Aplikasi Algoritma Greedy untuk Penentuan Pengumpulan Kartu G & G dalam Permainan Bully

Jason Rivalino - 13521008
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13521008@std.stei.itb.ac.id

***Abstrak*—Makalah ini akan membahas mengenai aplikasi penerapan Algoritma Greedy yang ada dalam mata kuliah Strategi Algoritma. Aplikasi yang ada terkait dengan *side mission* yang ada dalam permainan *open-world* Bully yaitu untuk pengumpulan seluruh kartu G & G. Adapun penerapan Algoritma Greedy yang diterapkan dalam *side mission* ini adalah untuk mencari optimasi minimasi dalam pencarian kartu G & G secara keseluruhan dalam peta *game*.**

Kata Kunci—Algoritma Greedy, TSP, optimasi, minimasi, Kartu G&G, Bully

I. PENDAHULUAN

Bermain *game* sendiri merupakan salah satu hal yang sering dilakukan oleh banyak orang pada zaman sekarang ini. Bermain *game* seringkali dilakukan sebagai *refreshing* untuk menghilangkan kejenuhan dan stress yang disebabkan oleh aktivitas yang dijalani sehari-hari. Selain itu, di dalam *game* sendiri terdapat beberapa tantangan tertentu yang membuat pemainnya menjadi tertantang untuk menyelesaikannya sehingga *game* memiliki manfaat juga untuk mengasah otak dan meingkatkan kemampuan *problem-solving*.

Dengan semakin berkembangnya zaman, terdapat semakin benyak jenis variasi *game* yang beredar di pasaran. Variasi-variasi *game* yang ada juga memiliki jenis tipe *genre* dan keunikannya sendiri yang membedakan antara satu *game* dengan *game* lainnya. Salah satu jenis *genre game* yang paling terkenal dan banyak peminatnya sekarang ini adalah *game* bergenre *open-world*.

Game *open-world* sendiri[1] adalah game dengan dunia virtual dengan pemainnya dapat bebas menjelajahi tempat-tempat yang ada dalam *game* dan bebas untuk bereksplorasi serta melakukan aktivitas apapun. Meskipun pemain dapat mengeksplorasi tempat permainan secara bebas, namun masih terdapat batasan tertentu dalam gamenya seperti fitur kondisi geografis yang tidak dapat dilewati. Dalam *game open-world* sendiri biasanya terdapat misi tertentu yang harus diselesaikan oleh pemainnya untuk menamatkan *game*, namun jika pemain merasa malas menyelesaikan misi, pemain dapat berpetualang, jalan-jalan, dan mengacau di dalam game.

Salah satu pengembang *game* dengan *genre open-world* yang paling terkenal di seluruh dunia adalah Rockstar Games yang sudah mengembangan seri *game open-world* terbesar di dunia yaitu Grand Theft Auto Series yang merupakan *game* dengan pemainnya dapat menjalankan misi sekaligus berpetualang dan membuat kekacauan kriminal di dalam *game* yang tentunya tidak dapat dilakukan di dunia nyata. Namun, selain mengembangkan Grand Theft Auto Series, Rockstar Games juga mengembangkan beberapa *game open-world* lainnya, seperti salah satunya adalah Bully yang akan dibahas dalam makalah kali ini.

Bully sendiri merupakan *game open-world* dengan karakter utamanya merupakan seorang murid SMA. Dalam permainan ini, pemain dapat melakukan berbagai macam aktivitas di dalam game seperti menjalankan misi utama, berpetualang membuat kerusuhan di sekolah ataupun kota, dan juga menyelesaikan berbagai misi tambahan untuk menamatkan *game* ini secara keseluruhan hingga 100%. Salah satu misi tambahan yang ada dalam game ini adalah misi *collectibles* dengan pemain harus mengumpulkan objek tertentu yang tersebar di seluruh peta permainan untuk mendapatkan *reward* atau *achievement* tertentu. Salah satu misi *collectibles* yang ada dalam *game* Bully ini sendiri adalah misi untuk mengumpulkan semua Kartu G & G yang tersebar di seluruh peta permainan yang bisa dikumpulkan dengan menggunakan Algoritma Greedy.

Pada makalah ini, penulis mencoba untuk membahas mengenai penerapan Algoritma Greedy untuk penentuan pengumpulan Kartu G & G yang ada dalam permainan Bully.

II. LANDASAN TEORI

A. Algoritma Greedy

Algoritma Greedy adalah algoritma yang memecahkan persoalan secara langkah per langkah. Algoritma ini merupakan algoritma yang paling populer untuk memecahkan persoalan pencarian solusi optimal yaitu maksimasi dan minimasi. Sesuai namanya yaitu *greedy*, yang berarti rakus, tamak, atau serakah, algoritma ini memiliki prinsip “take what you can get now”.

Dalam Algoritma Greedy, setiap langkah yang ada akan mengambil pilihan yang terbaik tanpa memperhatikan konsekuensi kedepannya (memilih optimum lokal) dengan harapan langkah sisanya akan berakhir menuju optimum global.

Namun, optimum global yang dihasilkan dari Algoritma Greedy tidak selalu merupakan solusi optimum terbaik (bisa merupakan solusi sub-optimum atau pseudo-optimum). Hal ini disebabkan karena Algoritma Greedy tidak beroperasi secara menyeluruh terhadap semua kemungkinan solusi dan terdapat beberapa fungsi seleksi yang berbeda. Namun, jika solusi terbaik mutlak tidak terlalu diperlukan, Algoritma Greedy dapat digunakan untuk menghasilkan hampiran solusi optimal.

Beberapa elemen yang terdapat pada Algoritma Greedy antara lain:

1. Himpunan kandidat (C): himpunan ini berisi kandidat elemen pembentuk solusi yang akan dipilih pada setiap langkah
2. Himpunan solusi (S): himpunan ini berisi kandidat yang sudah terpilih sebagai solusi persoalan.
3. Fungsi solusi: fungsi yang menentukan apakah kandidat yang dipilih sudah memberikan solusi. Fungsi ini akan mengembalikan nilai boolean yaitu true jika memberikan solusi lengkap dan false jika memberikan solusi belum lengkap.
4. Fungsi seleksi (selection function): fungsi yang memilih kandidat yang paling mungkin untuk mencapai solusi berdasarkan strategi greedy tertentu yang bersifat heuristik.
5. Fungsi kelayakan: fungsi yang memeriksa kelayakan kandidat yang terpilih apakah dapat dimasukkan ke dalam himpunan solusi atau tidak.
6. Fungsi objektif: fungsi yang mengoptimalkan solusi (memaksimumkan atau meminimumkan).

