Tugas Besar 1 IF2211 Strategi Algoritma

Pemanfaatan Algoritma Greedy dalam Aplikasi Permainan “Galaxio”

Disusun oleh:

Muhammad Naufal Nalendra (13521152)

Sulthan Dzaky Alfaro (13521159)

Muhammad Habibi Husni (13521169)

Program Studi Teknik Informatika

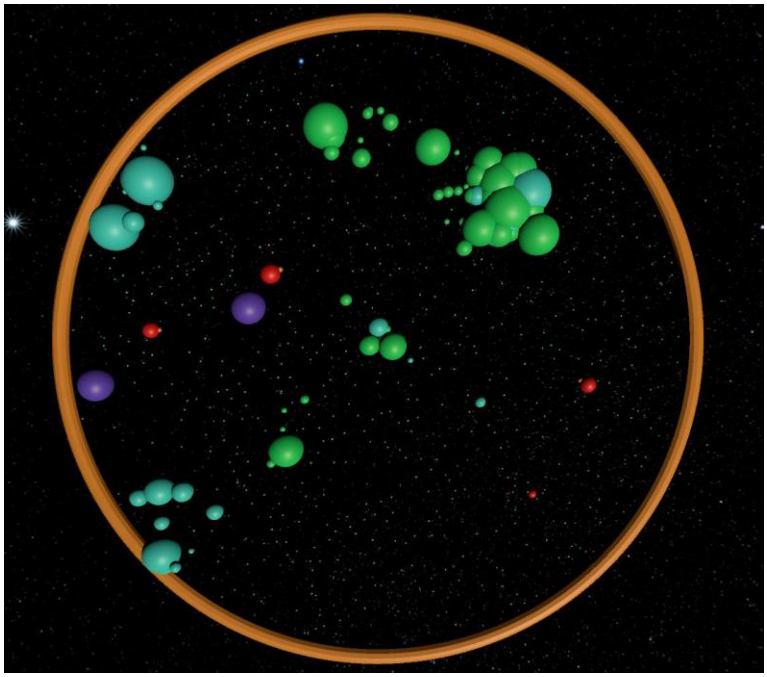
Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

BAB 1

Deskripsi Tugas

Galaxio adalah sebuah game battle royale yang mempertandingkan bot kapal anda dengan beberapa bot kapal yang lain. Setiap pemain akan memiliki sebuah bot kapal dan tujuan dari permainan adalah agar bot kapal anda yang tetap hidup hingga akhir permainan. Penjelasan lebih lanjut mengenai aturan permainan akan dijelaskan di bawah. Agar dapat memenangkan pertandingan, setiap bot harus mengimplementasikan strategi tertentu untuk dapat memenangkan permainan.



Adapun aturan-aturan yang ada di permainan ini, antara lain:

1. Map (Peta)

Peta adalah bidang kartesius yang membentang ke arah positif dan negatif. Peta hanya akan berurusan dengan bilangan bulat. Kapal Anda hanya dapat berada dalam posisi bilangan bulat x,y di peta. Pusat peta akan menjadi 0,0 dan tepi akan disediakan sebagai radius, putaran maksimum permainan akan sama dengan ukuran radius.

Peta berisi objek-objek berikut:

- Pemain - diri Anda sendiri dan lawan lainnya.
- Makanan - benda kecil yang perlu Anda kumpulkan untuk memperbesar ukurannya.
- Lubang cacing - dua titik di peta yang terhubung di kedua arah.
- Awan gas - zona berbahaya yang akan melukai Anda saat melintasi.
- Bidang asteroid - zona berbahaya yang akan memperlambat Anda saat melintasi.

Visibilitas

Seluruh peta akan terlihat setiap saat dan akan menyusut seiring berkurangnya batas.

Batas

Ukuran peta akan menyusut setiap tik permainan. Objek yang berada di luar peta di akhir permainan akan dihapus dari peta. Ini tidak termasuk pemain, berada di luar peta akan mengurangi ukuran pemain sebanyak 1 setiap centang permainan. Pengurangan ukuran peta adalah tindakan terakhir dari kutu permainan.

2. Objek

Semua objek diwakili oleh bentuk lingkaran dan memiliki titik pusat dengan koordinat X dan Y serta radius yang menentukan ukuran dan bentuknya.

Makanan

Peta akan tersebar dengan objek makanan berukuran 3 yang dapat dikonsumsi oleh pemain.

- Makanan tidak akan bergerak.
- Makanan akan dihapus jika berada di luar peta.
- Jika seorang pemain bertabrakan dengan makanan, ia akan menghabiskan makanan tersebut secara keseluruhan dan pemain akan bertambah dengan ukuran yang sama dengan makanannya.

Makanan Super

Saat makanan muncul di peta, ia memiliki peluang persentase untuk muncul sebagai makanan super. Makanan super meningkatkan "tingkat penyerapan" Anda sebagai nilai yang digerakkan oleh konfigurasi yang saat ini ditetapkan ke 2, artinya Anda mengonsumsi 2 kali ukuran makanan standar lain yang Anda konsumsi. Kemampuan ini tetap untuk sejumlah kutu, saat ini 5.

Nilai berikut mengonfigurasi fitur ini:

- ScoreRates (Item 6): kesempatan untuk menghasilkan makanan super.
- superfoodConsumptionRate: pengganda konsumsi untuk makanan super.
- superfoodEffectDuration: jumlah kutu yang efeknya bertahan.
- maxSuperfoodCount: jumlah maksimum makanan super yang dapat dihasilkan di peta.

Contoh : Anda memiliki bot ukuran 10 yang bertabrakan dengan makanan super dan bertambah 3 menjadi 13, pada centang berikutnya tidak bertabrakan dengan apa pun. Pada centang kedua

bertabrakan dengan makanan standar, ukuran bot sekarang bertambah 6 menjadi 19. Karena itu adalah centang kedua dengan tingkat konsumsi aktif, efeknya sekarang kedaluwarsa.

Lubang cacing

Lubang cacing ada berpasangan, memungkinkan kapal pemain masuk ke satu sisi dan keluar dari sisi lainnya. Lubang cacing akan tumbuh setiap centang permainan ke ukuran maksimum yang ditetapkan, saat dilalui, lubang cacing akan menyusut setengah ukuran kapal yang melintasinya.

- Traversal hanya dapat dilakukan jika lubang cacing lebih besar dari kapal.
- Melintasi lubang cacing itu seketika, tidak ada penalti yang diterapkan kepada pemain karena menggunakan lubang cacing.
- Momentum dan arah dipertahankan melalui lubang cacing.
- Jika salah satu ujung pasangan wormhole berada di luar peta, keduanya akan hancur.
- Pasangan lubang cacing tidak diberikan melainkan perlu dipetakan oleh pemain melalui coba-coba.

Awan Gas

Awan gas akan tersebar di sekitar peta dalam kelompok awan yang lebih kecil. Kapal dapat melintasi awan gas, namun begitu sebuah kapal bertabrakan dengan awan gas, ukurannya akan berkurang 1 setiap detik permainan. Setelah kapal tidak lagi bertabrakan dengan awan gas, efeknya akan hilang.

Bidang asteroid

Bidang asteroid akan tersebar di sekitar peta dalam kelompok awan yang lebih kecil. Kapal dapat melintasi medan asteroid, namun begitu kapal bertabrakan dengan awan asteroid, kecepatannya akan berkurang 2 kali lipat. Setelah kapal tidak lagi bertabrakan dengan medan asteroid, efeknya akan hilang.

Salvo torpedo

Saat ini muncul di peta, mereka telah diluncurkan dari kapal lain! Hati-hati dengan jalur mereka karena tabrakan dengan mereka akan mengurangi ukuran mereka saat ini dari ukuran Anda. Mereka juga bertabrakan dengan semua yang ada di peta, menyebabkan kerusakan pada apa pun yang menghalangi jalan mereka!

- Mereka melakukan perjalanan dalam lintasan garis lurus, dalam pos tertentu yang diwakili dalam keadaan mereka
- Mereka melakukan perjalanan dengan kecepatan 60
- Mereka mulai dengan ukuran 10
- Mereka melanjutkan selama mereka memiliki ukuran yang tersisa
- Ukuran dikurangi dari mereka ketika mereka bertabrakan dengan apa pun kecuali lubang cacing, serta dari apa pun yang bertabrakan dengan mereka
- Ukuran yang mereka kurangi dari apa yang mereka tabrak sama dengan ukuran objek yang lebih kecil yang terlibat dalam tabrakan

- Contoh:
- Jika salvo bertabrakan dengan makanan, dan makanan berukuran 3, dan salvo berukuran 10, pada akhir tumbukan makanan akan berukuran 0 dan dihilangkan, dan salvo akan berukuran 7
- Jika salvo bertabrakan dengan pemain berukuran 20, dan salvo berukuran 10, salvo akan berukuran 0 dan dihilangkan, dan pemain berukuran 10. Pemain yang menembakkan salvo akan mendapatkan ukuran 10 juga.
- Jika dua salvo bertabrakan, satu berukuran 5 dan satu berukuran 10, yang berukuran 5 akan berukuran 0 dan dihilangkan, dan yang lainnya berukuran 5 dan melanjutkan jalurnya.
- Jika mereka bertabrakan dengan bot, mereka mencuri ukuran yang mereka konsumsi dan mengembalikannya ke bot asli yang memecat mereka
- Mereka akan segera disingkirkan dari dunia segera setelah mereka keluar dari batas dunia.

