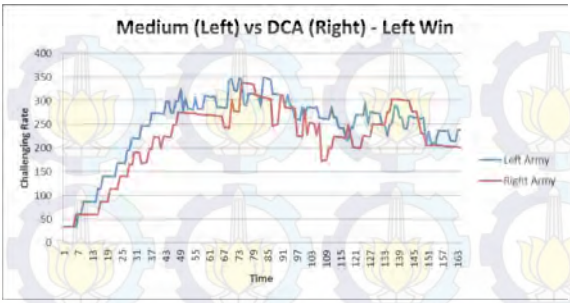

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

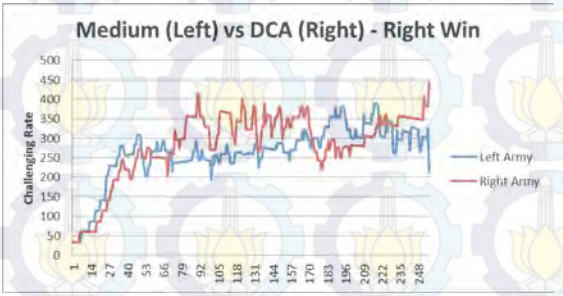
8. Hasil Pengujian AI tanpa DCA *Medium* Melawan AI dengan DCA

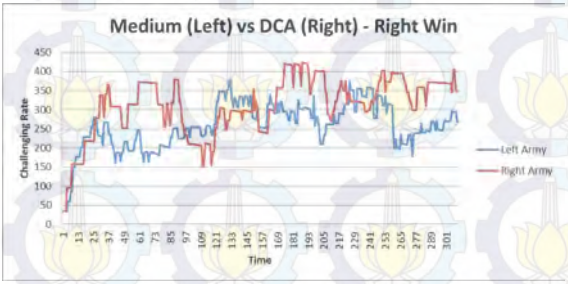
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	293	37,96	DCA
2	485	30,79	DCA
3	162	17,03	Medium
4	254	27,82	DCA
5	308	38,42	DCA
6	322	38,74	DCA
7	259	23,79	Medium
8	327	41,86	DCA
9	178	26,39	Medium
10	303	31,82	DCA
Mean	289,1	31,46	

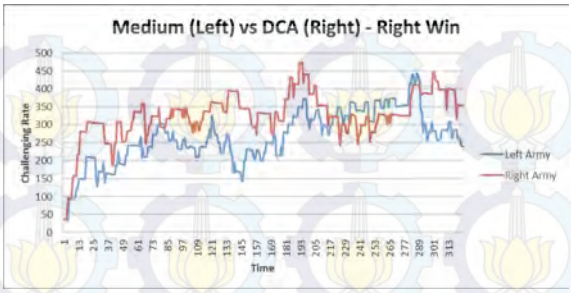


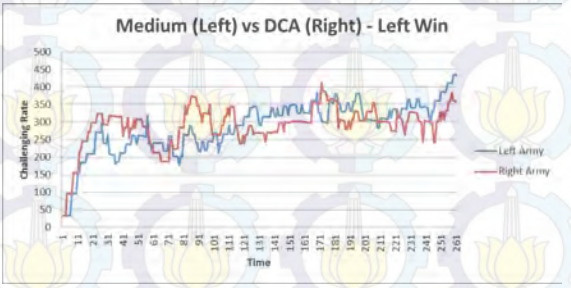


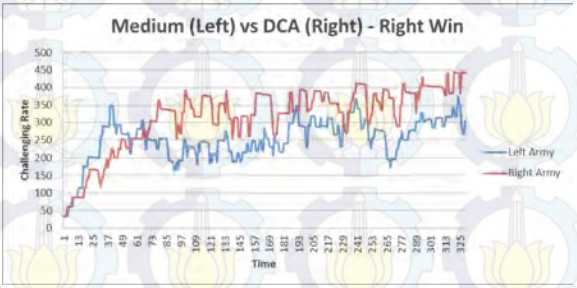

Medium (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

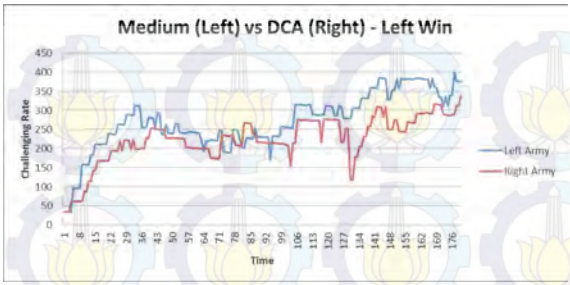

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Left Army
Right Army

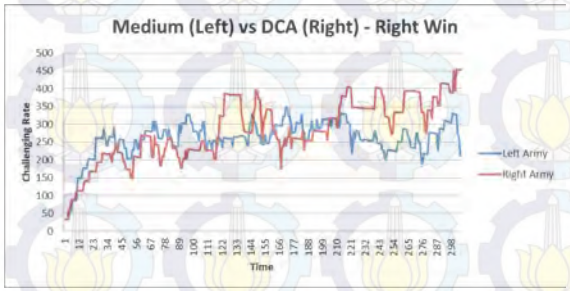

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

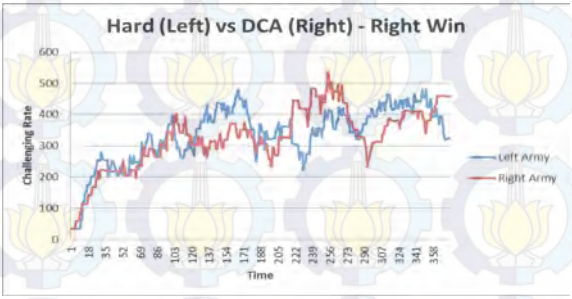

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

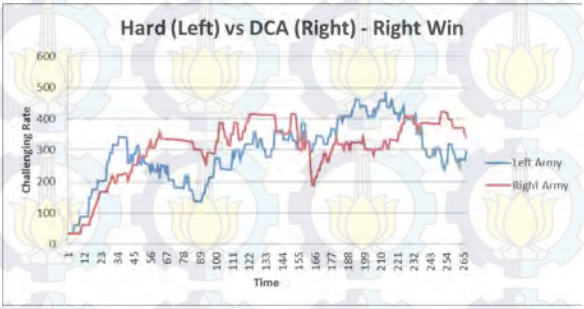

Medium (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

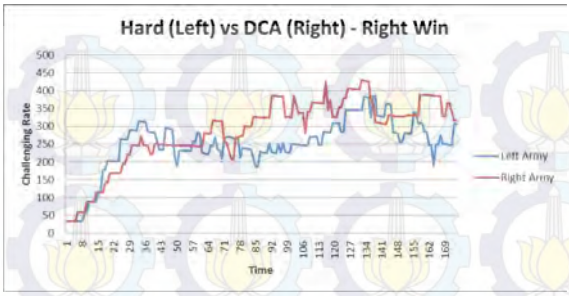

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

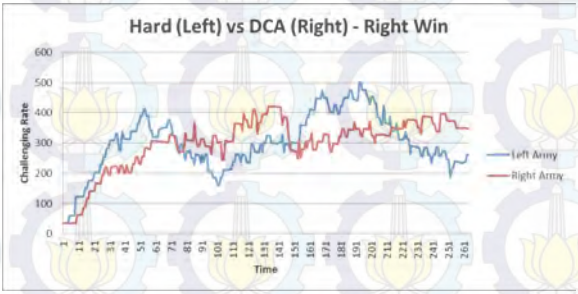
9. Hasil Pengujian AI tanpa DCA *Hard* Melawan AI dengan DCA

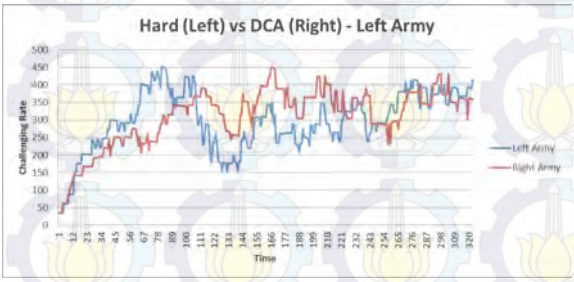
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	372	29,72	DCA
2	265	40,78	DCA
3	172	29,32	DCA
4	262	41,33	DCA
5	321	33,19	Hard
6	377	32,25	Hard
7	169	43,67	DCA
8	224	41,35	DCA
9	515	30,80	DCA
10	320	34,23	DCA
Mean	299,7	35,66	

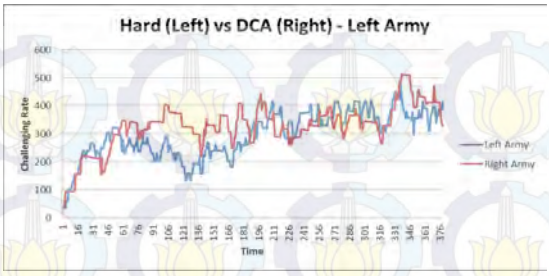


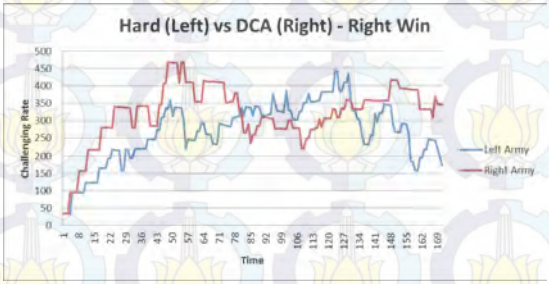


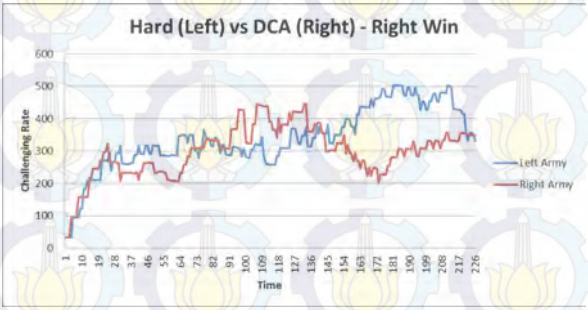

Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

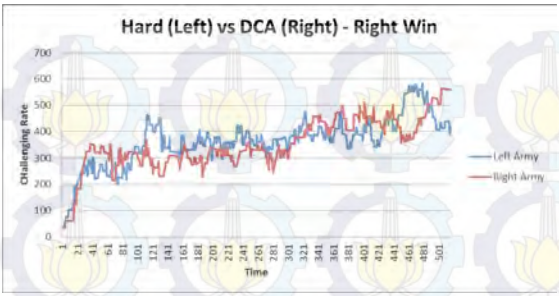

Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

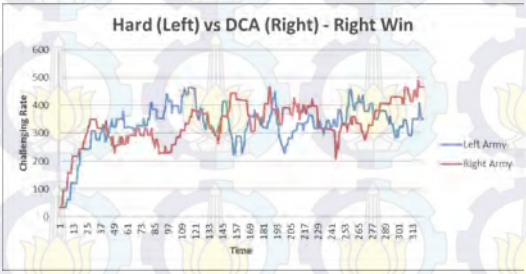

Hard (Left) vs DCA (Right) - Left Army
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Left Army
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

FINAL PROJECT – TE 141599

DYNAMIC SCENARIO WITH DCA (DYNAMIC CHALLENGING LEVEL ADAPTER) ON 2D REAL TIME STRATEGY TOWER DEFENSE GAME

Ramadhany Candra Arif Putra
NRP 2211100044

Advisors
Dr. Supeno Mardi Susiki N., S.T., M.T.
Christyowidiasmoro, S.T., M.T.

Departement of Electrical Engineering
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

ABSTRAK

Nama	:	Ramadhany Candra Arif Putra
Judul	:	Skenario Dinamis Menggunakan DCA (*Dynamic Challenging Level Adapter*) pada Permainan 2D Bergenre *Real Time Strategy Tower Defense*
Pembimbing	:	1. Dr. Supeno Mardi S. N., S.T., M.T. 2. Christyowidiasmoro, S.T., M.T.

Permainan *Real-time Strategy* (RTS) merupakan permainan yang bersifat kompetitif, antara pemain melawan pemain ataupun melawan *non-player character* (NPC). Banyak pengembang permainan menggunakan kemampuan NPC lawan yang terstruktur secara statis. Kemampuan tersebut tidak selalu dapat memberikan tingkat tantangan (*challenging rate* (CR)) yang sesuai ke berbagai tipe karakter pemain. Oleh karena itu dibutuhkan suatu kecerdasan buatan (AI) yang dapat mengatur kemampuannya terhadap berbagai tipe karakter pemain, sehingga dihasilkan pola permainan yang dinamis serta tingkat tantangan yang sesuai. Pada tugas akhir ini akan diimplementasikan *dynamic challenging level adapter* (DCA) sebagai mekanisme untuk beradaptasi terhadap unsur permainan dan menentukan pemilihan keputusan. Pemilihan keputusan akan menggunakan bantuan *decision tree* dan penyelesaian *knapsack problem* digunakan untuk memilih kombinasi pasukan yang tepat dalam mengatasi keadaan. Untuk mengukur performa sistem AI dengan DCA, dibuat AI yang didasari tingkat kesulitan yang umum digunakan yaitu mudah, sedang, dan sukar untuk menjadi lawan tanding. Hasil pengujian AI dengan DCA didapatkan rata-rata dari rata-rata selisih nilai CR tiap waktu untuk hasil pertarungan dengan berbagai tipe tingkat kesulitan adalah 34,02, dibandingkan dengan rata-rata untuk hasil pertarungan dengan tingkat kesulitan yang setara adalah 33,65 sehingga hanya terpaut sebesar 0,37. Sedangkan dengan tingkat kesulitan tidak setara didapatkan rata-rata terpaut sebesar 55,98 dari rata-rata tingkat kesulitan setara. Dengan hasil tersebut dapat disimpulkan penggunaan DCA dapat menjadi pengganti tingkat kesulitan yang statis namun tetap dengan tingkat tantangan yang sesuai bagi tiap tipe karakter pemain.

Kata Kunci: DCA, Kecerdasan Buatan, RTS, *Decision Tree*, *Challenging Rate*, *Knapsack Problem*.

ABSTRACT

Name	:	Ramadhany Candra Arif Putra
Title	:	*Dynamic Scenario Using DCA (Dynamic Challenging Level Adapter) on 2D Real Time Strategy Tower Defense Game*
Advisors	:	1. Dr. Supeno Mardi S. N., S.T., M.T. 2. Christyowidiasmoro, S.T., M.T.

Real-time Strategy (RTS) game is a game with competitive environment, that can be played by player versus player or non-player character (NPC). Many game developer implement enemy NPC with static pattern. Ability of static NPC not always give challenge to any kind of model player. Because of that, it needed an artificial intelligence (AI) that can maintain it's ability to any kind of model player, so it's produce dynamic flow of gameplay and balance in challenging level. In this final project will be implemented dynamic challenging level adapter (DCA) as mechanism for adapting of element in game environment and choose decision. Decision selection will be handled with decision tree to decide outcome for all condition, and solution of knapsack problem will be used for choosing unit combination to handle the condition. To see what AI NPC with DCA is capable of, another AI is created that implement simple mechanism with default difficulty level like easy, medium, and hard. This new AI will be enemy for AI with DCA. The result of implementation NPC with DCA give the average of average difference in CR each time was 34,02 as the result of battle NPC with DCA and without DCA with different difficulty level. Comparing the result of NPC without DCA battle with the same difficulty level, the average of average difference in CR each time was 33,65, adrift at 0.37. While with different difficulty level got the average adrift at 55,98 from the same difficulty level. With these result we can conclude the use of DCA can be a substitute for static difficulty level but still with the same challenging level to every type of model player.

Keywords: Artificial Intelligence, Challenging Rate, DCA, Decision Tree, Knapsack Problem, RTS.

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT atas segala limpahan berkah, rahmat, serta hidayah-Nya, penulis dapat menyelesaikan penelitian ini dengan judul : **Skenario Dinamis Menggunakan DCA (*Dynamic Challenging Level Adapter* pada Permainan 2D Bergenre *Real Time Strategy Tower Defense*)**.

Penelitian ini disusun dalam rangka pemenuhan bidang riset di Jurusan Teknik Elektro ITS, Bidang Studi Teknik Komputer dan Telematika, serta digunakan sebagai persyaratan menyelesaikan pendidikan S1. Penelitian ini dapat terselesaikan tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Keluarga, Ibu dan Bapak tercinta yang telah memberikan dorongan spiritual dan material dalam penyelesaian buku penelitian ini.
2. Bapak Dr. Tri Arief Sardjono, S.T., M.T. selaku Ketua Jurusan Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember.
3. Secara khusus penulis mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Dr. Supeno Mardi S. N. S.T., M.T. dan Bapak Christyowidiasmoro S.T., M.T. atas bimbingan selama mengerjakan penelitian.
4. Bapak-ibu dosen pengajar Bidang Studi Teknik Komputer dan Telematika, atas pengajaran, bimbingan, serta perhatian yang diberikan kepada penulis selama ini.
5. Seluruh teman-teman angkatan e-51 serta teman-teman *B201-crew* Laboratorium Bidang Studi Teknik Komputer dan Telematika.

