Algoritma A* untuk Menentukan Rute Tempuh pada Berbagai Medan di Permainan Simulasi Koloni

Fata Nugraha 13517109
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
fata@students.itb.ac.id

***Abstraksi*—Masalah penentuan rute terpendek sudah menjadi permasalahan umum di dunia informatika. Secara umum, masalah ini dapat dibagi menjadi pencarian rute terpendek tanpa bobot dan dengan bobot. Algoritma A* adalah salah satu algoritma yang digunakan untuk menyelesaikan masalah ini, dan makalah ini akan membahas penerapan algoritma A* untuk menentukan rute terpendek pergerakan kolonis pada permainan simulasi koloni RimWorld.**

***Kata kunci*—Algoritma A*, Graf, Kolonis, Medan Tempuh, RimWorld**

I. PENDAHULUAN

Salah satu masalah yang sering dibahas di dunia informatika adalah masalah pathfinding atau pencarian rute terpendek, karena masalah ini memiliki banyak penerapan yang serupa di dunia nyata dan penyelesaian masalah ini dengan algoritma brute force memiliki kompleksitas yang tinggi. Ada beberapa cara untuk menyelesaikan masalah pencarian jalur ini, seperti Algoritma Dijkstra, Uniform Cost Search (UCS), Greedy Best-First Search, dan A*.

Salah satu permainan yang memanfaatkan algoritma A* adalah sebuah permainan simulasi koloni yang bernama RimWorld. Pada permainan ini, setiap kolonis akan secara otomatis bergerak untuk menjalankan tugasnya masing-masing, dan sebisa mungkin akan mencari jalan terbaik yang bisa dilalui untuk menghemat waktu perjalanannya, misalnya dengan memilih rute melalui jalan beraspal daripada rute yang melalui rawa atau tanah berlumpur. Karena semua pergerakan yang dilakukan kolonis adalah pergerakan dari suatu posisi ke posisi lainnya, kita bisa melihat penerapan algoritma A* pada permainan ini.

Gambar 1.1 RimWorld
Sumber : https://steamcdn-a.akamaihd.net/steam/apps/294100/header.jpg?t=1541837948

II. TEORI DASAR

A. Definisi Graf

Graf adalah sebuah struktur diskrit yang terdiri dari simpul dan sisi yang menghubungkan simpul-simpul tersebut. Graf juga dapat didefinisikan sebagai pasangan himpunan simpul-simpul dan himpunan sisi yang menghubungkan sepasang simpul, dengan himpunan simpulnya adalah himpunan tak kosong. Artinya, syarat agar sebuah graf bisa terdefinisi adalah setidaknya graf tersebut memiliki satu buah simpul.

Berdasarkan ada atau tidaknya sisi gelang atau sisi ganda pada suatu graf, graf dibedakan menjadi dua kategori yaitu graf sederhana dan graf tidak sederhana. Graf sederhana adalah graf yang tidak mengandung sisi gelang atau sisi ganda. Kebalikannya, graf tak sederhana adalah graf yang mengandung sisi gelang atau sisi ganda.

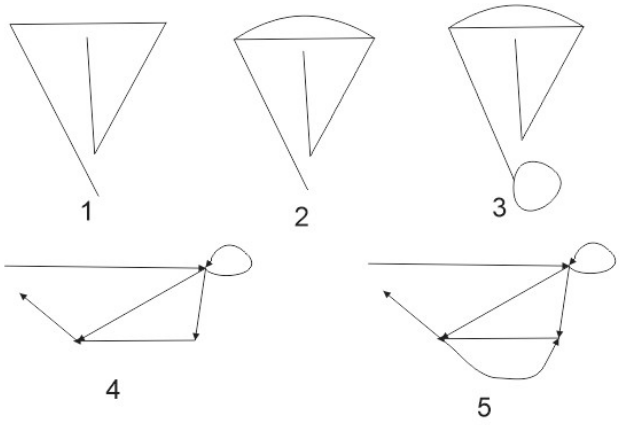
Berdasarkan keberadaan sisi berarah pada graf, graf juga dibagi menjadi dua kategori. Kategori yang pertama adalah graf tidak berarah, yaitu graf yang sisi-sisinya tidak memiliki arah. Kategori yang kedua adalah graf berarah, yaitu graf yang sisi-sisinya memiliki arah. Berdasarkan dua jenis

pengategorian ini, kita bisa membuat sebuah tabel yang berisikan jenis-jenis graf seperti berikut :

