Implementasi Greedy Best First Search dalam Permainan Pac-Man

Juan Louis Rombetasik *13519075*
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): mizuday5.7@gmail.com
E-mail (school): 13519075@std.stei.itb.ac.id

***Abstract*—Pac-Man adalah suatu permainan arcade yang sampai sekarang masih dikenang banyak orang. Konsep dari permainan Pac-Man sangat sederhana, pemain harus memakan semua titik kecil yang dinamakan *pac-dots* dalam suatu labirin tanpa tertangkap oleh 4 hantu. Hantu-hantu dalam Pac-Man memiliki perilaku tertentu. Perilaku ini dapat ditentukan dengan algoritma greedy best first search untuk hantu-hantu dapat mencari jalur ke pemain.**

Kata Kunci—Hantu, Perilaku, Greedy, Pac-Man, pac-dots

I. PENDAHULUAN

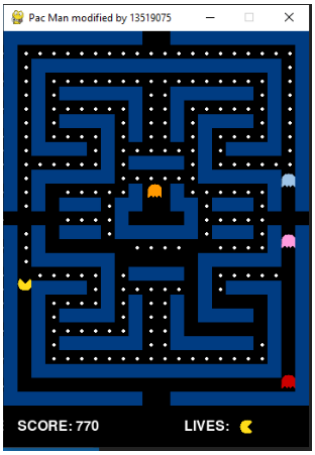
Pac-Man adalah sebuag permainan video yang dibuuat dan dikembangkan pertama kali oleh Namco di didireksi oleh Toru Iwatani dengan anggota 9 orang. Diluncurkan di Jepang pada 22 Mei 1980, hingga saat ini merupakan salah satu permainan video terpopuler sampai sekarang.

Dalam permain video Pac-Man, pemain bermain sebagai karakter kuning yang dinamakan Pac-Man. Pac-Man ditugaskan untuk menelusuri labirin dan mengumpulkan "memakan" titik-titik kecil yang dinamakan *pac-dots*. Permainan selesai dan berganti level jika semua *pac-dots* telah habis dikumpulkan. Pac-dots yang dimakan juga akan menaikkan skor.

Selama permainan Pac-Man akan diganggu oleh 4 hantu dalam labirin, yang masing-masing mempunyai nama Blinky, Pinky, Inky, dan Clyde, yang mempunyai perilakunya tersendiri untuk mengejar Pac-Man.

Pada awal permainan, Pac-Man akan mendapat 3 nyawa. Nyawa Pac-Man akan berkurang satu dan permainan akan kembali ke *state* semula tanpa *pac-dots* yang sudah termakan jika Pac-Man bersentuhan dengan salah satu dari keempat hantu. Permainan akan berakhir dengan kekalahan dari pemain jika nyawa menyentuh 0.

Hingga saat ini ada banyak sekali jenis variasi Pac-Man seperti salah satunya Pac-Man yang memiliki *item power-up* dimana jika dimakan akan menambahkan kecepatan, mengubah tingkah laku musuh dan lain-lain. Namun untuk pembahasan kali ini, Permainan Pac-Man yang dibahas adalah permainan Pac-Man sederhana tanpa *power up*



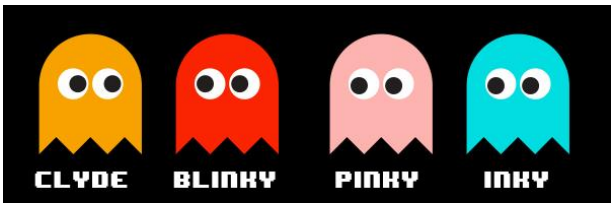
Gambar 1: Tampilan utama permainan Pac-Man
Sumber: dokumen-pribadi
https://github.com/mizuday/pac-man-best-first-search

II. PERMAINAN PAC-MAN

Permainan Pac-Man memiliki suatu pola permainan yang repetitif pada tiap level. Seiring dengan meningkatnya level, permainan akan menjadi lebih sulit dengan berkembangnya labirin dan kemampuan hantu dalam mengejar Pac-Man (kecepatan dan kepekaan bertambah). Pengembangan game Pac-Man memunculkan banyak side gameplay baru yang diimplementasikan, namun pada umumnya gameplay dan tokoh utama Pac-Man tetap sama.

