# Penentuan Strategi Bertarung Optimal Menggunakan Algoritma A* pada *Game Girls' Frontline*

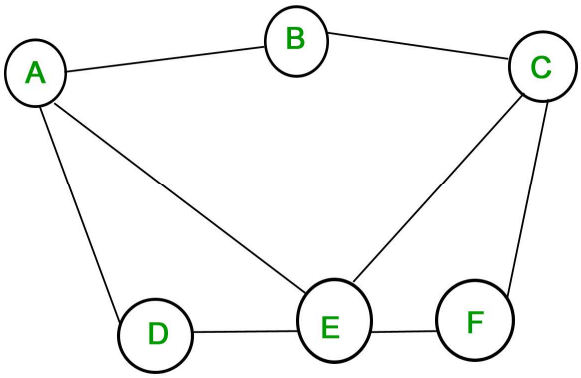
Arung Agamani Budi Putera / 13518005
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail : arung.agamani@gmail.com

***Abstraksi*—Girls' Frontline adalah game strategi buatan MICA Team yang menggunakan graf sebagai salah satu komponen utama dalam *gameplay*-nya. Pemain akan diberi kesempatan untuk menggerakkan unitnya pada graf yang tersedia untuk melawan musuh pada beberapa *node*, dan melakukan *capture* pada suatu *node* tertentu untuk memenangkan pertarungan.**

***Keywords—component; node, graph, BFS, greedy, turn.***

## I. PENDAHULUAN

### A. *Graf*




*Gambar 1. Graf*

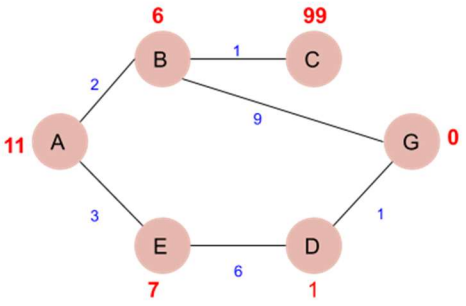
Sebuah desain dari *gameplay* pada suatu permainan sering kali menggunakan suatu abstraksi yang memudahkan pemain untuk melihat situasi dalam bentuk yang lebih sederhana. Beberapa bentuk abstraksi tersebut seringkali menggunakan suatu struktur yang memiliki permodelan yang dapat didefinisikan secara matematis. Salah satu abstraksi tersebut adalah graf. Penggunaan konsep graf dapat membuat permasalahan yang terdapat di-dunia nyata dapat didefinisikan dengan lebih mudah dan diselesaikan dengan memanfaatkan sifat-sifat maupun property yang dimiliki oleh graf itu sendiri. Salah satu contoh dari *real life problem* yang menggunakan konsep graf adalah *Traveling Salesman Problem*, yang apabila diterjemahkan ke dalam konteks graf, adalah persoalan mencari keberadaan *Hamiltonian Circuit* pada graf tersebut.

### B. *Girls' Frontline*


*Gambar 2. Poster Teaterikal Girls' Frontline*

Girls' Frontline adalah sebuah game buatan MICA Team yang berfokus pada membuat suatu tim yang disebut *Echelon* dari beberapa unit prajurit yang disebut *T-Dolls* (singkatan dari *Tactical Dolls*) yang setiap unitnya memiliki beberapa atribut yang dapat mendefinisikan seberapa kuat suatu unit. Kekuatan suatu *echelon* ditentukan dari beberapa mekanisme lainnya, seperti tipe serangan suatu unit, posisi suatu unit dalam tim, beberapa *attack modifier*, dan beberapa mekanisme yang mungkin belum dipublikasikan. Mekanisme-mekanisme tersebut akan menghasilkan sebuah nilai yang mendefinisikan kekuatan suatu tim. Nilai ini kedepannya akan disebut sebagai Power, dan akan menjadi fokus untuk merancang strategi untuk game ini.

### C. *Algoritma Route Planning*




*Gambar 3. Undirected Weighted Graph dengan nilai Heuristic*

Penulis mengkategorikan permasalahan pada makalah ini sebagai permasalahan optimasi. Hal ini karena ingin dicari rute dengan *cost* yang paling kecil, dan juga dengan resiko yang paling rendah. Terdapat beberapa algoritma yang sering digunakan dalam hal pencaharian rute, namun dapat dibagi menjadi dua kelompok besar, yakni *uninformed search* dan *informed search*.

*Uninformed search* adalah pencaharian rute yang tidak diberitahu informasi pendukung terkait tujuan yang sedang dicari, sehingga proses pencaharian hanya melihat pada informasi yang dimiliki sekarang. Informasi tersebut bisa saja beragam, dan pada akhirnya seluruh informasi tersebut diunifikasi sehingga menjadi satu besaran yang disebut *cost*. Algoritma yang termasuk dalam *uninformed search* antara lain Breadth First Search, Depth First Search, Uniform Cost Search.

*Informed search* adalah pencaharian rute di mana terdapat informasi terkait tujuan yang ingin dicapai. Umumnya informasi ini disebut sebagai nilai heuristik. Nilai ini memberi keterangan terkait penilaian terhadap suatu lokasi, di mana penilaian ini haruslah relevan dengan hal yang ingin dioptimisasi dalam proses pencahariannya. Nilai heuristik digunakan dalam algoritma pencaharian untuk menentukan titik yang akan dieksplorasi selanjutnya, sehingga desain dari heuristik yang baik akan memberi hasil yang lebih baik pula, untuk sebagian besar kasus. Algoritma yang termasuk dalam *informed search* antara lain Greedy Best First Search, dan A*.