Tabrakan Lubang Cacing

- Salvo torpedo berinteraksi dan melintasi lubang cacing dengan cara yang sama seperti yang dilakukan pemain
- Mereka akan keluar dari lubang cacing di ujung yang berlawanan dan melanjutkan perjalanan mereka
- Mereka mengonsumsi energi dari penjelajahan wormhole, dengan cara yang sama seperti yang dilakukan pemain

Supernova

Supernova adalah senjata yang sangat kuat, yang hanya muncul sekali per pertandingan! Pickup supernova akan muncul antara kuartal pertama dan kuartal terakhir game. Pickup ini akan memungkinkan pemain yang telah mengumpulkannya untuk menembakkan bom supernova!

Bom supernova bergerak seperti torpedo, tetapi tidak bertabrakan dengan apa pun saat bepergian. Pemain yang menembak kemudian dapat mengirim perintah lanjutan untuk meledakkan supernova, menyebabkan kerusakan besar pada pemain mana pun yang terjebak di zona ledakan! Selain itu, itu akan menelurkan awan gas dalam radius kejatuhannya.

Setelah pemain mengumpulkan pickup, itu tidak dapat digunakan oleh pemain lain. Pickup selanjutnya juga tidak akan muncul. Menghancurkan pemain yang saat ini memiliki pikap supernova akan menghancurkan pikap tersebut.

Teleportasi

Seorang pemain dapat meluncurkan teleporter ke suatu arah di peta. Teleporter bergerak ke arah itu dengan kecepatan 20 dan tidak bertabrakan dengan apapun. Pemain kemudian dapat menggunakan perintah lain untuk berteleportasi ke lokasi teleporter mereka. Itu akan hancur jika mencapai akhir peta.

Dibutuhkan pemain ukuran 20 untuk meluncurkan teleporter terlepas dari apakah mereka berteleportasi ke sana atau tidak. Seorang pemain mulai dengan satu teleporter dan mendapatkan yang lain setiap 100 kutu dan dapat memiliki maksimal 10 pada titik mana pun.

Nilai berikut mengonfigurasi fitur ini di file pengaturan aplikasi:

- Tingkat Biaya
- StartChargeCount
- MaxChargeCount
- Ukuran
- Kecepatan
- Biaya

3. Kapal

Bot Anda bermain sebagai pesawat ruang angkasa melingkar, yang memakan benda-benda planet dan kapal lain untuk memperbesar ukurannya. Kapal Anda dimulai dengan nilai-nilai berikut:

- Kecepatan - kapal Anda akan bergerak x posisi ke depan setiap detik permainan, di mana x adalah kecepatan Anda. Kecepatan Anda akan mulai dari 20 dan berkurang seiring pertumbuhan kapal.
- Ukuran - kapal Anda akan mulai dengan radius 10.
- Menuju - kapal Anda akan bergerak ke arah ini, antara 0 dan 359 derajat.

Kapal Anda tidak akan mulai bergerak sampai Anda memberinya perintah untuk melakukannya. Setelah diberikan perintah maju pertama, ia akan terus bergerak dengan kecepatan kapal saat ini dan menuju sampai perintah berhenti diberikan.

Ada ukuran minimal 5 untuk kapal, jika pada titik mana pun ukurannya lebih kecil dari ini, kapal akan dihapus dari peta dan dianggap tersingkir.

Kecepatan

Kecepatan menentukan seberapa jauh kapal Anda akan bergerak setiap tik permainan. Kapal Anda tidak akan bergerak saat berhenti atau perintah MAJU awal telah dikeluarkan, namun nilai kecepatan Anda akan tetap sama. Kecepatan berbanding terbalik dengan ukuran, kapal yang lebih besar akan bergerak lebih lambat. Kecepatan ditentukan dengan rumus berikut:

speedRatio = 200
speedRatio/bot.size = speed

Dengan hasil dibulatkan ke langit-langit dan dengan minimal 1. Perhatikan bahwa nilai 200 berasal dari Speeds.Ratio di appsettings.json dan dapat berubah selama penyeimbangan.

Afterburner

Afterburner akan meningkatkan kecepatan kapal Anda dengan faktor 2. Namun ini juga akan mulai mengurangi ukuran kapal Anda sebesar 1 per tick. Afterburner aktif dapat menyebabkan kehancuran diri jika ukuran kapal menjadi kurang dari 5.

Torpedo Materi Gelap

Kapal Anda dilengkapi dengan torpedo materi gelap. Ini dapat digunakan untuk memanipulasi ruang di sekitar Anda, dan memberikan muatan kepada musuh Anda.

Torpedo yang mengenai objek di dunia akan menghancurkan objek tersebut, dan ketika mereka mendarat di musuh Anda, Anda mencuri materi dari mereka secara proporsional dengan kerusakan yang terjadi pada mereka.

- Kapal Anda akan menerima 1 salvo charge setiap 10 tik.
- Kapal Anda hanya dapat membawa maksimal 5 muatan salvo sekaligus.
- Kapal Anda kemudian dapat menukar muatan salvo dengan ukuran 5 untuk menembakkan rentetan 10 torpedo ke arah yang Anda pilih.
- Menembakkan torpedo sementara kapal Anda kurang dari ukuran 10 dapat menyebabkan kapal Anda hancur sendiri jika ukurannya kurang dari 10.

Tameng

Untuk pertahanan kapal Anda dapat mengaktifkan perisainya. Setelah diaktifkan, itu dapat membelokkan torepedos, yang langsung memantul.

Sementara torpedo akan memantul ke arah yang berlawanan. Kapal Anda akan didorong ke arah yang biasanya dituju oleh torpedo.

Awan gas dan fenomioa lainnya tidak terpengaruh.

Setelah diaktifkan, perisai akan bertahan selama 20 kutu berikutnya dan akan menghabiskan ukuran 20 kutu. Mereka juga membutuhkan waktu isi ulang sebanyak 20 kutu.

4. Tabrakan

Tabrakan akan terjadi ketika dua objek tumpang tindih dengan setidaknya satu unit ruang dunia. Ini berarti bahwa objek dapat saling bersentuhan, tetapi saat mereka tumpang tindih dengan satu unit ruang dunia, mereka akan dianggap bertabrakan dan mekanika tabrakan akan berlaku.

Tabrakan kapal ke kapal

Saat kapal pemain bertabrakan, kapal dengan ukuran lebih besar akan memakan kapal yang lebih kecil dengan laju 50% dari ukuran kapal yang lebih besar hingga maksimum ukuran kapal yang lebih kecil.

Setelah tabrakan kapal, kedua arah kapal akan terbalik 180 derajat, mereka akan dipisahkan oleh 1 unit ruang dunia dan selanjutnya akan terus bergerak ke arah baru ini setelahnya. Ini akan mensimulasikan pantulan.

Tabrakan makanan

Ketika sebuah kapal bertabrakan dengan partikel makanan, kapal akan mengkonsumsi makanan secara keseluruhan dan akan bertambah besar ukurannya dengan ukuran makanan yang baru saja ditabraknya.

5. Game Tick Payload

Semua pemain akan menerima keadaan dunia, semua objek game, dan semua objek pemain di awal setiap centang. Muatan setiap centang game akan berisi informasi berikut:

```
{
  "World":                                                                    {
    "CenterPoint":                                                            {
      "X":                                                                    0,
      "Y":                                                                    0
    },
    "Radius":                                                                 1000,
    "CurrentTick":                                                            0
  },
  "GameObjects":                                                              {
    "8b77d46b-2844-48c4-a3f3-179de15776a3":    [    3,    0,    0,    2,    42,    225    ],
    "2b75d46b-2866-48f1-a4g5-179de15779o0":    [    3,    0,    0,    2,    234,    -900    ],
    "9b34d46b-4844-48g2-a3b2-179de15771m8":    [    3,    0,    0,    2,    -100,    189    ],
    ...
  },
  "PlayerObjects":                                                            {
    "ad672ef2-f6a7-404c-950a-a867c54c7de0":    [    10,    20,    0,    1,    42,    225,    1    ],
    "5f535caf-c3fc-4935-9d95-6e48b3680fd7":    [    20,    10,    90,    1,    234,    -900,    2    ],
    "e671e725-4bbb-4d4d-ad18-f013a567dfda":    [    40,    5,    180,    1,    -100,    189,    4    ],
    ...
  }
}
```

GameObjects

Daftar GameObjects berisi semua objek di peta. Daftar berisi panduan untuk setiap objek dan data objek.

Urutan data tidak akan berubah, dan adalah sebagai berikut:

- Ukuran - Jari-jari objek
- Kecepatan - Kecepatan yang dapat digerakkannya
- Heading - Arah yang dituju

- GameObjectType - Jenis objek

  2: Makanan

  3: Lubang cacing

  4: Awan Gas

  5: Lapangan asteroid

- Posisi X - Posisi x pada bidang kartesius
- Posisi Y - Posisi y pada bidang kartesius

PlayerObjects

Daftar PlayerObjects berisi semua objek pemain di peta. Daftar berisi panduan untuk setiap objek dan data objek.