Kesempurnaan hanya milik Allah SWT, untuk itu penulis memohon segenap kritik dan saran yang membangun. Semoga penelitian ini dapat memberikan manfaat bagi kita semua. Amin.

Surabaya, Juni 2015

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR PUSTAKA

[1] P. Marc, "Knowledge Acquisition for Adaptive Game AI," Science of Computer Programming, 2007. (Dikutip pada halaman 1).

[2] R. Hunicke, "AI for Dynamic Difficulty Adjustment in Games," National Conference on Artificial Intelligence, 2004. (Dikutip pada halaman 1).

[3] S.-H. Chang, "DCA: Dynamic Challenging Level Adapter for Real-time Strategy Games," IEEE 15th International Conference on Computanional Science and Engineering, 2012. (Dikutip pada halaman 2, 6, 7).

[4] D. Adams, "The State of RTS." http://www.ign.com/articles/2006/04/08/the-state-of-the-rts (diakses pada 28 Mei 2015), April 2006. (Dikutip pada halaman 5).

[5] S. Ontanon, "A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft," Computational Intelligence and AI in Games, IEEE Transaction, 2013. (Dikutip pada halaman 6).

[6] M. I. Chowdhury, "Bringing Auto Dynamic Difficulty to Commercial Games: A Reusable Design Pattern Based Approach," The 18th International Conference on Computer Games, 2013. (Dikutip pada halaman 6).

[7] A. E. Rhalibi, "Artificial Intelligence for Computer Games," International Journal of Computer Games Technology, 2009. (Dikutip pada halaman 7).

[8] J. R. Quinlan, "Simplifying Decision Tree," Artificial Intelligence Laboratory, MIT, 1987. (Dikutip pada halaman 8).

[9] C. Kingsford, "What are Decision Tree?." http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2701298 (diakses pada 1 Juni 2015), September 2008. (Dikutip pada halaman 8).

[10] S. Goddard, "Dynamic Programming 0-1 Knapsack Problem." http://www.cse.unl.edu/ goddard/Courses/CSCE310J (diakses pada 1 Juni 2015), Maret 2010. (Dikutip pada halaman 9).

BIOGRAFI PENULIS

Ramadhany Candra Arif Putra, lahir di Denpasar pada tanggal 16 Februari 1994. Ia menyelesaikan jenjang Sekolah Dasar di SD Negeri 12 Padangsambian pada tahun 2006, kemudian melanjutkan pendidikan SMP di SMPN 1 Denpasar dan lulus pada tahun 2008. Pada tahun 2011, penulis menyelesaikan pendidikan SMA di SMAN 1 Denpasar. Setelah lulus dari jenjang SMA, penulis melanjutkan pendidikan di Institut Teknologi Sepuluh Nopember dengan mengambil Jurusan Teknik Elektro. Pada semester kelima, penulis mengambil konsentrasi bidang studi Teknik Komputer dan Telematika dan aktif sebagai asisten laboratorium B201. Penulis Selama menempuh pendidikan kuliah, penulis bergabung dalam Himpunan Mahasiswa Teknik Elektro, Unit Kegiatan Mahasiswa (UKM) Robotika dan mengikuti beberapa kompetisi karya ilmiah seperti Program Kreatifitas Mahasiswa. Penulis tertarik dengan riset mengenai teknolgi *Game*, *Augmented Reality*, dan *Image Processing*.

BAB 1
PENDAHULUAN

Penelitian ini dilatar belakangi oleh berbagai kondisi yang menjadi acuan. Selain itu juga terdapat beberapa permasalahan yang akan dijawab sebagai luaran dari penelitian.

1.1 Latar belakang

Permainan *Real-time Strategy* (RTS) *Tower Defense* (TD) merupakan permainan yang memiliki komponen strategi dengan unsur membangun dan atau mengatur sumber daya dengan pola permainan yang bergerak secara *real-time* [1]. Permainan RTS TD memiliki tujuan untuk menghancurkan bangunan utama lawan dan juga mempertahankan bangunan utama pemain. Untuk meraih tujuan tersebut pemain akan ditantang dengan adanya pasukan lawan yang memiliki tujuan yang sama. Tantangan tersebut memiliki beberapa macam tingkat kesulitan yang dapat dipilih oleh pemain dengan menyesuaikan kemampuan pemain.

Tingkat kesulitan yang ada pada sebuah permainan RTS TD akan mempengaruhi tingkat kemampuan *non-player character* (NPC) sebagai musuh untuk dapat melawan pemain. Untuk mendapatkan kemampuan itu NPC diberi kecerdasan buatan (*Artificial Intelligence* atau AI). Untuk menyesuaikan tantangan terhadap berbagai karakter pemain, terdapat beberapa tingkat kesulitan dasar yang diberikan dalam permainan RTS TD. Tingkat kesulitan dasar pada umumnya dibagi menjadi tiga yaitu, *easy* (mudah), *medium* (sedang), dan *hard* (sukar). Namun dengan tingkat kesulitan dasar, pemain akan merasa permainan terlalu mudah atau terlalu susah [2]. Untuk menyelesaikan permasalahan tersebut banyak penelitian yang mengaplikasikan berbagai bentuk kecerdasan buatan. Salah satunya adalah dengan mengadopsi strategi yang sama dengan pemain. Namun dengan menggunakan kecerdasan buatan seperti itu, pola permainan akan selalu sama setiap pertandingan, sehingga hasil permainan akan selalu sama. Kecerdasan buatan seperti itu dianggap memiliki permasalahan permainan yang statis.

Dengan adanya permasalahan pola permainan yang sama atau statis, dikembangkan kembali kecerdasan buatan menggunakan me-

kanisme *Dynamic Challenging Level Adapter* (DCA). DCA merupakan mekanisme yang akan menentukan tindakan sesuai dengan aksi dan keadaan yang terjadi. Dengan penggunaan DCA, pola serta hasil permainan setiap pertandingan akan dinamis dan juga menyesuaikan tingkat tantangan (*Challenging Rate* atau CR) pemain. Pada penelitian [3] dicoba penerapan DCA pada permainan RTS Star Craft 2 (SC2) yang menganalisa lingkungan permainan dengan melihat bangunan yang dibangun serta ras pasukan yang ada didalam permainan. Pada penelitian tersebut dilakukan pengukuran dengan mengadu kemampuan kecerdasan buatan dengan DCA dan tanpa DCA. Kesimpulan yang didapat adalah tingkat tantangan permainan dengan penerapan DCA akan semakin seimbang, serta akan melipat gandakan waktu permainan dan juga pemenang akan ditentukan saat mendekati akhir permainan.

Dalam tugas akhir ini akan digunakan kecerdasan buatan dengan mengimplementasi DCA yang akan menganalisa unsur umum permainan serta parameter dalam lingkungan permainan. Hasil analisa akan digunakan untuk mengambil keputusan kapan dan pasukan apa yang harus dikeluarkan dalam suatu skenario serta mengontrol pengumpulan sumber daya (*resource*). Diharapkan dengan menggunakan DCA akan memberikan performa yang lebih baik serta tetap menjaga kedinamisan tingkat kesulitan.

1.2 Permasalahan

Penggunaan kecerdasan buatan dalam permainan RTS masih memiliki pola permainan statis. Hal tersebut dapat membuat pemain akan mudah menghafal atau memprediksi jalannya permainan sehingga tingkat tantangan akan menjadi dirasa mudah atau permainan akan dirasa terlalu susah karena pola permainan yang tidak sesuai dengan pengalaman pemain.

1.3 Tujuan

Penerapan kecerdasan buatan dengan mekanisme DCA dalam permainan RTS diharapkan akan membuat permainan menjadi lebih dinamis serta menyeimbangkan tingkat tantangan (CR) dan kesulitan dalam permainan. Pola dinamis yang ingin dicapai dalam tugas akhir ini adalah keadaan yang selalu berusaha mengimbangi pola permainan pemain sehingga jarak CR tidak berpaut terlalu ja-

uh. Hal tersebut akan membuat segala jenis pemain dapat bermain dengan tantangan dan kesulitan yang setara. Selain itu penggunaan algoritma ini dapat dimanfaatkan untuk penelitian lanjutan mengenai Artificial Inteligence serta optimasi dalam permainan RTS.

1.4 Batasan masalah

Untuk memfokuskan permasalahan yang akan diangkat maka dilakukan pembatasan masalah. Batasan-batasan masalah tersebut diantaranya adalah:

1. DCA akan mengontrol dalam skenario waktu dan tipe pasukan yang akan dimunculkan.
2. *Challenging Rate* (CR) atau tingkat tantangan akan mengacu pada persamaan 2.1.
3. Permainan akan berbasis pada prototipe permainan RTS TD.
4. Permainan akan menggunakan Unity3D sebagai *Game Engine*.

1.5 Sistematika Penulisan

Laporan penelitian tugas akhir ini tersusun dalam sistematika dan terstruktur sehingga lebih mudah dipahami dan dipelajari oleh pembaca maupun seseorang yang hendak melanjutkan penelitian ini. Sistematika laporan penelitian ini didasarkan dengan alur sebagai berikut.

BAB I PENDAHULUAN

Bab ini berisi uraian tentang latar belakang, permasalahan, tujuan penelitian, batasan masalah dan sistematika laporan.

BAB II TEORI PENUNJANG

Pada bab ini berisi tentang uraian secara sistematis teori-teori yang berhubungan dengan permasalahan yang dibahas pada pengerjaan tugas akhir ini. Teori-teori ini digunakan sebagai dasar dalam pengerjaan, yaitu informasi terkait, teori mengenai model permainan yang digunakan, algoritma yang akan digunakan, teori mengenai Unity3D sebagai *Game Engine*, dan teori-teori penunjang lainya.

BAB III DESAIN DAN IMPLEMENTASI

Bab ini berisi tentang penjelasan-penjelasan terkait sistem yang dibuat. Guna mendukung itu, digunakanlah blok diagram agar sistem yang akan dibuat dapat terlihat dan mudah dibaca untuk diim-

plentasikan pada pembuatan permainan. Selain itu digunakan juga gambar-gambar yang merepresentasikan bentuk permainan serta antar muka permainan yang dibuat.

BAB IV PENGUJIAN DAN ANALISA

Bab ini menjelaskan tentang pengujian yang dilakukan terhadap sistem hasil dari tugas akhir ini dan menganalisa sistem tersebut. Spesifikasi perangkat keras dan perangkat lunak yang digunakan juga disebutkan dalam bab ini. Sehingga ketika akan dikembangkan lebih jauh, spesifikasi perlengkapannya bisa dipenuhi tanpa harus melakukan uji coba perangkat lunak maupun perangkat keras lagi.

BAB V PENUTUP

Bab ini merupakan penutup yang berisi kesimpulan dari sistem perangkat lunak yang telah di buat dari hasil pengujian yang telah dilakukan. Saran dan kritik yang membangun untuk pengembangkan lebih lanjut juga dituliskan pada bab ini.

BAB 2
TINJAUAN PUSTAKA

Bab 2, akan dibahas mengenai teori penunjang dan perangkat lunak yang digunakan sebagai bahan acuan dan referensi agar tugas akhir ini menjadi lebih terarah

2.1 Permainan *Real-time Strategy*(RTS)

Real-time Strategy (RTS) merupakan kategori permainan strategi yang umumnya terfokus dalam pertarungan militer atau perang. Permainan RTS seperti Warcraft^{TM} atau Line Ranger^{TM} memerlukan pemain untuk mengontrol pasukan yang terdiri dari berbagai macam tipe dan mengalahkan pasukan lawan. Pertarungan tersebut terjadi dalam medan perang virtual yang bergerak secara *real-time*. Umumnya untuk meraih kemenangan dalam permainan RTS, pemain akan bergantung pada efisiensi mengumpulkan dan mengatur sumber daya atau *resource*, serta menggunakan dengan baik sumber daya tersebut untuk berbagai elemen aksi dalam permainan. Elemen aksi dalam permainan dapat berupa memunculkan pasukan, membuat bangunan untuk mengumpulkan sumber daya kembali, dan meningkatkan level pasukan dan atau bangunan [4].

Ciri-ciri game dengan genre *Real Time Strategy* adalah sebagai berikut :

1. Game RTS berjalan secara simultan, dimana lebih dari satu pemain dapat melakukan aksi dalam waktu yang bersamaan. Aksi tersebut terjadi secara duratif, yang artinya memerlukan beberapa waktu untuk menyelesaikannya.
2. Game RTS dimainkan secara *real-time*, artinya pemain memiliki waktu yang terbatas untuk menentukan pergerakan selanjutnya.
3. Sebagian besar game RTS ditampakkan secara parsial. Pemain hanya dapat melihat bagian peta yang telah dijelajahinya. Teknik ini disebut *fog-of-war*.
4. Sebagian besar game RTS bersifat non-deterministik. Tiap aksi mempunyai kesempatan untuk berhasil.
5. Game RTS mempunyai kompleksitas yang sangat tinggi baik dari segi ruang (*space*) maupun jumlah aksi yang tersedia pada

tiap siklus keputusan [5].

Dalam permainan RTS pasukan lawan menjadi tantangan bagi pemain untuk menyelesaikan misinya atau meraih kemenangan. Berhasil atau tidak misi pemain dilihat dari keberhasilan untuk menghancurkan bangunan utama musuh dan mempertahankan bangunan utama pemain dari serangan musuh. Lawan dalam permainan RTS dapat berupa sesama manusia atau melawan kecerdasan buatan.

2.2 *Dynamic Difficulty Adjustment* (DDA)

Dynamic difficulty adjustment (DDA) atau yang dapat dikenal dengan *dynamic game balancing* (DGB) merupakan suatu teknik untuk melakukan otomasi perubahan tingkat tantangan (*difficulty*) pada permainan digital yang *real-time*, berbasis pada kemampuan pemain atau usaha yang dilakukan pemain saat bermain untuk memberikan pemain pengalaman yang optimal atau sering disebut sebagai *flow* atau alur permainan [6].

Berbagai model DDA dikembangkan untuk masing-masing tipe permainan digital. Salah satunya adalah *dynamic challenging level adapter* (DCA) yang diimplementasikan pada permainan RTS. DCA merupakan mekanisme komputer untuk menentukan tindakan secara otomatis terhadap tingkah laku pemain yang beragam [3]. Pada dasarnya penggunaan DCA akan menyesuaikan tingkah laku komputer dengan memperhatikan tingkah laku pemain didalam permainan. Ide utama penggunaan mekanisme DCA ini adalah menganalisa unsur dasar dari permainan misalnya, tipe pasukan, jumlah pasukan, jumlah sumber daya, keadaan sumber daya, keadaan bangunan, dan unsur umum lain dalam permainan RTS. Proses mekanisme DCA diawali dengan mengekstraksi *event* yang pemain lakukan, lalu dilakukan pengolahan dan mengevaluasi apa tindakan yang seharusnya dilakukan. Dengan proses pertimbangan maka mekanisme DCA akan memilih skenario terbaik yang akan dilakukan dalam kurun waktu tersebut. Pada umumnya dalam permainan RTS terdapat dua unsur umum yang dapat dilakukan pertama adalah mengumpulkan sumber daya dan kedua adalah mengeluarkan pasukan. Kedua unsur ini yang akan menjadi roda putar dinamis dalam pengaplikasian DCA.

2.3 *Challenging Rate*(CR)

Challenging Rate (CR) atau tingkat tantangan merupakan suatu nilai yang digunakan untuk menentukan keseimbangan antara kemampauan AI dan kemampuan lawannya. Hal tersebut digunakan untuk mendapatkan nilai adaptif dan nilai keefektifan dari penggunaan DCA terhadap tingkat kemampuan lawan [3]. CR akan mengacu pada persamaan (2.1):

$$CR = \frac{A}{a} + \frac{H}{b} + \frac{P}{c} + \frac{C}{d} \tag{2.1}$$

Dalam persamaan (2.1), parameter A (Attack) merupakan nilai jumlah seluruh kekuatan serang dari pasukan, parameter H (Health Point) merupakan nilai jumlah seluruh life point pasukan dan bangunan yang ada, parameter P (Population) merupakan nilai jumlah seluruh populasi didalam permainan, dan parameter C (Cost) merupakan nilai pengeluaran keuangan atau resource yang digunakan. Parameter a, b, c, dan d merupakan nilai konstan untuk normalisasi persamaan (2.1) yang nilainya akan mengacu pada nilai normal dari status yang dituju.

2.4 *Artificial Intelligence* (AI)

Artificial Intelligence (AI) adalah kecerdasan yang ada pada sebuah mesin atau perangkat lunak yang dibuat dengan memiliki tujuan-tujuan tertentu. Dalam permainan digital, AI bertugas sebagai pemroses tingkah laku dan pengambil keputusan dari lawan dalam permainan atau dikenal dengan *nonplayer character* (NPC) [7] . Seiring dengan berkembangnya teknologi komputer dan permainan digital, telah banyak ditawarkan riset dan ide yang berhubungan dengan AI. Dalam genre permainan umum seperti *action games*, *role-playing games*, dan *strategy games*, tingkah laku dari AI biasanya diimplementasikan sebagai variasi dari aturan dasar yang telah ditentukan didalam permainan. Tetapi dengan adanya *machine-learning technique*, AI dapat berkembang dengan sendirinya. Contohnya dapat meningkatkan performa dengan mempelajari dari kesalahan dan kesuksesannya dan juga dapat beradaptasi de-

ngan kekuatan dan kelemahan dari pemain atau mempelajari lawan dengan meniru strategi permainan yang digunakan.

