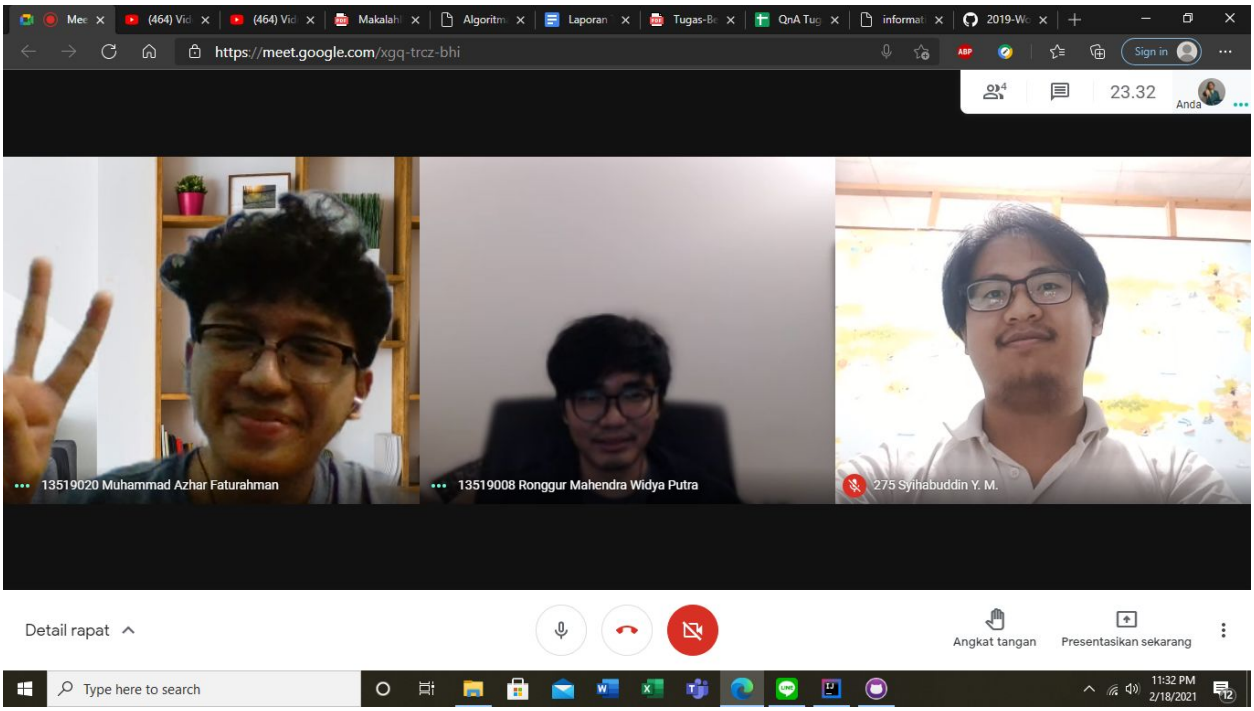
LAPORAN TUGAS BESAR I
IF2211 STRATEGI ALGORITMA

Semester II Tahun 2020/2021

Pemanfaatan Algoritma Greedy dalam Aplikasi Permainan “Worms”



Disusun Oleh :

Ronggur Mahendra Widya Putra (13519008)

Muhammad Azhar Faturahman (13519020)

Syihabuddin Yahya Muhammad (13519149)

Program Studi Teknik Informatika

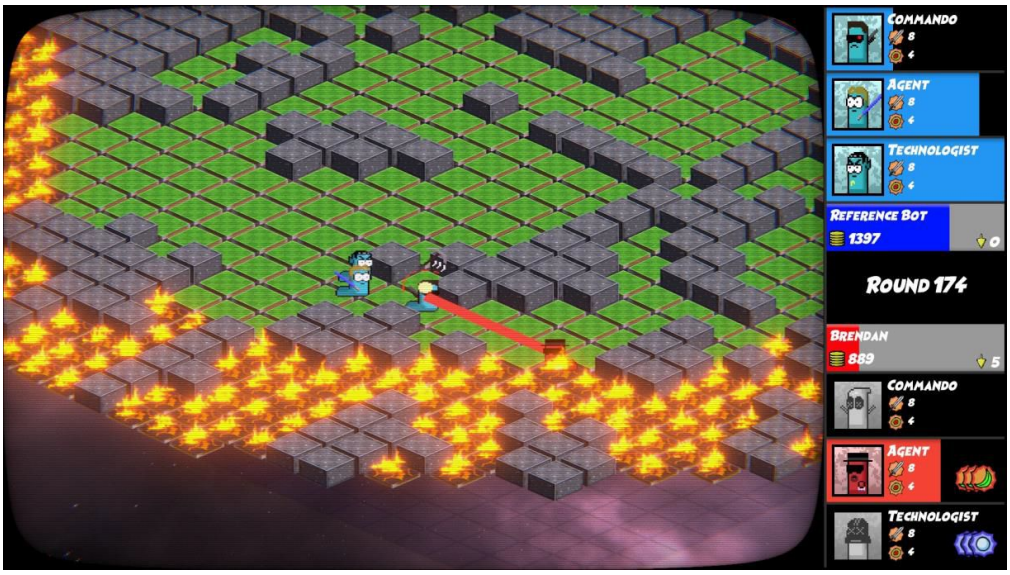
Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

BAB I

DESKRIPSI TUGAS

Worms adalah sebuah turned-based game yang memerlukan strategi untuk memenangkannya. Setiap pemain akan memiliki 3 worms dengan perannya masing-masing. Pemain dinyatakan menang jika ia berhasil bertahan hingga akhir permainan dengan cara mengeliminasi pasukan worms lawan menggunakan strategi tertentu.