Untuk pseudocode yang ada dalam Algoritma Greedy sendiri adalah sebagai berikut:

```
function greedy(C : himpunan_kandidat) → himpunan_solusi
{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy }
Deklarasi
  x : kandidat
  S : himpunan_solusi

Algoritma:
  S ← {}      { inisialisasi S dengan kosong }
  while (not SOLUSI(S)) and (C ≠ {} ) do
      x ← SELEKSI(C)      { pilih sebuah kandidat dari C }
      C ← C – {x}         { buang x dari C karena sudah dipilih }
      if LAYAK(S ∪ {x}) then      { x memenuhi kelayakan untuk dimasukkan ke dalam himpunan solusi }
          S ← S ∪ {x}      { masukkan x ke dalam himpunan solusi }
      endif
  endwhile
  {SOLUSI(S) or C = {} }

  if SOLUSI(S) then     { solusi sudah lengkap }
    return S
  else
    write('tidak ada solusi')
  endif
```

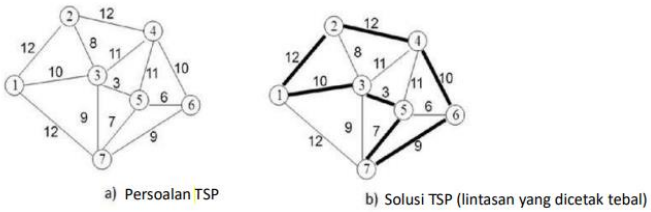
Gambar 2.1 Pseudocode Algoritma Greedy

(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf)

Terdapat beberapa contoh persoalan yang dapat diselesaikan dengan menggunakan Algoritma Greedy, antara lain persoalan penukaran uang, persoalan memilih aktivitas, minimasi waktu dalam sistem, persoalan knapsack dan TSP, penjadwalan job dengan tenggat waktu, pohon merentang minimum, lintasan terpendek, kode Huffman, dan pecahan Mesir[2].

B. *Travelling Salesman Problem* (TSP)

Travelling Salesman Problem atau TSP merupakan salah satu contoh persoalan yang berkaitan dengan graf. Persoalan ini sendiri memiliki deskripsi sebagai berikut "Diberikan n buah kota serta diketahui jarak antara setiap kota satu sama lain. Temukan perjalanan (tour) dengan jarak terpendek yang dilakukan oleh seorang pedagang sehingga ia melalui setiap kota tepat hanya sekali dan kembali lagi ke kota asal keberangkatan"[3]. Inti dari persoalan TSP sendiri adalah untuk menemukan sirkuit Hamilton yang memiliki bobot minimum. TSP sendiri masuk ke dalam persoalan dengan tipe NP-hard.



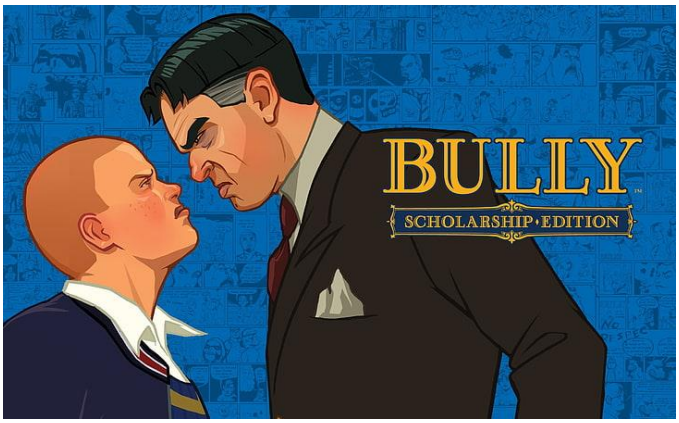
Gambar 2.2 Contoh persoalan TSP

(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Greedy-(2022)-Bag3.pdf)

Untuk penyelesaian dari persoalan TSP sendiri, persoalan ini sebenarnya bisa diselesaikan dengan menggunakan berbagai macam algoritma dan sering dijadikan contoh persoalan untuk menguji kecepatan dari sebuah algoritma. Untuk pencarian TSP dengan Algoritma Greedy sendiri memiliki kompleksitas yaitu sebesar $O(n^2 \log n)$[4].

C. Bully

Bully merupakan sebuah *video game* dengan *genre open-world* yang dikembangkan oleh Rockstar Games dan dirilis pada tanggal 17 Oktober 2006[5]. *Game* ini tersedia dalam berbagai macam *platform* seperti Playstation 2, Wii, Xbox 360, Windows PC, Android, dan iOS.



Gambar 2.3 Artwork Game Bully

(Sumber: https://www.wallpaperflare.com/bully-scholarship-edition-bully-scholarship-edition-application-wallpaper-pljtd)

Dalam *game* ini, pemain akan berpesan sebagai seorang anak SMA bernama Jimmy Hopkins yang baru saja pindah sekolah menuju Bullworth Academy setelah sebelumnya

mengalami *drop-out* dari berbagai macam sekolah lamanya akibat tindakan-tindakannya yang melanggar aturan. Latar tempat yang ada dalam *game* ini sendiri berada di sebuah kota fiktif bernama Bullworth.

Di dalam *game* ini, pemain dapat melakukan berbagai macam aktivitas seperti menjalankan cerita misi utama, mengikuti kelas akademik, berinteraksi dengan berbagai macam *circle* yang ada di sekolah, bekerja untuk mendapatkan uang tambahan, melakukan balapan sepeda, berpetualang mengelilingi kota, menolong atau membuat kekacauan di sekolah dan kota, dan berbagai macam aktivitas lainnya.

Untuk menamatkan game ini secara keseluruhan hingga mencapai 100%, pemain harus menamatkan seluruh cerita misi utama dan juga menamatkan misi-misi tambahan yang ada di dalam *game*. Salah satu misi tambahan di dalam *game* adalah misi *collectibles* dengan pemain harus mengumpulkan objek tertentu yang tersebar di seluruh peta permainan untuk mendapatkan *reward* atau *achievement*. Salah satu *collectibles item* yang ada dalam *game* ini adalah kartu G & G yang lokasinya tersebar di seluruh peta Bullworth.

D. *Side Mission G & G Cards*

Side Mission Grottos and Gremlins (G & G) Cards merupakan misi tambahan yang ada di dalam game Bully. Pada misi ini, pemain harus mengumpulkan seluruh kartu G & G yang tersebar di seluruh peta Bullworth. Total seluruh kartu yang harus dikumpulkan berjumlah 40 buah[6]. Misi ini perlu dilakukan oleh pemain jika ingin mencapai penamatan game 100%. Setelah mengumpulkan semua kartu yang ada, pemain akan mendapatkan hadiah yaitu berupa kostum Grotto Master yang bisa dipakai oleh pemain dalam permainan.

Gambar 2.4 Kartu G & G

(Sumber: https://www.youtube.com/watch?v=6Abu_XpauGY&t=134s)

Gambar 2.5 Kostum Grotto Master

(Sumber: https://www.youtube.com/watch?v=6Abu_XpauGY&t=134s)

III. PENCARIAN SOLUSI DENGAN ALGORITMA GREEDY

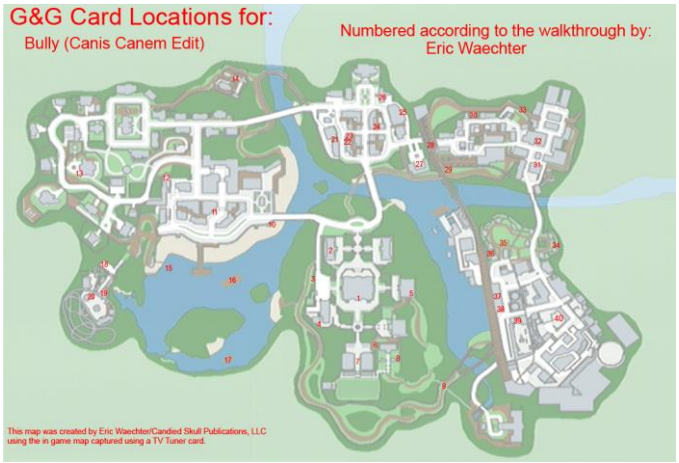
Untuk pencarian solusi dengan Algoritma Greedy, hal pertama yang harus dilakukan adalah dengan melakukan pemetaan terhadap semua lokasi G & G Cards yang ada pada peta kota Bullworth. Selain itu, perlu dilakukan juga penentuan lokasi awal untuk pencarian kartu. Dalam contoh makalah ini, pencarian dimulai dari posisi titik yang terdekat dengan sekolah dan tempat tinggal Jimmy Hopkins.

