LAPORAN TUGAS BESAR STRATEGI ALGORITMA (IF2211)

IMPLEMENTASI ALGORITMA GREEDY PADA BOT DIAMONDS

KELOMPOK 14:

MUHAMMAD ROYHAN A	**123140146**
DZAKY PRAMADHANI	**123140208**
SILVIA	**123140133**

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SUMATERA

2025

DAFTAR ISI

BAB I
DESKRIPSI TUGAS

1.1 Deskripsi Tugas

Diamonds merupakan suatu programming challenge yang mempertandingkan bot yang anda buat dengan bot dari para pemain lainnya. Setiap pemain akan memiliki sebuah bot dimana tujuan dari bot ini adalah mengumpulkan diamond sebanyak-banyaknya. Cara mengumpulkan diamond tersebut tidak akan sesederhana itu, tentunya akan terdapat berbagai rintangan yang akan membuat permainan ini menjadi lebih seru dan kompleks.

Untuk memenangkan pertandingan, setiap pemain harus mengimplementasikan strategi tertentu pada masing-masing bot-nya. Penjelasan lebih lanjut mengenai aturan permainan akan dijelaskan di bawah.Pada tugas pertama Strategi Algoritma ini, mahasiswa diminta untuk membuat sebuah bot yang nantinya akan dipertandingkan satu sama lain. Tentunya mahasiswa harus menggunakan strategi greedy dalam membuat bot ini.

1.2 Program Permainan Diamonds

1. Game engine, yang secara umum berisi:
    - Kode backend permainan, yang berisi logic permainan secara keseluruhan serta API yang disediakan untuk berkomunikasi dengan frontend dan program bot
    - Kode frontend permainan, yang berfungsi untuk memvisualisasikan permainan

2. Bot starter pack, yang secara umum berisi:
    - Program untuk memanggil API yang tersedia pada backend
    - Program bot logic (bagian ini yang akan kalian implementasikan dengan algoritma greedy untuk bot kelompok kalian)
    - Program utama (main) dan utilitas lainnya

1.3 Komponen-komponen permainan Diamonds

- **Diamonds**

  Untuk memenangkan pertandingan, kita harus mengumpulkan diamond ini sebanyak-banyaknya dengan melewati/melangkahinya. Terdapat 2 jenis diamond yaitu diamond biru dan diamond merah. Diamond merah bernilai 2 poin, sedangkan yang biru bernilai 1 poin. Diamond akan di-regenerate secara berkala dan rasio antara diamond merah dan biru ini akan berubah setiap regeneration.

- **Red Button/Diamond Button**

  Ketika red button ini dilewati/dilangkahi, semua diamond (termasuk red diamond) akan di-generate kembali pada board dengan posisi acak. Posisi red button ini juga akan berubah secara acak jika red button ini dilangkahi.

- **Teleporters**

  Terdapat 2 teleporter yang saling terhubung satu sama lain. Jika bot melewati sebuah teleporter maka bot akan berpindah menuju posisi teleporter yang lain.

- **Bots and Bases**

  Pada game ini kita akan menggerakkan bot untuk mendapatkan diamond sebanyak banyaknya. Semua bot memiliki sebuah Base dimana Base ini akan digunakan untuk menyimpan diamond yang sedang dibawa. Apabila diamond disimpan ke base, score bot akan bertambah senilai diamond yang dibawa dan inventory (akan dijelaskan di bawah) bot menjadi kosong.

- **Inventory**

  Bot memiliki inventory yang berfungsi sebagai tempat penyimpanan sementara diamond yang telah diambil. Inventory ini memiliki kapasitas maksimum sehingga sewaktu waktu bisa penuh. Agar inventory ini tidak penuh, bot bisa menyimpan isi inventory ke base agar inventory bisa kosong kembali.