Tugas besar kali ini bertujuan membuat sebuah bot terbaik untuk bermain permainan Worms dengan menggunakan algoritma greedy. Strategi greedy yang diimplementasikan harus dikaitkan dengan fungsi objektif dari permainan itu sendiri, yaitu memenangkan permainan dengan cara mengeliminasi seluruh worms lawan dengan senjata dan skill yang sudah disediakan dalam permainan. Implementasi pemain harus dapat dijalankan pada game engine yang telah disebutkan pada spesifikasi tugas besar, serta dapat dikompetisikan dengan pemain dari kelompok lain.

BAB II

LANDASAN TEORI

A. Algoritma Greedy

Algoritma Greedy adalah algoritma yang membentuk solusi langkah per langkah dengan mencari solusi terbaik lokal pada setiap langkahnya. Pada kebanyakan kasus, algoritma greedy tidak menghasilkan solusi yang paling optimal, namun menemukan solusi yang mendekati optimum dengan waktu yang cukup cepat. Algoritma Greedy memiliki prinsip "take what you can get now" artinya adalah algoritma akan mengambil solusi yang optimal pada langkah tersebut tanpa mempertimbangkan konsekuensinya pada langkah berikutnya.

Elemen elemen algoritma greedy :

1. Himpunan kandidat, C : berisi kandidat yang akan dipilih pada setiap Langkah (misal : simpul sisi di dalam graf , job, task, koin , benda , karakter , dsb).
2. Himpunan solusi, S : berisi kandidat yang sudah dipilih.
3. Fungsi solusi: menentukan apakah himpunan kandidat yang dipilih sudah memberikan solusi.
4. Fungsi seleksi selection function): memilih kandidat berdasarkan strategi greedy tertentu . Strategi greedy ini bersifat heuristik.
5. Fungsi kelayakan feasible): memeriksa apakah kandidat yang dipilih dapat dimasukkan ke dalam himpunan solusi (layak atau tidak).
6. Fungsi obyektif : memaksimumkan atau meminimumkan

Dengan menggunakan elemen elemen di atas, maka dapat dikatakan bahwa Algoritma greedy melibatkan pencarian sebuah himpunan bagian, S, dari himpunan kandidat, C; yang dalam hal ini, S harus memenuhi beberapa kriteria yang ditentukan, yaitu S menyatakan suatu solusi dan S dioptimisasi oleh fungsi obyektif.

B. Pemainan Worms

Dalam pertandingan game Worms terdapat 2 pemain yang masing-masing memiliki 3 cacing akan bermain melawan satu dengan yang lain. Tujuanya adalah menjadi pemain dengan cacing terakhir yang hidup.

Permainan ini dimainkan di peta dengan ukuran 33x33 sel. Ada 4 jenis sel yaitu :

1. AIR

   Pada sel AIR worm bisa dengan bebas pergi dan menembak melalui sel AIR.

2. DIRT

   Pada sel DIRT worm dibutuhkan untuk melakukan suatu perintah DIG agar sel DIRT menjadi sel AIR. Selagi sel ini masih sel DIRT worm tidak dapat pergi ke sel ini maupun menembak melalui sel tersebut.

3. DEEP_SPACE

   Pada sel ini worm tidak dapat pergi ke sel ini maupun menembak melalui sel ini.

4. LAVA

   Worm bisa menembah melalui sel LAVA serta pergi dengan bebas melalui sel LAVA namun worm yang berada di sel LAVA akan mendapatkan pengurangan nyawa.

Pada sel juga bisa terdapat POWERUP, POWERUP dapat diambil worm ketika pergi ke sel tersebut. POWERUP akan memulihkan 10 nyawa bagi worm yang mengambilnya.

Pada setiap babak, setiap pemain dapat memberikan 1 perintah untuk worm yang aktif. Worm yang aktif akan ditentukan oleh game. Semua perintah akan divalidasi oleh game sebelum dijalankan dan perintah yang tidak valid tidak akan melakukan apa apa dan akan mengurangi poin pemain tersebut. Perintah kedua pemain dijalankan pada waktu yang sama(dalam 1 giliran), dan tidak secara berurutan. Terdapat 7 perintah dengan spesifikasi seperti berikut :

1. MOVE

   Format untuk perintah MOVE adalah : move x y , dengan spesifikasi seperti berikut :

   - x dan y adalah koordinat sel yang akan dituju.
   - Worm dapat pergi ke semua sel AIR yang bersebelahan dengan sel worm tersebut.
   - Worm tidak dapat MOVE ke cell yang dihuni worm lain.
   - Worm tidak dapat MOVE ke sel DIRT atau DEEP SPACE.