Urutan data tidak akan berubah, dan adalah sebagai berikut:

- Ukuran - Jari-jari objek
- Kecepatan - Kecepatan yang dapat digerakkannya
- Heading - Arah yang dituju
- GameObjectType - Jenis objek

  1: Pemain

- Posisi X - Posisi x pada bidang kartesius
- Posisi Y - Posisi y pada bidang kartesius
- Efek Aktif - Efek Bitwise saat ini memengaruhi bot

  Ini adalah flag bit kumulatif, yang diwakili oleh:

  0 = Tidak berpengaruh

  1 = Afterburner aktif

  2 = Bidang Asteriod

  4 = Awan gas

  Misalnya, jika sebuah kapal memiliki ketiga efek, efek aktifnya adalah 7.

6. Perintah

Di setiap permainan centang setiap pemain dapat mengirimkan satu perintah untuk kapal mereka.

Semua perintah pemain divalidasi sebelum menjalankan perintah apa pun. Perintah yang tidak valid (mis. Sintaks yang tidak valid) mengakibatkan perintah diabaikan dan kapal Anda mempertahankan statusnya saat ini.

Semua perintah pemain dijalankan pada waktu yang sama (dalam satu centang permainan), dan tidak berurutan.

Semua Perintah

- MAJU
- BERHENTI
- START_AFTERBURNER
- STOP_AFTERBURNER

Struktur Komando

Perintah dapat dikirim ke mesin sesering yang Anda suka, tetapi mesin tidak menunggu perintah dari bot dan memproses status game dengan kecepatan yang ditentukan.

Pelari hanya akan mengizinkan maksimal satu perintah per centang.

Perintah Anda akan diajukan ke bot Anda untuk diproses selama centang berikutnya. Tindakan ini diproses di bawah FIFO (First in, First out), artinya tindakan terkirim Anda yang paling awal diproses terlebih dahulu. Namun, daftar tindakan ini hanya boleh sepanjang satu perintah.

Ini berarti bot Anda tidak perlu mengirim perintah setiap centang dan dapat memakan waktu selama ia ingin mengirim setiap perintah. Jangan ragu untuk menjalankan semua kecerdasan buatan pintar yang Anda suka! Perhatikan saja bahwa bot lain mungkin masih mengirimkan perintah karena tidak ada waktu tunggu.

Perintah: MAJU

Perintah ini akan mulai menggerakkan kapal Anda ke arah yang ditentukan dalam derajat.

Perintah: BERHENTI

Perintah ini akan berhenti menggerakkan kapal Anda hingga perintah gerakan lain dikeluarkan. Tidak ada interia, oleh karena itu kapal Anda langsung berhenti.

Perintah: START_AFTERBURNER

Perintah ini mengaktifkan afterburner kapal Anda. Perhatikan, kecepatan Anda hanya digunakan saat Anda sedang bergerak ke suatu arah, oleh karena itu afterburner hanya akan berpengaruh saat Anda sedang bergerak. Biaya afterburner akan selalu berlaku, jika efek afterburner aktif.

Perintah: STOP_AFTERBURNER

Perintah ini menonaktifkan afterburner kapal Anda.

Perintah: FIRE_TORPEDOES

Perintah ini menggunakan 1 muatan salvo dan ukuran 5 untuk mengirimkan salvo torpedo ke arah yang diberikan

Perintah: FIRE_SUPERNOVA

Perintah ini menggunakan pikap supernova untuk mengirimkan supernova ke pos yang diberikan

Perintah: DETONATE_SUPERNOVA

Perintah ini meledakkan supernova yang ada

Perintah: FIRE_TELEPORTER

Perintah ini menghabiskan 1 biaya teleportasi dan 20 ukuran untuk mengirim teleporter ke arah yang diberikan

Perintah: TELEPORTASI

Perintah ini memindahkan Anda ke teleporter yang ada jika ada.

Perintah: USE_SHIELD

Perintah ini mengaktifkan perisai Anda untuk membelokkan torpedo yang masuk, jika Anda memilikinya.

7. Akhir permainan

Kapal terakhir yang hidup adalah kapal pemenang dan dengan demikian bot pemenang. Jika dua kapal terakhir mati pada saat yang sama, kapal dengan skor tertinggi akan menjadi pemenangnya.

8. Skor

Skor akan disimpan untuk setiap pemain yang akan terlihat setelah pertandingan selesai. Ini hanya akan digunakan untuk memecahkan dasi.

- Mengkonsumsi kapal pemain lain = 10 poin
- Mengkonsumsi makanan = 1 poin
- Melintasi lubang cacing = 1 poin

Bab 2: Landasan Teori

2.1. Gambaran Algoritma Greedy secara Umum

Algoritma greedy adalah algoritma yang memecahkan persoalan secara langkah per langkah step by step dan bersifat *greedy* dalam mendefinisikan solusinya. Algortima ini berguna mencari solusi dari optimization problem seperti mencari nilai optimum global dari suatu permasalahan. Pada setiap langkah dari algoritma dicari pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan. Akan tetapi, algoritma ini tidak membolehkan backtracking sehingga harus mencari optimum lokal dari setiap langkahnya. Hal ini bertujuan bahwa langkah-langkah optimum lokal tersebut mengarah pada solusi dengan optimum global (global optimum).

Syarat dari penggunaan algoritma greedy adalah permasalahan tersebut memenuhi dua sifat berikut:

- Solusi optimal dari persoalan dapat ditentukan dari solusi optimal subpersoalan tersebut.
- Pada setiap persoalan, terdapat suatu langkah yang dapat dilakukan dimana langkah tersebut menghasilkan solusi optimal pada subpersoalan tersebut.

Algoritma greedy memiliki beberapa komponen yang perlu didefinisikan sebagai berikut:

- Himpunan kandidat (C): Berisi kandidat yang mungkin dipilih pada setiap langkahnya.
- Himpunan solusi (S): Berisi kandidat yang sudah terpilih sebagai solusi.
- Fungsi solusi (solution function): Menentukan apakah himpunan solusi yang dikumpulkan sudah memberikan solusi. (Domain: himpunan objek, Range: boolean).
- Fungsi seleksi (selection function): Memilih kandidat berdasarkan strategi greedy tertentu. Fungsi ini memiliki sifat heuristik (fungsi dirancang untuk mencari solusi optimum dengan mengabaikan apakah fungsi tersebut terbukti paling optimum secara matematis). (Domain: himpunan objek, Range: objek).
- Fungsi kelayakan (feasibility function): Memeriksa apakah kandidat yang terpilih oleh fungsi seleksi dapat dimasukkan ke dalam himpunan solusi. (Domain: himpunan objek, Range: boolean).
- Fungsi objektif (objective function): Memaksimumkan atau meminimumkan suatu parameter pada suatu persoalan. (Domain: himpunan objek, Range: himpunan objek).

Dengan menggunakan elemen/komponen di atas, algoritma greedy dapat didefinisikan sebagai berikut: "Algoritma greedy merupakan pencarian sebuah himpunan bagian S dari himpunan kandidat C, dimana S memenuhi kriteria kelayakan sebagai solusi paling optimum, yaitu S merupakan suatu himpunan solusi dan S dioptimisasi oleh fungsi objektif."

Skema umum algoritma greedy menggunakan pseudocode dengan pendefinisian elemen/komponennya 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 mememuhi 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 Skema Algoritma Greedy

Pada akhir tiap iterasi, solusi yang terbentuk adalah optimum lokal, dan paka akhir loop while-do akan ditemukan optimum global, tapi perlu diingat bahwa optimum global ini belum tentu merupakan solusi yang optimum melainkan sub-optimum atau pseudo-optimum. Hal ini karena algoritma greedy tidak melakukan operasi secara menyeluruh kepada semua kemungkinan yang ada serta banyaknya variasi fungsi seleksi yang bisa dipilih.

Dapat disimpulkan bahwa algoritma greedy cocok digunakan untuk permasalahan dimana hanya dibutuhkan solusi secara hampiran dan bukan solusi mutlak. Keuntungan dari algoritma ini adalah dapat menemukan solusi secara jauh lebih cepat dibanding algoritma brute force. Salah satu penggunaannya adalah pada Traveling Salesman Problem dimana solusi dapat ditemukan lebih cepat walaupun kalah optimal dengan algoritma brute force.

