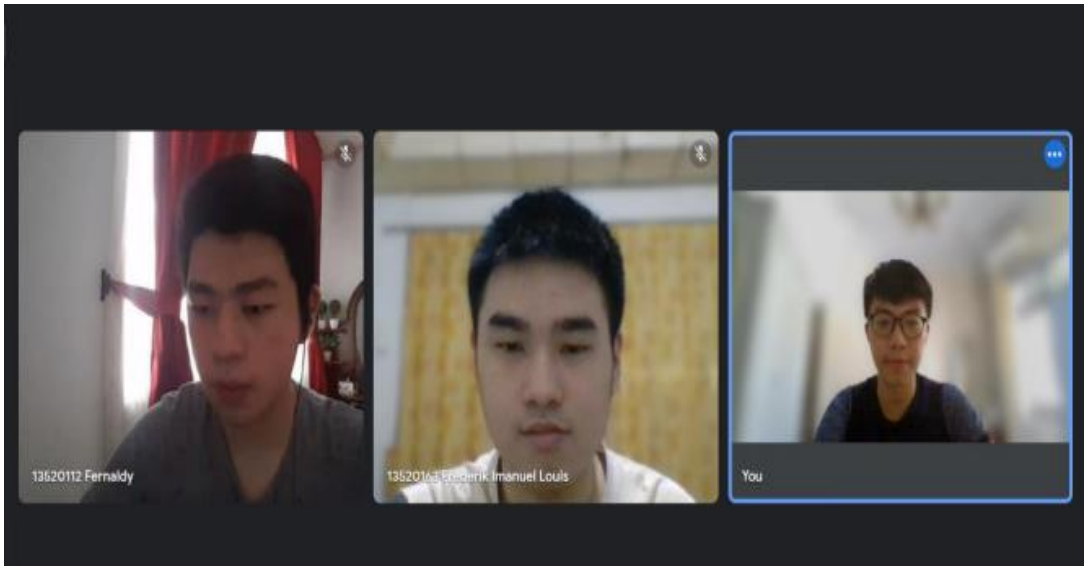
Pemanfaatan Algoritma *Greedy* dalam Aplikasi Permainan "Overdrive"

Dibuat Sebagai Tugas Besar 1

IF2211

Strategi Algoritma



Kelompok [Eurobeat Intensifies]

Louis Yanggara	**13520063**
Fernaldy	**13520112**
Frederik Imanuel Louis	**13520163**

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2022

BAB 1

DESKRIPSI TUGAS

Pada tugas besar kali ini, anda diminta untuk membuat sebuah bot untuk bermain permainan Overdrive yang telah dijelaskan sebelumnya. Untuk memulai, anda dapat mengikuti panduan singkat sebagai berikut.

1. Download latest release starter pack.zip dari tautan berikut https://github.com/EntelectChallenge/2020-Overdrive/releases/tag/2020.3.4
2. Untuk menjalankan permainan, kalian butuh beberapa requirement dasar sebagai berikut. a. Java (minimal Java 8): https://www.oracle.com/java/technologies/downloads/#java8 b. IntelIiJ IDEA: https://www.jetbrains.com/idea/ c. NodeJS: https://nodejs.org/en/download/
3. Untuk menjalankan permainan, kalian dapat membuka file “run.bat” (Untuk Windows dapat buka dengan double-click, Untuk Linux/Mac dapat menjalankan command “make run”).
4. Secara default, permainan akan dilakukan diantara reference bot (default-nya berbahasa Java) dan starter bot (default-nya berbahasa JavaScript) yang disediakan. Untuk mengubah hal tersebut, silahkan edit file “game-runner-config.json”. Anda juga dapat mengubah file “bot.json” dalam direktori “starter-bots” untuk mengatur informasi terkait bot anda.
5. Silahkan bersenang-senang dengan memodifikasi bot yang disediakan di starter-bots. Ingat bahwa bot kalian harus menggunakan bahasa Java dan di-build menggunakan IntelIiJ sebelum menjalankan permainan kembali. Dilarang menggunakan kode program yang sudah ada untuk pemainnya atau kode program lain yang diunduh dari Internet. Mahasiswa harus membuat program sendiri, tetapi belajar dari program yang sudah ada tidak dilarang.
6. (Optional) Anda dapat melihat hasil permainan dengan menggunakan visualizer berikut https://github.com/Affuta/overdrive-round-runner
7. Untuk referensi lebih lanjut, silahkan eksplorasi di https://github.com/EntelectChallenge/2020-Overdrive .