A* adalah salah satu algoritma pencaharian rute yang temasuk pada *informed search*, yakni informasi mengenai tujuan pencaharian diketahui dan digunakan dalam perhitungan untuk mencari rute. Terdapat algoritma pencaharian rute yang termasuk *informed search* juga, yakni Greedy Best First Search, namun algoritma ini tidak memperhitungkan terkait *cost* untuk mencapai tujuan. Hal ini yang membuat penulis memilih algoritma A* karena sesuai dengan tujuan yang ingin dicapai, yakni rute yang paling optimal secara *cost* dan dengan resiko paling rendah.

## II. ANALISIS PERMASALAHAN

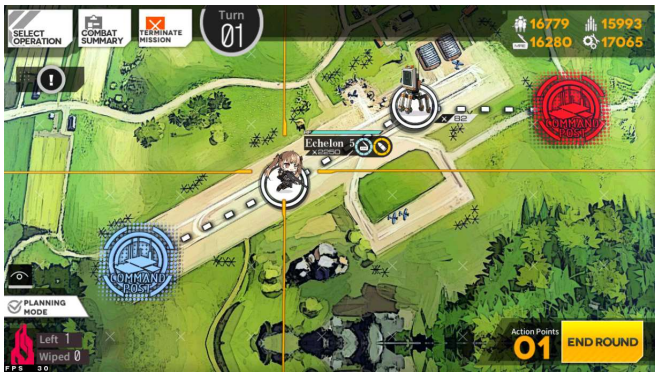
### A. *Mekanisme Permainan Girls' Frontline*


*Gambar 4. Tampilan permainan*

Permainan dimulai dengan memilih beberapa tim yang akan terjun ke lapangan untuk memulai pertempuran. Dibutuhkan minimal satu tim agar dapat memulai permainan, dan pada beberapa kesempatan, pemain dapat menerjunkan tim tambahan di tengah jalanannya permainan.


*Gambar 5. Pemilihan Echelon yang akan bertarung*

Permainan berjalan dengan sistem *turn-based*, yakni pemain dan musuh akan bergiliran melakukan suatu aksi sampai batas yang ditentukan. Setiap aksi dapat berupa memindahkan suatu tim ke node lainnya atau melakukan *deployment* untuk suatu tim ke node tertentu. Banyak aksi yang dapat dilakukan akan bertambah seiring ronde bertambah, di mana satu ronde adalah satu giliran pemain dan satu giliran musuh. Khusus untuk musuh, hanya diperbolehkan untuk melakukan satu aksi saja, namun aksi tersebut berlaku ke seluruh unit tim musuh yang berada dalam permainan, dengan beberapa pengecualian seperti *boss unit* yang akan tetap diam pada satu node sampai *boss unit* tersebut diusik oleh unit dari pemain.




*Gambar 6. Aksi bergerak*

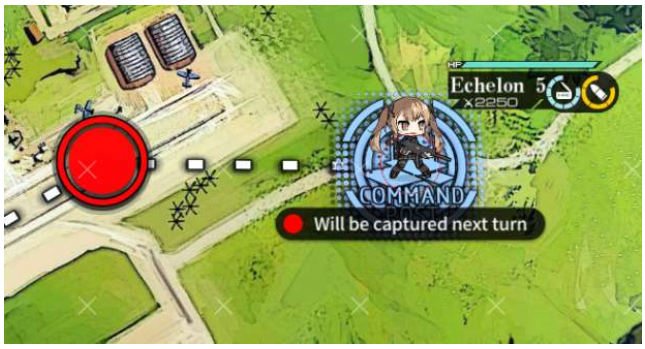
Saat pemain telah kehabisan aksi yang dapat dilakukan, maka pemain tidak akan memiliki hal lain yang dapat dilakukan selain untuk mengakhiri gilirannya. Hal ini dilakukan dengan menekan tombol "END TURN" yang terdapat pada pojok kanan bawah dari layar. Saat pemain mengakhiri gilirannya, maka setiap *node* yang terdapat unit tim dari pemain akan di-*capture* oleh pemain. Hal ini ditandai dengan berubahnya *node* menjadi warna biru. Hal yang serupa juga berlaku untuk musuh.


*Gambar 7. Aksi capture*

Setiap *node* yang dihuni oleh unit tim dari musuh akan di-*capture* oleh musuh untuk setiap akhir giliran dari musuh. Mekanisme *capture* ini akan memengaruhi hasil akhir dari permainan, di mana terdapat penambahan poin atau penghargaan apabila pemain dapat meng-*capture* seluruh *node* yang ada di dalam pertarungan.


*Gambar 8. Capture yang dilakukan oleh kedua sisi*

Permainan akan berlangsung sampai kondisi menang atau kalah terpenuhi. Kondisi menang dari setiap pertarungan adalah ter-*capture*-nya *node* dari markas musuh. Beberapa pertarungan mengharuskan pemain untuk mengalahkan *boss unit* terlebih dahulu agar dapat melakukan *capture* terhadap *node* markas musuh.




*Gambar 9. Capture terhadap markas musuh*

Sementara itu, kondisi kalah adalah saat seluruh unit tim yang dimiliki pemain dinyatakan kalah, atau *node* markas pemain dikuasai oleh musuh, ataupun pemain memutuskan untuk menyerah.