Pac-Man
Pac-Man adalah tokoh utama permainan, pemain dapat menggerakan Pac-Man ke atas/kiri/kanan/bawah sesuai keingan, pemain akan berhenti jika menabrak tembok/dinding labirin. Tujuan pemain adalah memakan semua *pac-dots* untuk berganti level. Setiap pac-dots yang termakan akan meningkatkan skor pemain. Skornya ini yang nantinya digunakan untuk membandingkan dengan pemain lainnya.

Hantu
Hantu adalah antagonis dari permainan Pac-Man. Pada permainan Pac-Man umumnya, hanya terdapat 4 hantu yaitu Blinky, Pinky, Inky, Clyde



Gambar 2: Tampilan utama permainan Pac-Man
Sumber: http://nickcassway.com/designblog/?p=2443

Keempat hantu tersebut bergerak secara otomatis dengan tujuan berkolisi atau bersentuhan dengan pemain. Untuk mencari rute ke pemain hantu harus pada menemukan jalan ke pemain, maka dari itu pergerakkan hantu dapat diatur dengan suatu algoritma *path-finding* yaitu salah satunya Greedy Best First Search. Tidak hanya itu, masing-masing hantu memiliki perilakunya tersendiri seperti berikut:

- Blinky (Red)
  Mencari jalan dengan pemain menggunkan Best First Search dan bergerak langsung pada jalur itu menuju pemain. "Mengejar pemain."

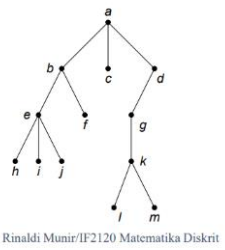
- Pinky (Pink)
  Mengecek arah pergerakan pemain dan memprediksi posisi pemain berdarkan dinding persimpangan terdekat, lalu menggunakan Best First Search ke tempat prediksi tersebut. "Memotong jalur pemain."

- Inky (Blue)
  Inky akan menggunakan Best First Search untuk melakukan perjalanan ke titik yang dibuat dengan merefleksikan posisi Blinky pada garis tegak lurus yang dibuat oleh titik prediksi Pinky dalam kaitannya dengan posisi Blinky, disesuaikan ke titik tanpa dinding. "Menjebak pemain."

- Clyde (Orange)
  Bergerak secara acak. "*Wandering Ghost.*"

III. TEORI DASAR

Dalam algoritma path-finding, proses pencarian dapat digambarkan dalam sebuah struktur pohon pencarian.

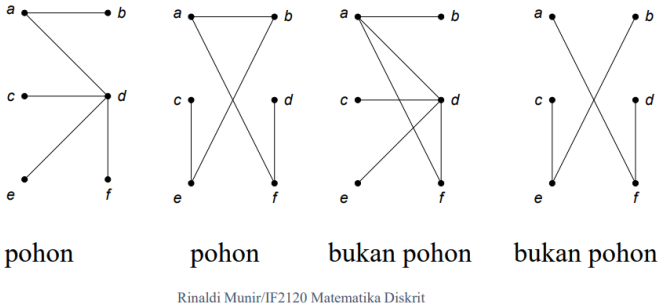


Gambar 3: Pohon
Sumber: Rinaldi Munir

Pencarian dimulai dari root menuju ke salah satu daun yang merupakan target / tujuan yang dicari. Pada path-finding, pohon pencarian dibuat seiring algoritma berjalan karena seluruh jalur belum diketahui.

A. *Pohon*

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit [1]. Suatu simpul/*node* pada pohon biasa digambarkan dengan titik atau lingkaran, simpul pada pohon bisa mengandung suatu nilai atau kondisi yang akan membawanya ke simpul lainnya. Suatu simpul biasanya dihubungkan dengan garis ke simpul lainnya.