IF2211 Strategi Algoritma - Tugas Besar 1 3 Strategi greedy yang diimplementasikan tiap kelompok harus dikaitkan dengan fungsi objektif dari permainan itu sendiri, yaitu memenangkan permainan dengan cara mencapai garis finish lebih awal atau mencapai garis finish bersamaan tetapi dengan kecepatan lebih besar atau memiliki skor terbesar jika kedua komponen tersebut masih bernilai imbang. Salah satu contoh pendekatan greedy yang bisa digunakan (pendekatan tak terbatas pada contoh ini saja) adalah menggunakan powerups begitu ada untuk mengganggu mobil musuh. Buatlah strategi greedy terbaik, karena setiap bot dari masing-masing kelompok akan diadu satu sama lain dalam suatu kompetisi Tubes 1 (TBD).

Strategi greedy harus dijelaskan dan ditulis secara eksplisit pada laporan, karena akan diperiksa pada saat demo apakah strategi yang dituliskan sesuai dengan yang diimplementasikan. Tiap kelompok dapat menggunakan kreativitas mereka dalam menyusun strategi greedy untuk

memenangkan 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 2

LANDASAN TEORI

DASAR TEORI

Algoritma *greedy* adalah suatu algoritma yang memecahkan persoalan secara langkah per langkah (*step by step*) sedemikian hingga,

pada setiap langkah:

1. mengambil pilihan terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan (prinsip "*take what you can get now"*)
2. dan "berharap" bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global.

Elemen-elemen dalam 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*): memiliki kandidat berdasarkan strategi *greedy* tertentu. Strategi *greedy* ini bersifat heuristik
5. Fungsi kelayakan (*feasible*): memerika 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.

Penggunaan Algoritma *Greedy* tidak selalu menghasilkan solusi optimum (terbaik), bisa jadi solusi yang diperoleh merupakan solusi sub-optimum atau *pseudo*-optimum. Hal itu disebabkan Algoritma *greedy* tidak beroperasi secara menyeluruh terhadap semua kemungkinan solusi yang ada dan terdapat beberapa fungsi seleksi yang berbeda sehingga kita harus memilih fungsi yang tepat jika ingin algoritma menghasilkan solusi optimal.

Algoritma *greedy* digunakan jika solusi terbaik mutlak tidak terlalu diperlukan, daripada menggunakan algoritma yang kebutuhan waktunya eksponensial untuk menghasilkan solusi yang terbaik.

CARA KERJA PROGRAM

Overdrive adalah permainan dimana 2 bot akan diadu dalam sebuah permainan lintasan 4 jalur dimana akan ada berbagai *powerups* dan *obstacle* yang menjadi tantangan bagi kedua bot. Pada setiap ronde, kedua bot akan diberi kesempatan untuk melaksanakan sebuah *command* secara bersamaan. Pemenang dari permainan ini adalah bot yang pertama sampai ke garis finish. Dalam permainan ini ada berbagai *command* yang dapat digunakan oleh bot, yaitu:

- NOTHING: bot tidak melakukan apapun.
- ACCELERATE: meningkatkan kecepatan ke tingkatan selanjutnya.
- DECELERATE: menurunkan kecepatan ke tingkatan sebelumnya.
- TURN_LEFT: pindah 1 blok ke kiri.
- TURN_RIGHT: pindah 1 blok ke kanan.
- USE_BOOST: menggunakan *boost* yang telah didapatkan sebelumnya.
- USE_OIL: menempatkan minyak tepat dibawah mobil mengakibatkan kecepatan setiap mobil yang lewat(kecuali mobil sendiri) turun 1 tingkat.
- USE_LIZARD: mobil akan melompati *lizard* yang ada pada lintasan.
- USE_TWEET: memunculkan *cyber truck* di tempat tertentu, jika ada mobil yang tersangkut di belakang *cyber truck* maka tidak mobil tersebut tidak dapat bergerak hingga ronde berakhir dan kecepatan mobil berubah menjadi 3.
- USE_EMP: menembakkan EMP ke arah depan, jika mengenai lawan akan menghentikan mereka sepanjang ronde dan menurunkan kecepatan lawan menjadi 3.
- FIX: mengurangi 2 poin *damage* dari mobil.