Saat unit tim pemain bertemu dengan tim musuh, bisa melalui tim pemain yang bergerak ke tim musuh atau sebaliknya, maka akan terjadi pertempuran pada *node* tersebut. Terdapat mekanisme yang kompleks saat terjadi pertempuran, ditandai dengan berubahnya mode game. Kedua tim akan bertarung untuk memperoleh pemenang dalam pertarungan tersebut. Tim yang kalah akan dipaksa untuk menyerah. Apabila tim yang kalah adalah tim dari pemain, maka tim tersebut diharuskan untuk "retreat" dan akan keluar dari permainan. Apabila tim yang kalah adalah tim dari musuh, maka tim musuh tersebut akan seketika lenyap dan *node* tempat terjadinya pertempuran akan menjadi milik pemain. Mekanisme *capture* tetap mengikuti mekanisme dasar, yakni baru berlaku apabila giliran berakhir.

Untuk menyederhanakan kasus, maka proses pertempuran ini akan disederhanakan dengan melihat perbedaan nilai Combat Effectiveness antara kedua tim. Tim yang memiliki

nilai Combat Effectiveness yang lebih tinggi akan otomatis menang, namun total Health Point yang dimiliki oleh tim tersebut akan berubah Berdasarkan perbedaan nilai Combat Effectiveness. Semakin kecil perbedaan nilai, maka besar damage yang diterima oleh tim yang menang akan semakin besar. Hal ini sejalan dengan kasus yang sering terjadi pada game ini dan cukup relevan untuk dijadikan suatu bentuk hasil penyederhanaan.

### *B. Rancangan Strategi*

Tujuan akhir dari permainan ini adalah untuk menguasai *node* markas musuh dengan menempuh langkah yang paling optimal. Definisi dari optimal yang digunakan adalah banyak aksi yang paling sedikit, dan total *damage* yang diterima oleh tim pemain yang paling sedikit.

Dengan contoh penyederhanaan kasus yang telah didefinisikan pada bagian II.A maka terlebih dahulu didefinisikan suatu prosedur yang mensimulasikan proses bertarung ini. Penulis akan menyebut prosedur ini sebagai Battle Procedure, B(p,e), di mana :

$$B(p : team, e : team)$$

Suatu tipe data $team$ didefinisikan memiliki atribut HealthPoint, MaxHealthPoint, dan CombatEffectiveness.

Hasil dari suatu pertarungan adalah satu tim yang menang dan satu tim yang kalah. Penentuan tim yang menang adalah melihat tim yang mana yang memiliki nilai CombatEffectiveness yang lebih besar. Setelah itu, tim yang menang akan mendapatkan pengurangan pada HealthPoint dengan ketentuan sebagai berikut

$$HP_{win} \leftarrow HP_{win} - MaxHP_{win} * \frac{CE_{win} - CE_{lose}}{100}$$

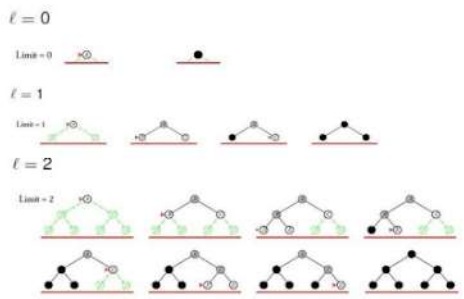
dengan :

- HP : Health Point
- MaxHP : Max Health Point
- CE : Combat Effectiveness

Prosedur ini akan dipanggil untuk setiap pertempuran yang berlangsung.

Setelah merancang definisi dari pertempuran, maka hal selanjutnya adalah memilih metode untuk menjalankan urutan aksi. Pada kasus kali ini, aksi yang dijalankan hanyalah aksi berpindah, yakni memindahkan unit tim dari *node* ke *node* lainnya. Bentuk dari perpindahan yang berada pada suatu representasi graf memberi petunjuk bahwa dapat dilakukan pencaharian rute dengan menggunakan algoritma yang terkait, yakni : BFS, DFS, UCS, Greedy Best First Search, maupun A*.

Karena mekanisme permainan yang membatasi jumlah aksi untuk setiap ronde, maka pembentukan pohon status untuk setiap algoritma yang ada akan dibatasi sampai kedalaman tertentu, yakni banyak aksi yang diperbolehkan. Hal ini hanya berlaku untuk pemain, sementara untuk setiap unit musuh hanya diperbolehkan untuk melakukan satu aksi saja, sehingga kedalaman maksimalnya adalah satu.




*Gambar 10. Ilustrasi kedalaman penelusuran graf*

Suatu aksi diberikan *cost* sebesar satu saja, dan hal ini adalah sama untuk seluruh aksi. Apabila diperlukan definisi untuk *cost function*, maka definisinya ialah "banyak aksi yang telah dilakukan untuk mencapai *node/state* ini dikali dengan nilai Combat Effectiveness musuh terbesar dalam permainan".

Hal ini akan memberi kesulitan saat memilih rute terbaik saat menggunakan algoritma BFS atau DFS, karena terdapat Batasan kedalaman dari pencaharian. Nilai dari *cost function* yang *uniform* untuk setiap aksi juga tidak akan membuat perubahan yang signifikan apabila digunakan algoritma UCS, sehingga diperlukan fungsi heuristik. Kehadiran dari fungsi heuristik membuat algoritma yang cocok untuk digunakan adalah Greedy Best First Search atau A* dan variannya.