2.5 *Decision Tree*

Decision tree atau pohon keputusan merupakan suatu cara menentukan keputusan dengan menghasilkan dua kemungkinan data yaitu ya atau tidak. Untuk menentukan keputusan, data yang digunakan sebagai masukan akan dibuat membentuk pohon keputusan dan aturan keputusan [8]. Pembentukan pohon keputusan yang dibuat didasari pada klasifikasi data yang digunakan sebagai masukan, sehingga pemilihan keputusan akan didasari oleh aturan yang disesuaikan dengan klasifikasinya [9]. Data masukan dibuat sebagai sekumpulan data yang telah disesuaikan dengan klasifikasinya masing-masing sehingga membentuk sebuah tabel. Nilai *entropy* didapatkan dengan menggunakan kumpulan data masukan yang akan digunakan untuk membentuk pohon keputusan. *Entropi* merupakan jumlah bit yang dibutuhkan untuk mengekstrak suatu kelas dari sejumlah data acak dalam suatu ruang sampel.

Dalam *decision tree* untuk menghasilkan hasil keputusan yang optimal dibutuhkan beberapa proses antara lain membuat tabel terdistribusi terpadu dengan menyatakan semua nilai kejadian pada setiap rule, menghitung tingkat independensi antara kriteria pada suatu rule, yaitu antara atribut dan target atribut dan mengeliminasi kriteria yang tidak perlu, yaitu yang tingkat independensinya tinggi [8]. Atribut dalam *decision tree* merupakan parameter yang dibuat sebagai kriteria dalam pembentukan pohon, sedangkan target atribut merupakan atribut yang menyatakan data solusi per data atau penentu keputusannya.

2.6 *Knapsack Problem*

Knapsack problem adalah permasalahan dalam menentukan opmitasi nilai kombinasi. Sebagai contoh permasalahannya adalah terdapat sekumpulan barang yang memiliki suatu nilai dan bobot tertentu, bagaimana kombinasi barang yang dapat dimasukan pada suatu tempat yang memiliki keterbatasan total bobot yang dapat ditampung, namun memiliki total nilai barang yang maksimal. Banyak model permasalahan lainnya untuk permasalahan ini namun intinya adalah terdapat suatu limit tertentu untuk dapat menam-

pung suatu data namun ingin memiliki hasil dengan nilai terbesar.

Permasalahan ini dibagi menjadi dua macam [10] yaitu:

1. 0-1 *Knapsack Problem*
   Pada permasalahan ini *item* atau barang yang dapat dipilih merupakan barang yang tidak dapat dipisahkan sehingga dapat diambil satu barang atau tidak. Permasalahan ini dapat diselesaikan dengan menggunakan pendekatan *dynamic programming*.
2. *Fractional Knapsack Problem*
   Pada permasalahan ini barang yang digunakan merupakan barang yang dapat dipisahkan atau dipecah sehingga dapat mengambil sebagian dari suatu barang. Permasalahan ini dapat diselesaikan menggunakan *greedy algorithm*.

Pendekatan penyelesaian yang paling umum digunakan adalah *dynamic programming*. Karena meskipun tingkat kerumitan perhitungan yang harus dilakukan untuk menyelesaikan *knapsack problem* teknik *dynamic programming* mampu melakukannya dengan tingkat kecepatan yang tinggi.

BAB 3
DESAIN DAN IMPLEMENTASI SISTEM

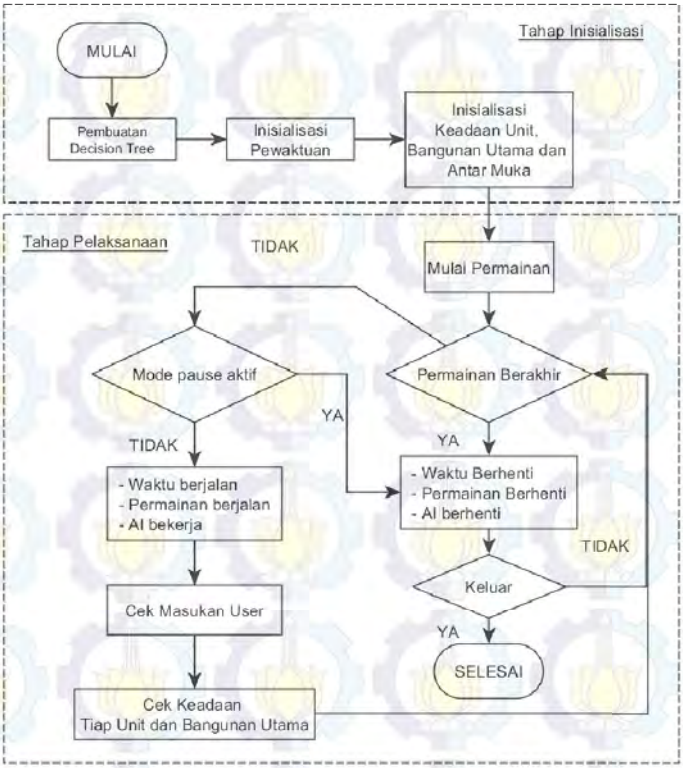
3.1 Desain Sistem

Desain sistem dalam tugas akhir ini dibagi menjadi tiga sistem besar yaitu desain sistem permainan *line defense*, desain sistem *non-player character* (NPC) lawan yang kedepannya akan disebut dengan kecerdasan buatan (AI) tanpa DCA, dan desain sistem NPC AI dengan DCA.

3.1.1 Desain Sistem Permainan *Line Defense*

Sistem permainan yang didesain merupakan sebuah permainan dengan genre *line defense*. Dalam sistem ini terdapat dua bagian utama yaitu tahap inisialisasi, dan tahap pelaksanaan seperti terlihat pada Gambar 3.1. Dalam tahap inisialisasi proses dimulai dengan membentuk atau mengolah *decision tree* sehingga dapat dihasilkan rumus untuk menentukan keputusan didalam sistem kecerdasan buatan. Selain itu dalam tahap inisialisasi ini disiapkan juga berbagai komponen sistem yang digunakan seperti inisialisasi pewaktuan, inisialisasi keadaan unit dan bangunan utama, dan inisialisasi antar muka. Tahap selanjutnya adalah pelaksanaan, yaitu tahap inti dari permainan ini bekerja. Pada tahap ini terjadi perputaran sistem yang akan terus mengecek hal-hal yang menjadi inti permainan antara lain: perintah masukan pemain, kalkulasi pewaktu, proses kecerdasan buatan, dan kejadian-kejadian yang meliputi unit pasukan dan bangunan utama yang menjadi unsur dalam permainan. Pada tahap pelaksanaan terdapat proses interupsi yang dapat digunakan untuk menghentikan permainan sejenak dan atau menghentikan permainan.

Proses pengecekan masukan pemain dibagi menjadi dua bagian, yang pertama masukan untuk mengatur tipe unit yang diinginkan dan yang kedua digunakan untuk mengatur kejadian didalam antar muka. Untuk pengaturan tipe unit dan macam tipe unit akan dijelaskan lebih detail pada sub Bab Unsur Permainan. Kejadian didalam antar muka terdapat dua macam yang digunakan yaitu yang pertama untuk mengatur wilayah pandang terhadap arena permainan dan yang kedua untuk melakukan penghentian sementara sistem atau dikenal dengan *pause* permainan. Proses pengecekan keadaan



Gambar 3.1: Desain sistem permainan

unit dan bangunan utama digunakan untuk menentukan kejadian yang terjadi didalam permainan, contohnya adalah penentuan apakah sedang dalam kondisi pertarungan atau tidak, dan pengecekan apakah permainan telah selesai atau belum. Penjelasan lebih detail mengenai kondisi permainan akan dijelaskan dalam sub bab tersendiri.

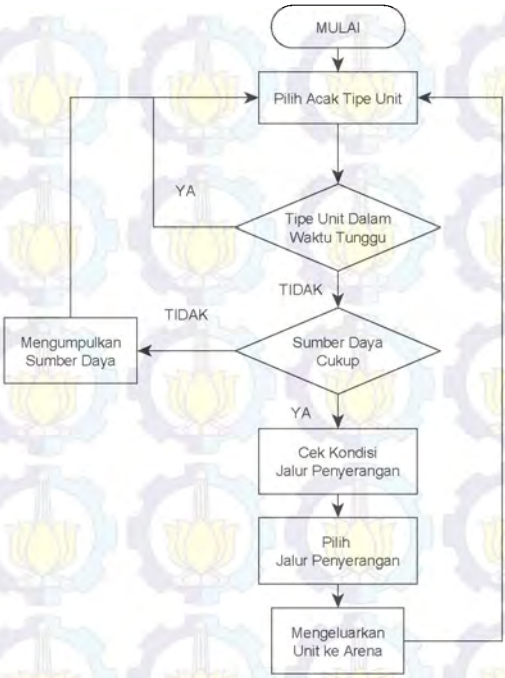
3.1.2 Desain Sistem NPC AI tanpa DCA

Non-player character (NPC) bertindak sebagai lawan didalam permainan yang digunakan dalam tugas akhir ini. Untuk dapat bekerja sesuai dengan fungsinya sebagai lawan, dibutuhkan kecerdasan

buatan (AI) yang akan berperan sebagai otak dari NPC tersebut. Dalam tugas akhir ini diterapkan proses AI seperti pada Gambar 3.2. AI dalam model ini akan bertindak sebagai NPC umum yang akan mengimplementasi pola acak bersyarat. Proses AI ini akan selalu dilaksanakan dalam interval 2 detik. Proses awal dimulai dari memilih tipe unit secara acak, selanjutnya mengecek apakah unit tersebut dalam waktu tunggu. Jika unit yang terpilih dalam waktu tunggu maka akan kembali memilih unit secara acak hingga mendapatkan unit yang tidak dalam waktu tunggu. Proses berikutnya akan mengecek apakah harga unit tersebut cukup dengan sumber daya yang dimiliki oleh NPC. Jika sumber daya kurang maka NPC akan memilih mengumpulkan sumber daya dahulu, jika tidak kurang maka akan dilakukan kalkulasi untuk menentukan jalur penyerangan mana yang akan dipilih. Proses pemilihan jalur penyerangan ini akan menggunakan syarat berupa probabilitas dari kepadatan suatu jalur penyerangan. Setelah itu maka akan dikeluarkan unit yang telah dipilih pada jalur penyerangan yang dipilih. Proses ini akan berjalan terus hingga pada sistem permainan (Gambar 3.1) menetapkan proses berhenti berjalan.

3.1.3 Desain Sistem NPC AI dengan DCA

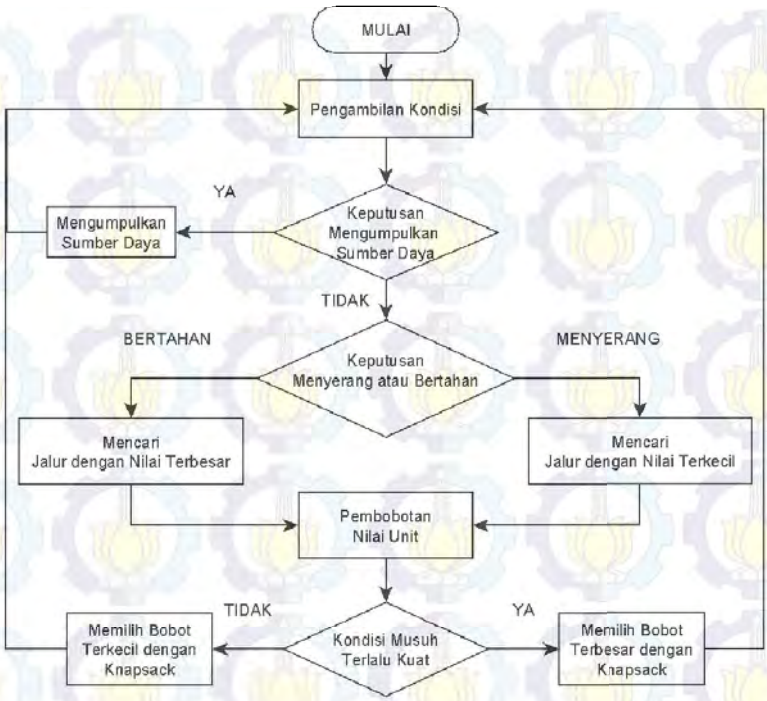
Proses pada pengontrol *non-player character* (NPC) dengan kecerdasan buatan (AI) menggunakan *Dynamic Challenging Level Adapter* (DCA) (selanjutnya disebut dengan AI DCA) digunakan untuk menghasilkan alur permainan yang menyeimbangkan keadaan lawannya. Proses dimulai dengan pengambilan data kondisi permainan yang akan dilakukan setiap 2 detik sekali. Kondisi permainan yang diambil meliputi kondisi *Challenging Rate* (CR) yang menerapkan persamaan (2.1), kondisi sumber daya yang dimiliki AI DCA, kondisi keadaan unit apakah terdapat unit yang sedang dalam waktu tunggu, serta kondisi nilai perbedaan antara CR yang dimiliki pasukan DCA dengan CR lawannya. Setelah pengambilan kondisi permainan data akan diolah untuk menentukan perintah apa yang akan dipilih antara menunggu untuk mengumpulkan sumber daya atau mengeluarkan pasukan. Pemilihan kondisi ini menggunakan algoritma *Decision Tree* dengan mengambil parameter dari data kondisi permainan. Jika kondisi yang dipilih adalah menunggu untuk mengumpulkan sumber daya maka kondisi akan selesai, sedangkan



Gambar 3.2: Desain sistem tanpa DCA

jika kondisi mengeluarkan unit yang dipilih maka akan dilanjutkan untuk pemilihan kondisi berikutnya.

Proses selanjutnya memilih antara mengelurkan unit dengan tujuan untuk bertahan(*defense*) atau menyerang(*attack*). Pemilihan ini juga menggunakan *decision tree* guna menentukan kondisi apa yang dipilih dengan parameter yang diambil adalah kondisi CR, jarak perbedaan nilai CR yang dimiliki pasukan DCA dengan lawannya, serta kondisi keadaan nilai kondisi pada jalur(*line*) dalam permainan. Setelah pemilihan maka tiap-tiap kondisi memiliki aksi yang berbeda, yaitu jika memilih menyerang maka akan dipilih jalur dengan nilai terkecil dan akan membobot pasukan yang ada dengan bobot yang mengutamakan kekuatan serang(*attack power*) dari tiap tipe pasukan, dan jika memilih bertahan maka akan dipilih jalur dengan nilai terbesar dan akan membobot pasukan yang ada



Gambar 3.3: Desain sistem DCA

dengan bobot yang mengutamakan ketahanan (*hit point*) dari tiap tipe pasukan. Setelah proses pembobotan maka akan dilakukan pengecekan apakah kondisi saat ini lawan berada pada kondisi nilai yang terlalu jauh dengan kondisi pasukan DCA, jika memang demikian maka akan dipilih pembobotan dengan nilai terkecil menjadi yang terbesar dan sebaliknya sedangkan jika tidak dalam kondisi tersebut maka pembobotan akan menggunakan nilai awal. Selanjutnya adalah pemilihan kombinasi tipe pasukan apa saja yang akan dikeluarkan menggunakan algoritma penyelesaian *knapsack problem* dengan parameter jumlah tipe pasukan, harga untuk mengeluarkan masing-masing pasukan, bobot masing-masing pasukan serta keadaan sumber daya saat ini. Proses tersebut merupakan proses terakhir sehingga akan kembali pada pengambilan kondisi permainan seper-

ti pada tahap awal. Desain sistem pada DCA diilustrasikan seperti pada Gambar 3.3.

3.2 Alur Kerja

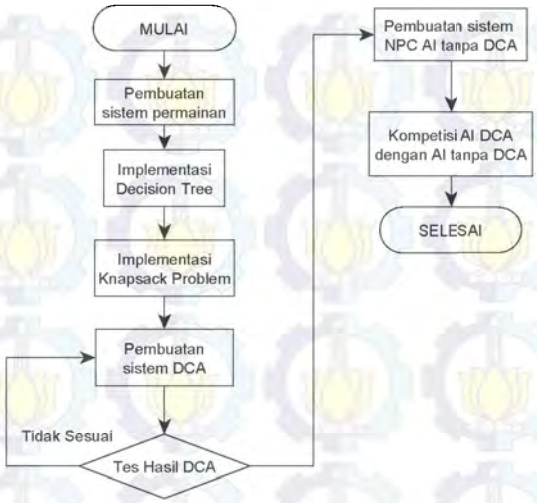
Alur kerja dalam pengerjaan tugas akhir ini terbagi oleh enam tahapan proses. Dalam masing-masing proses memiliki luaran yang dihasilkan dan dijadikan inputan untuk proses pada sistem permainan yang dibuat. Tahapan proses dari tugas akhir ini adalah sebagai berikut:

1. Pembuatan sistem permainan.
   Pembuatan sistem ini meliputi, unsur permainan, kondisi permainan, dan antar muka permainan.
2. Implementasi *decision tree* pada permainan.
   Implementasi *decision tree* digunakan untuk menentukan suatu luaran dari banyak inputan sekaligus.
3. Implementasi algoritma penyelesaian *knapsack problem* pada permainan.
   Implementasi *knapsack problem* digunakan untuk memilih kombinasi terbaik dari unit yang dibutuhkan pada keadaan tertentu.
4. Pembuatan sistem DCA.
   Dilakukan pembuatan sistem sesuai dengan desain sistem pada bab 3.1.
5. Tes *decision tree* dan algoritma penyelesaian *knapsack problem* pada sistem DCA.
   Dilakukan pengetesan pada hasil *decision tree* serta *knapsack problem* untuk menunjukan kesesuaian dengan hasil yang diharapkan.
6. Pembuatan sistem kecerdasan buatan NPC pasukan lawan.
   Pembuatan sistem kecerdasan buatan pasukan NPC digunakan untuk mencoba apakah sistem DCA telah sesuai dengan yang diharapkan.