Setelah dilakukan pemetaan, berikutnya adalah mencari setiap titik koordinat pada setiap lokasi kartu yang ada. Pencarian koordinat dilakukan dengan menggunakan bantuan dari *website* https://mapgenie.io/bully/maps/bullworth yang menyediakan aplikasi peta interaktif untuk *game* ini. Titik-titik koordinat ini nantinya akan diperlukan untuk proses perhitungan untuk pencarian rute dengan Algoritma Greedy.

Hasil pemetaan yang ada juga akan dibagi ke dalam empat bagian berdasarkan kondisi wilayah geografis pada peta untuk memudahkan pencarian solusi berdasarkan kondisi TSP. Pembagian pemetaannya sendiri adalah sebagai berikut:

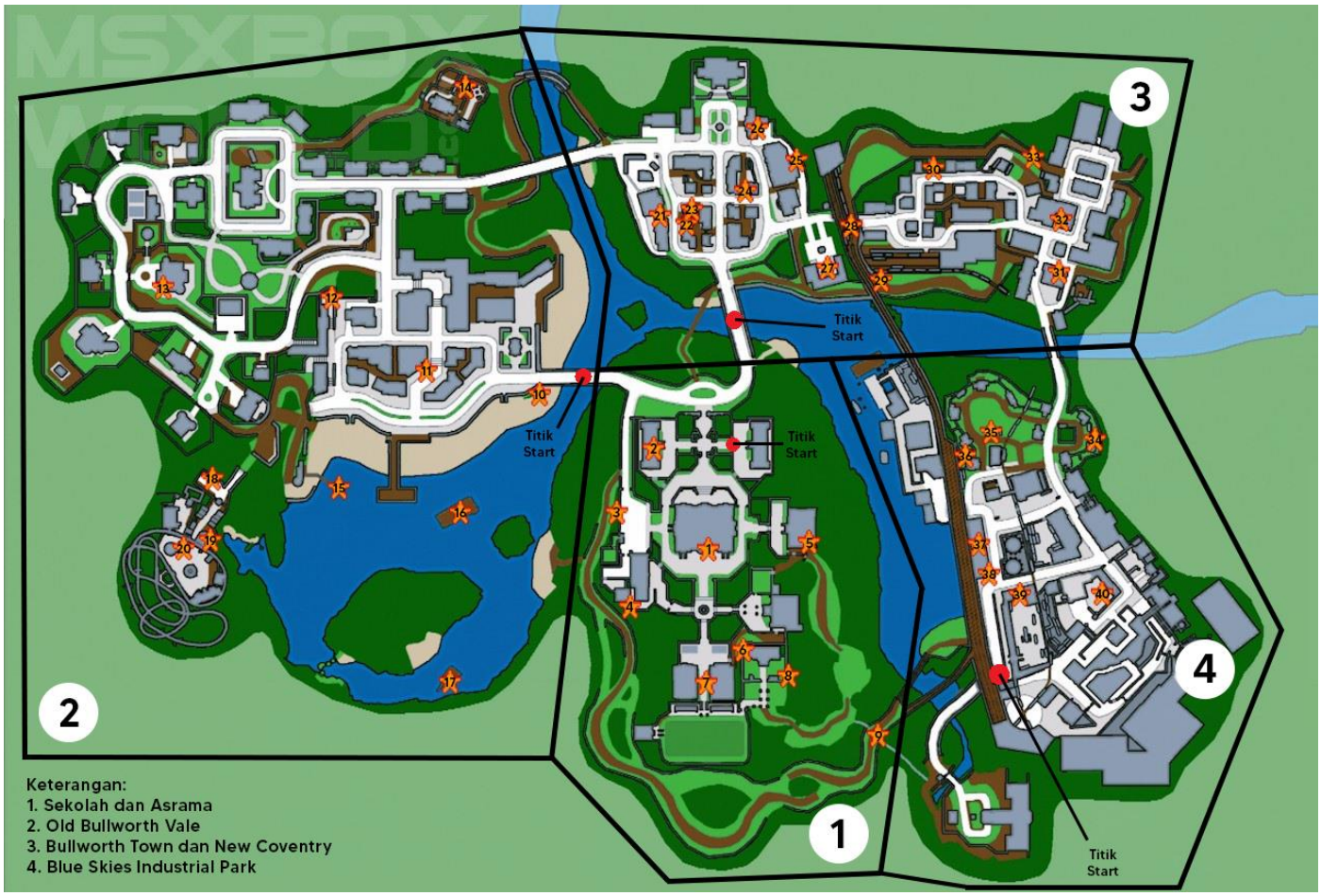
1. Pencarian kartu pada wilayah asrama dan sekolah: pencarian diawali dari bagian gerbang depan asrama laki-laki
2. Pencarian kartu pada wilayah Old Bullworth Vale: pencarian diawali dari posisi jembatan di sebelah kiri sekolah
3. Pencarian kartu pada wilayah Bullworth Town dan New Coventry: pencarian diawali dari posisi jembatan di sebelah kiri sekolah
4. Pencarian kartu pada wilayah Blue Skies Industrial Park: pencarian diawali dari kolong rel kereta

Untuk pemetaan dari keseluruhan kartu yang ada dan juga lokasi dari titik-titik koordinat pada kartu adalah sebagai berikut:



Gambar 3.1 Urutan Penomoran lokasi Kartu G & G pada peta

(Sumber: https://hilmykhoiri.wordpress.com/2013/07/05/peta-tempat-g-g-gnome-rubber-band-pada-game-bully-ps2/)



Gambar 3.2 Informasi lengkap pemetaan

(Sumber: http://yunuzzone.blogspot.com/2012/02/grottos-gremlins-card.html)

No.	Info Tempat	Koordinat X	Koordinat Y
1.	Start Location (Front Boy Dorms)	x=-0.6823566285177947	y=0.6931773185769998
2.	G & G Cards 1	x=-0.6908306068896195	y=0.6560155978259274
3.	G & G Cards 2	x=-0.7102114982045293	y=0.6902874963907095
4.	G & G Cards 3	x=-0.7224923632054185	y=0.6691401304319555
5.	G & G Cards 4	x=-0.7190979803525010	y=0.6374005055338330
6.	G & G Cards 5	x=-0.6546365231392883	y=0.6602412600027776
7.	G & G Cards 6	x=-0.6813123348844101	y=0.6243960869202425
8.	G & G Cards 7	x=-0.6922242824752516	y=0.6102020773937937
9.	G & G Cards 8	x=-0.6647455880106747	y=0.6119472177570486
10.	G & G Cards 9	x=-0.6314288212711290	y=0.5906712389110140