Tabel 2.1 Jenis-jenis graf

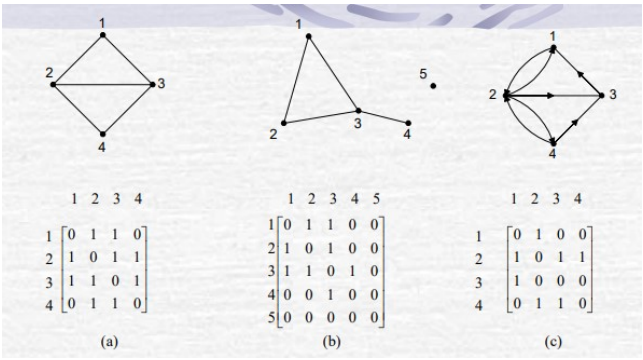
Jenis	Sisi	Sisi ganda	Sisi gelang
Graf sederhana	Tak berarah	Tidak	Tidak
Graf ganda	Tak berarah	Ya	Tidak
Graf semu	Tak berarah	Ya	Ya
Graf berarah	Berarah	Tidak	Ya
Graf ganda berarah	Berarah	Ya	Ya

Gambar di bawah ini secara berurutan menggambarkan graf sederhana (1), graf ganda (2), graf semu (3), graf berarah (4), dan graf ganda berarah (5).



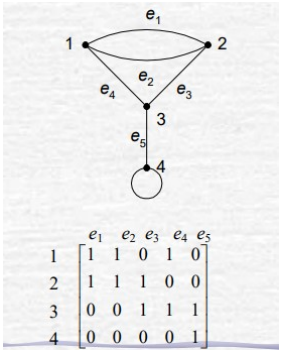
Gambar 2.1 Jenis-jenis graf
Sumber : Penulis

Sebuah graf bisa direpresentasikan dengan matriks ketetanggaan, matriks bersisian, dan senarai ketetanggaan. Berikut ini adalah contoh dari masing-masing representasi tersebut :

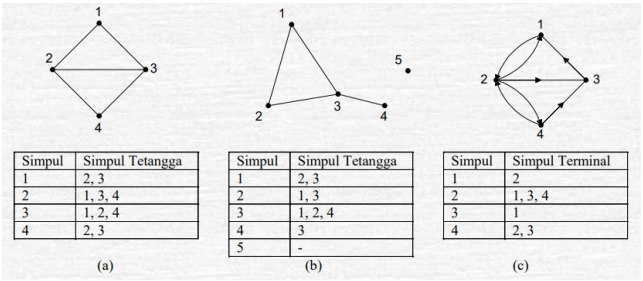


Gambar 2.2 Matriks ketetanggaan

Sumber :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf



Gambar 2.3 Matriks bersisian
Sumber :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf



Gambar 2.4 Senarai ketetanggaan
Sumber :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf

B. Algoritma A* (A-star)

Algoritma A* adalah salah satu algoritma yang banyak digunakan untuk kasus pencarian jalur di dalam suatu graf. Algoritma ini pertama kali ditemukan oleh Peter Hart, Nils Nilsson, dan Bertram Raphael pada tahun 1968. Algoritma ini banyak digunakan karena dapat menghasilkan jawaban terpendek dengan kompleksitas yang tidak terlalu tinggi.

Cara kerja algoritma ini adalah dengan memanfaatkan sebuah list yang menyimpan jumlah cost yang sudah dilalui (g(n)) ditambah fungsi heuristik yang memprediksi jarak dari titik terakhir ke titik tujuan (h(n)). Lalu untuk setiap iterasi, algoritma ini memilih jalur yang memiliki g(n) + h(n) terkecil.