2.2. Garis Besar Cara Kerja Bot Permainan Galaxio

Secara garis besar permainan galaxio memerlukan beberapa object permainan sebagai berikut:

GameObjects

Daftar Semua objek pada peta dan deskripsinya

- Size - Jari-jari sebuah objek
- Speed - Kecepatan yang dapat digerakkan sebuah objek
- Heading - Arah yang dituju
- GameObjectType - Jenis objek

  2: Makanan

3: Lubang cacing

4: Awan Gas

5: Lapangan asteroid

- Posisi X - Posisi x pada bidang kartesius
- Posisi Y - Posisi y pada bidang kartesius

PlayerObjects

Daftar PlayerObjects berisi semua objek pemain di peta beserta panduan untuk setiap objek

Urutan data tidak akan berubah, dan adalah sebagai berikut:

- Size - Jari-jari objek
- Speed - Kecepatan yang dapat digerakkan sebuah objek
- Heading - Arah yang dituju
- GameObjectType - Jenis objek

  1: Pemain

- Posisi X - Posisi x pada bidang kartesius
- Posisi Y - Posisi y pada bidang kartesius
- Efek Aktif - Efek yang saat ini memengaruhi bot

  Sebuah flag bit kumulatif, yang diwakili oleh:

  0 = Tidak berpengaruh

  1 = Afterburner aktif

  2 = Bidang Asteriod

  4 = Awan gas

Selain GameObjects dan PlayerObjects, terdapat beberapa objek secara spesifik yang mempengaruhi jalannya permainan:

Objects

- Food : Objek tidak bergerak yang menambah size kapal
- Super Food : Menambah pertambahan size untuk setiap food yang dimakan
- Wormholes : Pemain dapat masuk di satu ujung dan keluar di ujung yang lainnya
- Gas clouds : Dapat dilewati tapi mengurangi size kapal setiap tick
- Asteroid fields : Dapat dilewati tapi mengurangi speed kapal

- Torpedo salvo : Objek yang ditembakkan oleh kapal musuh dan dapat mengurangi size player
- Supernova : Serangan seperti torpedo tapi dapat dikendalikan kapan meledak
- Teleport : Mengurangi size player dan meluncurkan teleporter ke sebuah arah

Ship

- Speed : Menentukan seberapa jauh kapal bergerak setiap tick
- Afterburner : Meningkatkan kecepatan kapal sebanyak 2 kali lipat
- Dark matter torpedoes : Menembakkan torpedo dengan harga size 5
- Shield : Melindungi dari serangan torpedo

Bot akan melakukan analisis menggunakan algoritma yang ditentukan dengan mempertimbangkan berbagai mekanisme diatas. Selama jalannya permainan bot akan mencari langkah selanjutnya yang dikira terbaik untuk memenangkan permainan. Sebuah bot menjadi pemenang permainan jika semua musuh sudah lenyap dan bot tersebut menjadi satu-satunya yang bertahan.

2.3. Implementasi Algoritma Greedy ke dalam Bot Permainan Galaxio

Ada beberapa algoritma yang dapat digunakan sebagai algoritma bot. Beberapa algoritma yang dapat digunakan adalah algoritma brute force, algoritma greedy, algoritma tree traversal, dan lain-lain. Tentunya setiap algoritma ini memiliki kelebihan dan kekurangan masing masing. Pada algoritma brute force misalnya, dibutuhkan banyak waktu dan tenaga untuk menguji dan menemukan hasil yang optimal meskipun hasil yang ditemukan pasti adalah nilai optimal global. Sementara itu, algoritma greedy tidak membutuhkan banyak waktu dan tenaga tetapu membutuhkan strategi heuristik yang mumpuni untuk mendekati persoalan.

Strategi heuristik tidak bisa menghasilkan solusi optimum global secara pasti dan permbuktiannya membutuhkan waktu yang lama. Namun, kelebihan dari algoritma greedy dalam pemrograman bot ini jauh lebih besar dari kekurangannya. Jika dbandingkan dengan algoritma-algoritma lain, algoritma greedy lebih mudah dan tidak memerlukan sumber daya yang banyak.

Karena itu, penulis memilih menggunakan algoritma greedy untuk menyelesaikan permasalahan kali ini. Terdapat banyak kemungkinan solusi algoritma greedy yang dapat diterapkan sehingga penggunaan algoritma ini juga fleksibel. Walaupun tidak ada jaminan bahwa algoritma akan mencapai solusi global, harapan penulis algoritma yang dibuat dapat mencapai nilai-nilai optimum lokal yang menghampiri nilai optimum global.

2.4. Garis Besar Game Engine Permainan Galaxio

Game engine adalah framework perangkat lunak yang utamanya didesain untuk pengembangan video game dan biasanya termasuk library yang relevan dan program yang membantu. Di dalam sebuah game engine biasanya terdapat rendering engine, physics engine, sound engine, scripting language, animation engine, artificial intelligence, networking engine, streaming engine, memory management, threading support, localization support, scene graph, dan video support.

Pada permainan Galaxio game engine sudah disediakan sebelumnya oleh Entelect dan tersedia dalam starter-pack sehingga tidak perlu dilakukan compile ulang dan build project untuk menggunakan engine. Prerequisites atau requirements dari game engine ini adalah JDK (Java Development Kit) dengan versi minimal versi 11, Microsoft .NET Core 3.1 , serta adanya NodeJs dalam mesin. User dapat menggunakan file run.bat untuk menjalankan game engine pada windows atau menggunakan script makefile pada command prompt untuk Linux atau macOS.

Terdapat beberapa file .json seperti appsettings.json yang dibutuhkan game engine untuk mengubah berbagai parameter game engine serta beberapa detail dan state dari permainan seperti MinimumPlayerSize atau StartingPlayerSize. Beberapa parameter pada file appsettings.json dapat diubah tapi sebagian besar sebaiknya dibiarkan agar tidak tidak terjadi konflik atau perubahan pada game rules yang diterapkan

• appSettings.json:

→ "BotCount" : Menentukan jumlah maksimal bot dalam permainan
→ "TickRate" : Menentukan jumlah rata-rata tick dalam permainan
→ "MapRadius" : Menentukan ukuran map yang berupa lingkaran
→ "MapRadiusRatio" : Menentukan rasio untuk jari-jari map
→ "StartRadius": Menentukan radius saat game dimulai
→ "StartRadiusRatio": Menentukan rasio untuk radius saat game dimulai
→ "StartingPlayerSize": Menentukan size awal sebuah player saat game dimulai
→ "MinimumPlayerSize" : Menentukan size minimum sebuah player untuk tetap bermain
→ "ConsumptionRatio" : Menentukan berapa kali lipat dari makanan yang akan dicerna menjadi size
→ "ScoreRates": Mengatur score yang didapat sesuai performance dalam game
→ "Speeds": Mengatur kecepatan player atau objek
→ "Seeds": Menentukan bagaimana map akan di generate
→ "WorldFood": Mengatur bagaimana semua makanan di map akan diletakkan, ukuran, dll
→ "Wormholes": Mengatur parameter dari objek wormhole
→ "Afterburners": Mengatur parameter dari objek afterburner
→ "GasClouds": Mengatur parameter dari objek gas cloud
→ "AsteroidFields": Mengatur parameter dari objek asteroid field
→ "Torpedo": Mengatur parameter dari objek torpedo
→ "Shield": Mengatur parameter dari objek perisai
→ "Supernova": Mengatur parameter dari objek supernova
→ "Teleport": Mengatur parameter dari objek teleporter

Awalnya starter bot default belumlah lengkap, masih terdapat fitur fitur yang belum lengkap dan belum ditambahkan ke dalam folder src dari starter bot. Oleh karena itu penulis terlebih dahulu menambahkan beberapa fitur ke beberapa file java dan melakukan import seluruh

file ke BotService.java. Beberapa file yang belum lengkap merupakan PlayerActions.java dan ObjectTypes.java.

Setelah itu, penulis menggunakan algoritma greedy nya ke dalam BotService.java dimana penulis mengimplementasikannya ke dalam aksi aksi dari player. Setelah algoritma greedy starter bot telah dibuat , lalu dikompilasi agar perubahan tersebut dapat diimplementasikan pada file jar dari starter bot. Pada program ini telah disediakan build tools dare Apache Maven Project. Terdapat beberapa command dari build lifecycle yang dapat dijalankan oleh Maven, diantaranya adalah berikut ini:

• validate: Melakukan validasi apakah seluruh data dan informasi yang dibutuhkan untuk melakukan build project sudah terpenuhi.

• compile: Melakukan kompilasi terhadap source code yang telah dibuat oleh pemrogram.

• test: Melakukan testing menggunakan source code hasil kompilasi terhadap suatu framework yang sudah ditentukan sebelumnya.

• package: Melakukan packaging dari hasil kompilasi source code menjadi suatu file dengan format jar.

• verify: Melakukan tes verifikasi integrasi terhadap file jar yang telah dibangun (menentukan apakah project sudah siap untuk di-install).

• install: Melakukan instalasi project ke dalam local repository beserta dependencies project tersebut.

• deploy: Menyudahi build environment yang telah dibuat sebelumnya dan melakukan copy project ke dalam remote repositories.

Pada permainan Galaxio ini, untuk melakukan kompilasi dan mengubah ke jar, pengguna hanya perlu menggunakan compile dan install

Bab 3: Aplikasi Strategi Greedy

3.1. Pemetaan Elemen/Komponen Algoritma Greedy pada Bot Permainan Galaxio

3.1.1. Pemetaan Elemen/Komponen Algoritma Greedy pada Permasalahan Menghindari Obstacle (asteroid, warp, dan gas cloud)

Tujuan utama dari permainan Galaxio adalah menjadi satu-satunya kapal yang bertahan hingga akhir permainan. Mengikuti tujuan tersebut, permainan memiliki beberapa mekanisme untuk menambah atau mengurangi ukuran hingga habis.

Agar bot yang dirancang dapat menggunakan seluruh sumber daya yang dimiliki terutama ukuran kapal dengan baik, kapal harus dioptimisasi agar tidak menerima terlalu banyak

penalti yang mengurangi kesempatan untuk menang seperti damage pada kapal, pengurangan kecepatan, dan lain lain.