BAB II
LANDASAN TEORI

2.1 Algoritma Greedy

Algoritma greedy merupakan teknik paling sederhana dan populer untuk menyelesaikan masalah optimasi dengan mencari solusi terbaik. Algoritma greedy terdiri dari dua jenis, yaitu maksimisasi (maximization) dan minimisasi (minimization). Prinsip algoritma greedy adalah "ambil apa yang bisa Anda dapatkan sekarang!" atau memilih hal terbaik yang dapat dicapai tanpa melihat konsekuensi di masa mendatang. Algoritma tersebut membangun solusi langkah demi langkah, dan pada setiap langkah, keputusan optimal harus dibuat.

Keputusan yang telah diambil tidak dapat dikembalikan lagi ke langkah terakhir. Dengan memilih optimum lokal pada setiap langkah, algoritma greedy berharap dapat mengarah pada solusi optimum global. Meskipun berupaya mencapai optimum pada suatu waktu, algoritma greedy tidak selalu berhasil memberikan solusi optimal untuk semua masalah, tetapi justru terkadang menghasilkan solusi sub-optimal.

2.2 Cara Kerja Program Bot

Program bot dalam permainan Diamonds dirancang untuk mengumpulkan diamond sebanyak-banyaknya dengan strategi berbasis algoritma greedy. Dan bot akan beroperasi secara dinamis pada peta permainan yang terdiri atas diamond, teleporter, red button, dan base. Dalam implementasinya, bot dikelola menggunakan algoritma greedy yang berfokus pada pengambilan keputusan optimal pada setiap langkah.

Untuk mencapai tujuan dalam mengumpulkan berlian dalam pengambilan keputusan kami mengimplementasikan strategi greedy bot mengikuti urutan prioritas berikut:

1. **Mengambil berlian bot lain**

   Prioritas utama bot adalah mengambil bot lain yang membawa berlian dan berada dalam jangkauan untuk di serang(tackle). Dan Jika inventory bot penuh maka segera kembali ke base untuk menyimpan diamond dan mengosongkan inventory.Jika inventory bot penuh, bot akan segera kembali ke base untuk menyimpan diamond dan mengosongkan inventory.

2. **Kabur jika ada bot lain mendekat**

   Apabila terdeteksi ada bot lawan yang mendekat dan berpotensi untuk menyerang, bot memprioritaskan untuk bergerak menjauh dari musuh dan kabur. Tujuan dari strategi ini adalah untuk melindungi berlian yang sudah terkumpul di inventori dan menjaga kelangsungan hidup bot.

3. **Ambil berlian manual**

   Bot akan fokus untuk mengambil berlian yang tersebar di board jika tidak ada peluang untuk menyerang bot lain atau mendapat ancaman dari bot lawan. Bot akan bergerak menuju berlian terdekat atau yang memiliki nilai lebih tinggi.

4. **Fokus balik dulu**

   Apabila bot telah mengumpulkan sejumlah berlian di inventori atau kapasitas inventori mendekati penuh dan prioritas selanjutnya kembali ke *home base*. Menyimpan berlian ke *base* akan menambah skor bot secara permanen dan mengosongkan inventori sehingga bot siap untuk mengumpulkan berlian kembali. Strategi ini memastikan bahwa poin yang dikumpulkan aman dan tidak hilang jika terjadi *tackle* oleh bot lawan.

BAB III
APLIKASI STRATEGI GREEDY

3.1 Mapping Persoalan Diamonds ke Elemen Algoritma Greedy

Untuk Mengimplementasikan algoritma greedy pada permainan diamonds ini dapat dipetakan ke dalam elemen-elemen dasar greedy seperti berikut:

1. Himpunan Kadidat

   Mencakup semua aksi yang dilakukan oleh bot dan semua objek atau kondisi yang menjadi target aksi boy seperti: diamonds, Posisi bot lawan, posisi teleporter serta arah gerak yang tersedia.

2. Himpunan Solusi

   Aksi yang dipilih oleh bot selama permainan berlangsung untuk memaksimalkan skor terakhir serta status bot yang berhasil bertahan hidup hingga akhir permainan dengan poin maksimal.

3. Fungsi Solusi

   Untuk menentukan apakah aksi yang dipilih dapat membawa bot mendekati tujuan akhir yaitu mengumpulkan skor sebanyak mungkin. Misal Bot berhasil mengumpulkan jumlah berlian sebanyak mungkin dalam kondisi board yang dinamis.