Keseluruhan alur kerja dari tugas akhir ini dapat dilihat pada Gambar 3.4.

3.3 Pembuatan Sistem Permainan

Pembuatan sistem permainan dibagi menjadi tiga tahap yang akan membentuk suatu permainan secara utuh, tahap-tahap terse-



Gambar 3.4: Alur kerja

but meliputi unsur permainan, kondisi permainan, dan antar muka permainan.

3.3.1 Unsur Permainan

Pada model permainan yang dibuat unsur yang digunakan dalam permainan ini dibagi menjadi 4 unsur besar yaitu tipe pasukan, bangunan utama, jalur penyerangan, dan sumber daya. Masing-masing unsur dapat dijelaskan sebagai berikut:

1. Tipe Pasukan.
   Dalam model permainan yang digunakan ini terdapat 4 macam tipe pasukan masing-masing yaitu *Swordman*, *Archer*, *Assassin*, dan *Crusader* sesuai dengan Gambar 3.5. Masing-masing tipe pasukan tersebut memiliki status yang berbeda-beda sesuai dengan Tabel 3.1. Dari data Tabel 3.1 dapat dibagi menjadi dua poin penting yang akan digunakan untuk penentuan unit mana yang dibutuhkan yaitu kekuatan serang(*attack power*) dan ketahanan(*hit point*). Dua poin tersebut masing-masing merepresentasikan suatu keadaan yaitu, pada kekuatan serang merepresentasikan tingkat bahaya su-

atu unit terhadap lawannya, sedangkan nilai ketahanan merepresentasikan tingkat durabilitas dari suatu unit terhadap lawannya. Status lainnya merepresentasikan seberapa berharganya suatu unit yang dapat dilihat dari harga unit serta lamanya waktu tunggunya (*cooldown time*). Tiap tipe pasukan akan memiliki 4 buah *state* animasi yang menunjukan kondisi mereka saat ini yaitu keadaan *idle* atau tidak melakukan apa-apa, keadaan berjalan, keadaan menyerang, dan keadaan mati seperti pada Gambar 3.6.

Tabel 3.1: Status masing-masing tipe unit

Tipe Unit	HP (Hit Point)	ATK (Attack Power)	ASPD (ATK Speed)	ARNG (ATK Range)	COOLDOWN	COST
Swordman	750	110	1.3	1.3 s	1.5 s	150
Archer	500	110	2	2 s	2.3 s	200
Assassin	750	275	1.2	0.8 s	3 s	300
Crusader	1250	165	1.2	1.5 s	4 s	450

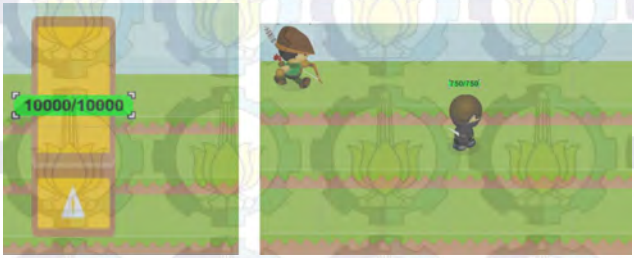
Gambar 3.5: Penggambaran masing-masing tipe unit dari kiri ke kanan: *Swordman*, *Archer*, *Assassin*, dan *Crusader*

Gambar 3.6: *State* animasi unit dari kiri ke kanan: *idle*, jalan, serang, mati

2. Bangunan Utama.
   Bangunan utama merupakan representatif dari tujuan permainan yaitu menghancurkan bangunan utama lawan dan atau

mempertahankan bangunan utama pemain. Dalam permainan ini bangunan utama akan diletakan pada tiap-tiap ujung jalur, dan juga sebagai posisi kemunculan suatu pasukan. Keadaan dari durabilitas bangunan utama akan diperlihatkan dengan nilai ketahanan yang ditampilkan langsung pada layar. Nilai ketahanan dari bangunan utama ini juga masuk hitungan dalam kondisi CR sesuai dengan formula 2.1.

3. Jalur Penyerangan.
   Untuk menyambungkan antara bangunan utama satu dengan yang lainnya dan untuk menentukan lajur perjalanan pasukan, dalam permainan ini diberikan 3 macam jalur penyerangan. Dengan menggunakan 3 macam jalur penyerangan maka dapat dipilih strategi penyerangan atau pertahanan yang digunakan saat permainan berlangsung. Jalur penyerangan ini juga menjadi salah satu parameter dalam penentuan keputusan didalam sistem DCA.



Gambar 3.7: Model bangunan utama(kiri), serta jalur penyerangan(kanan)

4. Sumber Daya.
   Dalam permainan dibutuhkan suatu perputaran alur, dalam permainan ini digunakan sumber daya sebagai salah satu parameter nilai yang menjadi bagian perputaran alur tersebut. Sumber daya dalam permainan ini akan bertambah seiring dengan waktu berjalan, dan akan digunakan untuk membayar saat ingin mengeluarkan pasukan. Sumber daya dalam permainan ini sangat berperan penting dalam perputaran alur karena untuk mencapai tujuan permainan pemain harus menghancurkan bangunan utama lawannya atau mempertahankan

bangunannya agar tidak hancur duluan dengan cara mengeluarkan pasukan untuk bertahan ataupun menyerang.

3.3.2 Kondisi Permainan

Dalam permainan ini terdapat dua macam kondisi yang merepresentasikan keadaan tujuan dalam permainan yaitu:

1. Kondisi Pertarungan.
   Merupakan kondisi pada saat suatu unit pemain berada pada jangkauan serang unit lawan yang berada pada jalur yang sama atau sebaliknya. Kondisi ini akan memicu *state* animasi serang pada unit yang telah memasuki jangkauan serangnya sehingga dapat melakukan pengurangan ketahanan kondisi lawannya. Kondisi ini dapat diperlihatkan pada *state* animasi seperti pada Gambar 3.6.
2. Kondisi Selesai.
   Kondisi ini dibagi menjadi dua yaitu kondisi kalah dan kondisi menang. Kondisi menang merupakan kondisi pada saat bangunan utama lawan berada pada *hit point* 0 dan bangunan utama pemain memiliki *hit point* lebih dari 0, sehingga dinyatakan pemain berhasil melakukan tujuan atau objektifnya. Sedangkan kondisi kalah merupakan kondisi yang berkebalikan dari sebelumnya yaitu bangunan utama pemain berada pada *hit point* 0 dan bangunan utama lawan memiliki *hit point* lebih dari 0, sehingga dinyatakan pemain gagal dalam melakukan objektifnya.

3.3.3 Antar Muka Permainan

Untuk dapat menampilkan informasi dan interaksi kepada pemain dibuatlah tampilan antar muka dalam permainan ini. Tampilan antar muka seperti pada Gambar 3.8, memiliki sembilan macam unsur informasi yang ditampilkan dijelaskan sesuai dengan nomor yang ada, yaitu:

1. Informasi *Hit Point* Bangunan Utama Pemain.
   Dalam informasi ini ditampilkan data kondisi bangunan utama pemain sehingga jika pemain tidak melihat secara langsung kondisi *hit point* pada bar bangunan utamanya ia dapat langsung melihat dari bagian informasi ini.
2. Informasi *Hit Point* Bangunan Utama Lawan.

Gambar 3.8: Tampilan antar muka yang digunakan

Dalam informasi ini ditampilkan data kondisi bangunan utama lawan sehingga pemain dapat memantaunya dari jauh tanpa harus melihat langsung bar *hit point* dari bangunan lawan.

3. Peta Kecil (*Mini-Map*).
   Informasi ini digunakan untuk menampilkan letak posisi dari unit yang telah berada pada arena permainan, dengan identifikasi warna hijau merupakan unit milik pemain sedangkan warna merah merupakan unit milik lawan. Selain itu juga ditampilkan di posisi jalur mana unit tersebut berada.
4. Pewaktu (*Timer*).
   Pewaktu digunakan untuk menampilkan informasi berapa lama suatu permainan telah berlangsung.
5. Bar *Hit Point* Bangunan Utama.
   Bar ini memberikan informasi keadaan *hit point* bangunan utama secara langsung, namun hanya ditampilkan tepat diatas bangunan utama yang bersangkutan sehingga jika tidak dapat melihat secara langsung *hit point* suatu bangunan utama dapat menggunakan informasi yang berada pada kiri atau kanan atas.
6. Keadaan Sumber Daya.
   Informasi ini menampilkan kondisi sumber daya saat ini yang berupa nilai dari total sumber daya. Informasi ini akan berubah-ubah seiring dengan waktu dan saat pemain mengeluarkan bangunan.

7. Tombol Unit.
   Informasi ini memberikan tampilan kepada pemain keadaan dari tiap unit berupa, tipe unit, harga untuk mengeluarkan, keadaan waktu tunggu, serta keadaan kecukupan sumber daya. Tipe unit ditampilkan dengan perbedaan bentuk dari tiap unit yang ada, harga ditampilkan langsung dengan nilai angkanya, sedangkan waktu tunggu dan kondisi kecukupan sumber daya ditunjukan dengan warna, merah untuk keadaan kurang cukup sumber daya, dan warna agak gelap menunjukan unit sedang dalam waktu tunggu. Informasi ini juga digunakan untuk memilih unit apa yang ingin dikeluarkan dengan meng-klik langsung gambar.
8. Tombol Pause.
   Antar muka ini ditujukan untuk melakukan penghentian permainan dan sehingga dapat keluar dari permainan.
9. Representatif Unit.
   Tampilan ini menunjukan kondisi unit yang ada di permainan, bar pada atas unit menunjukan keadaan *hit point* yang dimiliki unit tersebut. Sedangkan tampilan unit itu sendiri akan memberikan informasi mengenai kondisi atau *state* apa yang sedang dilakukan.

Seluruh antar muka ini bertujuan untuk dapat memberikan informasi yang jelas kepada pemain sehingga dapat mengatur strategi saat bermain.

3.4 Implementasi *Decision Tree*

Proses implementasi *decision tree* dimulai dengan penentuan kondisi yang akan menjadi parameter masukan. Pada sistem DCA ini digunakan dua buah *decision tree* yang pertama mengatur pemilihan kondisi apakah harus mengumpulkan sumber daya atau tidak, dan yang kedua untuk mengatur pemilihan kondisi apakah memilih kondisi menyerang atau bertahan.

3.4.1 *Decision Tree* Pemilihan Mengumpulkan Sumber Daya

Pada proses *decision tree* yang pertama, parameter yang digunakan sebagai masukan adalah kondisi-kondisi sebagai berikut:

1. Kondisi *Challenging Rate*(CR).
   Parameter ini akan melihat apakah kondisi CR pasukan NPC AI DCA berada pada nilai yang lebih dari atau kurang dari CR pasukan lawannya. Nilai ini menunjukan apakah lawan berada pada tingkat keaktifan tingi atau tidak dilihat dari seberapa banyak unit yang sudah dikeluarkan. Karena jika semakin aktif maka nilai CR akan terus meningkat.
2. Perbedaan nilai *Challenging Rate*(CR).
   Parameter ini akan menentukan seberapa jauh perbedaan nilai antara CR pasukan AI DCA dengan CR pasukan AI tanpa DCA. Perbedaan ini akan menggunakan nilai-nilai ambang yang ditentukan sesuai pada Tabel 3.2.

Tabel 3.2: Keterangan *decision tree* mengumpulkan sumber daya

Keterangan	1	2	3
DCR = Beda CR	Kecil (<50)	Sedang (>= 50 && <80	Besar (>= 80)
CR = Kondisi CR	Kurang dari Lawan	Lebih dari Lawan	
MON = Kondisi Sumber Daya	Kurang (< unit termurah)	Cukup (>= unit termurah)	
COOL = Kondisi waktu tunggu Unit	Ada	Tidak	
DEC = Keputusan	Mengumpulkan Sumber Daya	Mengelurakan Unit	

3. Kondisi waktu tunggu Unit.
   Parameter ini akan menentukan apakah terdapat unit yang berada pada waktu tunggu atau tidak. Kondisi ini dipilih karena jika terdapat unit dalam waktu tunggu maka unit yang ingin dikeluarkan ada kemungkinan sedang berada pada waktu tunggu. Sehingga unit tersebut tidak dapat digunakan untuk melakukan penyerangan atau pertahanan.
4. Kondisi sumber daya AI DCA.
   Parameter ini akan mengecek jumlah sumber daya yang dimiliki AI DCA apakah berada pada nilai ambang tertentu yang ditentukan sesuai pada Tabel 3.2. Pemilihan nilai ambang tersebut dikarenakan untuk dapat mengeluarkan sebuah

unit maka dibutuhkan minimal sumber daya yang cukup untuk membeli unit termurah.

Parameter masukan yang telah ditentukan selanjutnya akan digabung menjadi kumpulan data yang akan menghasilkan keputusan sesuai dengan hasil parameter masukan. Untuk kumpulan data pada *decision tree* yang pertama dapat dilihat pada Tabel 3.3. Kode angka pada tabel tersebut berasal dari Tabel 3.2.

Tabel 3.3: Hasil keputusan *decision tree* mengumpulkan sumber daya

DCR	CR	MON	COOL	DEC
1	1	1	1	1
1	1	1	2	1
1	1	2	1	1
1	1	2	2	2
1	2	1	1	1
1	2	1	2	1
1	2	2	1	1
1	2	2	2	2
2	1	1	1	1
2	1	1	2	1
2	1	2	1	2
2	1	2	2	2
2	2	1	1	1
2	2	1	2	1
2	2	2	1	1
2	2	2	2	2
3	1	1	1	1
3	1	1	2	1
3	1	2	1	2
3	1	2	2	2
3	2	1	1	1
3	2	1	2	1
3	2	2	1	1
3	2	2	2	1

3.4.2 *Decision Tree* Pemilihan Penyerangan atau Pertahanan

Pada proses *decision tree* yang kedua, parameter yang digunakan sebagai masukan adalah kondisi-kondisi sebagai berikut:

1. Kondisi *Challenging Rate*(CR).
   Parameter yang digunakan sama dengan *decision tree* pertama, yang akan melihat apakah kondisi CR pasukan AI DCA berada pada nilai yang lebih dari atau kurang dari CR pasukan AI tanpa DCA.
2. Perbedaan nilai *Challenging Rate*(CR).
   Parameter yang digunakan sama dengan *decision tree* pertama, yang akan menentukan seberapa jauh perbedaan nilai antara CR pasukan AI DCA dengan CR pasukan AI tanpa DCA. Perbedaan ini akan menggunakan nilai-nilai ambang yang ditentukan sesuai pada Tabel 3.5.
3. Kondisi Jalur.
   Parameter ini akan mengecek nilai dari sebuah jalur apakah sudah melebihi suatu ambang tertentu yang merupakan penanda bahwa jalur tersebut memiliki tingkat bahaya yang tinggi. Nilai ambang yang digunakan seperti pada Tabel 3.5. Nilai ambang tersebut dipilih karena dengan sejumlah pasukan tersebut sebuah jalur sudah memiliki tingkat ancaman yang tinggi.

Tabel 3.4: Keterangan *decision tree* menyerang atau bertahan

Keterangan	1	2	3
DCR = Beda CR	small (< 50)	Sedang (>= 50 && < 80)	Besar (> 80)
CR = Kondisi CR	Kurang Dari Lawan	Lebih Dari Lawan	
LINE = Kondisi Jalur (Total Unit in Line > 3)	Sedikit (1)	Sedang (2)	Banyak (3)
DEC = Keputusan	Menyerang	Bertahan	

Seperti pada *decision tree* pertama, akan dibuat kumpulan data yang akan digunakan untuk membentuk tabel untuk memberikan

hasil keputusan sesuai dengan yang masukannya. Untuk kumpulan data dari *decision tree* yang kedua dapat dilihat pada Tabel 3.5 dengan keterangan kode angka pada Tabel 3.4.

Tabel 3.5: Hasil keputusan *decision tree* menyerang atau bertahan

DCR	CR	LINE	DEC
1	1	1	1
1	1	2	2
1	1	3	2
1	2	1	1
1	2	2	1
1	2	3	1
2	1	1	1
2	1	2	1
2	1	3	2
2	2	1	1
2	2	2	1
2	2	3	1
3	1	1	1
3	1	2	2
3	1	3	2
3	2	1	1
3	2	2	1
3	2	3	1

Hasil dari pengambilan keputusan dari tiap *decision tree* akan digunakan sebagai masukan untuk sistem DCA yang dibuat.