A* Algorithm pseudocode

The goal node is denoted by node_goal and the source node is denoted by node_start

We maintain two lists: **OPEN** and **CLOSE**:

OPEN consists on nodes that have been visited but not expanded (meaning that sucessors have not been explored yet). This is the list of pending tasks.

CLOSE consists on nodes that have been visited *and* expanded (sucessors have been explored already and included in the open list, if this was the case).

```
1  Put node_start in the OPEN list with f(node_start) = h(node_start) (initialization)
2  while the OPEN list is not empty {
3    Take from the open list the node node_current with the lowest
4         f(node_current) = g(node_current) + h(node_current)
5    if node_current is node_goal we have found the solution; break
6    Generate each state node_successor that come after node_current
7    for each node_successor of node_current {
8      Set successor_current_cost = g(node_current) + w(node_current, node_successor)
9      if node_successor is in the OPEN list {
10       if g(node_successor) ≤ successor_current_cost continue (to line 20)
11     } else if node_successor is in the CLOSED list {
12       if g(node_successor) ≤ successor_current_cost continue (to line 20)
13       Move node_successor from the CLOSED list to the OPEN list
14     } else {
15       Add node_successor to the OPEN list
16       Set h(node_successor) to be the heuristic distance to node_goal
17     }
18     Set g(node_successor) = successor_current_cost
19     Set the parent of node_successor to node_current
20   }
21   Add node_current to the CLOSED list
22 }
23 if(node_current != node_goal) exit with error (the OPEN list is empty)
```

Gambar 2.5 Pseudocode Algoritma A*
Sumber : http://mat.uab.cat/~alseda/MasterOpt/AStar-Algorithm.pdf

Gambar 2.6 Perbandingan algoritma Dijkstra, Greedy Best-First, dan A*
Sumber : https://www.redblobgames.com/pathfinding/a-star/introduction.html

C. RimWorld

RimWorld adalah sebuah permainan simulasi koloni yang dikembangkan oleh Tynan Sylvester dan Ludeon Studios. Permainan ini bukanlah permainan kompetitif, tetapi permainan yang menggunakan kecerdasan buatan untuk menganalisis situasi permainan saat ini dan menentukan peristiwa yang akan terjadi berikutnya. Tujuan akhir dari permainan ini adalah membuat pesawat luar angkasa dan membawa koloni kita ke luar angkasa, tetapi pada kenyataannya kebanyakan pemain tidak melakukan hal ini karena butuh waktu yang sangat lama dan kesabaran yang tinggi untuk mengembangkan suatu koloni dari awal, dan pada akhirnya memilih untuk tetap mengembangkan lebih lanjut koloni yang sudah dibuatnya, atau membuat koloni baru yang terpisah dari koloni utamanya.

Pada awal permainan, kita diberikan peta dunia untuk memilih tempat awal dari koloni kita. Terdapat dua belas bioma berbeda pada permainan ini, yang dibagi menjadi tiga kategori, yaitu bioma hangat, bioma panas, dan bioma dingin. Sebuah bioma adalah sekumpulan tempat berbentuk segi-enam yang masing-masing tempatnya memiliki karakteristiknya sendiri, seperti pegunungan, sungai, laut, atau jalan raya yang menghubungkan antar wilayah. Peta awal dunia ini juga merupakan salah satu penerapan graf, tapi hal itu tidak akan dibahas di sini.

Setelahnya, kita bisa memilih tiga kolonis awal untuk koloni kita. Setiap kolonis memiliki kelebihan dan kelemahannya masing-masing, sehingga sangat disarankan untuk memilih kombinasi yang paling optimal dari pilihan yang ada. Tergantung pada cerita awal yang kita pilih, ada beberapa variasi cerita yang memberikan lima kolonis dengan sumber daya yang lebih sedikit, atau hanya satu kolonis dengan sumber daya yang lebih banyak. Setelah selesai memilih kolonis, simulasi koloni akan dimulai dari membuat tempat tinggal dan bertahan hidup dari alam sekitar.