Elemen / Komponen	Deskripsi Elemen
Himpunan Kandidat	Permutasi dari command FORWARD dan STOP, atau tidak menggunakan kedua command dalam sebuah turn
Himpunan Solusi	Kemungkinan permutasi dari command yang memastikan bahwa kapal yang dikendalikan bot dapat menghindari atau meminimisasi kerusakan yang ditimbulkan oleh berbagai obstacle
Fungsi Solusi	Memastikan bahwa permutasi dari command membuat bot bergerak seperti yang diharapkan (tidak memasuki area yang dipengaruhi oleh obstacles tanpa alasan dan hanya memasuki area obstacle jika diperintahkan)
Fungsi seleksi	Menyeleksi atau memilih command berdasar keadaan atau game state di suatu waktu yang ditentukan beserta strategi heuristik yang telah dipilih dan harus diikuti. Strategi heuristik yang diikuti adalah strategi yang dianggap paling optimal untuk setiap tick dan ronde. Untuk kasus ini yang paling penting adalah menentukan heading saat bertemu dengan sebuah obstacle
Fungsi kelayakan	Memastikan bahwa gerakan yang dilakukan bot untuk menghindari obstacle valid dan meminimisasi damage sehingga sekecil mungkin
Fungsi objektif	Mencari permutasi dari command yang membuat heading untuk menghindar optimal

Tabel 3.1.1.1 Komponen Algoritma Greedy pada Penghindaran Obstacle

3.1.2. Pemetaan Elemen/Komponen Algoritma Greedy pada Permasalahan Mengejar Musuh

Salah satu permasalahan yang ada dalam permainan galaxio adalah strategi dalam melakukan pengejaran terhadap musuh. Hal ini selaras dengan tujuan utama permainan yaitu mengalahkan semua bot lainnya.

Permainan galaxio memiliki sebuah mekanik yang mengatur ship-to-ship collision, yaitu saat sebuah kapal menabrak kapal yang lain. Jika hal ini terjadi, sesuai peraturan maka kapal yang lebih besar akan menyerap size dari kapal lawan yang lebih kecil sehingga dapat menyingkirkan kompetisi

Elemen / Komponen	Deskripsi Elemen
Himpunan Kandidat	Berisi permutasi dari beberapa command yang bisa dipilih untuk mengejar musuh, beserta himpunan bot musuh yang berada dekat kapal
Himpunan Solusi	Kemungkinan permutasi dari command yang mengarahkan bot untuk mengejar musuh dengan tepat dan aman
Fungsi Solusi	Memeriksa dan memastikan bahwa permutasi dari command membuat bot melakukan aksi sesuai yang diharapkan
Fungsi seleksi	Menyeleksi command sesuai game state di suatu waktu yang sesuai dengan strategi heuristik yang diharapkan (memilih musuh yang paling tepat untuk jadi sasaran)
Fungsi kelayakan	Memastikan musuh yang dipilih cocok menjadi sasaran dan sepadan dengan risiko untuk mengejarnya
Fungsi objektif	Menentukan langkah paling tepat sesuai dengan musuh yang dipilih dan kesempatan yang ditentukan

Tabel 3.1.2.1 Komponen Algoritma Greedy pada Pengejaran Musuh

3.1.3. Pemetaan Elemen/Komponen Algoritma Greedy pada Permasalahan Menembak Musuh (torpedo dan supernova)

Salah satu permasalahan lain pada permainan galaxio adalah strategi dalam melakukan penembakan torpedo ke arah musuh. Strategi yang dibuat harus dapat menghitung saat yang tepat untuk menembakkan torpedo beserta arah torpedo harus ditembakkan

Pada permainan galaxio kecepatan sebuah kapal berbanding lurus terhadap ukuran kapal. Dengan menimbang hal tersebut, penting bagi bot untuk dapat menembakkan torpedo dengan benar

Elemen / Komponen	Deskripsi Elemen
Himpunan Kandidat	Berisi permutasi dari beberapa command yang bisa dipilih untuk menembak musuh, beserta himpunan bot musuh yang berada di sekitar kapal
Himpunan Solusi	Kemungkinan permutasi dari command yang mengarahkan bot untuk menembak ke arah musuh dengan tepat
Fungsi Solusi	Memeriksa dan memastikan bahwa permutasi dari command membuat bot melakukan aksi sesuai yang diperintahkan
Fungsi seleksi	Menyeleksi command sesuai game state di suatu waktu yang sesuai dengan strategi heuristik yang diharapkan (memilih musuh yang paling tepat untuk jadi sasaran)
Fungsi kelayakan	Memastikan musuh yang dipilih cocok menjadi sasaran dan dapat diserang dengan akurat
Fungsi objektif	Menentukan langkah paling tepat sesuai dengan musuh yang dipilih dan lokasi musuh tersebut

Tabel 3.1.3.1 Komponen Algoritma Greedy pada Penembakan Musuh

3.1.4. Pemetaan Elemen/Komponen Algoritma Greedy pada Permasalahan Mencari Makanan

Salah satu permasalahan lain pada permainan galaxio adalah strategi dalam melakukan pengambilan makanan. Strategi yang dibuat harus dapat menghitung rute maupun heading yang paling optimal dalam mengambil makanan sebanyak-banyaknya

Elemen / Komponen	Deskripsi Elemen
Himpunan Kandidat	Berisi permutasi dari beberapa command yang bisa dipilih untuk mencari rute makanan terbaik
Himpunan Solusi	Kemungkinan permutasi dari command yang mengarahkan bot untuk mengambil makanan

Fungsi Solusi	Memeriksa dan memastikan bahwa permutasi dari command membuat bot mengambil food atau superfood yang terdekat
Fungsi seleksi	Menyeleksi command sesuai game state di suatu waktu yang sesuai dengan strategi heuristik yang diharapkan (mensorting food yang ada di sekitar kapal dan memilih yang terdekat
Fungsi kelayakan	Memastikan food atau superfood di sekitar tidak kosong dan aman diambil
Fungsi objektif	Menentukan langkah paling tepat sesuai dengan lokasi food dan superfood (diambil terdekat)

Tabel 3.1.4.1 Komponen Algoritma Greedy pada Mencari Makanan

3.1.5. Pemetaan Elemen/Komponen Algoritma Greedy pada Permasalahan Menghindar dan Melindungi Diri dari Musuh

Salah satu permasalahan lain pada permainan galaxio adalah strategi dalam menghindari serangan musuh serta melindungi diri. Strategi yang dibuat harus dapat memberi kapal kesempatan untuk merespons serangan musuh

Pada permainan galaxio musuh dapat menyerang kapal dengan cara mengejar (ship-to-ship collision) atau dengan menembakkan torpedo maupun supernova. Demi mencapai tujuan utama permainan, penting bagi bot untuk mengetahui cara mempertahankan diri

Elemen / Komponen	Deskripsi Elemen
Himpunan Kandidat	Berisi permutasi dari beberapa command yang bisa dipilih untuk menghindari serangan musuh
Himpunan Solusi	Kemungkinan permutasi dari command yang mengarahkan bot untuk menghindar atau melindungi diri
Fungsi Solusi	Memeriksa dan memastikan bahwa permutasi dari command membuat bot dapat mempertahankan diri (menggunakan shield, menyalakan afterburner, dll)

Fungsi seleksi	Menyeleksi command sesuai game state di suatu waktu yang sesuai dengan strategi heuristik yang diharapkan (Menentukan ancaman yang terbesar di tick tertentu)
Fungsi kelayakan	Memastikan dan mengecek jika objek yang di sekitar bukan ancaman atau ancaman yang bisa dihadapi
Fungsi objektif	Menentukan langkah paling tepat berdasar ancaman di sekitar dan command yang dapat digunakan

Tabel 3.1.5.1 Komponen Algoritma Greedy pada Menghindar dan Melindungi Diri

3.2. Eksplorasi Alternatif Solusi Algoritma Greedy pada Bot Permainan Galaxio

Terdapat berbagai solusi alternatif untuk algoritma greedy pada permainan galaxio. Salah satu faktornya adalah banyaknya elemen, objek, serta game state yang ada dalam permainan. Selain itu, semua elemen dan objek saling berkaitan sehingga perubahan pada satu elemen dapat menimbulkan perubahan pada state yang lain. Karena itu, setiap strategi dari permasalahan diatas juga saling berkaitan dan berinteraksi satu sama lain. Berikut adalah penjelasan lebih dalam megenai alternatif strategi dari algoritma greedy untuk tiap permasalahan.

3.2.1. Strategi Heuristik Menghindari Obstacle

Salah satu strategi heuristik yang penting untuk permainan galaxio adalah strategi untuk mengatur cara menghindari obstacle. Penting bagi kapal untuk menghindari berbagai obstacle yang ada karena tiap obstacle tersebut memiliki penalti atau efek yang tidak menguntungkan bagi bot seperti mengurangi size ataupun speed

Strategi pertama adalah menghindari seluruh obstacle bagaimanapun caranya sehingga bot akan sama sekali tidak melewati obstacle area yang ada. Semua obstacle seperti gas cloud maupun asteroid field masing-masing memiliki penalti seperti mengurangi size sebanyak satu per tick maupun mengurangi speed sebanyak dua per tick. Elemen size dan speed dari kapal tentunya penting untuk memenangkan permainan sehingga harus dijaga agar tidak berkurang. Tetapi tentunya menutup kemungkinan untuk memasuki area dari obstacle memiliki kekurangannya sendiri.