### *C. Rancangan Heuristik*

Mencapai kondisi menang memerlukan pemain untuk memindahkan unit timnya ke *node* markas musuh lalu mengakhiri giliran agar dapat dilakukan *capture* terhadap markas tersebut. Adanya batasan dari kedalaman pencaharian membuat dibutuhkannya heuristik untuk membantu pemilihan jalur yang paling efisien dan dengan implikasi *damage* terkecil ke pemain.

Pendefinisian heuristik yang akan digunakan adalah sebagai berikut :

- Seluruh nilai heuristik (h) di-set dengan nilai 0 untuk setiap *node* yang ada pada permainan.
- Nilai h dihitung dari *node* markas musuh, dengan nilai awal h = 0 apabila tidak ada *boss unit*, atau nilai awal h = $CE_{boss}$ apabila terdapat *boss unit*.
- Dilakukan penelusuran untuk setiap jalur yang mungkin dengan menggunakan algoritma BFS, dengan *start node* ialah *node* markas musuh. Setiap *node* yang dijelajahi akan di-set nilai h-nya dengan ketentuan sebagai berikut :

  $$h_{node} = h_{parent} + CE_{node}$$

  Apabila pada node tersebut sudah terdapat nilai h dari penelusuran dari node lain, maka diambil nilai h terbesar.
- Penelusuran dilakukan hingga algoritma BFS tadi menemui *node* markas pemain, sehingga *node* markas pemain dianggap sebagai tujuan pencaharian dari algoritma BFS tadinya. Nilai h dari node markas pemain akan diberi nilai h = inf

Selanjutnya, untuk setiap unit tim musuh yang ada di permainan, nilai h untuk setiap *node* yang terhubung dengan *node* tempat beradanya unit musuh ditambah dengan nilai $CE_{node}$.

Alasan penggunaan model heuristic ini adalah untuk membuat unit tim akan selalu memilih jalur dengan resiko terendah. Mengingat nilai cost function yang uniform, maka nilai f(n) untuk suatu node n adalah :

$$f(n) = c(n) + h(n)$$

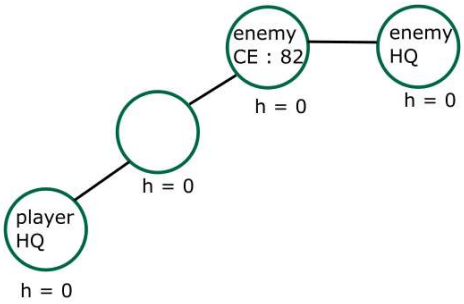
Sehingga, penggunaan algoritma A* akan menggunakan nilai fungsi f(n) sebagai parameter pencahariannya, dan akan selalu memilih node dengan nilai f(n) terkecil sebagai node yang akan diekspansi.

*D. Studi Kasus*

Kita ambil contoh kasus sederhana pada stage 1-1 Drill yang ada pada game, yang memberikan permainan seperti pada gambar berikut.

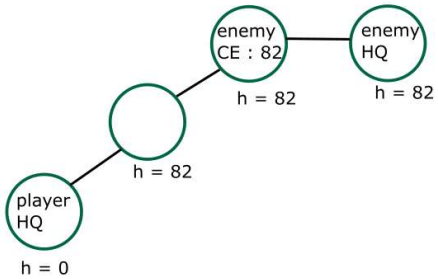

*Gambar 11. Stage 1-1*

Terdapat 4 buah *node*, satu unit tim dari musuh dan tidak ada *boss unit*. Representasi graf dengan nilai h pada kondisi awal adalah sebagai berikut.




*Gambar 12. Inisialisasi nilai h*

Karena tidak ada keberadaan *boss unit* pada *node* markas musuh, maka nilai h akan tetap seperti pada kondisi awal. Selanjutnya dilanjutkan untuk iterasi pada setiap unit tim musuh yang ada. Pada kasus kali ini hanya ada satu, sehingga nilai h setelah iterasi adalah sebagai berikut.



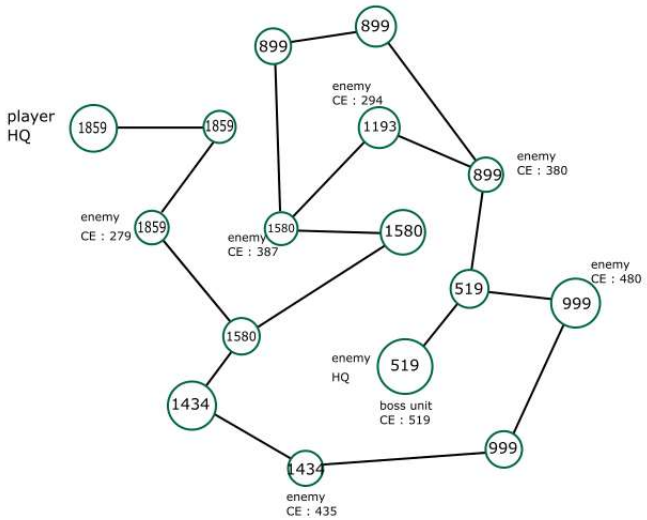

*Gambar 13. Nilai h untuk setiap node setelah perhitungan*

Untuk kasus ini maka akan sangat jelas bahwa hanya akan ada satu jalur yang paling optimal.

Studi kasus selanjutnya akan menggunakan permainan yang memiliki lebih banyak *nodes*.