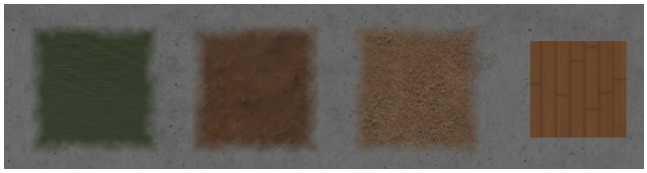
III. MEDAN TEMPUH YANG DAPAT DILALUI KOLONIS

Di permainan ini, ada berbagai macam medan yang tersebar di setiap wilayahnya, mulai dari tanah biasa, tanah berkerikil, tanah berlumpur, rawa, air dangkal, air dalam yang tidak bisa dilewati, sampai lantai buatan dari kayu atau semen yang bisa diletakkan di atas tanah yang tidak basah. Setiap medan tempuh memiliki pengali kecepatannya masing-masing, misalnya untuk setiap lantai buatan pengali kecepatannya adalah 100%. Sementara itu, tiga medan tempuh yang saya pilih untuk makalah ini adalah tanah biasa dengan pengali kecepatan sebesar 87%, tanah berlumpur dengan pengali kecepatan 52%, dan rawa dengan pengali kecepatan 36%. Masih ada banyak lagi variasi medan tempuh di permainan ini, tetapi saya memilih tiga medan tempuh ini karena jarak antar pengalinya yang cukup jauh.

Pada permainan ini, setiap kolonis akan bergerak secara otomatis sesuai dengan tugas yang diberikan pada mereka. Tetapi, jika kita mau, kita bisa menggerakkan setiap kolonis kita secara manual, misalnya, ketika sedang menghadapi serangan dari pihak musuh atau ketika kita akan menyerang musuh.

Permainan ini akan secara otomatis mencari jalur yang bisa ditempuh kolonis tersebut untuk sampai ke tempat yang hendak dituju. Selain itu, permainan ini juga secara otomatis mencari jalan dengan waktu tempuh yang paling singkat, dalam hal ini, kolonis akan berusaha menghindari medan tempuh dengan pengali yang lebih kecil dan memilih jalur memutar yang memiliki pengali kecepatan lebih besar. Tetapi ada kasus yang menyebabkan jalur dengan pengali yang lebih besar menghasilkan waktu tempuh yang lebih lama daripada melewati jalur dengan pengali yang lebih kecil, sehingga terkadang kolonis akan memilih jalur yang tidak memutar dan melalui tanah berlumpur atau rawa.

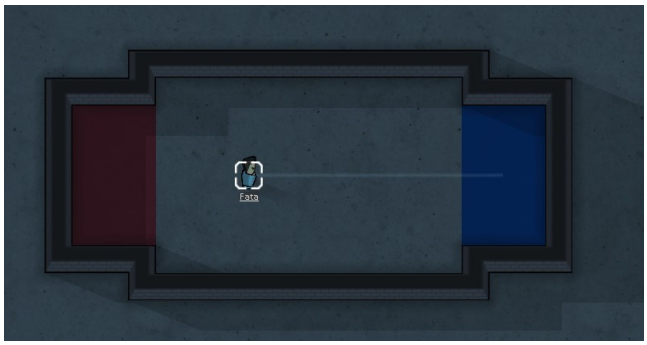
Terdapat tiga jenis medan tempuh yang digunakan pada percobaan kali ini. Medan tempuh yang pertama adalah rawa, dengan pengali kecepatan sebesar 36%, dilambangkan dengan warna hijau. Medan tempuh yang kedua adalah tanah berlumpur, dengan pengali kecepatan sebesar 52%, dilambangkan dengan warna coklat tua. Medan tempuh yang ketiga adalah tanah biasa, dengan pengali kecepatan sebesar 87%, dilambangkan dengan warna coklat muda. Pada gambar di bawah ini, kotak urutan keempat adalah lantai kayu, dengan pengali kecepatan 100%, sama dengan kondisi lingkungan di sekitarnya yang berlapis beton, dengan pengali kecepatan sebesar 100%.