- Jika 2 worm MOVE ke sel yang sama maka kedua worm akan menerima pengurangan nyawa dan akan mendapatkan kesempatan untuk bertukaran tempat atau diam ditempat.

2. DIG

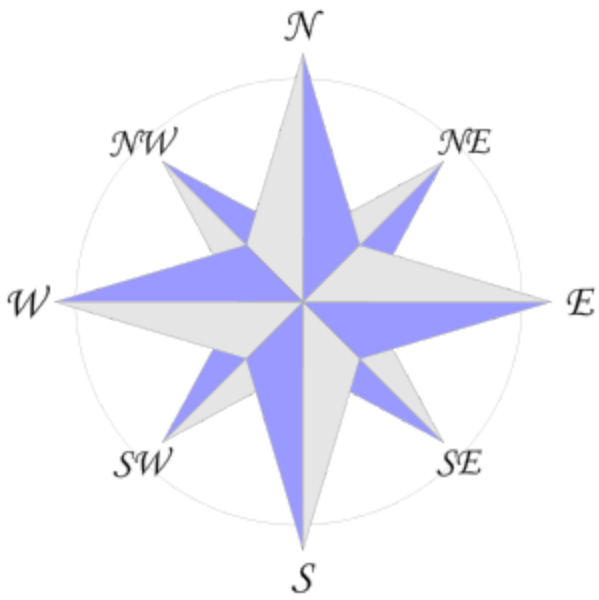
Format untuk perintah DIG adalah : dig x y , dengan spesifikasi seperti berikut:

- x dan y adalah koordinat sel yang akan di DIG.
- Worm dapat men DIG ke semua sel DIRT yang bersebelahan dengan sel worm tersebut.
- Men DIG suatu sel akan mengubah sel tersebut menjadi AIR.
- 2 worm men DIG sel yang sama adalah perintah yang valid.

3. SHOOT

Format untuk perintah SHOOT adalah : dig {arah} , dengan spesifikasi seperti berikut:

- {arah} adalah salah satu dari delapan arah mata angin : N (Utara), NE (Timur Laut), E (Timur), SE (Tenggara), S (Selatan), SW (Barat Daya), W (Barat), NW (Barat Laut)



- Jarak tembak dihitung dari euclidean distance. Untuk menghitung apakah suatu sel bisa ditembak, hitung euclidean distance dari posisi worm,

bulatkan ke bilangan bulat terdekat (floor), dan periksa apakah kurang dari sama dengan jarak maksimal tembak.

- Tembakan tidak menembus sel DIRT atau DEEP_SPACE
- Worm pertama yang berada dalam jangkauan tembakan akan kehilangan nyawa sesuai dengan settingan game pada game-config.json
- Jika nyawa worm mencapai 0, maka worm tersebut akan mati dan hilang dari map.
- Worm dapat menembak worm temannya.

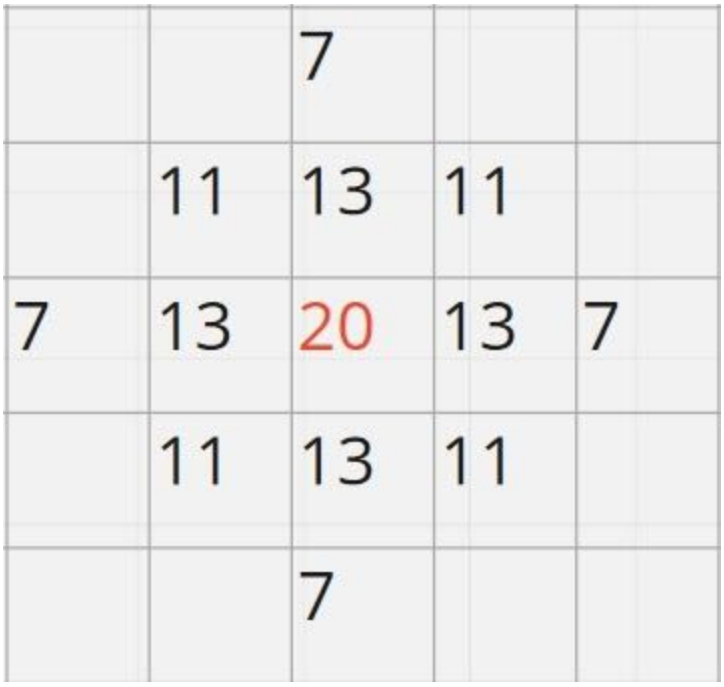
4. DO NOTHING

   Format untuk perintah DO NOTHING adalah : nothing. Player tidak akan melakukan apa apa ketika tidak melakukan apa apa. Perintah yang tidak valid dianggap sebagai perintah DO NOTHING dan jika suatu player melakukan 12 putaran DO NOTHING maka akan dianggap tidak valid dan didiskualifikasi dari game.