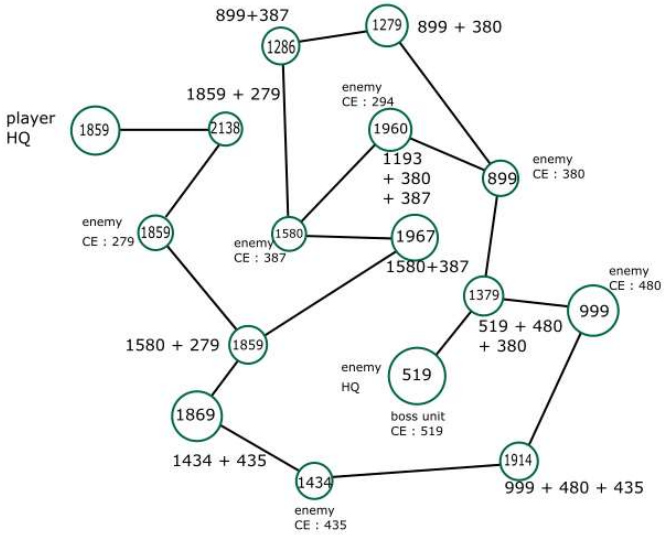

*Gambar 14. Stage 1-6*

Berikut adalah representasi graf setelah diberikan label dan melalui iterasi pertama.




*Gambar 15. Graf dengan nilai h untuk stage 1-6 : Iterasi awal*

Berikut adalah representasi graf setelah dilakukan penambahan nilai h pada setiap unit tim musuh yang ada.




*Gambar 16. Representasi graf dengan nilai h untuk stage 1-6. Setelah iterasi untuk setiap unit musuh.*

Adanya perubahan pada nilai h untuk setiap node yang bertetangga dengan unit tim musuh menandakan bahwa node tersebut lebih berbahaya apabila dikunjungi. Hal ini dikarenakan node tersebut adalah salah satu kandidat node yang akan menjadi tujuan bergeraknya unit tim musuh saat tiba giliran dari musuh untuk bergerak.

### E. *Implementasi A**

Implementasi untuk algoritma A* cukup *straightforward*, di mana digunakan fungsi f(n) yang telah didefinisikan pada bagian 2.C. Adapun beberapa hal yang memberi batasan sebagai respon terhadap mekanisme permainan adalah sebagai berikut.

1. Proses pencaharian rute hanya terbatas sampai n buah gerakan, di mana n adalah banyaknya aksi yang diperbolehkan pada saat giliran pemain. Banyaknya aksi ini dapat dilihat pada Heads-Up Display yang terdapat pada antarmuka permainan di pojok kanan bawah.
2. Pencaharian berhenti apabila telah dicapai *node* markas musuh ataupun sudah mencapai kedalaman maksimal yang diperbolehkan.
3. Apabila pada kedalaman maksimal masih tidak ditemui *node* markas musuh, maka dipilih *node* dengan nilai f(n) terkecil, lalu dilakukan *backtrack* untuk menentukan rute yang akan ditempuh. Pemilihan rute dengan tujuan akhir sebuah *node* dengan nilai f(n) terkecil akan memberi rute yang paling aman dan paling singkat.

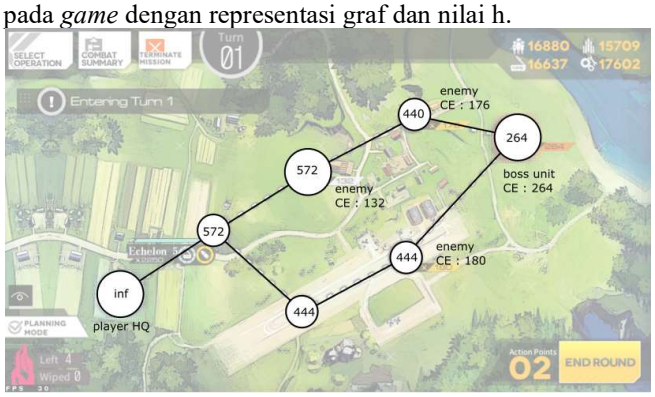
Adanya batasan ataupun ketentuan pada penggunaan algoritma A* untuk mekanisme permainan ini akan menjamin pemilihan rute akan bisa diaplikasikan langsung dengan permainan yang asli.

## III. IMPLEMENTASI STRATEGI

### A. *Immobile Enemy Case*

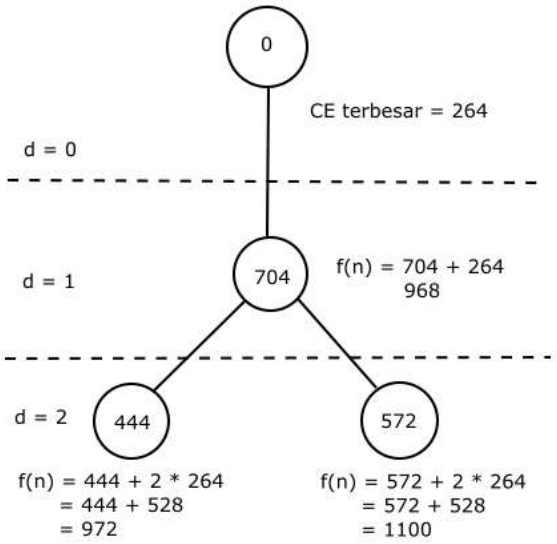
Sebelumnya telah dijelaskan bahwa unit musuh dapat melakukan pergerakan sebanyak satu kali untuk setiap giliran yang diberikan, untuk setiap unit musuh yang ada. Namun terdapat pengecualian untuk beberapa tipe unit musuh. Terdapat tipe unit yang *immobile* atau tidak bergerak/static, dan terdapat tipe unit yang hanya akan bergerak apabila telah mengalami pertempuran sebelumnya.