No.	Info Tempat	Koordinat X	Koordinat Y
1.	Start Location (Jembatan 1)	x=-0.7339770057955377	y=0.7171200748019544
2.	G & G Cards 10	x=-0.7515039527407907	y=0.7113837913199461
3.	G & G Cards 11	x=-0.7881380380868279	y=0.7213568759748625
4.	G & G Cards 12	x=-0.8245620953786954	y=0.7436009994076045
5.	G & G Cards 13	x=-0.8832699860243451	y=0.7492174281678672
6.	G & G Cards 14	x=-0.7718367662567971	y=0.8130983761912773
7.	G & G Cards 15	x=-0.8213324752495055	y=0.6789928549708435
8.	G & G Cards 16	x=-0.7754081955933145	y=0.6704141049165315
9.	G & G Cards 17	x=-0.7802472155254634	y=0.6105844395197124
10.	G & G Cards 18	x=-0.8683130760697679	y=0.6803168897974672
11.	G & G Cards 19	x=-0.8671585091240956	y=0.6592062510491274
12.	G & G Cards 20	x=-0.8777970188348831	y=0.6567323421563316

No.	Info Tempat	Koordinat X	Koordinat Y
1.	Start Location (Jembatan 2)	x=-0.6817731649855716	y=0.7391365227204432
2.	G & G Cards 21	x=-0.7072935342821154	y=0.7737503467854907
3.	G & G Cards 22	x=-0.6976769913303258	y=0.7734338711894395
4.	G & G Cards 23	x=-0.6976769913303258	y=0.7762660249021138
5.	G & G Cards 24	x=-0.6779875513944660	y=0.7825735561744409
6.	G & G Cards 25	x=-0.6613793131754164	y=0.7936875787733442
7.	G & G Cards 26	x=-0.6754555460852316	y=0.8029563933302057
8.	G & G Cards 27	x=-0.6463638168978321	y=0.7539908900649692
9.	G & G Cards 28	x=-0.6416860443756320	y=0.7626590303292033
10.	G & G Cards 29	x=-0.6324592453638331	y=0.7522315129475174
11.	G & G Cards 30	x=-0.6124686178808645	y=0.7926583543359982
12.	G & G Cards 31	x=-0.5668703747570589	y=0.7545794704131197
13.	G & G Cards 32	x=-0.5667660638855239	y=0.7724180054999863
14.	G & G Cards 33	x=-0.5832626161595726	y=0.7992810836853295

No.	Info Tempat	Koordinat X	Koordinat Y
1.	Start Location (under railway)	x=-0.5884509331891934	y=0.6136238695016516
2.	G & G Cards 34	x=-0.5555639706745410	y=0.6951375334578103
3.	G & G Cards 35	x=-0.5890773097942485	y=0.6966137831224444
4.	G & G Cards 36	x=-0.6004452706409040	y=0.6868705268337578
5.	G & G Cards 37	x=-0.5977878252479059	y=0.6573453876431046
6.	G & G Cards 38	x=-0.5954256515628629	y=0.6483777724118625
7.	G & G Cards 39	x=-0.5786061730781284	y=0.6409168092642545
8.	G & G Cards 40	x=-0.5640090603683348	y=0.6406863429521081

Tabel 3.1 Lokasi Koordinat dari setiap Kartu G & G dan juga Titik Awal Pencarian

(Sumber: Dokumen Penulis dengan data yang diambil dari https://mapgenie.io/bully/maps/bullworth)

Setelah berbagai macam informasi ini didapatkan, proses pencarian kartu dengan menggunakan Algoritma Greedy dapat dilakukan dengan tahapan sebagai berikut:

1. Menghitung jarak antara titik awal pertama dengan setiap titik lokasi kartu yang ada. Perhitungan dilakukan dengan menggunakan rumus Euclidean Distance yaitu sebagai berikut:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

2. Menentukan pemilihan lokasi kartu pertama berdasarkan nilai jarak terendah yang diperoleh (memilih dengan optimasi minimasi). Pemilihan lokasi kartu pertama ini akan menjadi solusi lokal awal dengan harapan bahwa solusi lokal ini akan mengarah kepada solusi global yaitu bisa mendapatkan semua kartu dengan jarak tempuh yang paling minimum.
3. Melakukan perhitungan kembali pada titik lokasi kartu yang belum dikunjungi dan memilih lokasi berdasarkan nilai jarak terendah hingga semua kartu telah didapatkan. (TSP tanpa kembali ke titik awal).
4. Opsional: Setelah semua kartu didapatkan, Jimmy Hopkins ingin kembali pergi ke tempat awalnya sehingga perlu dilakukan perhitungan jarak juga antara lokasi kartu yang terakhir dikunjungi dengan lokasi titik awal pertama (TSP dengan kembali ke titik awal).

Elemen-elemen Algoritma Greedy yang diterapkan dalam permasalahan ini adalah sebagai berikut:

1. Himpunan kandidat (C): semua lokasi kartu G & G yang ada dalam satu kondisi wilayah yang sudah dibatasi.
2. Himpunan solusi (S): kemungkinan solusi kartu G & G terdekat yang ingin diambil.
3. Fungsi solusi: memeriksa jarak antara satu titik dengan titik lainnya apakah jarak yang ada memiliki nilai paling kecil.
4. Fungsi seleksi: jarak terdekat dari himpunan lokasi kartu G & G yang tersisa.
7. Fungsi kelayakan: memeriksa apakah kartu G & G masih berada dalam satu wilayah.
8. Fungsi objektif: jarak terdekat yang bisa diperoleh untuk mengambil semua kartu G & G dan kembali ke tempat semula.

Untuk penjelasan detail penyelesaian solusi dengan Algoritma Greedy, pada makalah ini hanya akan dijelaskan pada pencarian di daerah Old Bullworth Vale. Untuk penjelasan pencarian pada keseluruhan peta akan dijelaskan pada video dengan lampiran di akhir laporan. Adapun, langkah-langkah penyelesaian persoalannya secara keseluruhan adalah sebagai berikut:

1. Perhitungan jarak titik awal dengan seluruh lokasi kartu dengan Euclidean Distance

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari titik awal (jembatan kiri sekolah)

	G&G 10	G&G 11	G&G 12	G&G 13	G&G 14	G&G 15
Jarak	0.01844	0.05433	0.09438	0.15270	0.10318	0.09531
	G&G 16	G&G 17	G&G 18	G&G 19	G&G 20	
Jarak	0.06243	0.11615	0.13929	0.14523	0.15598	

   Jarak minimum: 0.01844
   Titik terpilih: G & G 10
   Rute pengambilan kartu: Start – 10
   Jarak yang ditempuh: 0.01844

2. Perhitungan jarak Euclidean titik terakhir (kartu G & G 10) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 10

	G&G 11	G&G 12	G&G 13	G&G 14	G&G 15
Jarak	0.03797	0.07985	0.13709	0.10373	0.07696
	G&G 16	G&G 17	G&G 18	G&G 19	G&G 20
Jarak	0.04743	0.10482	0.12087	0.12688	0.13761