5. BANANA BOMB

   Format untuk perintah BANANA BOMB adalah : banana x y , dengan spesifikasi seperti berikut:

   - x dan y adalah koordinat sel yang akan di lempar BANANA BOMB.
   - Jarak lempar dihitung dari euclidean distance. Untuk menghitung apakah suatu sel bisa lempar, hitung euclidean distance dari posisi worm, bulatkan ke bilangan bulat terdekat (floor), dan periksa apakah kurang dari sama dengan 5.
   - BANANA BOMB dapat dilempar melewati dan pada sel DIRT.
   - Jika BANANA BOMB dilempar ke sel DEEP_SPACE maka BANANA BOMB akan hilang.
   - BANANA BOMB memiliki radius 2
   - Radius dan *damage* BANANA BOMB dapat direpresentasikan seperti berikut



- BANANA BOMB akan mengubah semua sel DIRT pada radiusnya menjadi sel AIR
- BANANA BOMB akan menghancurkan POWERUP pada radiusnya.
- BANANA BOMB dapat men *damage* worm teman.

6. SNOWBALL

   Format untuk perintah SNOWBALL adalah : snowball x y , dengan spesifikasi seperti berikut:

   - x dan y adalah koordinat sel yang akan di lempar SNOWBALL.
   - Jarak lempar dihitung dari euclidean distance. Untuk menghitung apakah suatu sel bisa lempar, hitung euclidean distance dari posisi worm, bulatkan ke bilangan bulat terdekat (floor), dan periksa apakah kurang dari sama dengan 5.
   - SNOWBALL dapat dilempar melewati dan pada sel DIRT.
   - Jika SNOWBALL dilempar ke sel DEEP_SPACE maka SNOWBALL akan hilang.
   - SNOWBALL memiliki radius 1
   - Semua worm yang ada pada radius SNOWBALL akan mendapatkan efek *FROZEN* untuk 5 ronde

- Semua worm yang memiliki efek *FROZEN* tidak akan merespon ke perintah namun tidak akan mendapatkan pengurangan poin dan bsia dihindari dengan menggunakan perintah SELECT
- SNOWBALL tidak menghancurkan POWERUP
- SNOWBALL dapat memberikan efek *FROZEN* pada worm teman.

7. SELECT

   Format untuk perintah SNOWBALL adalah : select {id worm} ; {perintah} , dengan spesifikasi seperti berikut:

   - {id worm} adalah id worm yang ingin kamu pilih

     * kamu hanya bisa memilih worm yang masih hidup
   - {perintah} adalah perintah yang ingin kamu lakukan pada worm yang telah kamu pilih.
   - Perintah select akan mengubah active worm menjadi id worm yang telah kamu pilih

Perintah - perintah tersebut dieksekusi dengan urutan berikut

1. SELECT
2. MOVEMENT
3. DIGGING
4. BANANA
5. SHOOTING
6. SNOWBALL

Semua worm memiliki peran, yang akan memberikan atribut dan senjata khusus. Cacing bisa mendapatkan satu dari profesi - profesi berikut:

1. Commando
   - Health : 150
   - Bisa menggunakan senjata dasar untuk menembak
2. Agent
   - Health : 100

- Bisa menggunakan senjata dasar untuk menembak
- Bisa menggunakan BANANA BOMB

3. Technologist
    - Health : 100
    - Bisa menggunakan senjata dasar untuk menembak
    - Bisa menggunakan SNOWBALL

Skor akan dipertimbangkan jika game seri:

- Jumlah ronde maksimal adalah 400 ronde
- Jika kedua worm mati dalam ronde yang sama

Jumlah skor diitung dari :

- Membunuh worm musuh memberikan 40 poin
- Men *damage* worm lain memberikan poin sebanyak *damage* yang dilakukan dikali 2
- Semua poin diatas akan mengurangi skor jika dilakukan ke worm sendiri
- Attack yang tidak kena memberikan 2 poin
- Bergerak memberikan 5 poin
- Men DIG memberikan 7 poin
- Memberikan efek FROZEN memberikan 17 poin
- Perintah DO NOTHING memberikan 0 poin
- Perintah yang tidak valid mengurangi poin dengan 4

Mulai dari ronde 100 akan ada sel LAVA yang datang dari luar map ke tengah map. Pada ronde 350 lava akan mengisi semua map kecuali 1 area kecil di tengah map dengan radius 4. Semua worm yang mengakhiri ronde di sel LAVA akan menerima *damage* sebesar 3 per ronde.