3.5 Implementasi *Spawning Unit* Menggunakan Algoritma Penyelesaian *Knapsack Problem*

Proses pemilihan secara otomatis suatu kombinasi data dibutuhkan dalam penerapan DCA pada model permainaan ini karena digunakan untuk menentukan kombinasi unit apa yang tepat dengan keterbatasan serta keadaan saat ini. Untuk itu digunakan model algoritma penyelesaian *knapsack problem*. Proses tersebut akan menentukan dari 4 macam unit dengan bobot yang berbeda

dan dengan keadaan sumber daya yang ada, berupa kombinasi unit yang pas dalam keadaan tersebut. Sebagai contoh dengan keadaan menyerang unit dengan bobot kekuatan serang yang tinggi akan didahulukan dibandingkan dengan unit yang memiliki bobot kekuatan serang yang sedikit. Begitu juga dengan keadaan bertahan sistem akan memilih unit dengan bobot ketahanan yang lebih untuk dikeluarkan dalam keadaan bertahan.

Dalam penyelesaian *knapsack problem* terdapat empat parameter inputan yaitu *value* atau nilai objek, *weight* atau bobot objek, macam objek atau benda, dan kapasitas dari kantong (*knapsack*). Dengan empat parameter tersebut untuk implementasi kedalam sistem DCA yang dibuat maka empat parameter itu masing-masing adalah

1. Nilai objek.
   Untuk nilai objek yang menjadi masukan adalah harga untuk mengeluarkan masing-masing tipe unit.
2. Bobot objek.
   Untuk bobot objek yang menjadi masukan adalah bobot tiap tipe unit yang telah diolah pada proses setelah pemilihan kondisi menyerang atau bertahan.
3. Macam objek.
   Merupakan nilai banyaknya tipe unit yang ada yaitu sejumlah empat.
4. Kapasitas.
   Kapasitas yang digunakan dan menjadi batasannya adalah jumlah sumber daya yang sedang dimiliki pada saat tersebut.

Setelah tiap masukan berhasil diolah maka hasil yang didapatkan adalah sebuah kombinasi tipe unit yang dibutuhkan sesuai dengan kondisi pada saat itu.

3.6 Implementasi NPC AI Tanpa DCA sebagai AI Tandingan

Dalam tugas akhir ini ingin dicapai suatu sistem AI yang dapat menyesuaikan kemampuannya terhadap berbagai kemampuan pemain. Oleh karena itu dibutuhkan AI lainnya yang akan di adu kemampuannya sehingga dapat diukur kemampuan AI dengan DCA apakah sudah dapat mengatur kemampuannya.

Untuk AI yang menjadi lawan akan disebut dengan AI tan-

dingan. AI tandingan ini akan mengikuti sistem dari NPC AI tanpa DCA, namun akan memiliki tingkat kesulitan yang dibedakan. Pada permainan pada umumnya tingkat kesulitan digunakan untuk memberikan tantangan kepada pemain. Pada AI ini untuk memberikan tingkat tantangan yang berbeda akan dibedakan menjadi tiga tingkat kesulitan yaitu mudah, sedang dan sukar.

Perbedaan dari tiap tingkat kesulitan tersebut terletak pada kecepatan sumber daya untuk bertambah. Pada kondisi normal kecepatan bertambahnya sumber daya yang telah didesain dalam sistem adalah 10 per 0.2 detik. Untuk memberikan perbedaan tingkat tantangan maka diubah jumlah penambahan sumber daya tersebut, masing-masing seperti berikut:

1. Tingkat Kesulitan Mudah: 8 sumber daya per 0.2 detik (40 sumber daya per detik).
2. Tingkat Kesulitan Sedang: 10 sumber daya per 0.2 detik (50 sumber daya per detik).
3. Tingkat Kesulitan Sukar: 12 sumber daya per 0.2 detik (60 sumber daya per detik).

Alasan memilih peningkatan sumber daya sebagai penentu tingkat kesulitan dikarenakan sumber daya menjadi bagian penting yang akan meningkatkan nilai ancaman. Nilai ancaman yang dimaksud adalah populasi unit yang dapat dimunculkan dalam kurun waktu tertentu. Karena untuk mengeluarkan sebuah unit dibutuhkan sumber daya maka jika penambahan sumber daya lebih banyak dari normal akan membuat lebih cepat untuk mengeluarkan unit baru atau lebih cepat mengeluarkan unit termahal. Dengan adanya unit didalam arena maka akan meningkatkan nilai ancaman terhadap lawannya karena unit tersebut pasti akan menyerang kearah bangunan utama lawannya.

Sebagai perbandingan secara matematis, jika permainan berlangsung dalam jangka waktu 3 menit (180 detik) dan tidak terjadi peperangan di arena dan tipe unit yang dikeluarkan adalah unit termurah (150 sumber daya) tanpa memikirkan waktu tunggu, maka dapat dirumuskan total jumlah populasi unit yang dapat dibuat dalam jangka waktu tersebut adalah:

1. Tingkat Kesulitan Mudah (40 sumber daya per detik)
   $TotalPasukan = (180 * 40)/150 = 48 unit$
2. Tingkat Kesulitan Sedang (50 sumber daya per detik)

$TotalPasukan = (180 * 50)/150 = 60unit$

3. Tingkat Kesulitan Sukar (60 sumber daya per detik)
   $TotalPasukan = (180 * 60)/150 = 72unit$

Sehingga dapat dilihat bahwa dengan tingkat kesulitan sukar maka jumlah unit yang dapat dihasilkan akan lebih banyak dibandingkan dengan tingkat kesulitan dibawahnya, begitu juga sebaliknya tingkat kesulitan mudah akan menghasilkan unit tersedikit sehingga ancamannya lebih rendah dari tingkat kesulitan lainnya.

BAB 4
PENGUJIAN DAN ANALISA

Bab ini akan dipaparkan hasil pengujian dan analisa dari desain sistem dan implementasi yang telah dibahas pada bab 3. Pengujian akan ditujukan untuk memperlihatkan apakah sistem permainan telah berjalan dengan baik serta implementasi kecerdasan buatan menggunakan DCA telah memberikan hasil yang optimal. Untuk dapat memperlihatkan hasil kemampuan kecerdasan buatan akan dilakukan pengukuran dengan memperlihatkan hasil grafik dan juga hasil kalkulasi rata-rata nilai CR tiap waktu dari grafik yang ditampilkan. Pengujian dilakukan menggunakan editor pada perangkat lunak Unity versi 5.0 pada sistem operasi Windows 8. Bentuk pengujian yang dilakukan penulis untuk menguji kemampuan AI dengan DCA adalah sebagai berikut.

4.1 Pengujian Kesesuaian Sistem Permainan

Pada pengujian kesesuaian sistem permainan, hal yang diamati adalah apakah permainan ini telah bekerja sesuai dengan desain sistem yang diinginkan. Pengujian ini dilakukan dengen mencoba seluruh proses dalam permainan dan mencoba fungsi-fungsi yang ada apakah telah berjalan dengan baik atau tidak. Hasil dari pengujian seluruh fungsi berjalan dengan sesuai seperti pada desain sistem. Hasil yang didapatkan dari pengujian kesesuaian sistem permainan ini terdapat pada Tabel 4.1.

Dari tabel dapat dilihat bahwa keseluruhan sistem permainan telah dapat berjalan dengan baik sehingga permainan dapat menjadi representatif dari permainan RTS *Tower Defense*.

4.2 Pengujian Sistem Kecerdasan Buatan

Pengujian sistem kecerdasan buatan dilakukan untuk memberikan data apakah kecerdasan buatan dengan DCA telah berhasil membuat kedinamisan tingkat kesulitan. Untuk dapat memperlihatkan hasil tersebut perlu dilakukan pengujian terlebih dahulu terhadap kecerdasan buatan tanpa DCA. Oleh karena itu pada pengujian sistem kecerdasan buatan akan dibagi menjadi dua bagian yaitu pengujian kemampuan AI tanpa DCA melawan AI tanpa DCA

Tabel 4.1: Hasil pengujian fungsi

No	Nama Jenis Fungsi	Fungsi dapat berjalan
1	Tombol Mengeluarkan Unit	Ya
2	Unit Idle	Ya
3	Unit Berjalan	Ya
4	Unit Menyerang	Ya
5	Unit Mati	Ya
6	Pewaktu	Ya
7	Tombol Pause	Ya
8	Kondisi Selesai	Ya

dan yang kedua adalah pengujian kemampuan AI dengan DCA melawan AI tanpa DCA. Untuk dapat mengukur apakah nilai yang dihasilkan telah seimbang akan dilakukan penghitungan rata-rata selisih nilai CR tiap waktu dari hasil tiap pertandingan. Penghitungan tersebut dilakukan dengan menselisihkan nilai CR antara pasukan kubu kiri dengan kubu yang kanan.

4.2.1 AI tanpa DCA dengan Tingkat Kesulitan Setara

Pengujian kemampuan tingkat kesulitan akan didasari dengan tiga macam tingkat kesulitan dasar yang digunakan yaitu mudah (*easy*), sedang (*medium*), dan sukar (*hard*). Dalam pengujian ini ditampilkan data berupa lama suatu pertandingan dan nilai dari *challenging rate* (CR). Pada pengujian ini diuji kondisi permainan antara dua kubu pasukan yang memiliki tingkat kesulitan yang sama. Pengujian ini bertujuan untuk memperlihatkan bagaimana kondisi permainan jika tingkat kemampuan yang digunakan setara, sehingga dapat digunakan untuk menyimpulkan apakah kedinamisan kecerdasan buatan menggunakan DCA sudah dapat menggantikan tingkat kesulitan yang statis. Proses pengujian tingkat kesulitan setara dibagi menjadi tiga macam percobaan sebagai berikut:

1. AI tanpa DCA *Easy* melawan AI tanpa DCA *Easy*.
   Pengujian dengan mengadu antara pasukan dengan tingkat

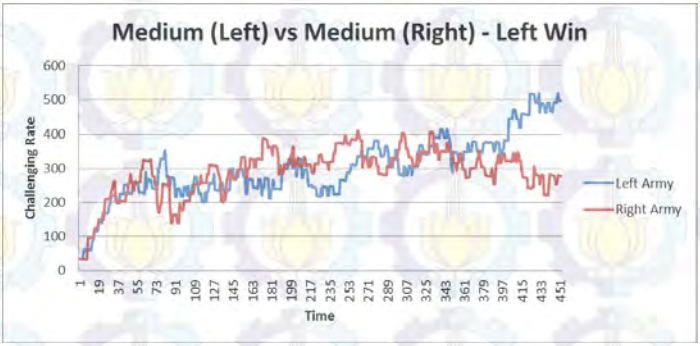
kesulitan *easy* melawan *easy*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.1. Dalam beberapa kali percobaan diambil sepuluh sampel data yang ditunjukan pada Tabel 4.2. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 290,4 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 27,46.

Gambar 4.1: Salah satu hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Easy*

Tabel 4.2: Hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Easy*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	222	12,91	Easy Left
2	251	27,96	Easy Right
3	255	34,37	Easy Right
4	475	23,94	Easy Right
5	308	33,90	Easy Left
6	318	30,48	Easy Right
7	232	35,02	Easy Left
8	181	16,16	Easy Left
9	223	35,58	Easy Left
10	439	24,26	Easy Right
Mean	290,4	27,46	

2. AI tanpa DCA *Medium* melawan AI tanpa DCA *Medium*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *medium* melawan *medium*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.2. Dalam beberapa kali percobaan diambil sepuluh sampel data yang ditunjukan pada Tabel 4.3. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 321 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 32,66.



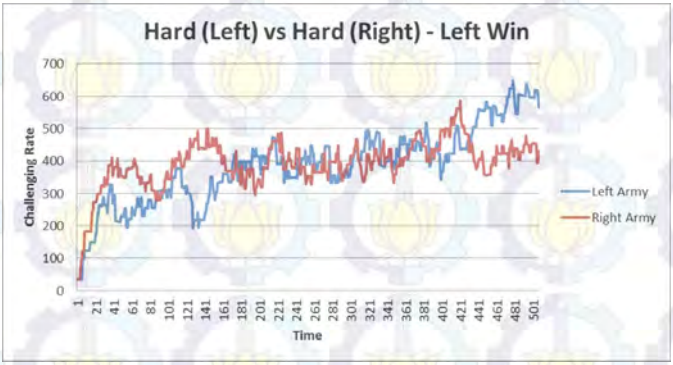
Gambar 4.2: Salah satu hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Medium*

3. AI tanpa DCA *Hard* melawan AI tanpa DCA *Hard*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *hard* melawan *hard*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.3. Dalam beberapa kali percobaan diambil sepuluh sampel data yang ditunjukan pada Tabel 4.4. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 321,8 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 40,84.

Dari pengujian tingkat kesulitan setara dapat dirata-rata nilai rata-rata selisih nilai CR tiap waktu adalah sebesar 33,65 seperti pada Tabel 4.5. Nilai tersebut terjadi karena nilai tingkat tantangan yang dihasilkan selalu setara, karena keadaan yang dimiliki antar kubu tidak memiliki perbedaan parameter yang berbeda, sehingga

Tabel 4.3: Hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Medium*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	450	19,14	Medium Left
2	254	26,35	Medium Left
3	404	41,53	Medium Right
4	526	25,49	Medium Left
5	128	35,23	Medium Left
6	192	30,67	Medium Right
7	271	49,12	Medium Right
8	445	35,99	Medium Left
9	202	32,60	Medium Left
10	338	30,50	Medium Left
Mean	321	32,66	



Gambar 4.3: Salah satu hasil pengujian AI tanpa DCA *Hard* melawan AI tanpa DCA *Hard*

proses ini dapat menjadi contoh model hasil pertandingan antara pemain yang memiliki kemampuan yang sama dengan sistem yang dibangun dalam suatu permainan.

4.2.2 AI tanpa DCA dengan Tingkat Kesulitan Berbeda

Untuk memberikan data mengenai ketidakseimbangan dalam permainan, maka dilakukan pengujian dengan mengadu NPC de-

Tabel 4.4: Hasil pengujian AI tanpa DCA *Hard* melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	506	37,13	Hard Left
2	384	44,38	Hard Right
3	179	49,30	Hard Right
4	200	46,88	Hard Left
5	303	47,71	Hard Right
6	358	39,95	Hard Right
7	327	43,34	Hard Left
8	380	25,56	Hard Left
9	367	41,17	Hard Left
10	214	32,95	Hard Left
Mean	321,8	40,84	

Tabel 4.5: Hasil pengujian AI tanpa DCA tingkat kesulitan setara

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Keterangan
1	290,4	27,46	Easy vs Easy
2	321	32,66	Medium vs Medium
3	321,8	40,84	Hard vs Hard
Mean	311,07	33,65	

ngan tingkat kesulitan yang berbeda. Hal ini perlu dilakukan karena dibutuhkan suatu ambang pembanding antara nilai yang dianggap sudah cukup seimbang dan belum seimbang. Dengan mengadu NPC dengan tingkat kesulitan berbeda akan didapatkan nilai ambang untuk data yang belum seimbang. Pengujian ini akan dilakukan dengan membandingkan tiga kombinasi tingkat kesulitan dasar yaitu:

1. AI tanpa DCA *Easy* melawan AI tanpa DCA *Medium*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *easy* melawan *medium*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.4. Sama seperti pada pengujian sebelumnya diambil sepuluh sampel data yang ditunjukan pada Tabel 4.6. Dalam tabel ter-

sebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 159,6 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 74,00.

Gambar 4.4: Salah satu hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Medium*

Tabel 4.6: Hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Medium*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	160	70,17	Medium
2	231	62,35	Medium
3	160	89,12	Medium
4	120	82,90	Medium
5	127	88,90	Medium
6	138	77,41	Medium
7	149	60,43	Medium
8	160	58,45	Medium
9	111	92,68	Medium
10	240	57,55	Medium
Mean	159,6	74,00	

2. AI tanpa DCA *Easy* melawan AI tanpa DCA *Hard*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *easy* melawan *hard*. Dalam pengujian ini didapatk-

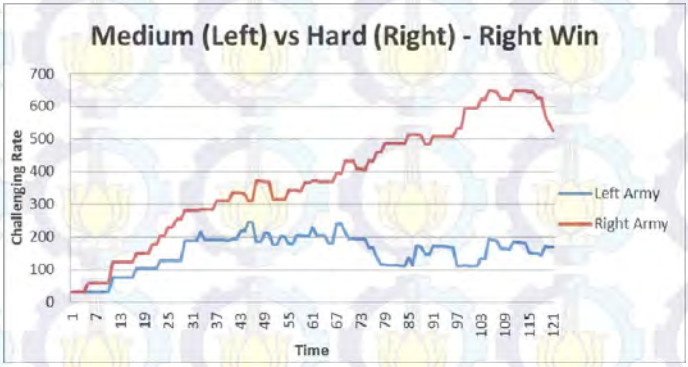
an salah satu hasil grafik yang dapat dilihat pada Gambar 4.5. Pengujian dilakukan berulang kali, dengan sepuluh sampel data memberikan hasil data yang ditunjukan pada Tabel 4.7. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 120,40 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 104,09.