Kali ini akan digunakan contoh dari stage 1-2 untuk mendemonstrasikan alur kerja strategi dengan unit tim musuh hanya berupa unit *immobile*. Berikut adalah peta dari 1-2 pada *game* dengan representasi graf dan nilai h.




*Gambar 17. Tampilan permainan stage 1-2 dan graf dengan nilai h, superimposed.*

Terdapat keterangan bahwa giliran kali ini memperbolehkan pemain untuk melakukan 2 aksi. Dapat dilihat j Dengan demikian, kedalaman maksimal untuk giliran kali ini ialah 2. Pembuatan pohon status dengan memanfaatkan algoritma A* akan menghasilkan pohon status sebagai berikut.

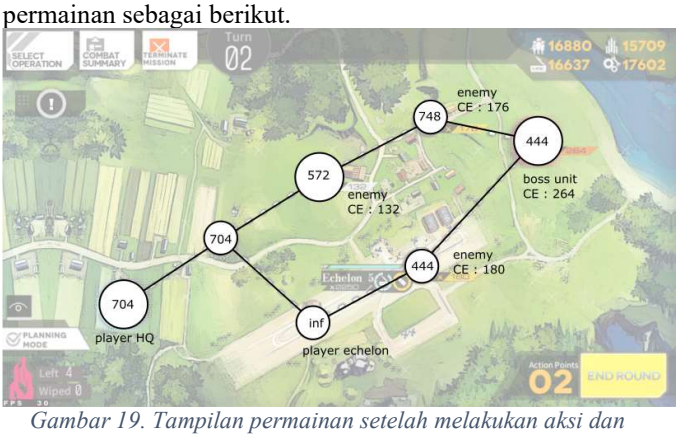



*Gambar 18. Pohon ruang status untuk gambar 17*

Hasil pada giliran ini akan memberi keputusan berupa pemindahan unit tim pemain dari markas pemain ke node 704

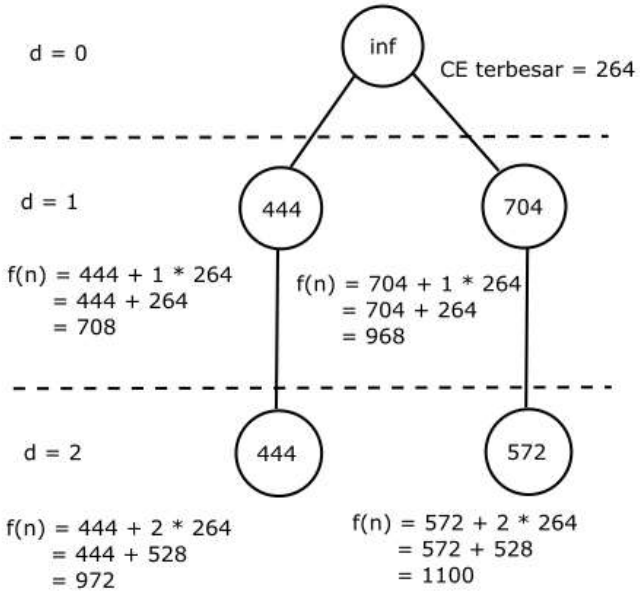
lalu ke node 444. Setelah ini, giliran pemain akan diakhiri karena aksi sudah mencapai batas, lalu node 704 akan dilakukan capture. Saat sampai pada giliran dari musuh, node yang di-duduki oleh musuh juga dilakukan capture, lalu giliran musuh diakhiri.

Selanjutnya, tiba giliran pemain lagi, dan prosedur penghitungan nilai h dilakukan kembali. Diperoleh kondisi permainan sebagai berikut.




*Gambar 19. Tampilan permainan setelah melakukan aksi dan representasi graf dengan nilai h, superimposed*

Posisi mulainya pencaharian diubah ke posisi unit tim pemain berada sekarang, lalu dilakukan pencaharian sampai kedalaman d = 2, sesuai jumlah aksi yang diperbolehkan.




*Gambar 20. Pohon ruang status untuk gambar 19*

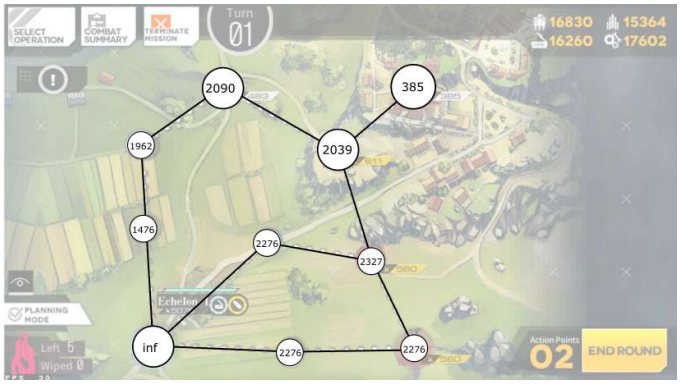
Pohon status ini menunjukkan bahwa rute terbaik yang dipilih adalah dari posisi saat ini ke node 444, lalu ke node 444 selanjutnya. Unit tim pemain akan melakukan pertarungan dengan unit tim yang berada pada node tujuan.