   Jarak minimum: 0.03797
   Titik terpilih: G & G 11
   Rute pengambilan kartu: Start – 10 – 11
   Jarak yang ditempuh: 0.05641

3. Perhitungan jarak Euclidean titik terakhir (kartu G & G 11) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 11

	G&G 12	G&G 13	G&G 14	G&G 15	G&G 16
Jarak	0.04268	0.09913	0.09318	0.05382	0.05251
	G&G 17	G&G 18	G&G 19	G&G 20	
Jarak	0.11105	0.09007	0.10053	0.11052	

   Jarak minimum: 0.04268
   Titik terpilih: G & G 12
   Rute pengambilan kartu: Start – 10 – 11 – 12
   Jarak yang ditempuh: 0.09909

4. Perhitungan jarak Euclidean titik terakhir (kartu G & G 12) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 12

	G&G 13	G&G 14	G&G 15	G&G 16
Jarak	0.05898	0.08723	0.06469	0.08816
	G&G 17	G&G 18	G&G 19	G&G 20
Jarak	0.14020	0.07694	0.09454	0.10188

   Jarak minimum: 0.05898
   Titik terpilih: G & G 13
   Rute pengambilan kartu: Start – 10 – 11 – 12 – 13
   Jarak yang ditempuh: 0.15807

5. Perhitungan jarak Euclidean titik terakhir (kartu G & G 13) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 13

	G&G 14	G&G 15	G&G 16	G&G 17
Jarak	0.12845	0.09364	0.13358	0.17272
	G&G 18	G&G 19	G&G 20	
Jarak	0.07051	0.09144	0.09265	

   Jarak minimum: 0.07051
   Titik terpilih: G & G 18
   Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18
   Jarak yang ditempuh: 0.22858

6. Perhitungan jarak Euclidean titik terakhir (kartu G & G 18) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 18

	G&G 14	G&G 15	G&G 16	G&G 17	G&G 19	G&G 20
Jarak	0.16413	0.04700	0.09343	0.11233	0.02114	0.02542

   Jarak minimum: 0.02114
   Titik terpilih: G & G 19
   Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18 – 19
   Jarak yang ditempuh: 0.24792

7. Perhitungan jarak Euclidean titik terakhir (kartu G & G 19) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 19

	G&G 14	G&G 15	G&G 16	G&G 17	G&G 20
Jarak	0.18102	0.04992	0.09243	0.09959	0.01092

   Jarak minimum: 0.01092
   Titik terpilih: G & G 20
   Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18 – 19 – 20
   Jarak yang ditempuh: 0.26064

8. Perhitungan jarak Euclidean titik terakhir (kartu G & G 20) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 20

	G&G 14	G&G 15	G&G 16	G&G 17
Jarak	0.18889	0.06069	0.10330	0.10791

   Jarak minimum: 0.06069
   Titik terpilih: G & G 15
   Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18 – 19 – 20 – 15
   Jarak yang ditempuh: 0.32133

9. Perhitungan jarak Euclidean titik terakhir (kartu G & G 15) dengan seluruh lokasi kartu yang belum diambil

   Dari hasil perhitungan, didapatkan hasil sebagai berikut:

   Jarak dari G & G 15

	G&G 14	G&G 16	G&G 17
Jarak	0.14294	0.04672	0.07980

   Jarak minimum: 0.04672
   Titik terpilih: G & G 16
   Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18 – 19 – 20 – 15 – 16
   Jarak yang ditempuh: 0.36805

10. Perhitungan jarak Euclidean titik terakhir (kartu G & G 16) dengan seluruh lokasi kartu yang belum diambil

    Dari hasil perhitungan, didapatkan hasil sebagai berikut:

    Jarak dari G & G 16

	G&G 14	G&G 17
Jarak	0.14273	0.06003

    Jarak minimum: 0.06003
    Titik terpilih: G & G 17
    Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18 – 19 – 20 – 15 – 16 – 17
    Jarak yang ditempuh: 0.42808

11. Perhitungan jarak Euclidean titik terakhir (kartu G & G 16) dengan satu-satunya kartu yang belum diambil yaitu G & G 14. Jarak yang didapat adalah 0.20269

    Jarak minimum: 0.20269
    Titik terpilih: G & G 14
    Rute pengambilan kartu: Start – 10 – 11 – 12 – 13 – 18 – 19 – 20 – 15 – 16 – 17 – 14
    Jarak yang ditempuh: 0.63077

12. Perhitungan jarak pengambilan kartu terakhir untuk kembali ke titik awal semula. Perhitungannya adalah 0.63077 + 0.10318 yaitu sebesar 0.73395

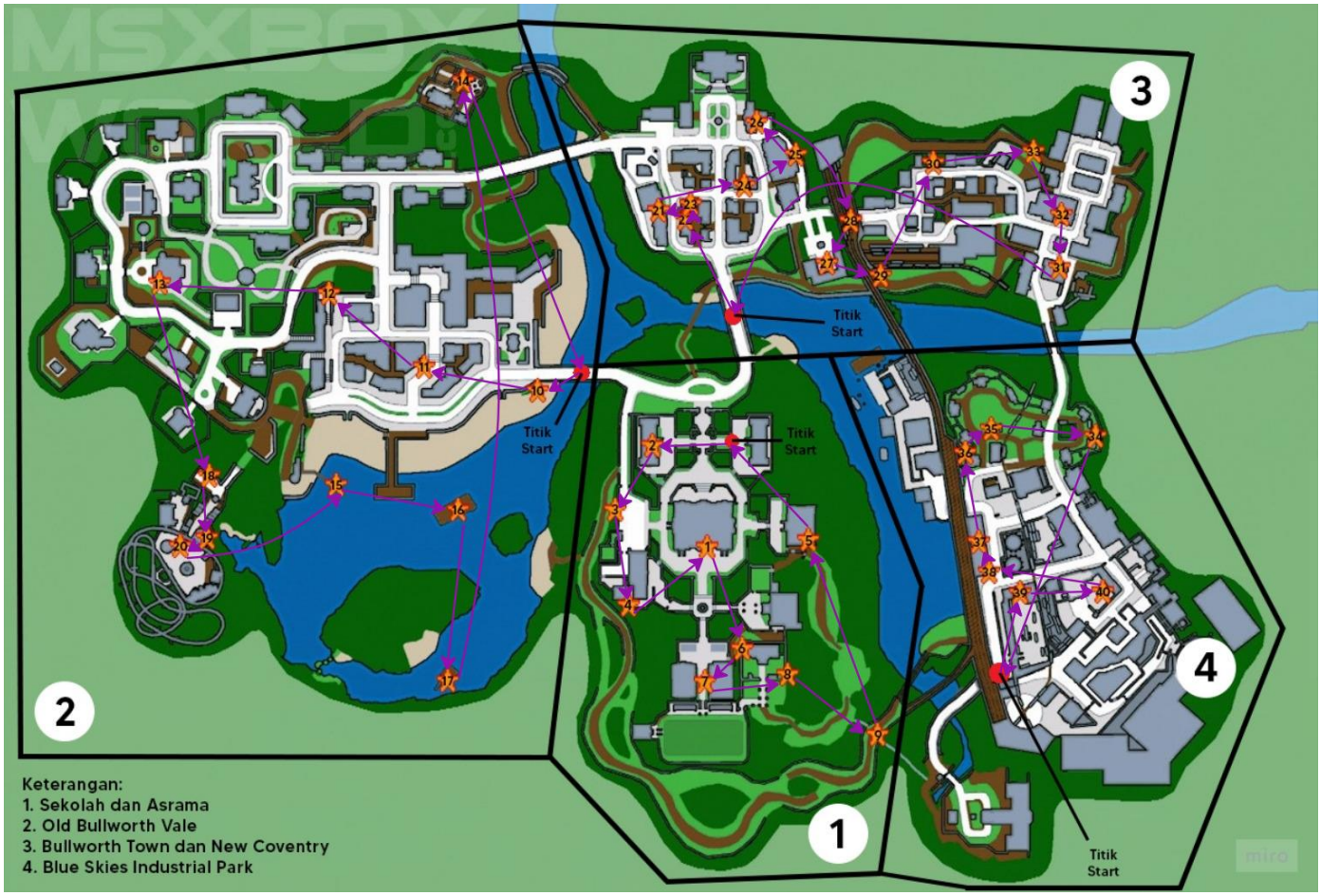
Dengan demikian, urutan rute pengambilan kartu yang didapat adalah melalui (Start – 10 – 11 – 12 – 13 – 18 – 19 – 20 – 15 – 16 – 17 – 14 – Start) dengan jarak yang ditempuh adalah sebesar 0.63077 untuk TSP tanpa kembali ke titik awal dan 0.73395 untuk TSP keseluruhan.