4. Fungsi Seleksi

   Memilih kandidat terbaik pada setiap langkah dan berdasarkan prioritas yang ditentukan yaitu:

   - Mengambil berlian bot lain
   - Kabur jika ada bot lain mendekat
   - Ambil berlian manual
   - Fokus balik dulu

5. Fungsi Kelayakan

   Memeriksa apakah kandidat yang dipilih sudah layak untuk dimasukkan ke dalam himpunan solusi atau aksi yang dipilih dapat dieksekusi misal: Apakah inventory sudah penuh, apakah langkah valid, dan lainya.

6. Fungsi Objektif

   Fungsi yang akan dioptimalkan adalah total skor berlian yang berhasil disimpan di home base, dengan skor meningkat saat diamond berhasil dikumpulkan dan disimpan di base.

3.2 Eksplorasi Alternatif Solusi Greedy

Dalam permainan Diamonds, beberapa alternatif strategi *greedy* dapat dipertimbangkan, meskipun mungkin tidak diimplementasikan:

1. Greedy berdasarkan prioritas diamond yang terdekat dan mengabaikan jenis diamond, Ini dapat mengurangi *overhead* komputasi dalam menentukan target.
2. Greedy berdasarkan prioritas diamond dengan nilai poin tertinggi bertujuan untuk memaksimalkan perolehan poin per langkah dan per waktu yang dihabiskan. Hal ini mencoba menyeimbangkan antara nilai berlian dangan upaya yang dibutuhkan untuk mencapainya.
3. Greedy berdasarkan penargetan bot lawan yang memiliki banyak diamond dengan mengejar dan menyerang, jika berhasil ini dapat memberikan keuntungan poin yang cepat.
4. Greedy berdasarkan meminimalkan Risiko dan memaksimalkan Pengamanan Poin, sangat berhati-hati dengan keamanan bot dan poin yang di dapat. strategi ini dapat meminimalkan kerugian dan memastikan bot tetap berada dalam permainan lebih lama.

3.3 Analisis Efisiensi dan Efektivitas

Efisiensi:

1. Kompleksitas komputasi relatif rendah, biasanya O(N), di mana N adalah jumlah objek (robot lain, berlian) di papan yang dipertimbangkan pada setiap gerakan.
2. Keputusan cepat berdasarkan keadaan dan solusi terdekat yang ada dalam situasi tersebut, tanpa pencarian solusi yang ekstensif atau komputasi yang panjang.

Efektivitas:

1. Mengambil diamond bot lain

   Sangat efektif dalam mengumpulkan poin yang signifikan dan cepat. Ketika berhasil menangani bot musuh, tetapi bot tidak hanya mencuri berlian yang dibawa musuh, tetapi juga dapat meningkatkan skornya sendiri. Namun, efektivitas ini disertai risiko. Jika penanganan gagal atau bot benar-benar dipukul balik oleh musuh, ada potensi kehilangan berlian yang sudah terkumpul.
2. Kabur jika ada Bot lain mendekat

   Menjaga bot tetap hidup dan menjaga berlian yang telah terkumpul dalam inventaris. Efektivitasnya terletak pada pencegahan kerugian dan menjaga bot dalam permainan untuk terus mengumpulkan berlian.
3. Ambil berlian manuaL

   Bagus dalam mengumpulkan poin dan berkontribusi paling besar terhadap perolehan skor bot. Efektivitasnya dioptimalkan dengan memprioritaskan berlian merah dengan nilai lebih tinggi (2 poin) daripada berlian biru (1 poin).
4. Fokus balik dulu

   Efektif dalam mengamankan poin yang sudah diperoleh. Dengan menyimpan berlian secara teratur di markas, efektivitasnya sangat tinggi dalam konservasi poin dan persiapan untuk akumulasi lebih lanjut.

3.4 Strategi Greedy yang Dipilih

strategi algoritma greedy yang diputuskan untuk digunakan dalam program bot adalah:

1. Ambil Diamond:

   Hitung jarak Manhattan ke semua diamond.