BAB III

PEMANFAATAN ALGORITMA GREEDY

A. Elemen - elemen Algoritma greedy pada permainan WORM

Dalam permainan WORM, setiap pemain akan memiliki 3 worm dimana setiap worm memiliki ID, tipe, dan senjatanya masing masing. Tujuan dari permainan ini adalah membunuh semua cacing lawan atau mendapatkan skor tertinggi jika permainan berakhir dengan seri. Dalam setiap ronde, pemain dapat memberikan perintah kepada cacing yang terpilih atau menggunakan selection token untuk memilih cacing lain. Dari sini dapat kita simpulkan bahwa elemen greedy permainan WORM sebagai berikut.

a. Himpunan Kandidat :
   Himpunan perintah yang dapat dilakukan meliputi move, dig, shoot, do nothing, banana bomb, snowball, dan select.
b. Himpunan Solusi :
   Himpunan perintah yang terpilih dan akan dilaksanakan
c. Fungsi Solusi :
   Memeriksa perintah yang terpilih merupakan 1 perintah bukan select disertai ataupun tidak disertai perintah select.
d. Fungsi Seleksi :
   Memilih perintah terbaik berdasarkan ronde, map, ID cacing yang bermain, cacing pemain, dan cacing lawan.
e. Fungsi Kelayakan :
   Memeriksa apakah perintah yang dipilih dapat dilakukan(e.g: perintah select ketika selection token > 0)
f. Fungsi Objektif :
   Membunuh semua cacing musuh dan/atau mendapatkan skor terbesar.

B. Solusi greedy yang dapat diimplementasikan

Terdapat banyak hal yang dapat dipertimbangkan pada sistem greedy yang bisa dibuat, dalam bagian ini akan dijelaskan greedy berdasarkan aspek yang ada beserta fungsi seleksinya.

a. Greedy by health

   Pada algoritma ini, dalam memilih perintah terbaik, akan dipertimbangkan keadaan cacing musuh, kemudian akan dipilih perintah untuk mengejar dan membunuh worm musuh yang memiliki health paling rendah.

   Algoritma ini akan mencek semua worm yang dimiliki oleh musuh, dan mencari worm musuh yang memiliki darah terendah. Oleh karena itu efisiensi algoritma ini adalah O(n) untuk mengiterasi list worm lawan.

   Efektivitas dari algoritma ini tidak terlalu baik terutama saat di awal pemain yang mana semua worm musuh memiliki darah yang cenderung sama dan besar.

b. Greedy by position

   Algoritma ini akan memilih perintah terbaik dengan mencari posisi musuh yang paling dekat dengan worm yang sedang digerakkan pada saat ini. Posisi musuh didapatkan dari daftar worm yang dimiliki oleh musuh, kemudian jarak worm musuh dengan worm pemain sekarang dihitung dengan *euclidean distance* dan dipilih worm dengan *distance* terkecil.

   Efisiensi dari algoritma ini adalah O(n) karena harus mengiterasi list worm yang dimiliki oleh musuh, lalu menghitung *euclidean distance*, dan memilih worm musuh dengan *distance* terkecil.

   Efektivitas algoritma ini cukup baik dengan pertimbangan terdapat dua worm pemain yang mengarah ke satu worm musuh.

c. Greedy by score

   Algoritma ini akan memilih perintah yang memberikan score terbanyak baik itu dari mendamage musuh, atau move dan dig.Algoritma ini akan semua kemungkinan aksi yang memberikan poin dan akan melakukan aksi yang memberikan poin terbanyak. Algoritma ini memiliki efisiensi O(1) dimana n adalah perintah yang valid karena Algoritma ini selalu menghitung semua perintah yang valid dan perintah yang valid selalu konstan.

Dari percobaan yang kami lakukan, kami menyimpulkan bahwa algoritma ini tidak efektif karena permainan sangatlah jarang mencapai ronde 400 atau berakhir dengan seri.

d. Greedy by profession

Pada algoritma ini, semua worm akan menargetkan untuk membunuh worm musuh yang memiliki profesi tertentu baik itu *Commander*, *Agent* ataupun *Technologist.* Algoritma ini akan berusaha untuk membunuh worm musuh dengan prioritas yang paling tinggi. Algoritma ini memiliki kompleksitas O(n,m) = nlogn * m dimana n adalah besar map, dan m adalah jumlah musuh, karena algortma ini menggunakan algoritma dijkstra untuk setiap musuh.

e. Greedy by damage

Di Algoritma ini, akan dipilih perintah terbaik dengan cara menghitung *damage* terbesar kepada musuh yang dapat dilakukan oleh worm saat ini, menggunakan semua senjata seperti tembakan, serangan banana bomb, atau snow ball.

Efisiensi dari algoritma ini bergantung pada senjata yang dimiliki oleh worm sekarang, sehingga kompleksitas algoritma berada di sekitar O(n) dengan n banyaknya senjata yang dimiliki worm.

Efektivitas algoritma bergantung dari banyaknya musuh, semakin banyak musuh yang berkumpul di satu tempat akan membuat damage yang diberikan banana bomb lebih besar dan lebih efektif dalam memberikan serangan.

f. Greedy by regroup

Algoritma ini akan memilih perintah terbaik dengan prioritas berkumpul terlebih dahulu. Jika semua cacing sudah berkumpul, maka ketiga cacing akan menyerang cacing musuh bersama sama. Cacing yang diserang dapat ditentukan melalui health ataupun posisi.

Pada algoritma ini, digunakan algoritma dijkstra untuk berkumpul dengan efisien sehingga efisiensinya adalah O(nlog(n)) dimana n merupakan ukuran luas map.