Gambar 4.5: Salah satu hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Hard*

Tabel 4.7: Hasil pengujian AI tanpa DCA *Easy* melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	109	115,09	Hard
2	114	103,99	Hard
3	121	99,53	Hard
4	159	90,93	Hard
5	112	97,16	Hard
6	114	108,85	Hard
7	134	109,37	Hard
8	112	103,67	Hard
9	84	101,28	Hard
10	145	111,06	Hard
Mean	120,40	104,09	

3. AI tanpa DCA *Medium* melawan AI tanpa DCA *Hard*.
   Pengujian dengan mengadu antara pasukan dengan tingkat kesulitan *medium* melawan *hard*. Dalam pengujian ini didapatkan salah satu hasil grafik yang dapat dilihat pada Gambar 4.6. Pengujian dilakukan berulang kali, dengan mengambil sepuluh sampel data dihasilkan data yang ditunjukan pada Tabel 4.8. Dalam tabel tersebut ditampilkan rata-rata waktu permainan yang dilakukan adalah 143,9 detik, dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 90,82.



Gambar 4.6: Salah satu hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Hard*

Dari hasil pengujian ini dapat disimpulkan bahwa dengan mengadu AI dengan tingkat kesulitan yang berbeda, maka hasil yang didapatkan adalah NPC dengan tingkat kesulitan yang lebih tinggi akan selalu mendominasi permainan. Hal ini dapat dilihat dari hasil grafik yang ada dan rata-rata dari rata-rata selisih nilai CR tiap waktu sebesar 89,63 yang ditampilkan pada Tabel 4.9. Data CR dari NPC yang menggunakan AI dengan tingkat kesulitan lebih tinggi akan memiliki CR yang meningkat secara cepat sehingga nilai selisih yang dihasilkan besar, sedangkan dengan tingkat kesulitan yang sama NPC akan memiliki nilai CR yang stabil atau nilai selisih yang kecil. Untuk parameter waktu permainan, dengan mengadu tingkat kesulitan yang berbeda, lama waktu bermain akan lebih cepat di-

Tabel 4.8: Hasil pengujian AI tanpa DCA *Medium* melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	119	101,56	Hard
2	223	78,26	Hard
3	129	108,13	Hard
4	118	110,35	Hard
5	138	82,24	Hard
6	112	90,23	Hard
7	145	77,07	Hard
8	182	72,10	Hard
9	114	103,84	Hard
10	159	84,40	Hard
Mean	143,9	90,82	

bandingkan dengan mengadu tingkat kesulitan yang setara. Sesuai

Tabel 4.9: Hasil pengujian AI tanpa DCA tingkat kesulitan tidak setara

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Keterangan
1	159,6	74,00	Easy vs Medium
2	120,40	104,09	Easy vs Hard
3	143,9	90,82	Medium vs Hard
Mean	141,3	89,63	

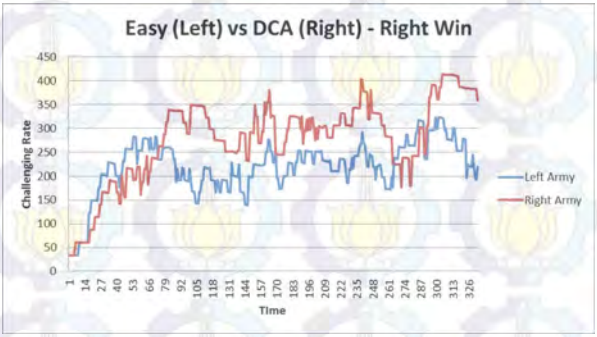
dengan persamaan 2.1 yang digunakan untuk mengukur tingkat tantangan (CR), parameter populasi sangat menentukan kenaikan dari nilai CR karena dengan penambahan populasi maka akan otomatis meningkatkan nilai serangan dan durabilitas. Penambahan populasi yang sangat tinggi ini diakibatkan karena model perbedaan tingkat kesulitan yang digunakan mengakibatkan perbedaan kecepatan suatu kubu untuk mengeluarkan unit untuk menyerang atau bertahan.

4.2.3 Pengujian Kemampuan AI dengan DCA Melawan AI tanpa DCA

Pengujian ini dilakukan untuk memperlihatkan bagaimana kemampuan dari AI dengan DCA apakah dapat memberikan keseimbangan dalam tingkat kesulitan atau tidak. Pengujian ini dilakukan dengan mengadu AI menggunakan DCA dengan AI tanpa DCA yang memiliki tiga macam tingkat kesulitan. Proses pengujian sama seperti pada proses pengujian sebelumnya, dan dihitung juga hasil rata-rata selisih nilai CR tiap waktu agar dapat dilihat apakah kemampuan AI menggunakan DCA akan dapat memberikan keseimbangan terhadap tingkat kesulitan. Pengujian yang dilakukan adalah sebagai berikut:

1. Pengujian Kemampuan AI dengan DCA melawan AI tanpa DCA *Easy*.

   Salah satu hasil pengujian ditampilkan dalam grafik sesuai pada Gambar 4.7. Pengujian dengan mengadu AI dengan DCA melawan AI tanpa DCA dilakukan berkali-kali dengan mengambil sepuluh sampel yang dapat dilihat pada Tabel 4.10.



Gambar 4.7: Salah satu hasil pengujian AI dengan DCA melawan AI tanpa DCA *Easy*

   Dari tabel tersebut didapatkan data rata-rata waktu permainan untuk melawan AI tanpa DCA *easy* adalah 252 detik, dan dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 37,49.

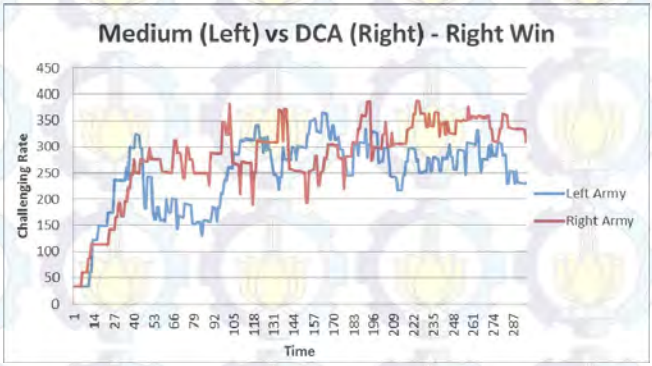
2. Pengujian Kemampuan AI dengan DCA melawan AI tanpa

Tabel 4.10: Hasil pengujian AI tanpa DCA *Easy* melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	331	39,12	DCA
2	138	39,12	DCA
3	250	34,17	DCA
4	241	30,20	Easy
5	193	43,23	DCA
6	242	34,11	DCA
7	391	44,92	DCA
8	246	41,31	Easy
9	242	27,95	Easy
10	250	40,79	DCA
Mean	252	37,49	

DCA *Medium*.

Salah satu hasil pengujian ditampilkan dalam bentuk grafik sesuai dengan Gambar 4.8. Pengujian dengan mengadu AI dengan DCA melawan AI tanpa DCA dilakukan berkali-kali dengan mengambil sepuluh sampel yang dapat dilihat pada Tabel 4.11.



Gambar 4.8: Salah satu hasil pengujian AI dengan DCA melawan AI tanpa DCA *Medium*

Dari tabel tersebut didapatkan data rata-rata waktu perma-

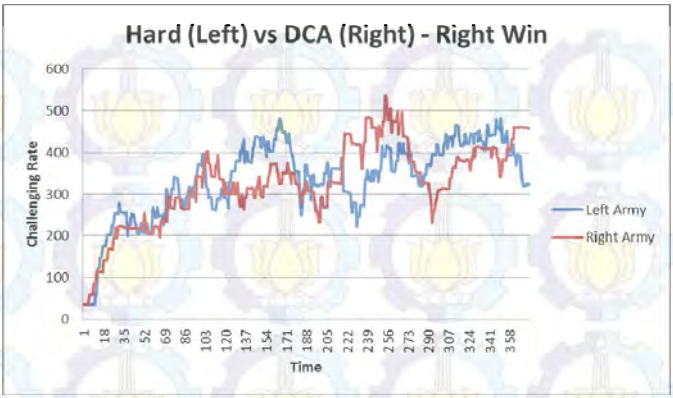
Tabel 4.11: Hasil pengujian AI tanpa DCA *Medium* melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	293	37,96	DCA
2	485	30,79	DCA
3	162	17,03	Medium
4	254	27,82	DCA
5	308	38,42	DCA
6	322	38,74	DCA
7	259	23,79	Medium
8	327	41,86	DCA
9	178	26,39	Medium
10	303	31,82	DCA
Mean	289,1	31,46	

inan untuk melawan AI tanpa DCA *medium* adalah 289,10 detik, dan dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 31,46.

3. Pengujian Kemampuan AI dengan DCA melawan AI tanpa DCA *Hard*.
   Salah satu hasil pengujian ditampilkan dalam bentuk grafik sesuai dengan Gambar 4.9. Pengujian dengan mengadu AI dengan DCA melawan AI tanpa DCA dilakukan berkali-kali dengan mengambil sepuluh sampel yang dapat dilihat pada Tabel 4.12.
   Dari tabel tersebut didapatkan data rata-rata waktu permainan untuk melawan AI tanpa DCA *Hard* adalah 299,7 detik, dan dengan rata-rata hasil dari tiap rata-rata selisih nilai CR tiap waktu pada setiap pertandingan adalah 35,66.

Pengujian dengan menggunakan sistem DCA memberikan hasil yang terlihat seimbang dengan menghasilkan rata-rata selisih nilai CR tiap waktu yang lebih kecil dibandingkan pada pengujian dengan tingkat kesulitan yang berbeda. Hasil rata-rata dari rata-rata selisih nilai CR tiap waktu dari ketiga pengujian tersebut sebesar 34,02 yang ditampilkan pada Tabel 4.13. Sedangkan dengan pengujian AI tanpa DCA yang setara didapatkan rata-rata sebesar 33,65



Gambar 4.9: Salah satu hasil pengujian AI dengan DCA melawan AI tanpa DCA *Hard*

Tabel 4.12: Hasil pengujian AI tanpa DCA *Hard* melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	372	29,72	DCA
2	265	40,78	DCA
3	172	29,32	DCA
4	262	41,33	DCA
5	321	33,19	Hard
6	377	32,25	Hard
7	169	43,67	DCA
8	224	41,35	DCA
9	515	30,80	DCA
10	320	34,23	DCA
Mean	299,7	35,66	

sedangkan AI tanpa DCA yang berbeda tingkat kesulitan memiliki rata-rata 89,63. Dari ketiga data tersebut dapat dilihat bahwa penggunaan DCA telah mampu menggantikan fungsi tingkat kesulitan statis karena hanya terpaut dalam 0,37 pada rata-rata selisih nilai CR tiap waktu.

Tabel 4.13: Hasil pengujian AI tanpa DCA melawan AI menggunakan DCA

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Keterangan
1	252	37,49	DCA vs Easy
2	289,1	31,46	DCA vs Medium
3	299,7	35,66	DCA vs Hard
Mean	280,27	34,02	

Hal ini diakibatkan karena dengan menggunakan DCA, sistem akan mempertahankan kondisi nilai tingkat tantangan (CR) sesuai dengan syarat atau aturan pada *decision tree* yang diimplementasikan pada sistem DCA. Sesuai dengan persamaan 2.1 untuk mengukur CR, DCA akan mengatur jumlah populasi yang dikeluarkan dan juga memilih unit mana yang terbaik untuk dikeluarkan sehingga CR yang dihasilkan tidak akan terpaut terlalu jauh dengan lawannya. Dengan *decision tree* juga, DCA mengatur keadaan sehingga lawan tidak akan memenangkan pertandingan dengan mudah dilihat dari hasil pemenang yang tidak selalu sama. Dari pengujian dua macam model pengujian antara kecerdasan buatan tanpa DCA dan dengan DCA dapat dilihat bahwa hasil menggunakan DCA sudah mendekati hasil pada kecerdasan buatan dengan tingkat kesulitan yang sama. Dari data tingkat kemenangan DCA masih memiliki probabilitas kemenangan lebih tinggi yaitu 73.33% (8 kali kalah dalam 30 data sampel). Hal ini kemungkinan diakibatkan oleh kemampuan DCA masih terlalu tinggi untuk ditandingkan dengan AI tanpa DCA yang telah didesain untuk saat ini. Keterangan mengenai grafik dan data sampel yang lebih detail akan ditampilkan pada bagian lampiran dari buku tugas akhir ini.

BAB 5
PENUTUP

5.1 Kesimpulan

Dari hasil implementasi dan pengujian kemampuan kecerdasan buatan dengan DCA yang sudah dilakukan dapat ditarik beberapa kesimpulan:

1. Hasil pengujian antara AI tanpa DCA dengan tingkat tantangan yang setara akan memberikan hasil kemenangan acak antara satu kubu dengan yang lain. Karena keadaan tiap kubu yang seimbang maka penentuan kemenangan tidak memiliki faktor penentu yang pasti dan memberikan hasil yang acak. Waktu yang dibutuhkan untuk menentukan pemenang cukup lama yaitu rata-rata 311,07 detik atau sekitar 5 menit 11 detik.
2. Hasil pengujian antara AI tanpa DCA dengan tingkat tantangan yang berbeda akan selalu menghasilkan kemenangan pada kubu yang menggunakan tingkat kesulitan yang lebih tinggi. Sebagai contoh tingkat kesulitan *hard* akan selalu menang melawan tingkat kesulitan *medium* dan *easy*. Karena tingkat kesulitan yang lebih tinggi memiliki keunggulan dalam kemampuan membuat pasukan lebih cepat. Waktu yang dibutuhkan untuk menentukan kemenangan lebih cepat dari ketika melawan tingkat tantangan yang setara yaitu rata-rata 141,3 detik atau sekitar 2 menit 21 detik.
3. Hasil pengujian antara AI tanpa DCA dengan tingkat kesulitan setara memiliki rata-rata dari rata-rata selisih nilai CR tiap waktu sebesar 33,65 dan dengan AI tanpa DCA dengan tingkat kesulitan berbeda memiliki rata-rata sebesar 89,63. Dari nilai tersebut dapat disimpulkan dengan mengadu kemampuan yang setara (dalam hal ini tingkat kesulitan yang setara) rata-rata dari rata-rata selisih nilai CR tiap waktu yang dihasilkan akan lebih kecil dan terpaut sebesar 55,98 dari kemampuan yang tidak setara.
4. Dengan penggunaan AI menggunakan DCA melawan berbagai macam tingkat kesulitan memberikan rata-rata dari rata-rata selisih nilai CR tiap waktu sebesar 34,02. Dengan nilai tersebut dibandingkan dengan nilai rata-rata pada AI tanpa DCA

dengan tingkat kesulitan setara memiliki perbedaan nilai yang kecil yaitu 0,37. Sehingga dapat disimpulkan AI menggunakan DCA telah berhasil untuk memiliki kemampuan yang memberikan kesetaraan atau keseimbangan tingkat kesulitan, meskipun lawannya berbeda tingkat kemampuan. Waktu yang dibutuhkan untuk menentukan kemenangan rata-rata adalah 280,27 detik atau sekitar 4 menit 36 detik.

5. Kemampuan AI dengan DCA masih tinggi untuk mendapatkan kemenangan. Dengan 30 data sampel, AI dengan DCA hanya mengalami kekalahan sebanyak delapan kali sehingga tingkat kemenangannya masih tinggi yaitu sebesar 73,33%. Hal ini dapat diakibatkan oleh kemampuan pemilihan keputusan yang masih belum banyak sehingga terdapat keadaan yang masih belum terkondisi.

5.2 Saran

Untuk pengembangan lebih lanjut mengenai tugas akhir ini dan untuk memberikan hasil pengujian yang optimal, disarankan untuk melakukan beberapa langkah lanjutan:

1. Pembuatan *decision tree* dengan aturan (*rule*) dan atribut yang lebih detail dan parameter masukan yang lebih banyak, agar keputusan yang dipilih lebih presisi.
2. Membuat model tipe unit yang lebih variatif, sehingga terdapat lebih banyak kombinasi unit pasukan yang dapat dibentuk.
3. Perombakan model alur kerja sistem DCA dengan mencoba mengganti pengambilan keputusan mengunakan *decision tree* menjadi *behaviour tree*.
4. Implementasi DCA pada model permainan yang lain seperti RTS *Tower Defense* yang berbasis *wave* atau gelombang serangan musuh.