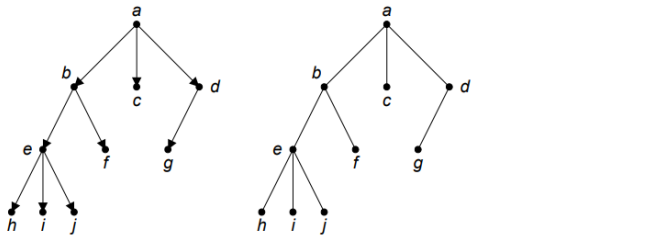


Gambar 4: Representasi Pohon
Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf

Pohon juga merupakan graf dengan sifat khusus dengan teorema, Misalkan G = (V, E) adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini adalah ekivalen:

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki m = n – 1 buah sisi.Dendro (alam)
4. G tidak mengandung sirkuit dan memiliki m = n – 1 buah sisi
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

Suatu Pohon yang satu buah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah sehingga menjadi graf berarah dinamakan pohon berakar (*rooted tree*) [2].



Gambar 5: Contoh pohon berakar
Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf

Terminologi pohon berakar:

1. Anak (*child* atau *children*) dan Orangtua (*parent*)
   Anak adalah simpul yang menjadi suksesor suatu simpul akarnya, sedangkan orangtua adalah predesesor dari anak. Sebagai contoh, simpul b, c, d adalah anak-anak simpul a,dan a adalah orangtua dari anak-anak itu.
2. Lintasan
   Lintasan adalah sisi-sisi atau garis yang menhubungkan satu simpul ke simpul tertentu. Contohnya lintasan dari a ke j adalah a, b, e, j dengan panjang lintasan
3. Saudara kandung (*siblings*)
   Saudara kandung adalah relasi antara simpul yang memiliki orangtua sama. Contohnya f adalah saudara kandung e.
4. Upapohon (*subtree*)
   Upapohon adalah pohon lebih yang kecil yang terkandung di dalam sebuah pohon. Contohnya adalah upapohon e h i j
5. Derajat
   Derajat sebuah simpul adalah jumlah anak yang dikandung pada sebuah simpul. Contohnya adalah b mempunyai anak e dan f, maka derajat simpul b adalah dua.
6. Daun (*leaf*)
   Daun adalah simpul berderajat nol atau tidak mempunyai anak. Simpul h i j f dan g adalah contoh daun.
7. Simpul dalam (*internal nodes*)
   Simpul yang mempunyai anak dan orang tua disebut simpul dalam. Contohnya adalah simpul b e dan d.
8. Aras (*level*) atau Tingkat
   Aras atau tingkat adalah jumlah predesesor yang dimiliki suatu simpul hingga simpul akar. Contohnya simpul g memiliki tingkat 2.
9. Tinggi (*height*) atau Kedalaman (*depth*)
   Aras maksimum dari suatu pohon disebut tinggi ataukedalaman. Contohnya pohon pada gambar 3 (b) mempunyai tinggi 3.

Pohon berakar yang setiap simpul cabangnya mempunyai paling banyak n buah anak disebut pohon n-ary. Salah satu contoh pohon n-ary adalah pohon biner yaitu dengan n = 2.

B. Greedy Best First Search

Greedy Best First Search atau juga sering disebut Best First Search adalah algoritma yang menggunakan fungsi evaluasi *f(n)* untuk setiap node dimana

f(n) = *h(n)* -> dimana *h(n)* adalah fungsi estimasi (heuristik) cost dari n ke tujuan dalam kasus ini *h(n)* bisa menjadi nilai

a. Euclidian distance dari hantu ke pemain

b. Manhattan distance dari hantu ke pemain

Eucludian distance menghasilkan nilai jarak dari titik awal sampai tujuan dengan fungsi pitagoras.

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \sqrt{(q_1-p_1)^2+(q_2-p_2)^2}.$$

Manhattan distance mendapatkan hasil jaraknya hanya dengan menjumlahkan perbedaan absis dan ordinat antara titik awal dan tujuan.

$$d_1(\mathbf{p},\mathbf{q}) = |p_1-q_1| + |p_2-q_2|$$

Fungsi heuristik yang sesuai adalah Manhattan distance, karena lebih mendekati nilai biaya sesungguhnya (yaitu untuk mencapai lokasi tujuan, hantu-hantu harus mengikuti belokanbelokan dan tidak bisa pergi menembus tembok). Fungsi Manhattan distance juga tidak akan pernah overestimate dan maka dari itu adalah admissible.