Strategi kedua yang dapat diimplementasikan adalah membuat bot tidak menghindari obstacle sepenuhnya dan hanya secara parsial. Pada strategi pertama bot dibuat agar tidak dapat memasuki obstacle sama sekali, tetapi hal ini juga mengurangi fleksibilitas gerakan bot. Terdapat beberapa kasus dimana memasuki area obstacle adalah optimal lokal yang harus diraih, misalnya saat terkepung oleh musuh dan tidak ada jalan keluar yang lain

3.2.2. Strategi Heuristik Mengejar Musuh

Salah satu strategi heuristik yang penting untuk permainan galaxio adalah strategi untuk mengatur metode mengejar musuh. Demi mengejar tujuan utama yaitu kemenangan, kapal harus memiliki algoritma untuk mengejar dan menghabisi musuh. Pada beberapa instansi sangat penting bagi bot untuk memiliki kemampuan menentukan target dan menyingkirkan musuh dari kompetisi dengan cepat agar mendapat kesempatan lebih besar untuk memenangkan permainan

Strategi pertama yang dapat digunakan adalah melakukan pengejaran pada target terdekat dan menghancurkan musuh dengan cepat. Strategi ini berfokus pada konsep *lightning warfare* atau perang kilat, yaitu menghabisi musuh-musuh yang lebih lemah secara cepat agar dapat menggunakan lebih banyak dari ruang dan sumber daya yang tersisa untuk mengalahkan musuh-musuh yang lain. Jika bot dapat menghabisi musuh dengan cepat pada awal permainan maka bot akan mendapat pertambahan size secara drastis sejak awal permainan sehingga berada pada posisi diuntungkan untuk memenangkan permainan

Strategi kedua yang bisa diimplementasikan adalah mengejar sebuah target hanya jika tidak banyak musuh maupun obstacle di sekitar. Saat bot berfokus pada suatu target dan bergerak mengejar, hal ini akan membuat bot rentan akan serangan dari musuh yang lain. Selain itu, jika bot mengejar musuh terlalu jauh terdapat kemungkinan bahwa bot sendiri bisa terkepung oleh musuh yang datang setelahnya. Beberapa poin diatas adalah dasar pemikiran dari strategi ini

3.2.3. Strategi Heuristik Menembak Musuh

Salah satu strategi heuristik yang penting untuk permainan galaxio adalah strategi untuk mendekati cara dan langkah menembakkan senjata ke musuh. Dalam permainan galaxio terdapat beberapa senjata jarak jauh yang disediakan mulai dari torpedo, supernova, hingga teleporter dapat digunakan. Oleh karena itu, penting bagi bot untuk menguasai dan menggunakan semua senjata tersebut sesuai potensialnya

Strategi pertama yang bisa diimplementasikan adalah menembakkan torpedo ke arah musuh hanya jika trajectory antar kedua kapal relatif kosong serta size kapal mencukupi. Hal ini dikarenakan torpedo memiliki harga size lima sedangkan serangan torpedo bernilai sepuluh dan berkurang satu setiap objek yang dilewati. Agar serangan torpedo yang diluncurkan memiliki nilai serangan yang lebih tinggi dan senilai dengan harganya, akan sangat menguntungkan jika trajectory antar kedua kapal tidak memiliki objek yang menghalangi

Strategi kedua yang bisa diimplementasikan hampir sama dengan strategi pertama dengan perbedaannya terletak pada minimal size yang harus dua kali lebih besar dari harga menggunakan senjata. Hal ini dikarenakan terdapat kemungkinan bahwa tembakan akan meleset dan size yang digunakan menjadi sia sia. Selain itu, adanya cadangan sumber daya

memungkinkan kapal untuk melakukan aksi command lain dalam sekejap sehingga membuat kapal dapat beraksi secara fleksibel

Strategi ketiga adalah ketika bot mendapat senjata supernova. Bot akan menggunakan senjata supernova dan menembakkan supernova ke kumpulan musuh dengan syarat melebihi jarak tertentu. Dasar dari pemikiran ini adalah kekuatan supernova yang memiliki radius serangan besar. Namun, radius yang besar juga berarti bahwa ada kemungkinan untuk bot terkena serangan sendiri. Hal seperti itu lah yang harus dicegah dalam program

3.2.4. Strategi Heuristik Mencari Makanan

Salah satu strategi heuristik yang penting untuk permainan galaxio adalah strategi untuk mencari rute optimal untuk mengambil makanan. Food dan superfood dalam permainan dapat dikonsumsi oleh kapal dan menyebabkan size bot bertambah. Pertambahan size ini sangat penting karena merupakan sumber daya utama bagi kapal untuk melakukan berbagai aksi

Strategi pertama adalah mengambil rute makanan mulai dari makanan yang terdekat. Hal ini dapat diimplementasikan menggunakan sorting objek food dan superfood yang ada di sekitar kapal, lalu bergerak menuju makanan yang dinilai jaraknya paling dekat. Strategi ini dapat memaksimalkan makanan yang diambil secara efektif

Strategi kedua yang dapat diimplementasikan adalah mengelompokkan kelompok makanan dan pergi ke kelompok makanan yang paling besar. Sama seperti strategi pertama, diperlukan sorting terhadap objek makanan yang sudah dibagi menjadi beberapa kelompok sesuai lokasinya. Perbedaan yang mendasar dari strategi pertama adalah prioritas yang diambil. Strategi ini mengutamakan mengambil makanan sebanyak-banyaknya dengan mengelompokkan lokasi objek makanan dengan lokasi makanan relatif terhadap lokasi kapal. Hal ini berbeda dengan strategi pertama yang mengutamakan kecepatan mendapat makanan dalam jangka pendek

3.2.5. Strategi Heuristik Menghindar dan Melindungi Diri

Salah satu strategi heuristik yang penting untuk permainan galaxio adalah strategi untuk bertahan hidup dan menghindari serta berlindung dari bahaya. Elemen ini sangat penting karena Tujuan utama permainan adalah menjadi satu-satunya yang bertahan hidup. Dengan tujuan tersebut, kapal harus memiliki kemampuan untuk menghindar atau melarikan diri dari ancaman yang mendekat serta melindungi diri dari serangan torpedo dan semacamnya

Strategi Pertama yang dapat diimplementasikan adalah mengaktivasi shield saat diserang oleh torpedo jika size cukup. Serangan torpedo merupakan serangan jarak jauh utama yang dimiliki oleh kapal sehingga merupakan ancaman kelas tinggi. Sebuah kapal dapat menembakkan torpedo beberapa kali sehingga dapat menghancurkan kapal musuh dalam waktu singkat. Strategi untuk menangkal torpedo musuh adalah menggunakan shield yang dapat diaktivasi dengan harga size 20. Kekurangan dari sebuah shield adalah

harganya yang mahal sehingga bisa digunakan untuk melakukan banyak command yang lain

Strategi Kedua yang dapat digunakan adalah strategi *hit and run* atau menghindar dengan afterburner dan menembakkan torpedo di sela-selanya. Mekanisme speed pada permainan galaxio bergantung pada size kapal dimana speed berbanding terbalik dengan size. Strategi ini mengantisipasi strategi kapal musuh untuk mengejar kapal. Karena musuh kemungkinan besar hanya mengejar kapal lain jika ukuran kapal lawan lebih kecil, maka hampir dapat dipastikan bahwa kapal yang dikejar akan lebih cepat dibanding kapal yang mengejar. Oleh karena itu, kami mengimplementasikan strategi untuk menghindar ke arah yang berlawanan disela dengan menembakkan torpedo ke kapal yang mengejar. Jika strategi ini digunakan dengan benar, kapal yang dikejar dapat menggunakan peluang tersebut menyerang balik kapal yang mengejar

3.3. Analisis Efisiensi dari Kumpulan Solusi Algoritma Greedy

Permainan Galaxio ini memiliki banyak variabel yang dapat mempengaruhi jalannya permainan, Setiap kapal dapat mengetahui game state serta game object pada tick tertentu. Hal ini memungkinkan tiap kapal untuk mengetahui variabel-variabel penting seperti ukuran kapal musuh, kecepatan musuh, benda-benda yang terdapat dalam lapangan, dan lain sebagainya. Tiap-tiap strategi algoritma Greedy yang kami implementasikan memiliki kompleksitas waktunya masing-masing.

Pada pemanggilan metode hindariMusuh, dilakukan pengecekan masing-masing musuh yang ada pada list permainan secara linear sehingga kompleksitasnya adalah O(n). Setelah itu dilakukan pengurutan list musuh berdasarkan jarak secara menaik untuk menentukan musuh yang akan ditembak dengan torpedo secara O(n log n), sehingga kompleksitas total metode ini adalah O(n log n).

Pada pemanggilan metode hindariTorpedo, dilakukan pengurutan list torpedo berdasarkan jarak dengan kapal secara menaik dengan kompleksitas O(n log n). Setelah itu dilakukan pengiterasian list tersebut untuk mencari torpedo yang akan menjadi kandidat untuk dihindari. Kompleksitas total metode ini adalah O(n log n).

Pada pemanggilan metode ambilMakanan dan ambilSuperfood, juga dilakukan pengurutan list berdasarkan jarak dengan pemain dengan algoritma sorting O(n log n). Setelah itu dilakukan iterasi untuk mencari kadidat makanan/superfood yang valid secara O(n). Kompleksitas total metode ini adalah O(n log n).