Gambar 3.1 Contoh Medan Tempuh yang digunakan
Sumber : Penulis

IV. UJI KASUS

Pada kasus pertama, kolonis diperintahkan untuk bergerak lurus dari karpet merah ke karpet biru. Karena seluruh medan tempuh yang ada adalah beton, maka kolonis langsung mengambil jalur yang terpendek, yaitu garis lurus menuju tujuan.

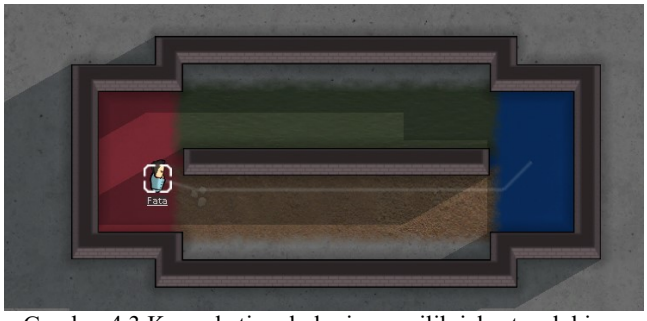


Gambar 4.1 Kasus pertama, kolonis langsung bergerak lurus menuju tujuan
Sumber : Penulis

Pada kasus kedua, terdapat medan tempuh rawa yang menghalangi jalur lurus kolonis untuk mencapai tujuan. Karena kolonis merasa bahwa waktu tempuh garis lurus akan lebih lama daripada waktu tempuh memutar, maka kolonis mengambil jalan memutar untuk menuju ke titik tujuan.

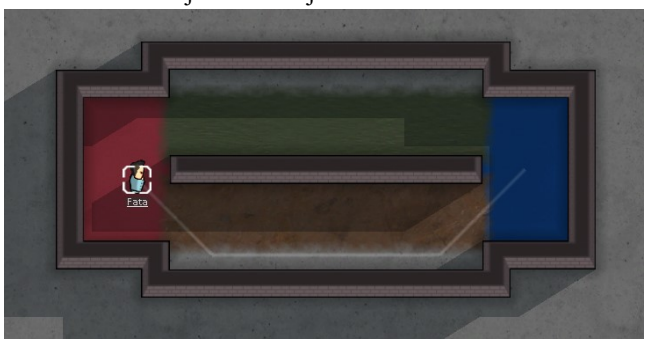
Gambar 4.2 Kolonis memilih jalan memutar untuk menghindari rawa
Sumber : Penulis

Pada kasus ketiga, terdapat dua buah medan tempuh yang dapat dilalui oleh kolonis, yaitu rawa dan tanah biasa. Karena kolonis merasa bahwa waktu tempuh melalui rawa (kecepatan 36%) akan lebih lama daripada waktu tempuh melalui tanah biasa (kecepatan 87%), maka kolonis mengambil jalur tanah biasa untuk menuju ke titik tujuan.



Gambar 4.3 Kasus ketiga, kolonis memilih jalur tanah biasa untuk mencapai tujuan
Sumber : Penulis

Pada kasus keempat, terdapat dua buah medan tempuh yang dapat dilalui oleh kolonis, yaitu rawa dan tanah berlumpur. Karena kolonis merasa bahwa waktu tempuh melalui rawa (kecepatan 36%) dan waktu tempuh melalui tanah berlumpur (kecepatan 52%) lebih lama daripada melalui jalan memutar di pinggir yang masih menggunakan beton (kecepatan 100%, dengan jarak yang lebih jauh), maka kolonis mengambil jalur beton untuk menuju ke titik tujuan.