Best First Search mengekspansi node yang tampak paling dekat ke tujuan. Berikut adalah pseudocode Best First Search yang diambil dari https://www.geeksforgeeks.org/best-first-search-informed-search/

```
Best-First-Search(Grah g, Node start)
    1) Create an empty PriorityQueue
       PriorityQueue pq;
    2) Insert "start" in pq.
       pq.insert(start)
    3) Until PriorityQueue is empty
          u = PriorityQueue.DeleteMin
          If u is the goal
             Exit
          Else
             Foreach neighbor v of u
                If v "Unvisited"
                    Mark v "Visited"
                    pq.insert(v)
             Mark u "Examined"
End procedure
```

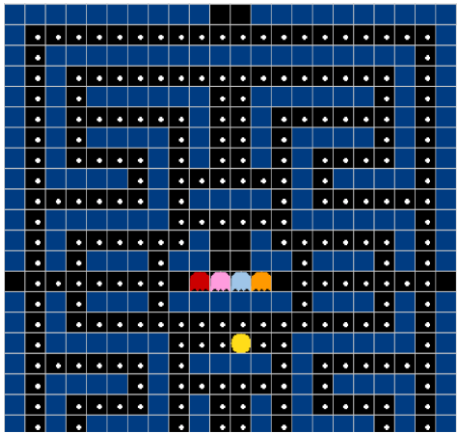
IV. Implementasi Greedy Best First Search Pada Pergerakkan Pac-Man

Banyak algoritma yang dapat digunakan dalam mencari jalur dari hantu ke pemain. Walaupun tidak selalu menghasilkan jalur yang efektif algoritma ini menghasilkan jalur yang cukup optimal dengan iterasi paling sedikit sehingga menghemat banyak waktu komputasi. Selain itu, untuk meringankan komputasi, pencarian jalur setiap hantu dilakukan tidak setiap pemain melakukan perubahan pergerakkan

A. Pembentukan Pohon Pencarian pada peta Pac-Man

Dalam suatu algoritma *path-finding.* Proses pencarian dapat digambarkan dengan sebuah struktur pohon pencarian.

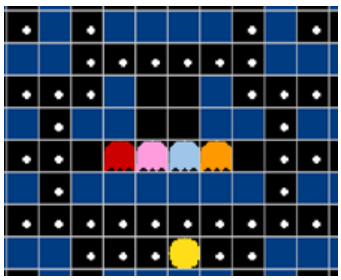
Pencarian dimulai dari root menuju ke salah satu daun yang merupakan target / tujuan yang dicari. Pada path-finding, pohon pencarian dibuat seiring algoritma berjalan karena seluruh jalur belum diketahui.



Gambar 6: Visualiasi grid pac-man
Sumber: Dokumen Pribadi

Pada permainan Pac-Man, suatu simpul bisa digambarkan dengan grid yang terlihat di gambar 6 dan simpul-simpul tetangganya adalah adalah grid yang beredekatan dengan grid tersebut (atas, bawah, kiri dan kanan). Pada kasus path-finding kali ini, simpul root pada adalah lokasi/grid masing hantu berada dan simpul tujuan adalah pada lokasi/grid Pac-Man berada.

Untuk contoh pembangkitan simpul best first search pada pac man, perhatikan contoh di bawah ini yang mengambil hantu merah sebagai patokan.