LAMPIRAN

1. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI tanpa DCA *Easy*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	222	12,91	Easy Left
2	251	27,96	Easy Right
3	255	34,37	Easy Right
4	475	23,94	Easy Right
5	308	33,90	Easy Left
6	318	30,48	Easy Right
7	232	35,02	Easy Left
8	181	16,16	Easy Left
9	223	35,58	Easy Left
10	439	24,26	Easy Right
Mean	290,4	27,46	


Easy (Left) vs Easy (Right) - Right Win
Axis Title
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

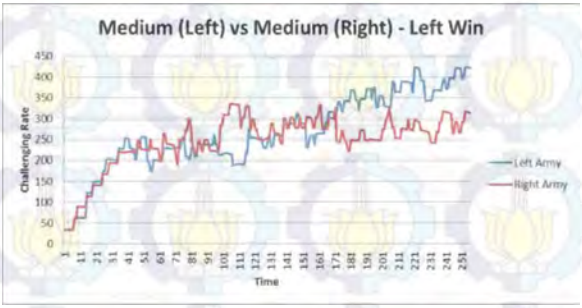

Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Easy (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

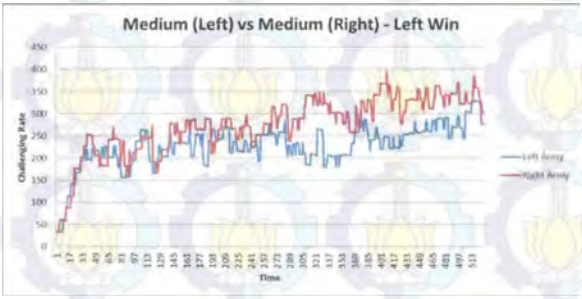

Easy (Left) vs Easy (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

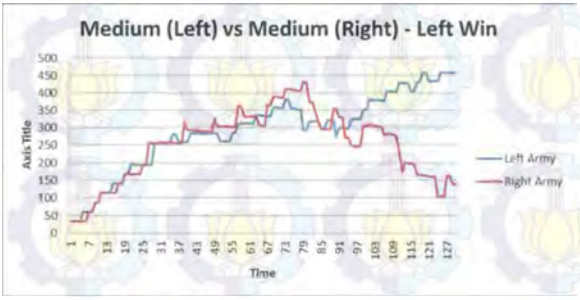
2. Hasil Pengujian AI tanpa DCA *Medium* Melawan AI tanpa DCA *Medium*

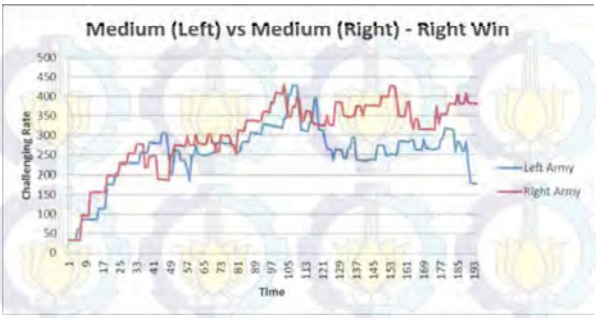
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	450	19,14	Medium Left
2	254	26,35	Medium Left
3	404	41,53	Medium Right
4	526	25,49	Medium Left
5	128	35,23	Medium Left
6	192	30,67	Medium Right
7	271	49,12	Medium Right
8	445	35,99	Medium Left
9	202	32,60	Medium Left
10	338	30,50	Medium Left
Mean	321	32,66	

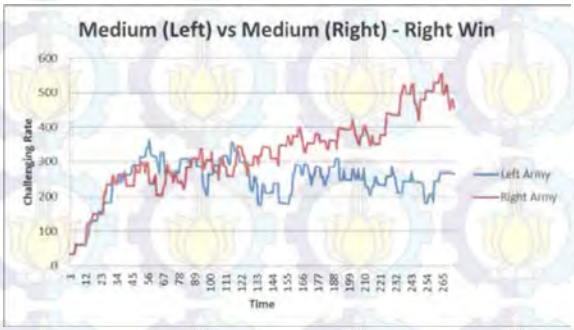


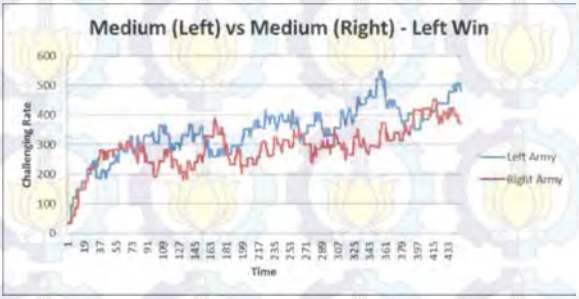

Medium (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

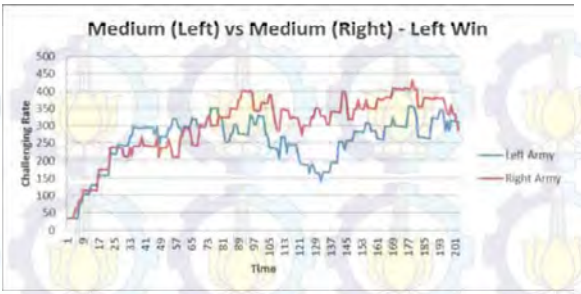

Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time


Medium (Left) vs Medium (Right) - Left Win
Axis Title
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

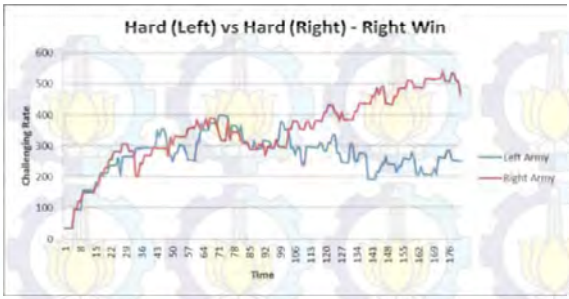

Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Medium (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

3. Hasil Pengujian AI tanpa DCA *Hard* Melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\|R_i - L_i\|)}{n}$	Pemenang
1	506	37,13	Hard Left
2	384	44,38	Hard Right
3	179	49,30	Hard Right
4	200	46,88	Hard Left
5	303	47,71	Hard Right
6	358	39,95	Hard Right
7	327	43,34	Hard Left
8	380	25,56	Hard Left
9	367	41,17	Hard Left
10	214	32,95	Hard Left
Mean	321,8	40,84	


Hard (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

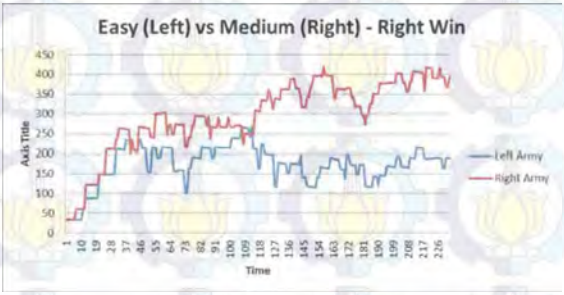

Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

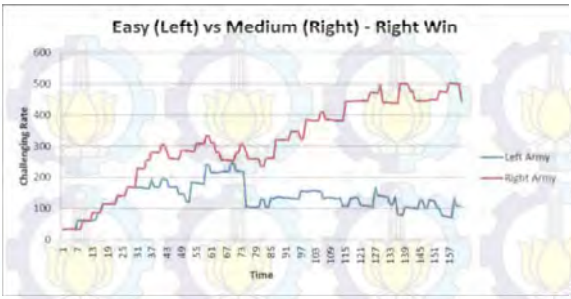

Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs Hard (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

4. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI tanpa DCA *Medium*

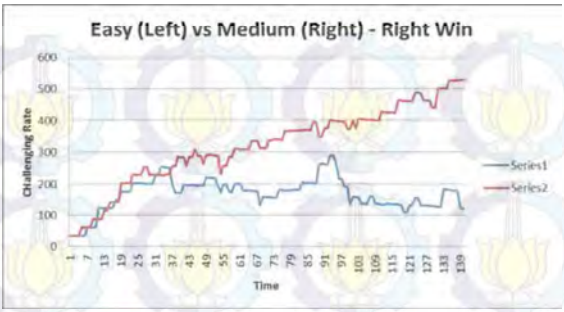
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	160	70,17	Medium
2	231	62,35	Medium
3	160	89,12	Medium
4	120	82,90	Medium
5	127	88,90	Medium
6	138	77,41	Medium
7	149	60,43	Medium
8	160	58,45	Medium
9	111	92,68	Medium
10	240	57,55	Medium
Mean	159,6	74,00	



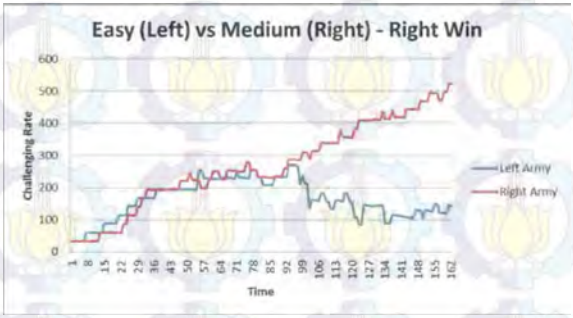

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

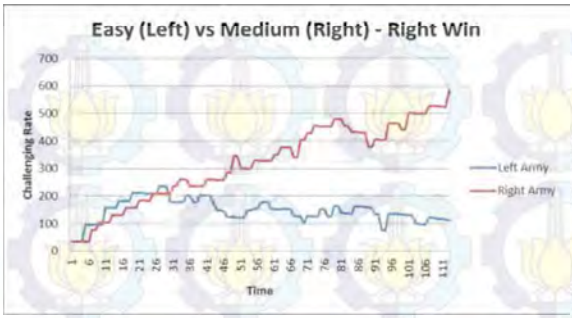

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

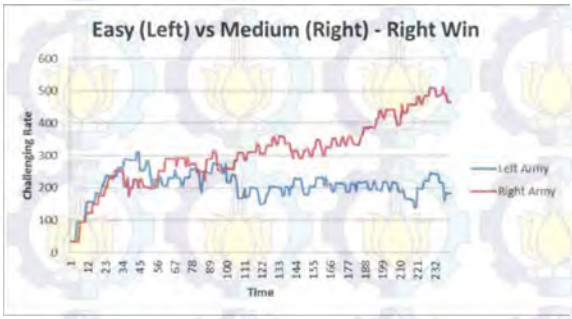

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Series1
Series2


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

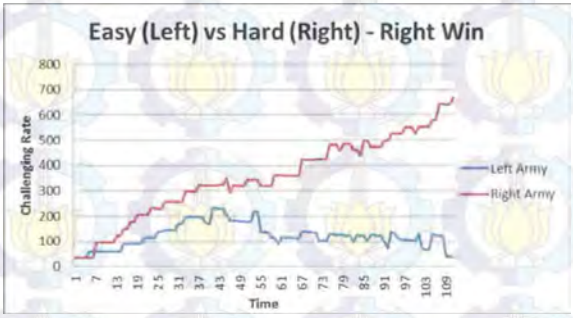

Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Medium (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

5. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI tanpa DCA *Hard*

No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	109	115,09	Hard
2	114	103,99	Hard
3	121	99,53	Hard
4	159	90,93	Hard
5	112	97,16	Hard
6	114	108,85	Hard
7	134	109,37	Hard
8	112	103,67	Hard
9	84	101,28	Hard
10	145	111,06	Hard
Mean	120,40	104,09	




Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

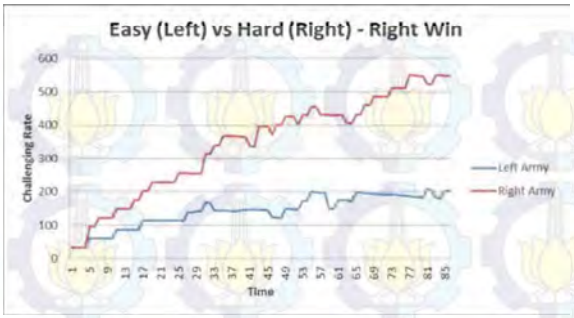

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

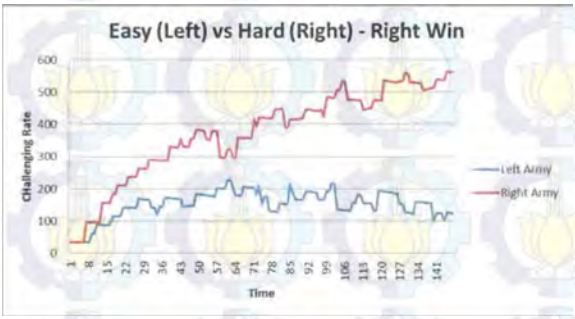

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

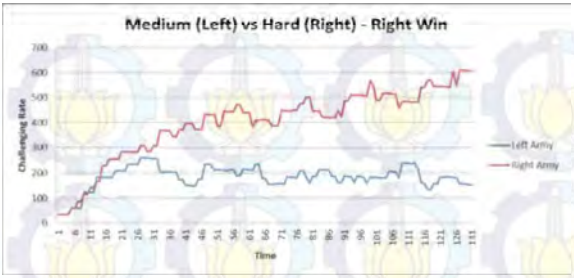

Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

6. Hasil Pengujian AI tanpa DCA *Medium* Melawan AI tanpa DCA *Hard*

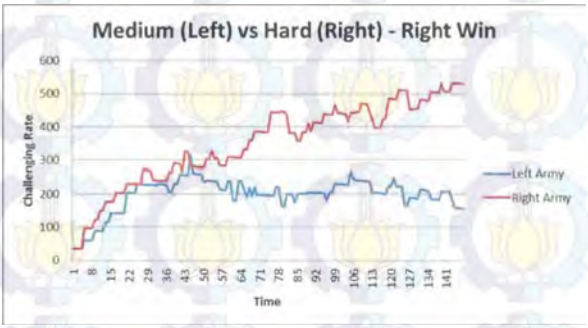
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	119	101,56	Hard
2	223	78,26	Hard
3	129	108,13	Hard
4	118	110,35	Hard
5	138	82,24	Hard
6	112	90,23	Hard
7	145	77,07	Hard
8	182	72,10	Hard
9	114	103,84	Hard
10	159	84,40	Hard
Mean	143,9	90,82	

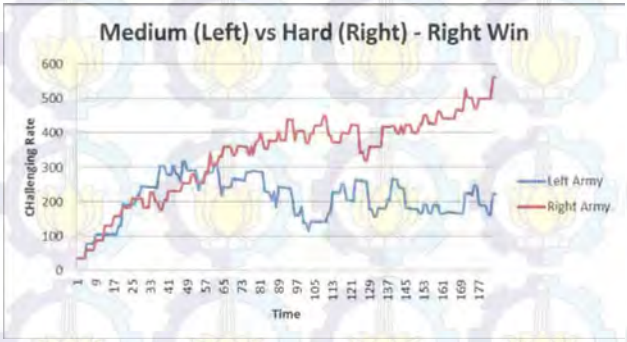

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

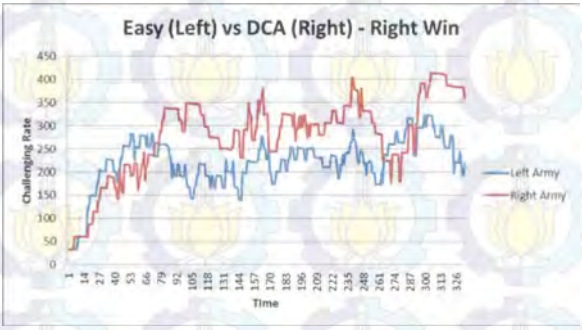

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

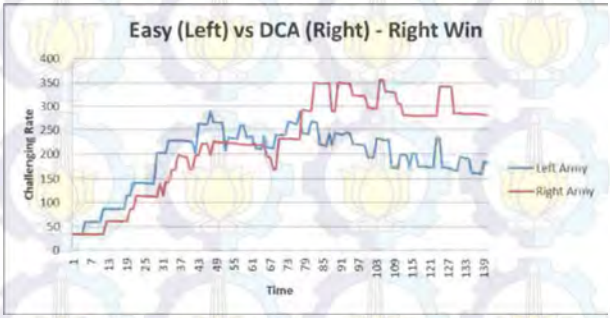

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

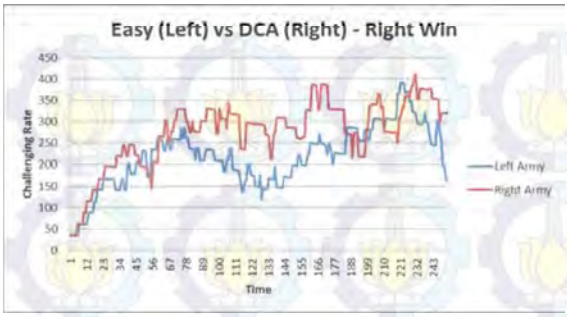

Medium (Left) vs Hard (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

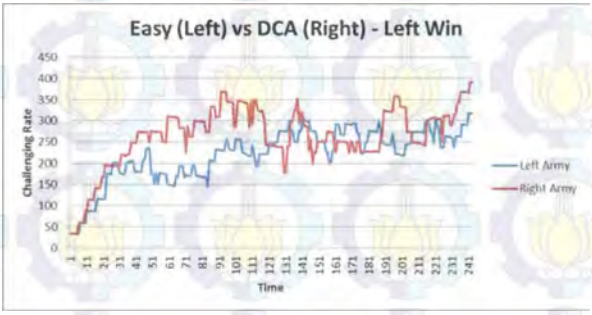
7. Hasil Pengujian AI tanpa DCA *Easy* Melawan AI dengan DCA

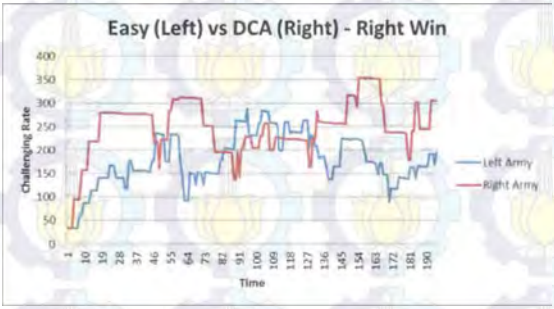
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	331	39,12	DCA
2	138	39,12	DCA
3	250	34,17	DCA
4	241	30,20	Easy
5	193	43,23	DCA
6	242	34,11	DCA
7	391	44,92	DCA
8	246	41,31	Easy
9	242	27,95	Easy
10	250	40,79	DCA
Mean	252	37,49	