Algoritma *greedy* akan diimplementasikan pada bot dengan tujuan untuk membuat pilihan yang paling efektif pada suatu ronde tanpa mempertimbangkan ronde-ronde selanjutnya. Namun untuk memilih *command* ada beberapa hal yang dapat dijadikan pertimbangan yaitu *obstacle* serta *powerups* berjarak 20 blok ke depan dan 5 blok ke belakang, selain itu kita juga memiliki informasi posisi lawan serta kecepatan lawan. Dengan menggunakan informasi tersebut, akan diambil pilihan yang terbaik pada ronde tersebut.

Untuk menjalankan *game engine*, pastikan dahulu konfigurasi bot yang digunakan sudah sesuai dengan yang diinginkan yaitu bot yang akan diuji dengan bot lain yang akan menjadi lawan. Setelah setelan bot sesuai maka cukup jalankan file run.

BAB 3

APLIKASI STRATEGI *GREEDY*

MAPPING PERSOALAN *OVERDRIVE* MENJADI ELEMEN *GREEDY*

1. Himpunan kandidat: pada setiap langkah, ada 12 *command* yang dapat dipilih sesuai dengan kondisi terbaik pada saat itu.
2. Himpunan solusi: *command-command* yang sudah terpilih hingga permainan berakhir.
3. Fungsi solusi: memeriksa apakah langkah-langkah yang sudah diambil dapat memberi kemenangan melawan *bot* lain.
4. Fungsi seleksi: Prioritaskan menghindari setiap *obstacle*, *fix* mobil, *boost*, menggunakan *power up* selain boost. Jika tidak ada hal yang dapat dilakukan pada ronde itu, maka *accelerate* saja.
5. Fungsi kelayakan: memeriksa apakah *command* yang dipilih tidak saling bertabrakan ataupun tidak sesuai dengan aturan permainan.
6. Fungsi obyektif: waktu tempuh (jumlah ronde) yang minimal serta *score* yang maksimum.

EKSPLORASI ALTERNATIF SOLUSI GREEDY

Dalam permainan Overdrive ada beberapa alternatif *greedy* yang dapat diambil yaitu:

1. *Greedy by speed*: Alternatif solusi ini akan selalu memprioritaskan penambahan kecepatan hingga ke *maximum state* jika sudah tidak dapat dilakukan percepatan/*accelerate* baru mempertimbangkan penggunaan *command* lain. Strategi ini mencoba untuk meminimalkan waktu yang diperlukan untuk mencapai garis *finish*.
2. *Greedy by score*: Alternatif solusi ini akan selalu memprioritaskan penggunaan *command* yang memberi penambahan *score.* Strategi ini mencoba untuk memaksimalkan *score* agar menang ketika keadaan berimbang(sampai ke garis *finish* bersama dengan lawan).
3. *Greedy by powerups*: Alternatif solusi ini akan selalu memprioritaskan penggunaan *powerups* yang diperoleh *bot*. Strategi ini mencoba untuk memanfaatkan setiap *powerups* yang ada dengan harapan untuk menghambat lawan.
4. *Greedy by damage*: Alternatif solusi ini akan selalu memprioritaskan penggunaan *command* fix jika *bot* sudah terkena damage. Strategi ini mencoba untuk mencapai garis *finish* lebih cepat dengan mengurangi penalti kecepatan.

ANALISIS EFISIENSI DAN EFEKTIVITAS DARI ALTERNATIF YANG DIRUMUSKAN

Setiap alternatif yang dirumuskan tentulah memiliki kelebihan dan kekurangan masing-masing, yaitu:

1. *Greedy by speed*: Menurut kami alternatif ini merupakan yang paling efektif dan efisien sesuai dengan tujuan dari permainan yaitu mencapai garis finish dengan secepat mungkin melalui alternatif ini *bot* akan selalu berusaha menambah kecepatan hingga maksimum dan mendahului lawan