C. Solusi greedy terbaik

Dari seluruh solusi greedy yang ada, kelompok kami memilih untuk menggunakan algoritma greedy by ganking commander and regroup. Pada strategi ini, cacing agent dan technologist akan menyerang commander lawan, sedangkan cacing commander akan berusaha untuk berkumpul dengan agent dan technologist. Jika cacing commander lawan sudah mati, maka semua cacing yang tersisa akan memburu cacing lawan yang paling dekat.

Hal yang menjadi pertimbangan adalah dalam permainan ini jika kedua pemain memiliki total health yang sama, sedangkan jumlah cacing yang berbeda, maka pemain dengan jumlah cacing yang lebih sedikit akan lebih unggul. Oleh karena itu, keberadaan cacing commander yang memiliki health paling besar merupakan ancaman yang sangat besar. Selain itu, walaupun kompleksitas algoritma ini cukup besar, dalam permainan worm kompleksitas algoritma tidak terlalu berpengaruh.

Algoritma yang digunakan dalam mengimplementasikan strategi greedy yang dipilih adalah sebagai berikut

1. HuntAndKill
   Algoritma untuk mencari rute terdekat ke posisi dimana worm tersebut bisa menembak worm musuh.
2. Regroup
   Algoritma untuk mengumpulkan semua worm teman di suatu area
3. BasicShoot
   Algoritma ini digunakan untuk menembak musuh.
4. Retreat
   Algoritma ini digunakan untuk menjauh jadi musuh selagi keluar dari garis tembak worm musuh.
5. Positioning
   Algoritma ini digunakan untuk mencari posisi yang bagus pada saat melawan musuh.
6. DigAndMoveTo

Algoritma ini mencari rute terbaik untuk pergi ke suatu tempat.

7. getPowerUp

   Algoritma ini digunakan untuk pergi ke power up.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

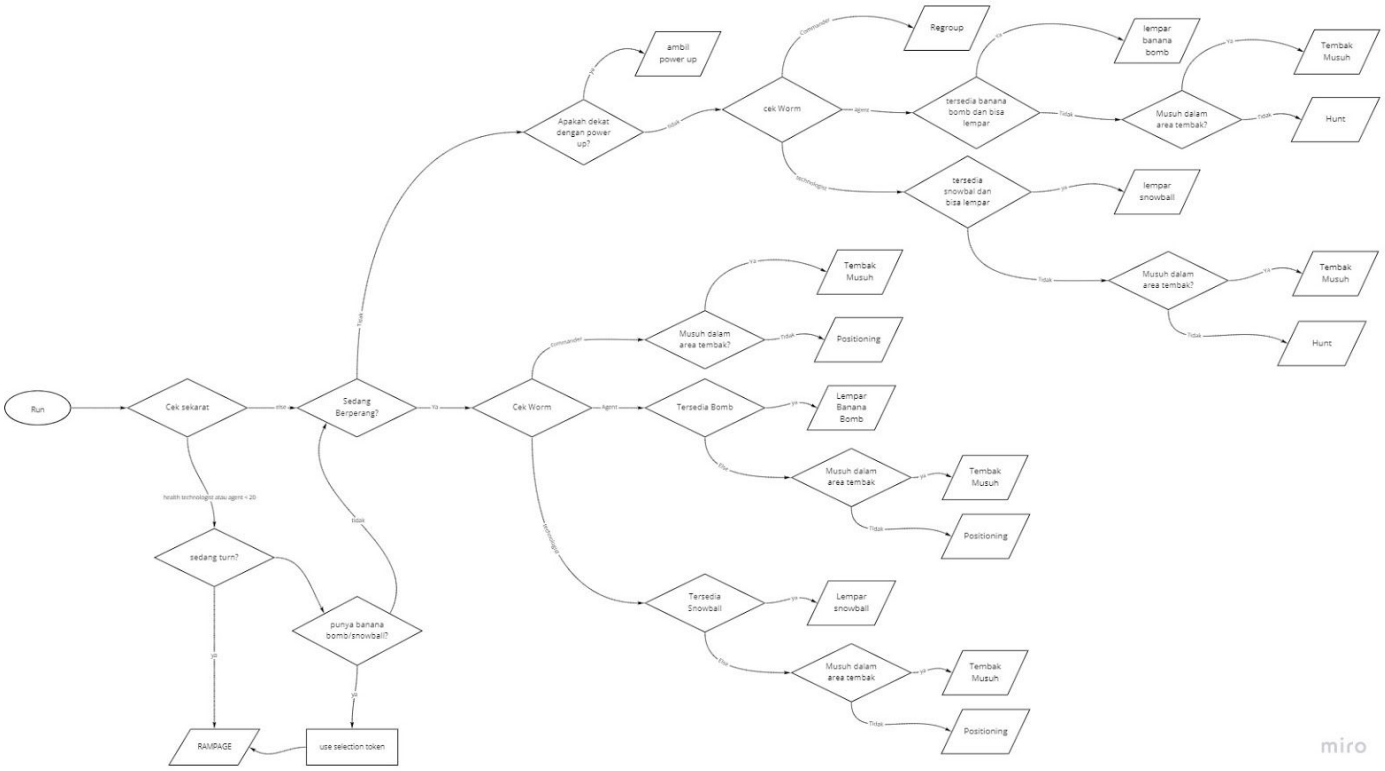
A. Pseudo Code

```
function run(gameState : GameState) -> Himpunan_Solusi
{ Mengembalikan Command terbaik untuk dilakukan bergantung pada
  Command dan GameState saat ini}

        Deklarasi
        cmd : Himpunan_Solusi

        Algoritma
        if (WormSekarangSekarat dan TidakBeku) then
            if (wormId = 2 and tersediaBananaBomb) then
                cmd <- lemparBananaBomb
            if (wormId = 3 and tersediaSnowballs) then
                cmd <- lemparSnowballs
        if (WormLainSekarat dan TidakBeku) then
            if (wormLainId = 2 and tersediaBananaBomb) then
                cmd <- selectWormIniDanLemparBananaBomb
            if (wormLainId = 3 and tersediaSnowballs) then
                cmd <- selectWormIniDanLemparSnowballs


        if (SedangBerperang) then
            if (wormId = 1) then
                if (musuhDalamAreaTembak) then
                    cmd <- tembakMusuh
                else
                    cmd <- positioning
            if (wormId = 2) then
                if (tersediaBananaBomb) then
                    cmd <- lemparBananaBomb
                else
                    if (musuhDalamAreaTembak) then
                        cmd <- tembakMusuh
                    else
                        cmd <- positioning
```

```
            if (wormId = 3) then
                if (tersediaSnowballs) then
                    cmd <- lemparStersediaSnowballs
                else
                    if (musuhDalamAreaTembak) then
                        cmd <- tembakMusuh
                    else
                        cmd <- positioning
```

```
else    { tidak berperang }
    if (adaPowerUpDiDekatWorm) then
        cmd <- ambilPowerUp

    if (wormId = 2 or wormId = 3) then
        if (wormId = 2 and tersediaBananaBomb) then
            cmd <- lemparBananaBomb
        if (wormId = 3 and tersediaSnowballs) then
            cmd <- lemparSnowballs

        if (musuhDalamAreaTembak) then
            cmd <- tembakMusuh
        if (musuhHuntMasihHidup)
            cmd <- hunt(CommanderMusuh)

        if (wormPemainBerkumpul) then
            if (wormId = 2 and tersediaBananaBomb) then
                cmd <- lemparBananaBomb
            else if (wormId = 3 and tersediaSnowballs) then
                cmd <- lemparSnowballs
            else
                cmd <- huntAndKill

        else
                cmd <- regroup

    else {wormId = 1}
        cmd <- regroup
```