Gambar 7: Pencarian oleh hantu merah
Sumber: Dokumen Pribadi

Jika titik (0,0) adalah grid pojok kiri bawah pada gambar 9, dan posisi pemain pada saat diam adalah (5,0), posisi Blinky (merah) adalah (3,3). Blinky mempunyai dua pilihan bergerak yaitu ke kiri atau kenan. Berdasarkan fungsi heuristiknya node di sebelah kanan mempunyai nilai heuristic lebih kecil (5 dibandingkan dengan 7) maka seharusnya dipilih yang 5 namun karena ditempati maka node disebelah kiri lah yang di pilih, lalu dibangkitkan tetantangganya, karena hanya satu tetangga yaitu node sebelah kiri (node di sebelah kanan sudah visited), maka terdapat 3 node yang dibangkitkan yaitu kiri, atas, dan bawah yang masing masing heuristiknya adalah 9, 9 dan 7 maka node yang di sebelah bawah yang di pilih) lalu dibangkitkan 1 node tetangga (bawah) lalu dibangkitkan dua node tetangga yaitu kiri dan kanan yang masing-masing heuristiknya adalah 7 dan 5, dipilih yang paling kecil yaitu kanan, lalu dibangkitkan 2 node tetangga yaitu kanan dan bawah (4 dan 4), karena masing memiliki heuristic yang sama maka dibuatlah prioritas yaitu kiri kanan, atas, dan bawah. Maka dipilihlah node kanan karena memiliki prioritas lebih tinggi dari bawah. Begitu terus seterusnya hingga merah berada di posisi (5,1) yang tegak lurus dengan node tujuan. Dibangkitkan dua node yaitu bawah dan kanan, yang masing masing heuristiknya 0 dan 1, maka dipilih node bawah karena heuristic lebih kecil, lalu pencarian diberhentikan karena sudah sampai di tujuan.

B. Implementasi

Pada awal pembuatan, terdapat sedikit permasalahan jika menggunakan Greedy Best First Search dikarenakan metode pencarian ini tidak complete. Permasalahannya adalah jika ada jalan buntu maka pencarian di hentikan (tidak ada node tetangga lagi).

Gambar 8: Pencarian dihentikan saat mencapai jalan buntu
Sumber: Dokumen Pribadi

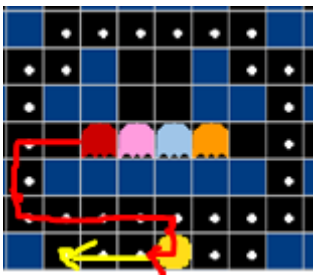
Permasalahan ini di solve dengan menghilangkan tembok yang menjadi penyebab jalan buntu tersebut seperti dilihat pada gambar 8. Greedy Best First Search pada permainan diimplementasikan dengan pseudocode di bawah:

```
Path-Best-First(Grah g, Node start, Node tujuan)
    1) Buat empty list untuk menyimpan jalur
       ArrayList list;
    2) Create an empty PriorityQueue
       {Berdasarkan fungsi heuristic manhattan}
       PriorityQueue pq;
    3) Insert "start" in pq.
       pq.insert(start)
    4) Until pq is empty
          u = pq.DequeuMin
          If u is the goal
             Exit
          Else
             Foreach neighbor v of u
                If v "Unvisited" && v !Occupied(ghost)
                    Mark v "Visited"
                    list.insert(v)
                    pq.insert(v)
             Mark u "Examined"
    return list;
End function
```

Masing-masing hantu memiliki *behaviour* yang berbeda maka dari itu dibuat pula 3 implementasi yang berbeda sesuai dengan perilaku hantu yaitu:

1. Mengejar Pemain (Blinky/merah)

Pada awal permainan, pemain dan hantu diam di tempat maka dari itu pencarian belum di lakukan. Pada saat pemain mulai melakukan pergerakan (memilih arah bergerak), Hantu akan melakukan pencarian best first search ke grid (simpul) dimana pemain melakukan gerakan lalu hantu akan terus bergerak lurus sesuai arah pemain melakukan gerakan.

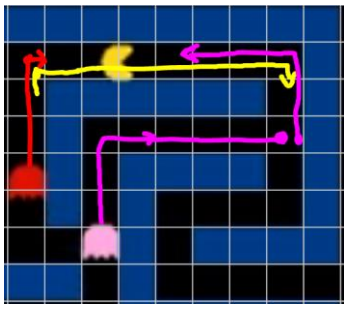


Gambar 9: Pencarian oleh hantu merah
Sumber: Dokumen Pribadi

Jika titik (0,0) adalah grid pojok kiri bawah pada gambar 9, dan posisi pemain pada saat diam adalah (4,0), posisi blinky (merah) adalah (2,3) maka pada saat kuning bergerak ke kiri. Merah akan melakukan pencarian greedy best first search ke posisi pemain melakukan gerakan(*move*) yaitu ke (4,0) lalu melakukan gerakan ke kiri (sesuai dengan gerakan pemain). Jika pemain melakukan perubahan arah pergerakkan, maka hantu akan melakukan kalkulasi ulang jalur untuk dilaluinya