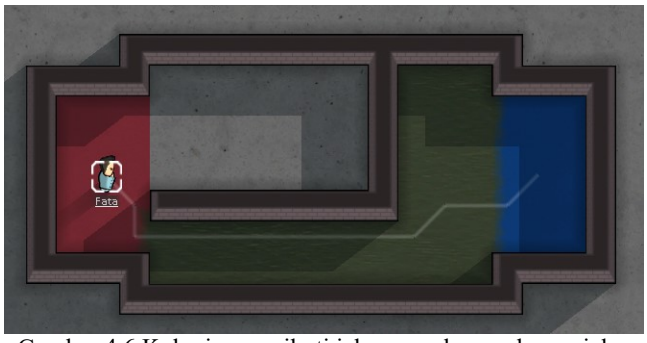


Gambar 4.4 Kasus keempat, kolonis memilih jalur beton dan memutar untuk mencapai tujuan
Sumber : Penulis

Pada kasus kelima, terdapat sebuah jalur yang dibuat dari lantai kayu (kecepatan 100%) dan medan tempuh di sekitarnya adalah rawa (kecepatan 36%). Berdasarkan gambar tersebut, terlihat jelas kolonis berusaha mengikuti jalur lantai kayu yang dibuat, karena jalur itu adalah jalur yang memiliki waktu tempuh paling singkat.

Gambar 4.5 Kolonis mengikuti jalur lantai kayu untuk menghindari rawa
Sumber : Penulis

Pada kasus keenam, terdapat sebuah wilayah yang dilapisi beton sedangkan wilayah lainnya adalah rawa. Karena kolonis mendeteksi bahwa tidak mungkin bisa mencapai tujuan jika melalui daerah berlapis beton, maka kolonis memutuskan untuk melalui jalur rawa, meskipun kecepatannya menjadi lebih lambat.



Gambar 4.6 Kolonis mengikuti jalur rawa karena hanya jalur itu yang bisa digunakan untuk mencapai tujuan
Sumber : Penulis

V. KESIMPULAN

Permainan simulasi koloni RimWorld adalah salah satu contoh permainan yang memanfaatkan algoritma A* untuk pergerakan kolonisnya. Permainan ini sebenarnya masih jauh lebih kompleks dan masih menerapkan banyak algoritma-algoritma lain. Makalah ini masih bisa dikembangkan lebih lanjut, karena pada makalah ini hanya membahas interaksi kolonis dengan medan tempuh yang ada di peta, belum membahas interaksi ketika kolonis bertemu objek bergerak atau kolonis lainnya lalu rute awal yang sudah dipilih terhalangi, sehingga kolonis tersebut harus mencari rute baru dari tempat ia berada sekarang.

VI. UCAPAN TERIMA KASIH

Pertama-tama, saya ucapkan puji syukur ke hadirat Tuhan Yang Maha Esa karena atas rahmat-Nya saya dapat menyelesaikan makalah ini. Lalu saya juga berterima kasih kepada Dr. Nur Ulfa Maulidevi ST.M.Sc., Dr. Masayu Leylia Khodra ST.MT., dan Dr. Ir. Rinaldi Munir, MT. selaku tim dosen mata kuliah IF2211 Strategi Algoritma yang telah memberikan bimbingan dan arahan sehingga saya dapat menyelesaikan mata kuliah ini sebaik mungkin. Tidak lupa saya juga mengucapkan terima kasih kepada orang tua, teman-teman, dan seluruh pihak yang tidak dapat saya sebutkan satu per satu.

REFERENSI

[1] Munir, Rinaldi. 2009. Diktat Kuliah IF2211 Strategi Algoritma. Bandung.

[2] Slide Kuliah Graf IF2120, informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2015-2016/Graf%20(2015).pdf

[3] Slide Kuliah A* IF2211 Strategi Algoritma, http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/A-Star-Best-FS-dan-UCS-(2018).pdf

[4] rimworldwiki.com, diakses pada tanggal 25 April 2019

[5] rimworldgame.com, diakses pada tanggal 25 April 2019

[6] https://www.redblobgames.com/pathfinding/a-star/introduction.html, diakses pada tanggal 25 April 2019

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, 26 April 2019

Fata Nugraha 13517109