B. FlowChart



C. Struktur Data

Pada implementasi algoritma kami, digunakan beberapa struktur data dengan spesifikasi sebagai berikut :

- Gamestate

  Struktur data Gamestate yang menyimpan semua data yang mewakili keadaan game diantaranya adalah urutan ronde, ronde maksimal permainan, ukuran map, worm yang aktif, urutan doNothing data MyPlayer, data Opponent, dan Map.

- MyPlayer

  Struktur data MyPlayer menyimpan data-data player diantaranya adalah Id Player, *score* player, health total worm, Worm milik player, dan jumlah selection token

- MyWorm

  Struktur data MyWorm terdiri atas id untuk menyimpan identitas pemain, score untuk menyimpan skor pemainan, health untuk menampilkan total darah/nyawa yang dimiliki pemain, worms untuk menyimpan semua worm, dan token untuk menggunakan selection token.

- Opponent

  Struktur data Opponent menyimpan data-data musuh diantaranya adalah Id musuh, *score* musuh, dan Worm milik musuh.

- Bomb

  Struktur data Bomb menyimpan atribut yang dimiliki bom, seperti jumlah dan jangkauan.

- PairBomb

  Struktur data PairBomb menyimpan dua buah nilai yaitu position dan integer. Struktur data ini digunakan dalam pencarian posisi pelemparan bom yang paling ideal. Position mewakili posisi pelemparan bom paling ideal, sementara integernya mewakili damage yang akan terjadi jika bom dilempar ke position.

- Position

  Position terdiri atas koordinat dari suatu entitas pada map, terdiri atas x untuk menyatakan basis dan y untuk menyatakan ordinat.

- PowerUp

  Struktur data ini berguna untuk menyatakan *power up* yang tersedia pada yang tersedia pada pemainan. Terdiri atas PowerUpType yang menyatakan tipe *power up* dan value yang menyatakan besaran dari *power up* tersebut.

- Cell

  Cell adalah suatu struktur data yang ada pada setiap node pada *map.* Didalamnya terdapat koordinat X *cell*, koordinat Y *cell*, type *cell*, dan *powerUp* yang terdapat pada *cell* tersebut.

- ModifiedCell

  ModifiedCell adalah suatu struktur data yang dibuat untuk memenuhi kebutuhan algoritma path planning didalamnya terdapat *cell* dari koordinat itu

sendiri, *cell* dari koordinat sebelum sel ini, suatu boolean yang menandai apakah ModifiedCell ini sudah di iterasi, dan jarak minimum ke *cell* ini.