Asumsikan peta memiliki skala 1:5 dari kenyataan dalam *game*. Maka dapat disimpulkan bahwa Jimmy Hopkins harus menempuh jarak sebesar 3.15385 km untuk mendapatkan seluruh kartu G & G yang tersebar di kota Bullworth Vale dan menempuh jarak sebesar 3.66975 km untuk mendapatkan seluruh kartu dan kembali ke tempat semula.

Dengan cara pencarian yang sama, bisa didapatkan juga urutan pencarian kartu beserta besar jarak yang ditempuh dari pencarian kartu yang dilakukan pada daerah sekolah dan asrama, Bullworth Town dan New Coventry, dan juga Blue Skies Industrial Park dengan hasil masing-masing sebagai berikut:

1. Sekolah dan asrama
   Rute pengambilan kartu: Start – 2 – 3 – 4 – 1 – 6 – 7 – 8 – 9 – 5 – Start
   Jarak untuk mendapatkan keseluruhan kartu: 0.30954 (1.54770 km jika memakai skala 1:5)
   Jarak untuk mendapatkan kartu dan kembali ke titik awal: 0.35259 (1.76295 km jika memakai skala 1:5)
2. Bullworth Town dan New Coventry
   Rute pengambilan kartu: Start – 22 – 23 – 21 – 24 – 25 – 26 – 28 – 27 – 29 – 30 – 33 – 32 – 31 – Start
   Jarak untuk mendapatkan keseluruhan kartu: 0.31888 (1.59440 km jika memakai skala 1:5)
   Jarak untuk mendapatkan kartu dan kembali ke titik awal: 0.43482 (2.17410 km jika memakai skala 1:5)
3. Blue Skies Industrial Park
   Rute pengambilan kartu: Start – 39 – 40 – 38 – 37 – 36 – 35 – 34 – Start
   Jarak untuk mendapatkan keseluruhan kartu: 0.16338 (816.9 meter jika memakai skala 1:5)
   Jarak untuk mendapatkan kartu dan kembali ke titik awal: 0.25128 (1.25460 km jika memakai skala 1:5)

Dari hasil ini, didapatkan untuk pemetaan rute keseluruhan untuk proses mendapatkan kartu G & G yang tersebar di kota Bullworth adalah sebagai berikut:



Gambar 3.3 Rute keseluruhan untuk mencari kartu G & G berdasarkan wilayah dengan menggunakan Algoritma Greedy

(Sumber: http://yunuzzone.blogspot.com/2012/02/grottos-gremlins-card.html)

Selain itu, persoalan penyelesaian pencarian kartu G & G yang tersebar di seluruh kota Bullworth ini juga dapat diselesaikan dengan menggunakan program Python dengan kode program yang ada adalah sebagai berikut:

```python
import math

# Fungsi untuk menghitung jarak antara dua titik dengan rumus Euclidean
def euclidean_distance(x1, y1, x2, y2):
    return math.sqrt(math.pow(x2 - x1, 2) + math.pow(y2 - y1, 2))

# Fungsi untuk menyelesaikan TSP dengan algoritma Greedy
def tsp_greedy(points):
    # Inisiasi variabel awal
    n = len(points) # Banyaknya titik
    visited = [False] * n # Untuk menandai titik
    visited[0] = True # Titik awal selalu dikunjungi
    tour = [0] # List untuk menyimpan urutan titik yang dikunjungi
    current = 0 # Titik awal
    total_distance = 0 # Total jarak yang ditempuh


    # Loop sampai semua titik dikunjungi
    for i in range(n - 1):
        # Mencari titik terdekat yang belum dikunjungi
        min_distance = float('inf')
        nearest = None
        for j in range(n):
            if not visited[j]:
                dist = euclidean_distance(points[current][0], points[current][1],
                                          points[j][0], points[j][1])
                if dist < min_distance:
                    min_distance = dist
                    nearest = j

        # Mengunjungi titik terdekat
        visited[nearest] = True # Jika dikunjungi, maka True
        tour.append(nearest) # Menambahkan titik ke list tour
        current = nearest
        total_distance += min_distance # Menambahkan jarak ke total jarak

    # Menghitung jarak total yang ditempuh untuk mendapatkan kartu
    # (TSP tanpa kembali ke titik awal)
    all_tour = tour.copy()
    all_card_dist = total_distance

    # Menghitung jarak total yang ditempuh untuk mendapatkan kartu
    # dan kembali ke tempat awal (TSP dengan kembali ke titik awal)
    tour.append(0)
    total_distance += euclidean_distance(points[current][0], points[current][1],
                                         points[0][0], points[0][1])

    return all_tour, all_card_dist, tour, total_distance
```

Gambar 3.4 *Screenshoot Source Code* Program

(Sumber: Dokumentasi Penulis dengan referensi dari https://github.com/albertofermer/Cuarto/blob/f4a9364b9ef0aab7c05b3491c46fd40d86919e8f/2%C2%BA%20Cuatrimestre/%5BMByHB%5D%20Modelos%20Bioinspirados%20y%20Heur%C3%ADsticas%20de%20B%C3%BAsquedas/Pr%C3%A1cticas/Pr%C3%A1ctica4_AlgoritmosHormigas/Greedy/greedy.py#L1)

Pada kode program ini, terdapat dua buah fungsi, yaitu fungsi untuk melakukan perhitungan jarak dengan menggunakan Euclidean Distance dan juga fungsi untuk pencarian rute TSP dengan menggunakan Algoritma Greedy. Pada fungsi yang dibuat ini, fungsi TSP akan menerima input berupa list koordinat, lalu ada variable yang digunakan untuk menghitung jumlah koordinat, menyimpan urutan titik dalam array, menandai titik yang sudah dikunjungi, dan menghitung bobot jaraknya.