2. *Greedy by score*: Menurut kami alternatif ini kurang efektif karena satu-satunya cara untuk *win by score* adalah jika kedua *bot* melewati garis *finish* secara bersamaan yang menurut kami sangat jarang terjadi.
3. *Greedy by powerups*: Menurut kami alternatif ini kurang efektif karena penggunaan *powerups* tidak selalu berguna untuk diri sendiri maupun merugikan lawan, malahan ada kemungkinan bahwa *powerups* yang digunakan malah menyebabkan pengurangan *score* karena tidak sesuai kondisi.
4. *Greedy by damage*: Menurut kami alternatif ini cukup efektif karena mengurangi penalti kecepatan yang diterima, namun alternatif ini menjadi kurang efektif jika *damage* yang diterima sudah melebihi 2 karena menyebabkan *bot* berhenti selama lebih dari satu *round* hanya untuk meningkatkan *speed* sebesar 1(dari 8 ke 9 pada *damage* 3 ke 2).

STRATEGI GREEDY YANG DIPILIH

Dari berbagai alternatif strategi greedy, kami memutuskan untuk mengutamakan *Greedy by speed*. Strategi tersebut dipilih akibat kondisi menang terlebih dahulu mempertimbangkan kecepatan mencapai garis finish, sehingga *Greedy by score* merupakan solusi yang kurang efektif. *Greedy by power up* juga kurang efektif karena mengumpulkan power up sebanyak mungkin tidak secara langsung mempercepat sampai ke garis finish. *Greedy by damage* dapat membantu mencapai garis finish lebih cepat dengan mengurangi penalti kecepatan, tetapi pada akhirnya, kami memutuskan bahwa *Greedy by speed* memiliki potensi yang lebih kuat untuk mencapai garis finish lebih cepat. Rincian prioritas strategi greedy yang diambil adalah sebagai berikut:

1. Jika mobil sedang boosting, prioritaskan menghindari obstacle untuk menjaga kecepatan:
    a. Jika jalur sekarang ada obstacle dan jalur kiri dan kanan tidak ada obstacle, pilih jalur kiri atau kanan yang memiliki power up lebih baik. Output Command: TURN_LEFT atau TURN_RIGHT
    b. Jika jalur sekarang ada obstacle dan jalur kiri atau kanan tidak ada obstacle, pilih jalur yang tidak ada obstacle. Output Command: TURN_LEFT atau TURN_RIGHT
    c. Jika jalur sekarang, kiri, dan kanan ada obstacle dan memiliki power up lizard, gunakan lizard. Output Command: USE_LIZARD
    d. Jika jalur sekarang tidak ada obstacle, lanjut ke prioritas berikutnya
2. Jika mobil sedang atau akan menabrak mobil musuh, dan player memiliki power up lizard, hunakan lizard. Output Command: USE_LIZARD.
3. Jika mobil memiliki damage lebih besar dari atau sama dengan tiga, perbaiki mobil agar memiliki kecepatan maksimum lebih baik. Output Command: FIX.
4. Jika mobil memiliki damage lebih besar dari atau sama dengan satu, dan player memiliki power up boost, perbaiki mobil untuk persiapan boost. Output Command: FIX.

5. Jika mobil memiliki damage nol, jalur tengah tidak memiliki obstacle untuk 15 block kedepan, player memiliki power up boost, gunakan boost. Output Command: USE_BOOST
6. Jika kecepatan mobil nol, tambah kecepatan. Output Command: ACCELERATE.
7. Jika mobil akan menabrak obstacle jika kecepatan ditambah
    a. Coba gunakan power up Tweet, EMP, atau Oil:
        i. Jika player memiliki power up EMP dan EMP akan mengenai posisi lawan sekarang, maka gunakan EMP. Output Command: USE_EMP.
        ii. Jika player memiliki power up Tweet dan kecepatan mobil lebih besar dari atau sama dengan enam:
            1. Jika lane lawan dan player berbeda, gunakan Tweet di lane lawan pada koordinat (opponent.lane, opponent.block+opponent.speed+3). Output Command: USE_TWEET Y X.
            2. Jika lane player dan lawan sama serta player tidak akan melewati koordinat (opponent.lane, opponent.block+opponent.speed+3), maka gunakan tweet. Output Command: USE_TWEET Y X.
        iii. Jika player memiliki power up Oil dan kecepatan mobil lebih besar dari atau sama dengan 9, jika kecepatan mobil lebih besar dari atau sama dengan 6 dan lawan berada tepat di belakang player, maka gunakan Oil. Output Command: Use_Oil.
    b. Jika lane kiri dan kanan tidak ada obstacle, maka pilih lane kiri atau kanan yang memiliki power up lebih optimal dengan prioritas: BOOST, LIZARD, EMP, TWEET, OIL. Output Command: TURN_LEFT atau TURN_RIGHT.
    c. Jika player tidak akan menabrak jika kecepatan tidak ditambah, maka tidak perlu melakukan apa-apa. Output Command: DO_NOTHING.
    d. Jika player tidak akan menabrak jika kecepatan dikurangi, maka kurangi kecepatan. Output Command: DECELERATE.
    e. Jika player memiliki power up Lizard, gunakan Lizard. Output Command: USE_LIZARD.
    f. Jika lane tengah memiliki tembok (wall), tetapi lane kiri atau kanan tidak memiliki tembok, pilih salah satu dari lane tersebut. Output Command: TURN_LEFT atau TURN_RIGHT.
8. Coba gunakan powerup Tweet, EMP, atau Oil seperti pada ketentuan (8.a.). Output Command: USE_TWEET, USE_EMP, atau USE_OIL.
9. Jika semua kondisi diatas gagal, tambah kecepatan mobil. Output Command: ACCELERATE.