   Pilih diamond merah terdekat; jika tidak ada, ambil biru.
2. Tackle Bot Lawan:

   Jika bot lawan memiliki inventory > 0 dan jarak ≤ 2 langkah, prioritaskan tackle.
3. Avoidance:

   Jika bot lawan mendekat (jarak < 3) dan inventory tidak kosong, kabur ke base.
4. Manajemen Inventory:

   Jika inventory penuh atau waktu hampir habis, kembali ke base.

BAB IV
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi algoritma greedy

1. Inisialisasi:
    - **directions**: daftar arah gerak (kanan, bawah, kiri, atas)
    - **goal_position**: posisi tujuan saat ini (bisa berlian, base, atau bot lain)
    - **current_direction**: arah gerak default saat roaming
    - **chased_timer**: hitung waktu dikejar bot lain
    - **last_position**: posisi terakhir untuk deteksi stuck
    - **run:** counter untuk mekanisme dodge dan kabur
    - **diamond_counter_cd**: cooldown pengambilan berlian setelah sampai di area diamond
    - **stuck_counter**: hitung berapa lama posisi tidak berubah (stuck)

2. Fungsi next_move(current_bot, board):
    - Update posisi dan properti bot
    - Deteksi bot musuh dalam radius tertentu (radius = 2) Jika ada, tingkatkan **chased_timer**; jika tidak reset
    - Update cooldown **diamond_counter_cd** jika **> 0**
    - Deteksi stuck jika posisi tidak berubah selama 2 tick. Jika stuck, ubah arah dan reset goal
    - Jika dikejar lebih dari 5 tick: Set goal ke base dan gerak pulang
    - Jika bot sudah punya ≥4 berlian: Jika ada bot musuh dekat dan run ≤ 2, Jika tidak, langsung pulang ke base
    - Cek bot musuh yang membawa berlian banyak dan berada dekat (≤4 blok) Jika ada, kejar bot tersebut untuk mencuri berlian
    - Jika cooldown **diamond_counter_cd = 0**: Cari diamond terbaik berdasarkan banyaknya diamond sekitar dan jarak terdekat Set goal ke diamond tersebut
    - Jika cooldown > 0 tapi ada diamond dekat (radius ≤ 2), ambil diamond itu

- Jika goal sudah ditentukan, hitung arah ke goal dan gerak ke sana
- Jika tidak ada goal, lakukan roaming dengan mengganti arah secara acak Return arah gerak (delta_x, delta_y)

4.2 Struktur Data

1. **list** untuk **directions**:

   self.directions = [(1, 0), (0, 1), (-1, 0), (0, -1)]. Ini adalah sebuah **list** yang berisi *tuple*. Setiap *tuple* merepresentasikan perubahan koordinat (delta X, delta Y) untuk setiap arah pergerakan dasar (kanan, atas, kiri, bawah). Struktur ini memungkinkan bot untuk dengan mudah mengubah arah pergerakannya secara siklis (roaming) atau memilih arah tertentu.

2. **Optional[Position]** untuk **goal_position** dan **last_position:**

   self.goal_position: Optional[Position] = None

   self.last_position: Optional[Position] = None

   Position adalah objek khusus dari *framework* game yang kemungkinan memiliki atribut **x** dan **y**. Penggunaan **Optional** menunjukkan bahwa atribut ini bisa saja **None** (tidak ada posisi tujuan atau posisi terakhir yang terekam) atau sebuah objek **Position**. Ini penting untuk kasus ketika bot belum menentukan tujuan atau baru memulai.

3. int untuk **counter dan timer:**

   self.current_direction, self.chased_timer, self.run, self.diamound_counter_cd, self.stuck_counter Variabel-variabel ini menggunakan tipe data integer untuk menyimpan nilai-nilai diskrit seperti indeks arah, hitungan waktu, atau hitungan kejadian (misalnya, jumlah kali dodge atau stuck). Ini adalah cara efisien untuk melacak status internal bot.

4. Objek **GameObject** untuk **board_bot**:

   Mewakili bot yang sedang dimainkan. Objek ini berisi informasi seperti posisi (**position**) dan properti (**properties**, yang mencakup jumlah berlian yang dimiliki, posisi base, dll.). Ini adalah struktur data kompleks yang disediakan oleh *framework* game.