Program kemudian melakukan pengecekan *loop* selama banyaknya koordinat untuk mencari titik-titik yang ingin dikunjungi. Jika titik yang ada belum dikunjungi, program akan melakukan perhitungan jarak Euclidean dan mencari jarak yang terkecil. Setelah didapat titik dengan jarak Euclidean terkecil, kunjungi titik tersebut dan kemudian masukkan titik ke dalam array list urutan titik dan kemudian mengganti titik pengecekan sebelumnya menjadi titik terbaru yang dikunjungi tersebut dan berulang hingga semua titik sudah dikunjungi.

Setelah itu, program akan menyimpan semua titik yang sudah dilalui beserta jarak tempuhnya. Untuk TSP dengan kembali ke titik awal, terdapat tour.append(0) yang memasukkan titik awal ke dalam list dan jarak tempuh akan ditambah dengan jarak dari titik terakhir yang dikunjungi dengan titik awal.

Untuk pengujian dari program ini, terdapat list koordinat dari berbagai posisi kartu G & G yang dibagi berdasarkan wilayah. Lalu menjalankan fungsi TSP dan mencetak hasil keluarannya. Berikut adalah kode programnya:

```python
# Aplikasi pemakaian fungsi

# Lokasi koordinat di sekolah dan asrama
points1 = [(-0.6823566285177947,0.6931773185769998),(-0.6908306068896195,0.6560155978259274),
           (-0.7102114982045293,0.6902874963907095),(-0.7224923632054185,0.6691401304319555),
           (-0.7190979803525010,0.6374005055338330),(-0.6546365231392883,0.6602412600027776),
           (-0.6813123348844101,0.6243960869202425),(-0.6922242824752516,0.6102020773937937),
           (-0.6647455880106747,0.6119472177570486),(-0.6314288212711290,0.5906712389110140)]

# Lokasi kooridnat di Old Bullworth Vale
points2 = [(-0.7339770057955377,0.7171200748019544),(-0.7515039527407907,0.7113837913199461),
           (-0.7881380380868279,0.7213568759748625),(-0.8245620953786954,0.7436009994076045),
           (-0.8832699860243451,0.7492174281678672),(-0.7718367662567971,0.8130983761912773),
           (-0.8213324752495055,0.6789928549708435),(-0.7754081955933145,0.6704141049165315),
           (-0.7802472155254634,0.6105844395197124),(-0.8683130760697679,0.6803168897974672),
           (-0.8671585091240956,0.6592062510491274),(-0.8777970188348831,0.6567323421563316)]

# Lokasi koordinat di Bullworth Town dan New Coventry
points3 = [(-0.6817731649855716,0.7391365227204432),(-0.7072935342821154,0.7737503467854907),
           (-0.6976769913303258,0.7734338711894395),(-0.6976769913303258,0.7762660249021138),
           (-0.6779875513944660,0.7825735561744409),(-0.6613793131754164,0.7936875787733442),
           (-0.6754555460852316,0.8029563933302057),(-0.6463638168978321,0.7539908900649692),
           (-0.6416860443756320,0.7626590303292033),(-0.6324592453638331,0.7522315129475174),
           (-0.6124686178808645,0.7926583543359982),(-0.5668703747570589,0.7545794704131197),
           (-0.5667660638855239,0.7724180054999863),(-0.5832626161595726,0.7992810836853295)]

# Lokasi koordinat di Blue Skies Industrial Park
points4 = [(-0.5884509331891934,0.6136238695016516),(-0.5555639706745410,0.6951375334578103),
           (-0.5890773097942485,0.6966137831224444),(-0.6004452706409040,0.6868705268337578),
           (-0.5977878252479059,0.6573453876431046),(-0.5954256515628629,0.6483777724118625),
           (-0.5786061730781284,0.6409168092642545),(-0.5640090603683348,0.6406863429521081)]

# Menjalankan fungsi tsp_greedy
tourAllCards, distanceAllCards, tourTSP, distanceTSP = tsp_greedy(points2)

# Menampilkan hasil
print("Rute untuk mendapatkan semua kartu:", tourAllCards)
print("Jarak untuk mendapatkan semua kartu:", distanceAllCards)
print("Rute untuk mendapatkan semua kartu dan kembali ke titik awal:", tourTSP)
print("Jarak untuk mendapatkan semua kartu dan kembali ke titik awal:", distanceTSP)
```

Gambar 3.5 *Screenshoot* Main Program

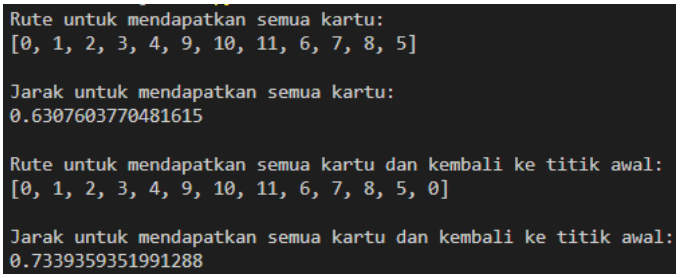
(Sumber: Dokumentasi Penulis dengan data koordinat yang diambil dari https://mapgenie.io/bully/maps/bullworth)

Jika program dijalankan, maka akan menampilkan hasil sebagai berikut:

```
Rute untuk mendapatkan semua kartu:
[0, 2, 3, 4, 1, 6, 7, 8, 9, 5]

Jarak untuk mendapatkan semua kartu:
0.30955414642461543

Rute untuk mendapatkan semua kartu dan kembali ke titik awal:
[0, 2, 3, 4, 1, 6, 7, 8, 9, 5, 0]

Jarak untuk mendapatkan semua kartu dan kembali ke titik awal:
0.35260281883852496
```

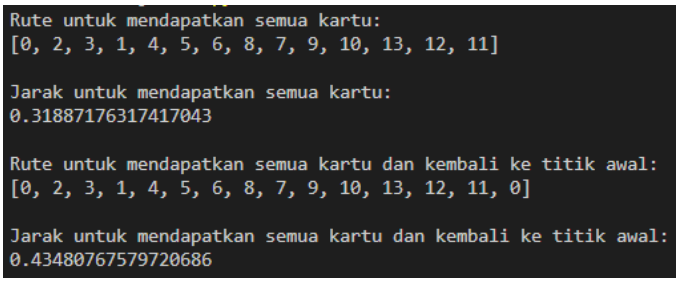
Gambar 3.6 *Screenshoot* Test Menjalankan Program untuk Pencarian Kartu G & G di sekolah dan asrama

(Sumber: Dokumentasi Penulis)