*Gambar 21. Kondisi menang.*

Karena nilai CE unit tim jauh lebih besar dari nilai CE unit tim lawan (2250 > 264), (2250 > 180), maka unit pemain akan menang, lalu giliran akan diakhiri karena sudah menghabiskan aksi yang tersedia. Sistem akan melakukan capture, lalu menyatakan pemain sebagai pemenang karena telah berhasil meng-*capture* markas musuh.

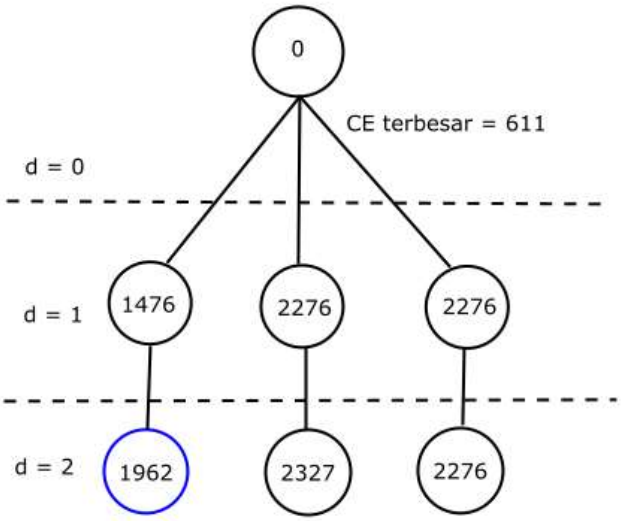
### *B. Mobile Enemy Case*

Pada kasus ini, terdapat beberapa unit musuh yang akan bergerak apabila sudah tiba pada giliran musuh. Adanya pergerakan ini akan memengaruhi nilai h yang terdapat pada seluruh *node* yang ada pada permainan, sehingga prosedur penghitungan nilai h dilakukan kembali. Adapun pergerakan dari setiap unit musuh tidak dapat diprediksi dengan akurasi 100%, namun memiliki kecenderungan untuk bergerak mendekati unit tim pemain terdekat. Berikut adalah peta dari permainan pada stage 2-1 dengan representasi graf dan nilai h.




*Gambar 22. Tampilan permainan untuk stage 2-1 dan representasi graf dengan nilai h, superimposed*

Dengan banyak aksi maksimal untuk giliran ini adalah 2, maka pohon ruang statusnya ialah.

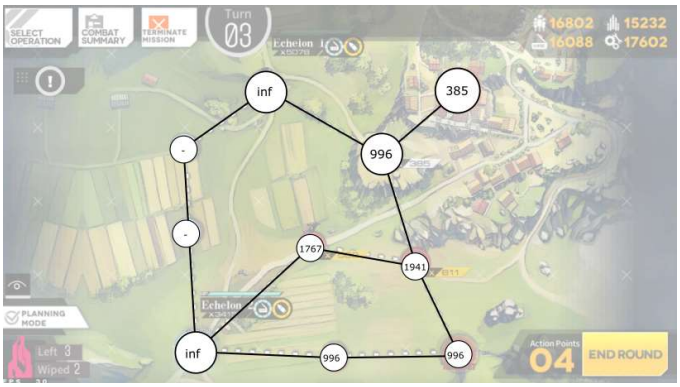



*Gambar 23. Pohon ruang status untuk gambar 22*

Diperoleh langkah yang ditempuh adalah dengan menggerakkan unit ke node 1476 lalu ke node 1962. Kondisi permainan sekarang adalah sebagai berikut.


*Gambar 24. Tampilan permainan setelah aksi dilakukan.*

Pada kondisi ini, terdapat dua unit tim musuh yang posisinya bertetangga dengan *node* markas pemain, sehingga diperlukan langkah untuk melindungi markas pemain. Hal ini memerlukan penggunaan satu aksi yakni menerjunkan satu unit tim baru ke *node* markas pemain, sehingga jatah aksi yang dapat dilakukan hanyalah satu aksi saja untuk unit tim pertama tadi. Karena hanya tersisa satu aksi dan hanya terdapat satu *node* yang terhubung langsung dengan posisi unit tim pertama, maka hasil dari algoritma tentu saja akan memilih node 2090 pada gambar …. Setelah mengakhiri giliran dan melanjutkan permainan hingga giliran musuh juga selesai, kondisi permainan sekarang adalah sebagai berikut.

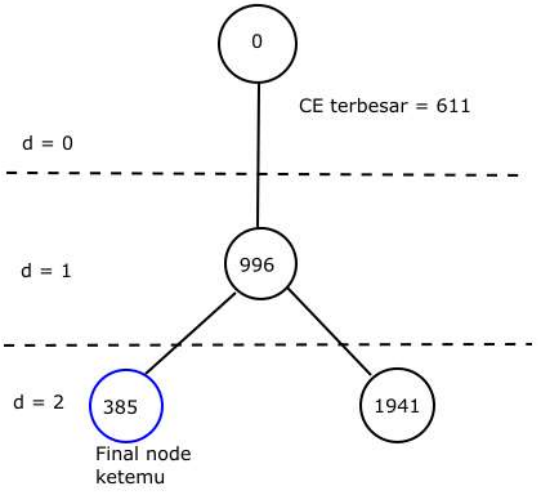



*Gambar 25. Tampilan permainan setelah penerjunan unit baru dan aksi pindah.*

Node yang telah dilewati diberi tanda "-" untuk menandakan bahwa node tersebut berada di luar perhitungan. Untuk giliran kali ini, diberikan 4 kesempatan untuk melakukan aksi, yang berlaku untuk keseluruhan unit tim, bukan untuk masing-masing unit tim pemain. Kondisi ini memberikan banyak kemungkinan langkah yang dapat ditempuh, mengingat aksi yang diperbolehkan bukan hanya menggerakkan unit tim.