BAB 4

IMPLEMENTASI DAN PENGUJIAN

PSEUDOCODE

procedure Overdrive()

{Menyelesaikan permainan overdrive secepat mungkin}

DEKLARASI

ALGORITMA

```
if (sedang boost) then
    if (lane sekarang tidak bersih dalam jangkauan boost) then
        if (lane kiri dan lane kanan bersih) then
            belok ke lane lain dengan mengutamakan lane yang ada power up
        else if (lane kanan bersih) then
            belok ke lane kanan
        else if (lane kiri bersih) then
            belok ke lane kiri
        else if (punya power up lizard) then
        p   akai lizard
        endif
    endif
else if (akan menabrak lawan dan punya power up lizard) then
    pakai lizard
else if (damage mobil >= 3) then
    fix
else if (damage mobil >= 1 dan punya power up boost) then
    fix
else if (damage mobil == 0 dan lane tengah bersih dalam jangkauan boost dan punya power up boost dan tidak sedang boost) then
    boost
```

else if (kecepatan == 0) then

accelerate

else if (lane sekarang bersih dengan kecepatan sekarang dan lane sekarang tidak bersih dengan kecepatan setelah accelerate) then

if (punya power up yang efektif apabila dipakai) then

pakai power up

endif

else if (lane sekarang tidak bersih dengan kecepatan setelah accelerate) then

if (lane kiri bersih dan lane kanan bersih) then

belok ke lane lain dengan mengutamakan lane yang ada power up

else if (lane sekarang bersih tanpa accelerate) then

do nothing

else if (lane kiri bersih) then

belok ke lane kiri

else if (lane kanan bersih) then

belok ke lane kanan

else if (speed > 6 dan lane sekarang bersih setelah decelerate) then

decelerate

else if (punya power up lizard) then

lizard

else if (tidak akan menerima damage >= 1 setelah accelerate dan damage sekarang < 2) then

accelerate

else if (akan menerima damage >= 1 setelah do nothing dan tidak akan menerima damage >= 1 di lane kiri dan kanan) then

belok ke lane lain dengan mengutamakan lane yang ada power up

else if (akan menerima damage >= 1 setelah do nothing dan tidak akan menerima damage >= 1 di lane kanan) then

belok ke lane kanan

else if (akan menerima damage >= 1 setelah do nothing dan tidak akan menerima damage >= 1 di lane kiri) then

belok ke lane kiri

endif

else if (punya power up yang efektif apabila dipakai) then

pakai power up

accelerate

endif

{Saat mencari lane dengan power up, prioritas power up adalah boost, lizard, emp, tweet, lalu oil}

{Power up EMP efektif apabila block mobil < block lawan dan selisih lane mobil dengan lane lawan <= 1}

{Power up Tweet efektif apabila speed mobil >= 6 dan mobil tidak akan menabrak posisi penempatan tweet dengan kecepatan sekarang}

{Posisi penempatan Tweet adalah block lawan + speed lawan + 3}

{Power up Oil efektif apabila (speed mobil 9 atau 15) atau (speed mobil >= 6 dan lawan akan menabrak mobil dengan kecepatannya sekarang)}