- CellType

  CellType adalah struktur data yang terdapat di dalam *cell*. CellType bisa memiliki 4 jenis tipe yaitu *AIR, DIRT, DEEP_SPACE*, dan *LAVA*.

- Direction

  Direction adalah Struktur data yang digunakan pada perintah Shoot. Direction bisa memiliki 8 jenis arah mata angin yaitu N (Utara), NE (Timur Laut), E (Timur), SE (Tenggara), S (Selatan), SW (Barat Daya), W (Barat), NW (Barat Laut).

- PowerUpType

  Struktur data ini merupakan kelanjutan dari struktur data PowerUp. PowerUp yang tersedia hanya *HEALTH_PACK*.

- Command

  Struktur data ini merupakan induk dari semua struktur data yang mewakili perintah yang akan dijalankan.

- DigCommand

  Struktur data ini merupakan child dari class Command, Terdapat atribut X dan Y yang menyatakan koordinat yang akan di dig oleh worm yang aktif.

- DoNothingCommand

  Struktur data ini merupakan child dari kelas Command. Fungsi dari kelas ini adalah memberikan perintah *do nothing* untuk menghindari hukuman karena perintah yang salah.

- MoveCommand

  Struktur data ini merupakan child dari class Command, Terdapat atribut X dan Y yang menyatakan koordinat yang akan dituju oleh worm yang aktif.

- ShootCommand

  Struktur data ini merupakan child dari class Command, Terdapat atribut Direction yang menyatakan arah dari Shoot yang akan dilakukan oleh worm yang aktif.

- ThrowBananaCommand

ThrowBananaCommand merupakan child dari class Command yang berfungsi untuk melempar *Banana Bomb* pada lokasi tertentu. Terdapat atribut x dan y yang menyatakan koordinat untuk melempar *Banana Bomb*.

- ThrowSnowballCommand

  Struktur data ini merupakan child dari class Command. Kelas ini berfingsi untuk memberikan perintah melempar *snowball* pada lokasi tertentu. Terdapat atribut x dan y yang menyatakan koordinat untuk melempar snowball.

- UseToken

  Struktur data ini merupakan child dari *class* Command. Terdapat atribut ID yang menyatakan ID dari worm yang akan dipilih dan cmd yaitu perintah yang akan dilakukan oleh worm tersebut.

D. Analisis

Dalam pengujian algoritma ini, kami menguji algoritma kami melawan reference bot yang sudah disediakan. Total pengujian yang dilakukan adalah 50 kali, dimana bot kami menang 39 kali dan kalah 11 kali. Dari angka tersebut didapat persentase kemenangan bot kami 78%. Dari 50 kasus tersebut nilai optimal didapat ketika agent dan technologist mencapai commander lawan di saat yang bersamaan. Semakin lama jeda antara technologist dan agent mencapai commander lawan, semakin buruk hasil yang didapat. Kasus terburuk terjadi ketika snowball technologist habis digunakan untuk melawan commander saja. Secara umum strategi yang kita gunakan sudah optimal untuk melawan bot lawan.

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

Algoritma Greedy merupakan algoritma yang sangat cocok digunakan untuk menyusun strategi kemenangan pada permainan WORM. pada bab 3 penulis telah menjelaskan 6 algoritma greedy yang bisa dipakai. Penulis menemukan bahwa kombinasi dari algoritma - algoritma greedy dasar tersebut merupakan solusi yang paling efektif untuk memenangkan permainan WORM.

B. Saran

Terkait dengan topik terkait, kami menyarankan beberapa hal untuk diperhatikan sebagai berikut:

- Dalam algoritma greedy yang dipakai, penulis menyarankan untuk memprioritaskan menyerang worm yang healthnya masih banyak daripada membunuh suatu worm, karena matinya suatu worm malah akan memperkuat pemain.
- Penulis juga menyarankan untuk membuat strategi khusus pada situasi 1 lawan 1 karena pada situasi tersebut penulis mengamati banyak potensi strategi yang efektif pada situasi tersebut
- Penulis juga mengamati bahwa potensi dari *technologist* pada tahap akhir permainan sangat lah besar sedangkan pada awal permainan sangat minim sehingga penulis menyarankan untuk memaksa *technologist* musuh menggunakan *snowball*-nya di awal permainan dan untuk menjaga *snowball* tim agar bisa digunakan pada akhir permainan.

DAFTAR PUSTAKA

Hart, P., Nilsson, N., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, *4*(2), 100–107. https://doi.org/10.1109/tssc.1968.300136

Jungnickel D. (1999) The Greedy Algorithm. In: Graphs, Networks and Algorithms. Algorithms and Computation in Mathematics, vol 5. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-03822-2_5

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271.

Diakses pada tanggal 19 Februari 2021 https://github.com/EntelectChallenge/2019-Worms/blob/develop/game-engine/game-rules.md

Diakses pada tanggal 19 Februari 2021 https://docs.oracle.com/en/java/javase/15/