Pada pemanggilan metode tembakTorpedo, kejarMusuh, dan tembakSupernova juga diperlukan algoritma sorting karena kandidat aksi yang diutamakan adalah objek yang paling dekat dengan pemain. Metode-metode ini juga memiliki kompleksitas O(n log n).

Pada algoritma ambilSupernova dan ledakkanSupernova, hanya diperlukan pengecekan kandidat aksi secara linear sehingga kompleksitas kedua metode ini adalah O(n).

3.4. Analisis Efektivitas dari Kumpulan Solusi Algoritma Greedy

3.4.1. Efektivitas Strategi Heuristik Menghindari Obstacle

Strategi pertama merupakan basis yang bagus untuk memulai pemikiran tentang cara menghindari obstacle, berusaha sebisa mungkin untuk tidak memasuki area gas cloud maupun asteroid berarti bot akan menjaga size dan speed miliknya agar tidak berkurang tanpa alasan bagus.

Strategi kedua memfokuskan pada fleksibilitas bot dan memberi bot opsi untuk memasuki berbagai area berbahaya seperti gas cloud maupun asteroid. Jika bot terkepung di pinggir map misalnya, maka bot dapat menggunakan gas cloud untuk melarikan diri.

Setelah menimbang pilihan kami, kami memilih untuk menggunakan utamanya strategi pertama. Hal ini karena saat kapal bergerak, perpindahannya tidak pasti serta saat size kapal besar maka speed nya akan lamban. Memasuki area gas cloud atau asteroid dianggap tidak sepadan risiko yang diterima.

3.4.2. Efektivitas Strategi Heuristik Mengejar Musuh

Strategi pertama berfokus pada pencarian pertarungan dan kompetisi secara cepat agar bot dapar berkapitulasi pada kemenangan pertama dan mengalahkan semua musuh lain dengan mudah. Konsep diatas juga bisa disebut *snowballing* atau mengeluarkan effort besar di awal agar dapat mudah menyelesaikan sisa rintangan

Strategi kedua lebih berfokus pada keamanan bot dan pertahanan diri sehingga berusaha menjauhi konflik di awal dan mengumpulkan berbagai resource yang adal terlebih dahulu. Keuntungan dari strategi ini adalah pada fase game setelah fase awal, ada kemungkinan bahwa state game saat itu menguntungkan bot. Misalnya ada beberapa bot musuh yang saling menghancurkan atau berada pada posisi yang lemah. Bot dapat berkapitulasi pada momen ini untuk meraih kemenangan.

Strategi ketiga merupakan strategi yang memiliki prioritas tinggi dan perlu digunakan pada awal permainan. Hal ini karena senjata supernova hanya ada satu buah selama jalannya permainan dan pastinya rentan untuk direbut oleh musuh. Agar bot dapat mengamankan supernova diberikan berbagai opsi mulai teleport dan afterburner untuk merebut supernova secepat mungkin

Dihadapkan ketiga strategi diatas, kami memutuskan untuk mengimplementasikan gabungan dari strategi kedua dan ketiga. Kami putuskan menggunakan strategi kedua setelah melihat banyaknya bot lain yang cenderung mengambil strategi pertama. Sementara itu, strategi ketiga dapat diimplementasikan sementara pada awal permainan dan bot dapat berganti prioritas saat objektif gagal diraih

3.4.3. Efektivitas Strategi Heuristik Menembak Musuh

Strategi pertama mendesain bot agar menggunakan torpedo secara efektif dan efisien. Trajectory dari torpedo ke arah musuh menjadi fokus utama yang perlu dioptimisasi. Hal ini karena karena harga dari torpedo konstan sementara value dari torpedo berkurang satu setiap objek yang dilewati. Hal ini lah yang berusaha untuk dihindari

Strategi kedua mirip dengan strategi pertama dengan pembeda terletak pada size minimal yang lebih besar. Tujuan dari perubahan ini dibanding strategi pertama adalah adanya cadangan sumber daya pada kapal untuk melakukan berbagai strategi lain misal tembakan meleset atau terjadi kesalahan dalam pengambilan keputusan. Adanya sisa sumber daya untuk digunakan dapat digunakan sebagai bakal dari rencana cadangan atau *damage control*

Kami memilih untuk menggabungkan beberapa aspek dari strategi pertama dengan strategi kedua agar mencapai hasil yang maksimal. Menggunakan gabungan strategi maka torpedo akan ditembakkan secara efektif dan efisien

3.4.4. Efektivitas Strategi Heuristik Mencari Makanan

Strategi pertama adalah langsung mensorting semua makanan yang ada di sekitar kapal berdasar jaraknya lalu mengambil dari makanan yang terdekat. Maksud dan tujuan dari strategi ini adalah membuat bot mengambil makanan mulai dari yang ada di dekatnya tanpa memedulikan kelanjutan dari arah yang diambil

Strategi kedua memiliki maksud utama mencari tahu rute pengambilan makanan yang terbaik sebelum bergerak ke daerah tersebut untuk mengambil makanan. Perbedaan mendasar dibanding strategi pertama adalah adanya penentuan jangka panjang untuk memilih rute yang diambil. Pada strategi ini, bot sudah tahu jika tempat yang dituju memiliki banyak makanan

Diantara kedua strategi, kami lebih memilih untuk menggunakan strategi pertama untuk kemudahan komputasi karena lebih efisen untuk dilakukan dari strategi kedua.

3.4.5. Efektivitas Strategi Heuristik Menghindar dan Melindungi Diri

Strategi pertama berfokus pada perlindungan dari serangan jarak jauh. Metode atau cara yang digunakan adalah ketika bot mendeteksi adanya torpedo yang mendekat, selanjutnya bot akan memasang shield jika size mencukupi atau jika tidak maka melakukan manuver menghindar

Strategi kedua mengambil kasus saat bot dikejar oleh musuh yang lebih besar. Taktik yang diimplementasikan adalah taktik *hit and run* yaitu menghindari musuh dengan pergi ke arah sebaliknya sambil menembakkan torpedo di sela sela pelarian. Menggunakan cara ini bot dapat menghindari bahaya sekaligus melakukan serangan balik ke musuh dengan peluang untuk menyerang balik musuh jika size menjadi kurang dari size bot

Melihat kedua strategi diatas, kami memutuskan untuk menggabungkan dan mengimplementasikan keduanya pada bot. Hal ini karena kedua strategi memiliki sinergi yang bagus

3.5. Strategi Greedy yang Digunakan pada Program Bot

Strategi heuristik yang penulis ambil sebagai algoritma *greedy* utama dari program bot game Galaxio ini merupakan gabungan dari beberapa kumpulan solusi yang telah penulis buat pada subab 3.2. Penulis mengambil beberapa strategi, mulai dari *Defense* serta *Attack.* Agar seluruh startegi dapat digabung menjadi suatu algoritma *greedy* yang optimal, pengurutan strategi harus dilakukan. Apabila tidak dilakukan pengurutan, prioritas aksi akan menjadikan algoritma tidak optimal. Oleh karena itu dilakukan pengurutan agar mendapatkan algoritma yang paling optimum. Untuk mencari algoritma yang paling optimum, penulis melakukan uji coba pada program dan memformulasikan urutan mana yang paling optimum yang dapat diimplementasikan sebagai algoritma *greedy*.

Setelah melakukan berbagai uji coba pada game ini, penulis berhasil mendapatkan strategi yang cukup optimal. Urutan prioritas aksi dari startegi yang telah di formulasikan oleh penulis sebagai berikut:

- Mengindar torpedo, cara menghindar kami dengan cara bergerak ke arah 90 derajat counterclockwise dari arah torpedo menyerang. Selain itu apabila sudah dekat dengan bot, bot akan mengaktifkan afterburner. Apabila size mencukupi, bot akan mengkatifkan shield untuk menghindari serangan musuh.
- Menghindari musuh, Apabila size musuh lebih besar dan musuh bergerak dengan cepat dan bot memiliki size yang cukup, bot akan mengaktifkan afterburner. Apabila shield tidak aktif dan size bot mencukupi, bot akan menghindar sekaligus menyerang musuh dengan torpedo.
- Kejar musuh, mengejar musuh dilakukan apabila size bot lebih besar dari size musuh terdekat.
- Ambil superfood, mengambil superfood terdekat lebih diprioritaskan agar bot bertambah besar lebih cepat.
- Ambil makanan, mengambil makanan biasa dilakukan dengan mengambil makanan terdekat
- Serang lawan, Menyerang lawan dengan torpedo dilakukan apabila musuh terdekat tidak memakai shield.

Strategi ini cukup optimal dalam permainan ini. Hampir semua aspek permainan ditangani oleh algoritma ini. Walaupun menurut penulis sudah cukup optimal, algoritma *greedy* ini tidaklah sempurna dan butuh peningkatan lagi.

Bab 4: Implementasi dan Pengujian

4.1. Implementasi Algoritma Greedy pada Bot Permainan Galaxio

Implementasi algoritma greedy terdapat pada file BotService.java. Terdapat beberapa fungsi untuk merealisasikan algoritma greedy yang telah disusun. Berikut ada beberapa fungsi yang digunakan untuk merealisasikan algoritma greedy.

4.1.1. Fungsi Algoritma Greedy

Ada beberapa fungsi yang dibuat untuk mengimplementasikan hasil algoritma greedy kami. Fungsi fungsi inilah yang menentukan apakah bot yang telah dibuat optimal atau tidak. Berikut fungsi fungsi yang digunakan untuk merealisasikan hasil algoritma greedy kami.