Untuk mempermudah kasus, unit tim kedua (yang paling barusan diterjunkan) dijadikan sebagai unit pertahanan saja, yakni membuatnya berjaga di markas pemain. Hal ini membuat pemain dapat fokus ke unit tim pertama yang sudah berada di sisi atas dari peta.

Penelusuran pohon pencaharian status dengan kedalaman maksimal 4 akan memberi hasil sebagai berikut.




*Gambar 26. Pohon ruang status untuk gambar 25*

Karena final node, yakni node markas musuh, telah ketemu, maka pencaharian dihentikan. Sehingga langkah yang ditempuh adalah ke node 996 lalu ke node 385, lalu akhiri giliran. Kondisi ini akan memberikan winning condition, sehingga permainan berakhir.


*Gambar 27. Kondisi menang*

Hasil ini memperlihatkan bahwa ada beberapa *edge cases* yang perlu untuk dicakup dalam rancangan strategi kita. Beberapa diantaranya adalah pembuatan mode defensive maupun mode ofensif, dan pergerakan/aksi yang melibatkan lebih dari satu unit tim.

Penggunaan dari algoritma A* dengan definisi heuristik yang telah dibahas membuktikan bahwa sebuah strategi untuk *game* yang melibatkan penggunaan graf sebagai komponen utamanya dapat dibuat dengan menggunakan beberapa algoritma pencaharian rute, di mana yang dibahas pada makalah ini adalah algoritma A*.

## IV. Hasil dan Kesimpulan

### A. *Hasil*

Penggunaan algoritma A* dalam menentukan rute terbaik untuk memenangkan permainan terbukti ampuh untuk memberikan hasil yang terbilang optimum. Terdapat beberapa kasus yang belum sempat ditanggulangi, seperti adanya unit tim musuh yang berpotensi merebut *node* markas pemain sehingga terjadi kondisi kalah, ataupun adanya unit tim yang tidak melakukan apa-apa.

Penggunaan dari heuristik ini juga bersifat non-deterministik, karena perpindahan dari unit musuh tidak dapat diprediksi dengan tepat. Terlebih pada tingkat kesulitan yang lebih tinggi, musuh dapat menerjunkan unit baru ke dalam permainan, dan unit tersebut memiliki nilai Combat Effectiveness yang acak. Hal ini menunjukkan bahwa mekanisme permainan yang semakin kompleks memerlukan rancangan strategi juga menjadi semakin kompleks, dengan memerhitungkan keberadaan beberapa tipe musuh.

Poin-poin yang dapat dikembangkan dalam rancangan strategi ini adalah pendefinisian yang lebih baik untuk *cost function*, karena definisi dari *cost function* yang ada sekarang hanya berdasar pada banyak langkah yang ditempuh. Beberapa variabel yang dapat dimasukkan dalam perhitungan adalah tipe musuh, *health point* yang dimiliki oleh unit tim saat itu, perhitungan resiko yang lebih cermat.

### B. *Kesimpulan*

Algoritma penelusuran graf dapat digunakan untuk mencari solusi optimal untuk permasalahan yang melibatkan graf. Pada kasus untuk algoritma A*, pendefinisian *cost function* dan fungsi heuristik yang baik dan benar akan memberi keputusan yang optimal dengan kompleksitas waktu maupun ruang yang lebih kecil.

## Tautan Video Youtube

Penulis membuat video yang berisi penjelasan terhadap makalah ini beserta demonstrasi secara langsung terhadap aplikasi strategi ini. Video tersebut dapat diakses melalui tautan berikut. https://www.youtube.com/watch?v=jUGV1lYz96g

## V. Ucapan Terima Kasih

Penulis mengucapkan terima kasih yang sebesar-besarnya kepada Allah SWT yang telah memberi saya kesehatan dan kesempatan dalam melalui serangkaian prosesi perkuliahan IF2211 Strategi Algoritma, hingga dapat menjalankan UAS dan menyelesaikan makalah ini. Ucapan terima kasih penulis sampaikan kepada seluruh tim dosen IF2211, yakni Bapak Dr. Ir. Rinaldi Munir, MT., Ibu Dr. Nur Ulfa Maulidevi, ST, M.Sc., dan Ibu Dr. Masayu Leylia Khodra, ST., MT. Berkat bimbingan beliau penulis bisa menyelesaikan makalah ini dengan baik.

Ucapan terima kasih juga penulis sampaikan kepada seluruh pihak yang telah membantu penulis dalam menyelesaikan makalah ini. Mulai dari orang tua, kerabat, sampai ke teman-teman Teknik Informatika ITB Angkatan 2018, dan teman-teman kelas K2.

## Referensi

[1] Munir, Rinaldi. *Route/Path Planning Using A Star and UCS*, http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/A-Star-Best-FS-dan-UCS-(2018).pdf, diakses 30 April 2020

[2] Combat – Girls' Frontline Wiki, https://en.gfwiki.com/wiki/Combat, diakses 01 Mei 2020

## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 02 Mei 2020

Arung Agamani Budi Putera - 13518005