2. Memotong Pemain (Pinky/*pink*)

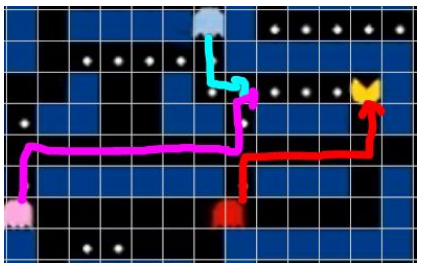
Pada awal permainan, pemain dan hantu diam di tempat maka dari itu pencarian belum di lakukan. Pada saat pemain mulai melakukan pergerakan (memilih arah bergerak), hantu akan melakukan pencarian best first search ke grid (simpul) prediksi yaitu simpul simpangan pertama terdekat dari pemain lalu melakukan best first search ke lokasi hantu blinky untuk mengapit pemain. Jika pemain melakukan perpindahan arah pergerakan (berbelok) atau jika Pinky sudah berpapasan dengan Blinky maka hantu akan melakukan pencarian ulang lagi seperti diatas



Gambar 10: Pencarian oleh hantu pink
Sumber: Dokumen Pribadi

3. Menjebak Pemain (Inky/biru)

Pada awal permainan, pemain dan hantu diam di tempat maka dari itu pencarian belum di lakukan. Pada saat pemain mulai melakukan pergerakan (memilih arah bergerak), hantu akan melakukan pencarian best first search ke grid (simpul) refleksi dari tujuan merah lalu melakukan best first search ke simpul prediksi sesuai yang dilakukan oleh pinky best first search ke lokasi hantu blinky untuk mengapit pemain. Karena hal inilah sering kali kita melihat biru dan merah berada di sisi bersebelahan (lihat gambar 10, biru sisi atas dan merah dari sisi bawah). Jika pemain melakukan perpindahan arah pergerakan (berbelok) maka hantu akan melakukan pencarian ulang lagi seperti diatas



Gambar 11: Pencarian oleh hantu biru
Sumber: Dokumen Pribadi

Ada beberapa kekurangan dari algoritma penjebakan yang dilakukan oleh inky, yaitu salah satunya adalah ketika melakukan perubahan arah maka biru akan melakukan reposisi ulang yang dan hal ini mengakibatkan algoritma penjebakan kurang efektif jika memiliki banyak lekukan dan persimpangan, namun sangat efektif di lorong panjang (pinggiran)

V. KESIMPULAN

Hantu dalam permainan Pac-Man dapat diimplementasikan dengan berbagai algoritma salah satunya adalah Greeedy Best First Search, algoritma ini tidak menjamin jalur optimal namun menjamin kecepatan komputasi. Semakin luasnya area permainan maka maka algortima seperti breadth first search menjadi kurang efektif karena hampir mengecek semua kemungkinan. Level (area) design Pac-Man yang sangat cocok dengan algoritma best first search adalah area yang tidak memiliki jalan buntu dan area open field. Sementara jika level/area yang memilki design seperti maze dan labirin dimana memiliki jalan buntu tidak disarankan menggunakan best first search karena greedy best first search adalah pencarian yang tidak *complete.*

LINK GITHUB PERMAINAN

https://github.com/mizuday/pac-man-best-first-search

REFERENCES

[1] Rinaldi Munir. 2020. Pohon (Bag.1). Diakses pada 9 Mei 2021 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf pukul 09.16

[2] Rinaldi Munir. 2020. Pohon (Bag.2). Diakses pada 9 Mei 2021 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf pukul 16.20

[3] Rinaldi Munir. 2021. Penentuan Rute (Bag.1). Diakses pada 9 Mei 2021 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian1-2021.pdf pukul 19.20

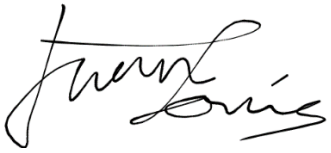
[4] Rinaldi Munir. 2020. Graf (Bag.1). Diakses pada 9 Mei 2021 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf pukul 08.00

[5] https://www.geeksforgeeks.org/best-first-search-informed-search/

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.

Tangerang Selatan, 11 Mei 2021



Juan Louis Rombetasik 13519075