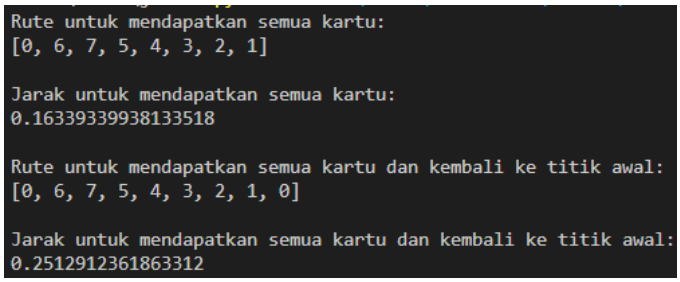
Gambar 3.7 *Screenshoot* Test Menjalankan Program untuk Pencarian Kartu G & G di Old Bullworth Town

(Sumber: Dokumentasi Penulis)



Gambar 3.8 *Screenshoot* Test Menjalankan Program untuk Pencarian Kartu G & G di Bullworth Town dan New Coventry

(Sumber: Dokumentasi Penulis)



Gambar 3.9 *Screenshoot* Test Menjalankan Program untuk Pencarian Kartu G & G di Blue Skies Industrial Park

(Sumber: Dokumentasi Penulis)

Dari hasil yang ditampilkan dalam program, hasil yang ada sudah sangat sesuai dengan perhitungan yang dilakukan pada proses manual sebelumnya, baik untuk rute perjalanan maupun untuk jarak yang ditempuh. Hanya mungkin terdapat beberapa informasi sedikit sebagai berikut:

1. Untuk rute pencarian kartu G & G pada Old Bullworth Town, didapat hasil yaitu [0, 1, 2, 3, 4, 9, 10, 11, 6, 7, 8, 5, 0]. Untuk hasil ini, angka-angka selain 0 perlu ditambah sejumlah 9 untuk menyesuaikan dengan penomoran pada peta yang dibuat. Sehingga rute yang benar adalah [0, 10, 11, 12, 13, 18, 19, 20, 15, 16, 17, 14, 0]. Rute ini sudah sesuai dengan pencarian manual yang dilakukan diatas sebelumnya.
2. Untuk rute pencarian kartu G & G pada Bullworth Town dan New Coventry, didapat hasil yaitu [0, 2, 3, 1, 4, 5, 6, 8, 7, 9, 10, 13, 12, 11, 0]. Untuk hasil ini, angka-angka selain 0 perlu ditambah sejumlah 20 untuk menyesuaikan dengan penomoran pada peta yang dibuat. Sehingga rute yang benar adalah [0, 22, 23, 21, 24, 25, 26, 28, 27, 29, 30, 33, 32, 31, 0]. Rute ini sudah sesuai dengan pencarian manual yang dilakukan diatas sebelumnya.
3. Untuk rute pencarian kartu G & G pada Old Bullworth Town, didapat hasil yaitu [0, 6, 7, 5, 4, 3, 2, 1]. Untuk hasil ini, angka-angka selain 0 perlu ditambah sejumlah 33 untuk menyesuaikan dengan penomoran pada peta yang dibuat. Sehingga rute yang benar adalah [0, 39, 40, 38, 37, 36, 35, 34]. Rute ini sudah sesuai dengan pencarian manual yang dilakukan diatas sebelumnya.
4. Untuk jarak yang ditempuh, jarak yang ada sudah sangat hampir mendekati. Perbedaan jarak yang sedikit dapat terjadi akibat permasalahan pembulatan pada perhitungan yang dilakukan secara manual.

Untuk hasil yang ada ini, masih terdapat kondisi yang sebenarnya bukan merupakan solusi optimal untuk proses pencarian kartu G & G secara keseluruhan karena Algoritma Greedy sendiri memang tidak melakukan pencarian secara menyeluruh terhadap seluruh kemungkinan solusi karena hanya mencari berdasarkan jarak terdekatnya saja. Namun, Algoritma Greedy cukup membantu dalam proses penyelesaian permasalahan yang cukup rumit seperti contohnya pada permasalahan TSP untuk mencari semua kartu G & G yang tersebar di peta Bullworth.

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Algoritma Greedy merupakan algoritma yang dapat dipergunakan untuk berbagai macam persoalan yang melibatkan optimasi minimasi ataupun maksimasi. Seperti pada permasalahan dalam makalah ini, Algoritma Greedy dapat dipergunakan untuk menyelesaikan permasalahan *Travelling Salesman Problem* untuk mencari minimasi optimal untuk mengunjungi semua titik lokasi kartu G & G dengan jarak yang paling minimum.

Hasil yang didapatkan melalui pencarian dengan menggunakan Algoritma Greedy bisa saja bukan merupakan solusi optimal dikarenakan beberapa kondisi tertentu. Meskipun begitu, Algoritma Greedy cukup bermanfaat karena dapat menghasilkan solusi dari permasalahan yang cukup rumit dengan cara yang cukup mudah dan waktu yang singkat.

B. Saran

Saran yang ada terkait pengerjaan tugas ini adalah mungkin untuk pemilihan algoritma, bisa dipertimbangkan dengan lebih baik lagi untuk pencarian agar dapat menghasilkan solusi yang lebih optimal.

TAUTAN VIDEO PENJELASAN

Untuk video penjelasan dari makalah ini, video dapat diakses melalui link berikut:

https://youtu.be/NDo2R5n4mA0

UCAPAN TERIMA KASIH

Penulis mengucapkan puji syukur kepada Tuhan Yang Maha Esa karena atas berkat dan rahmatnya, penulis dapat menyelesaikan makalah ini. Penulis juga menyampaikan terima kasih kepada keluarga dan teman-teman yang telah membantu dan memberikan dukungan kepada penulis dalam proses pembuatan makalah ini. Tidak lupa juga, penulis menyampaikan terima kasih terhadap Bapak dan Ibu dosen pengampu mata kuliah IF2211 Strategi Algoritma yang senantiasa membimbing dan mendidik dalam mata kuliah ini. Terakhir, penulis juga meminta maaf jika makalah yang ada masih terdapat kesalahan dan belum sempurna.

REFERENCES

[1] Pecinta Games, “Apa yang Dimaksud Open World dalam Video Game?”, https://www.pecinta.games/2020/10/apa-itu-game-open-world.html, 2020. [Diakses pada tanggal 13 Mei 2023].

[2] Program Studi Teknik Informatika STEI-ITB, "Homepage Rinaldi Munir," 2021. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf [Diakses pada tanggal 13 Mei 2023].

[3] Program Studi Teknik Informatika STEI-ITB, "Homepage Rinaldi Munir," 2021. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag3.pdf [Diakses pada tanggal 13 Mei 2023].

[4] Interview Bit, “Travelling Salesman Problem (TSP)”, https://www.interviewbit.com/blog/travelling-salesman-problem/, 2023. [Diakses pada tanggal 22 Mei 2023]

[5] Fox News, “Publisher: 'Bully' Video Game Has Positive Message”, https://web.archive.org/web/20130926031407/http://www.foxnews.com/story/2006/10/17/publisher-bully-video-game-has-positive-message/, 2006. [Diakses pada tanggal 13 Mei 2023]

[6] Y. Hopkins. “Grottos & Gremlins Card”, http://yunuzzone.blogspot.com/2012/02/grottos-gremlins-card.html, 2012. [Diakses pada tanggal 13 Mei 2023]

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, 22 Mei 2023

Jason Rivalino
13521008