STRUKTUR DATA YANG DIGUNAKAN

Bot diimplimentasikan sebagai suatu kelas bot dengan atribut yang menyimpan state permainan sesuai spesifikasi *Overdrive*. Selain itu, kelas bot memiliki method public “run” yang digunakan untuk menjalankan bot, dimana bot diberikan game state round tersebut dan mengeluarkan command yang sesuai. Selain itu, terdapat beberapa method private dalam bot yang berfungsi membantu method “run” untuk melaksanakan tugasnya. Method tersebut antara lain “getBlocksInFront” yang mengeluarkan list of blocks sesuai dengan posisi yang diminta, “hasPowerUp” yang mengecek apakah suatu power up spesifik tersedia, “LaneClean” yang mengecek apakah suatu list of blocks dari satu lane tidak memiliki obstacle sama sekali, “Tankable” yang mengecek cost damage jika melalui suatu lane, “MudCount” yang mengecek jumlah mud dalam suatu list of blocks, “IsCrashing” yang mengecek apakah player akan menabrak opponent, “PowerGreed” yang secara greedy memiliki untuk belok ke kiri atau kanan berdasarkan power up yang terdapat pada lane tersebut, “TryPower” yang mencoba menggunakan power up berdasarkan strategi tertentu, “UsePower” yang mengeluarkan command power up sesuai input, serta “NextSpeed” dan “PrevSpeed” yang masing-masing menghitung kecepatan player jika melakukan Accelerate atau Decelerate.

Selain kelas bot, terdapat beberapa kelas lain yang berfungsi menyederhanakan proses pengembangan bot dengan mengenumerasi game state, command, map, ataupun power up dalam terminologi yang lebih mudah digunakan. Kelas yang mengenumerasi state permainan maupun map antara lain Direction, PowerUps, State, dan Terrain. Kelas yang menyimpan properti tiap entitas maupun state permainan antara lain Car, GameState, Lane, dan Position. Kelas yang mengeluarkan command sebagai output string antara lain AccelerateCommand, BoostCommand, ChangeLaneCommand, DecelerateCommand, DoNothingCommand, EmpCommand, FixCommand, LizardCommand, OilCommand, dan TweetCommand.

ANALISIS SOLUSI YANG DIGUNAKAN

Pada *starter-kit*, terdapat *basic bot* yang dapat digunakan sebagai perbandingan terhadap *bot* yang telah kami kembangkan. Setelah dicoba, kami mendapati kesimpulan bahwa solusi kami jauh lebih baik dibandingkan dengan *bot* yang sudah ada baik dari kecepatan untuk mencapai garis *finish* maupun dari jumlah *score* yang diperoleh sehingga menurut kami solusi yang dikembangkan sudah cukup efektif dan efisien untuk memperoleh hasil yang maksimal. Berikut ini adalah hasil dari pertandingan antara *bot* yang dikembangkan kami dengan *bot* awal yang sudah diberikan.

```
Completed round: 143
****************************************************************************************************
Game Complete
Checking if match is valid
======================================
The winner is: A - [Eurobeat Intensifies]

A - [Eurobeat Intensifies] - score:321 health:0
B - CoffeeRef - score:94 health:0

======================================
****************************************************************************************************
```

BAB 5

KESIMPULAN DAN SARAN

KESIMPULAN

Dari hasil percobaan, kami mengambil kesimpulan bahwa strategi *greedy by speed* sangat efektif karena sesuai dengan tujuan dari permainan yaitu mencapai garis *finish* dengan secepat mungkin. Algoritma yang digunakan juga cukup sederhana meskipun dengan cukup banyak *conditional* untuk mempertimbangkan kemungkinan serta kondisi yang ada pada suatu ronde.

SARAN

Untuk memperoleh hasil yang lebih optimal dan strategi yang terbaik, dapat dicoba setiap pendekatan dan diadu satu sama lain untuk mengetahui strategi mana yang terbaik. Namun hal tersebut memerlukan cukup banyak waktu untuk pengembangan algoritma dan pengujian.

DAFTAR PUSTAKA

"Algoritma Greedy (Bagian 1)" oleh Rinaldi Munir
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf

A. Levitin, *Introduction to the design & analysis of algorithms*. Boston: Pearson, 2012.

LINK GITHUB

https://github.com/fernaldy112/Tubes1_-Eurobeat-Intensifies-