- HindariMusuh(PlayerAction playeraction, Posisition tujuan)

  Fungsi ini digunakan untuk menghindar musuh. Jika terdapat beberapa musuh yang berada didekat kapal/bot, bot akan menghindar dengan cara tembak musuh terdekat dengan torpedo yang tidak sedang memakai shield dan jika size dan stok torpedo mencukupi.

- hindariTorpedo(PlayerAction playeraction)

  Fungsi ini digunakan untuk menghidar torpedo yang ditembakkan dari player lain. Fungsi ini akan memberikan aksi kepada bot untuk mengaktifkan shield jika size mencukupi dan torpedo sudah dekat. Alternatif lain yaitu mengaktifkan afterburner. Jika tidak hidup, bot akan bergerak 90 derajat dari arah torpedo datang.

- ledakkanSupernova (PlayerAction playeraction)

  Fungsi ini digunakan untuk meledakkan supernova yang sebelumnya sudah didapatkan. Peledakan ini dilakukan di dekat tepi peta atau bom akan mengenai lawan.

- AmbilMakanan (PlayerAction playeraction, Posisition tujuan)

  Fungsi ini digunakan untuk mengambil makanan terdekat tetapi tidak menyentuh batas arena

- AmbilSuperfood (PlayerAction playeraction, Posisition tujuan)

  Fungsi ini digunakan untuk mendapatkan superfood terdekat tetapi tidak menyentuh batas arena.

- TembakTorpedo (PlayerAction playeraction)

  Fungsi ini digunakan untuk menembak torpedo ke player lain yang terdekat.

- KejarMusuh (PlayerAction playeraction)

Fungsi ini digunakan untuk mengejar musuh yang sizenya lebih kecil dari size bot kami. Apabila range cukup jauh, bot akan menggunakan teleport untuk mengejar player lain. Apabila jarak cukup dekat, bot akan mengejarnya

- ambilSupernova (PlayerAction playeraction)

  Fungsi ini akan memerintah bot untuk mengambil supernova. Apabila dekat, bot akan menggunakan afterburner. Jika supernova muncul dengan jarak yang jauh dengan bot, bot akan menggunakan teleport.

- tembakSupernova (PlayerAction playeraction)

  Fungsi ini akan menembak supernova ke lawan yang paling jauh agar tidak mengenai bot kami.

4.2. Penjelasan Struktur Data pada Bot Permainan Galaxio

Struktur data pada permainan Galaxio berbentuk class. Class tersebut dapat dikelompokkan menjadi 4 kategori atau bagian. Bagian pertama adalah Enums yang berisi berbagai abilities dan game objects yang mengatur mekanisme permainan, Models yang berisi file-file yang menentukan cara kerja berbagai objek dan parameter dalam game, Services yang berisi file BotService.java sebagai file utama dimana algoritma greedy diterapkan pada bot dan berisi cara bot bekerja dalam permainan, serta file Main.java yang berguna menjalankan permainan.

Di bawah ini adalah penjelasan lebih dalam mengenai beberapa class yang terdapat pada game galaxio beserta files yang ada :

A. Kategori Enums

Berikut adalah beberapa file pada kategori enums :

i) Effects.java

Berisi objek objek yang ada dalam game yang dapat berefek ke player. Afterburner memberikan speed 2 kali pada player, Asteroidfield merupakan area asteroid yang dapat mengurangi speed player, Gascloud merupakan area gas yang dapat mengurangi size player, superfood menambah size player lebih besar dari makanan biasa, dan shield dapat melindungi player dari serangan torpedo.

ii) ObjectTypes.java

Berisi objek objek yang ada di dalam game. Objek objek tersebut antara lain, player, food(makanan), wormhole(mirip teleporter), gascloud, asteroidfield, torpedosalvo (torpedo untuk menyerang), superfood, supernova pickup (mengambil supernova), supernova bomb(menembak supernova), teleporter(untuk teleport ke area lain), shield(melindungi player dari serangan torpedo).

iii) PlayerActions.java

Berisi aksi aksi yang dapat dilakukan player dalam game Galaxio. Aksi aksi tersebut yaitu forward(maju), stop(berhenti), startafterburner(menyalakan after burner), stopafterburner(mematikan after burner), firetorpedos(menembak torpedo), firesupernova(menembak supernova), detonatesupernova(meledakkan supernova), fireteleport(menembakkan teleport), teleport(memindahkan posisi player ke teleport yang sudah ditembak), activateshield(mengaktifkan pelindung).

B. Kategori Models

Berikut adalah beberapa file pada kategori models :

i) GameObject.java

Dalam program ini terdiri dari objek objek yang ada dalam game. Dalam program ini juga terdapat informasi informasi objek tersebut, antara lain speed objek, arah bergerak objek, posisi objek, dan lain lain.

ii) GameState.java

Dalam file ini terdapat program yang me list semua objek yang ada di map game. Mulai dari makanan, superfood, player lain, dan lain lain

iii) GameStateDto.java

Hampir sama dengan GameState.java, file ini berisi program yang mencatat semua objek dalam game.

iv) PlayerAction.java

Dalam file ini terdapat program yang digunakan untuk menentukan aksi bot dalam game. Program ini terdapat ID player, aksi yang akan dilakukan, dan heading atau arah pergerakan aksi tersebut. Aksi yang dapat digunakan tersedia di kategori Enums bagian PlayerActions.java

v) Posisition.java (struktur data tambahan)

Dalam file ini terdapat program yang digunakan untuk mendapatkan posisi dari objek pada game, baik musuh, makanan, kapal sendiri, dan lain lain.

vi) World.java

File ini merupakan informasi map maupun waktu di game ini. Dalam file ini terdapat program yang dapat memberikan informasi titik pusat map, waktu dalam game, dan radius.

C. Kategori Services

i) BotService.java

File ini berisi implementasi-implementasi algoritma greedy yang telah kami buat ke dalam bot game. Strategi strategi yang telah dibuat dibentuk menjadi aksi aksi player agar pergerakan bot menjadi optimal apabila melawan player lain.

D. File Main.java

Class bawaan dari permainan Galaxio. Class ini digunakan untuk memulai dan menjalankan permainan dan merupakan komponen yang penting

4.3. Pengujian Bot serta Analisis Performansi Bot Permainan Galaxio

Pengujian bot dilakukan dengan melawankan bot kami dengan bot default dari game. Kami menggunakan 3 bot dari bot default yang nantinya dipertandingkan dengan bot kami.

Sebuah pengujian tentunya tidak luput dari faktor-faktor yang mengganggu, begitu pula game galaxio yang memiliki berbagai faktor randomness dan keberuntungan. Oleh karena itu, kami mengambil sampel dari percobaan sebanyak 10 kali. Hasil dari pengujian tersebut dapat dilihat sebagai berikut:

Autowin	Win	Win	Lose	Win	Lose	Win	Win	Lose	Win	Win
Bot Default	Lose	Lose	Win	Lose	Win	Lose	Lose	Win	Lose	Lose

Setelah melakukan ujicoba terhadap 3 bot default dari game, didapat winrate yang dihasilkan dari bot kami yaitu 70% dimana terjadi kekalahan 3 kali dan kemenangan 7 kali di 10 pertandingan. Kekalahan terjadi disebabkan oleh asumsi kapal secara greedy yang kurang tepat dengan yang sebenarnya.

Bab 5: Kesimpulan dan Saran

5.1. Kesimpulan

Pada akhirnya, kelompok kami berhasil mengimplementasikan algoritma greedy untuk bot dari permainan galaxio. Algoritma greedy menjadi hasil yang terbaik untuk digunakan karena dengan mencari optimum lokal pada tiap langkah menyebabkan pencarian optimum global semakin mudah. Jika dibandingkan dengan algorima brute force maka penggunaan algoritma ini lebih efisien dan efektif. Hal ini karena kita tidak bisa mengantisipasi seluruh langkah yang dapat diambil lawan sehingga pengimplementasian algoritma brute force sangat sulit.

Tentunya dalam menggunakan strategi greedy memerlukan teknik heuristik sehingga algoritma yang dibuat semakin optimal untuk meraih objektif. Penggunaan teknik ini memudahkan pencarian optimum lokal dari setiap kondisi yang ada agar dapat mencapai

kemenangan dalam permainan. Hal ini terbukti dari hasil uji dan implementasi pada bot yang dapat meraih hasil yang cukup baik.

5.2. Saran

Setelah menyelesaikan tugas ini, kami memiliki beberapa saran yang dapat membantu dalam pengerjaan tugas ini:

• Pembagian tugas dan scheduling yang lebih baik agar tugas dapat dikerjakan dengan lancar dan cepat

• Pembuatan laporan sebaiknya dilakukan lebih cepat dan bersamaan dengan pembuatan algoritma bot agar dapat selesai dengan lebih cepat

• Pengujian pada bot dapat ditingkatkan dengan menggunakan map yang statis dan tidak random, disertai dengan peletakan posisi bot yang tetap agar pengujian bot tidak banyak terpengaruh oleh faktor RNG

Link Penting

Link github disini: https://github.com/SulthanDA28/Tubes1_AutoWin.git

Link demo disini: Demo Tugas Besar 1 IF2211 - Strategi Algoritma

Daftar Pustaka

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

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag2.pdf

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