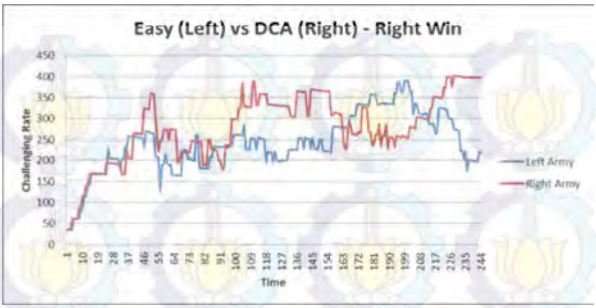


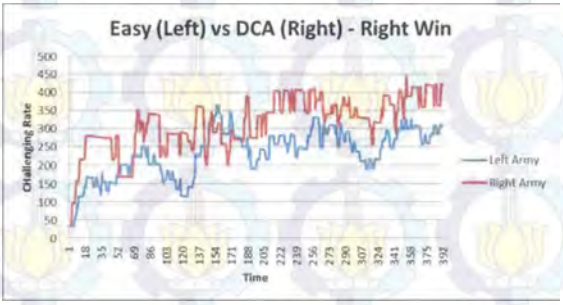


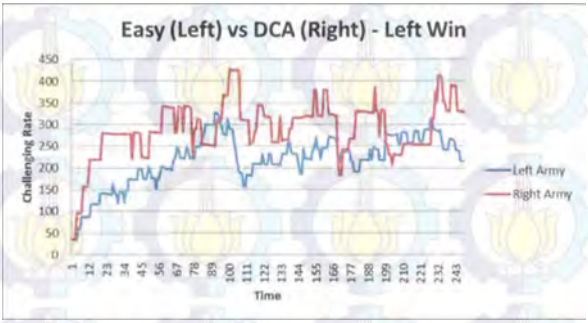

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

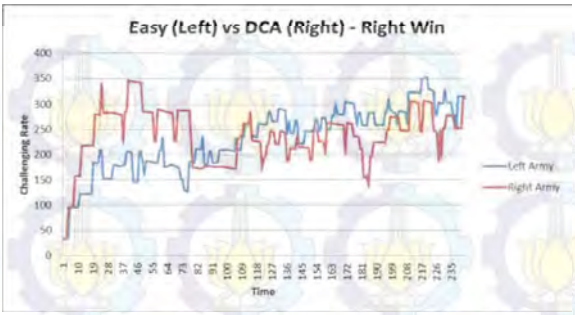

Easy (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

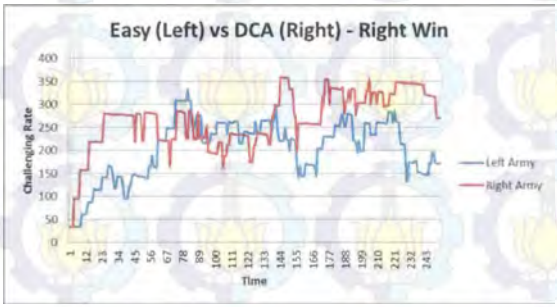

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

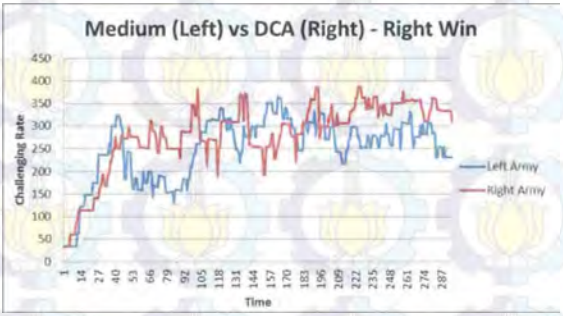

Easy (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

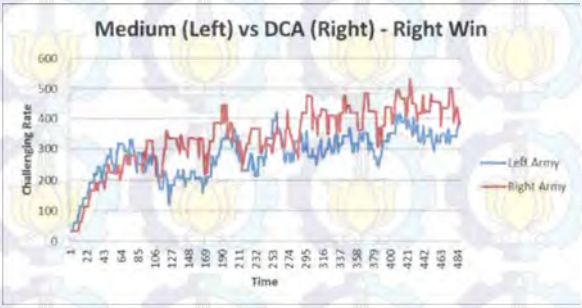

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

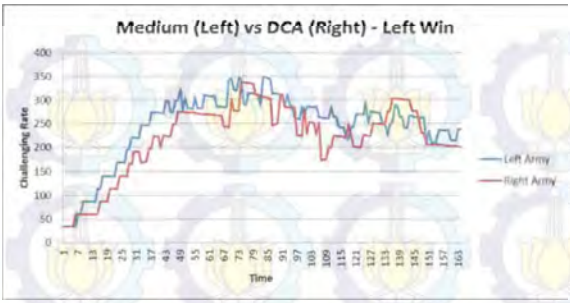

Easy (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

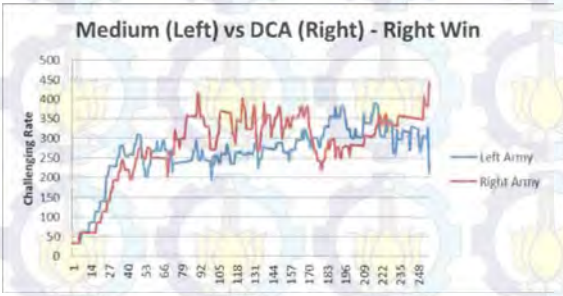
8. Hasil Pengujian AI tanpa DCA *Medium* Melawan AI dengan DCA

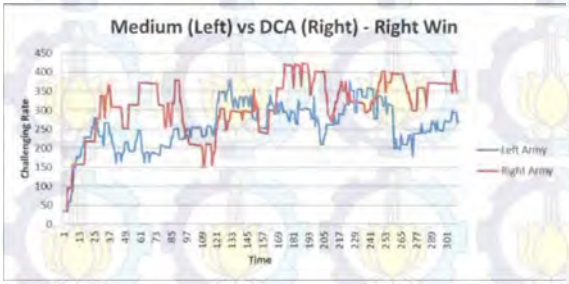
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	293	37,96	DCA
2	485	30,79	DCA
3	162	17,03	Medium
4	254	27,82	DCA
5	308	38,42	DCA
6	322	38,74	DCA
7	259	23,79	Medium
8	327	41,86	DCA
9	178	26,39	Medium
10	303	31,82	DCA
Mean	289,1	31,46	

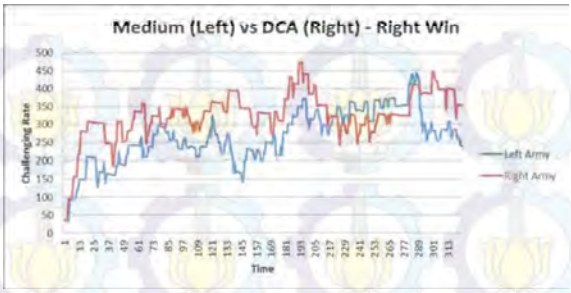


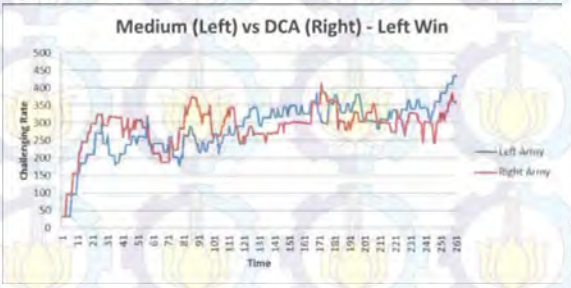


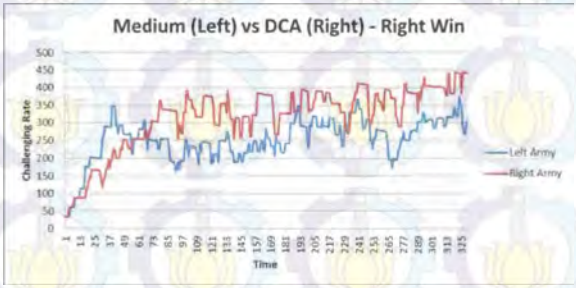

Medium (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

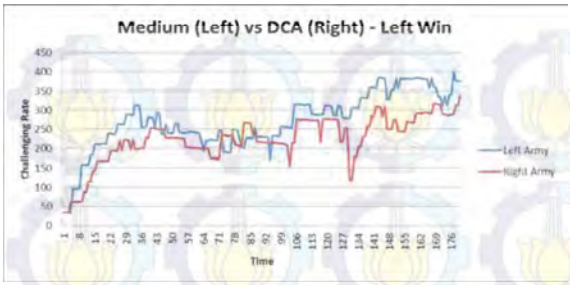

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Left Army
Right Army

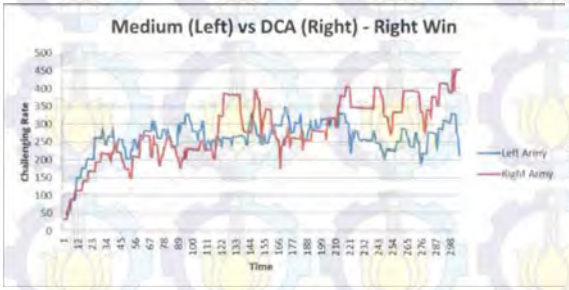

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army

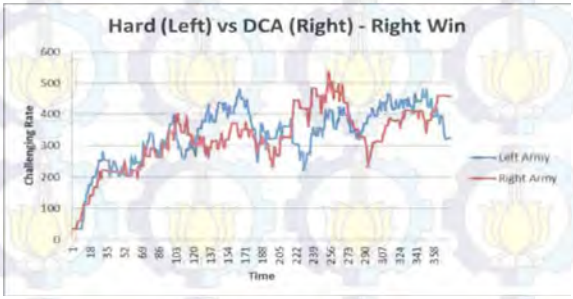

Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

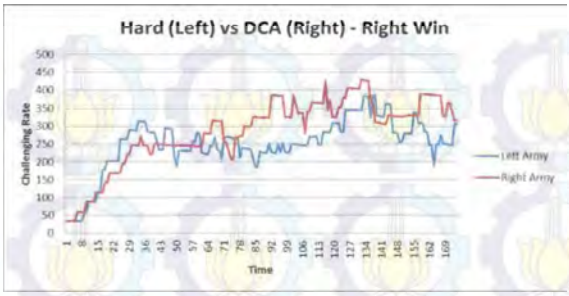

Medium (Left) vs DCA (Right) - Left Win
Challenging Rate
Time
Left Army
Right Army


Medium (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

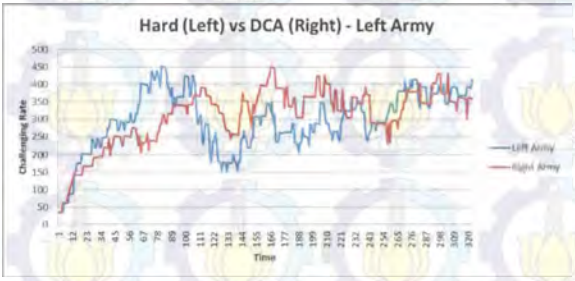
9. Hasil Pengujian AI tanpa DCA *Hard* Melawan AI dengan DCA

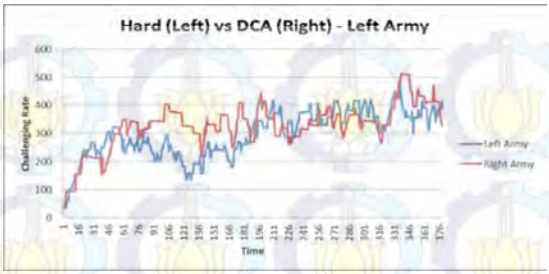
No	Waktu Pertandingan (detik)	$\frac{\sum_{i=1}^{n}(\lvert R_i - L_i \rvert)}{n}$	Pemenang
1	372	29,72	DCA
2	265	40,78	DCA
3	172	29,32	DCA
4	262	41,33	DCA
5	321	33,19	Hard
6	377	32,25	Hard
7	169	43,67	DCA
8	224	41,35	DCA
9	515	30,80	DCA
10	320	34,23	DCA
Mean	299,7	35,66	

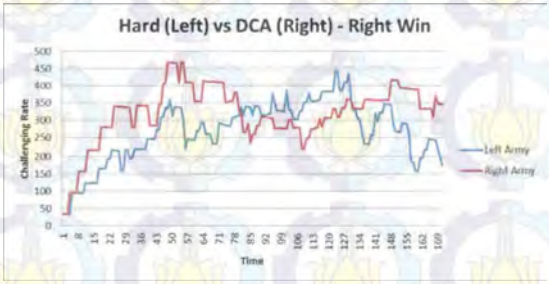



Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

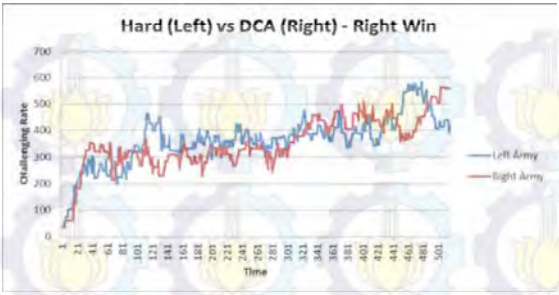

Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

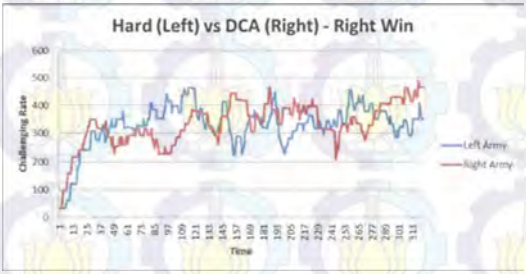

Hard (Left) vs DCA (Right) - Left Army
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Left Army
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army


Hard (Left) vs DCA (Right) - Right Win
Challenging Rate
Time
Left Army
Right Army

DAFTAR PUSTAKA

[1] P. Marc, "Knowledge Acquisition for Adaptive Game AI," Science of Computer Programming, 2007. (Dikutip pada halaman 1).

[2] R. Hunicke, "AI for Dynamic Difficulty Adjustment in Games," National Conference on Artificial Intelligence, 2004. (Dikutip pada halaman 1).

[3] S.-H. Chang, "DCA: Dynamic Challenging Level Adapter for Real-time Strategy Games," IEEE 15th International Conference on Computanional Science and Engineering, 2012. (Dikutip pada halaman 2, 6, 7).

[4] D. Adams, "The State of RTS." http://www.ign.com/articles/2006/04/08/the-state-of-the-rts (diakses pada 28 Mei 2015), April 2006. (Dikutip pada halaman 5).

[5] S. Ontanon, "A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft," Computational Intelligence and AI in Games, IEEE Transaction, 2013. (Dikutip pada halaman 6).

[6] M. I. Chowdhury, "Bringing Auto Dynamic Difficulty to Commercial Games: A Reusable Design Pattern Based Approach," The 18th International Conference on Computer Games, 2013. (Dikutip pada halaman 6).

[7] A. E. Rhalibi, "Artificial Intelligence for Computer Games," International Journal of Computer Games Technology, 2009. (Dikutip pada halaman 7).

[8] J. R. Quinlan, "Simplifying Decision Tree," Artificial Intelligence Laboratory, MIT, 1987. (Dikutip pada halaman 8).

[9] C. Kingsford, "What are Decision Tree?." http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2701298 (diakses pada 1 Juni 2015), September 2008. (Dikutip pada halaman 8).

[10] S. Goddard, "Dynamic Programming 0-1 Knapsack Problem." http://www.cse.unl.edu/ goddard/Courses/CSCE310J (diakses pada 1 Juni 2015), Maret 2010. (Dikutip pada halaman 9).

BIOGRAFI PENULIS

Ramadhany Candra Arif Putra, lahir di Denpasar pada tanggal 16 Februari 1994. Ia menyelesaikan jenjang Sekolah Dasar di SD Negeri 12 Padangsambian pada tahun 2006, kemudian melanjutkan pendidikan SMP di SMPN 1 Denpasar dan lulus pada tahun 2008. Pada tahun 2011, penulis menyelesaikan pendidikan SMA di SMAN 1 Denpasar. Setelah lulus dari jenjang SMA, penulis melanjutkan pendidikan di Institut Teknologi Sepuluh Nopember dengan mengambil Jurusan Teknik Elektro. Pada semester kelima, penulis mengambil konsentrasi bidang studi Teknik Komputer dan Telematika dan aktif sebagai asisten laboratorium B201. Penulis Selama menempuh pendidikan kuliah, penulis bergabung dalam Himpunan Mahasiswa Teknik Elektro, Unit Kegiatan Mahasiswa (UKM) Robotika dan mengikuti beberapa kompetisi karya ilmiah seperti Program Kreatifitas Mahasiswa. Penulis tertarik dengan riset mengenai teknolgi *Game*, *Augmented Reality*, dan *Image Processing*.