5. Objek **Board** untuk **board**:

   Mewakili kondisi papan permainan secara keseluruhan. Objek ini berisi daftar semua bot lain (**board**.**bots**) dan semua berlian yang ada di papan (**board**.**diamonds**). Ini adalah struktur data utama yang memberikan gambaran lengkap tentang lingkungan permainan.

6. **list** objek **GameObject** untuk **board**.**bots** dan **board**.**diamonds**:

   Ini adalah **list** yang berisi objek-objek lain (**GameObject** untuk bot dan **Diamond** untuk berlian). Bot mengiterasi melalui daftar ini untuk menemukan target, mendeteksi musuh, atau mencari berlian.

4.3 Analisis Desain Solusi Algoritma Greedy dan Pengujian

Keberhasilan Strategi Greedy

- Pengumpulan berlian secara efisien
  Bot selalu mengambil berlian yang tidak hanya dekat tapi juga berikan berlian lain di sekelompok besar, jadi waktu perjalanan lebih maksimal dan mendapatkan lebih banyak berlian serentak.
- Menghindari musuh saat jumlah berlian sudah banyak
  Dengan sistem kabur dan dodge, bot mampu menghindari kehilangan berlit untuk diserang bot lain.
- Mengejar musuh yang membawa berlian banyak
  Bot juga mencoba untuk mengambil resiko dengan mengejar lawan yang sudah

mengumpulkan beberapa berlian dalam jarak yang memungkinkan untuk meningkatkan peluang mendapatkan tambahan berlian.

- Penanganan stuck
  Bot dapat mendeteksi posisi yang tidak bergerak dalam beberapa tick dan berubah arah agar tidak terjebak melalui loop lokasi yang sama.

Keterbatasan dan Kondisi Gagal Mendapatkan Nilai Optimal

- Keputusan lokal tanpa prediksi masa depan
  Greedy algoritma memilih batu berlian terbaik sekarang tanpa mempertimbangkan berlian berikutnya secara lebih dalam, sehingga bot bisa melewatkan jalan yang lebih baik secara keseluruhan.
- Cooldown pengambilan berlian
  Selama cooldown menyala, bot bisa melewatkan lebih jauh berlian tetapi berpotensi memberi return yang lebih besar karena perhatiannya masih pada daerah kecil, sehingga berlian yang didapat mungkin bukan maksimal.
- Kejar-kejaran dengan bot lain yang agresif
  Jika ada banyak bot musuh agresif yang mengejar, bot dapat menjadi terlalu sering bergoyang arah dan kehilangan fokus dalam pengumpulan berlian, mengalahkan efisiensi.
- Terjebak di area terbatas
  Jika berlian dan musuh terlalu padat di daerah kecil, bot kadang stuck meskipun sudah ada mekanisme stuck counter, membuat waktu perjalanan tidak efektif.

Pengujian

Pengujian dilakukan dengan beberapa skenario unik:

1. Skenario berlian tersebar merata dan sedikit musuh
   - Bot bisa memilih jalur pengumpulan berlian yang cepat dan efisien.
   - Memperoleh nilai tinggi dengan cepat.

2. Skenario bot dikejar musuh saat sudah membawa berlian banyak
    - Mekanisme dodge dan pulang efektif mengurangi risiko kehilangan berlian.
    - Skenario ini menunjukkan hasil cukup optimal.
3. Skenario musuh membawa berlian banyak dalam radius pengejaran
    - Bot dapat mengejar dan merebut berlian lawan, meningkatkan hasil akhir.
4. Skenario diamond padat dan cooldown aktif
    - Bot masih fokus mengambil berlian yang dekat, namun menganggap berlian lain yang potensial lebih menguntungkan kalah.
5. Skenario stuck di area sempit
    - Bot mengubah arah setelah stuck counter aktif, namun masih kurang optimal karena area sempit membuat pilihan terbatas.

BAB V
KESIMPULAN DAN SARAN

Dari analisis pendekatan greedy dalam permainan Diamonds, dapat disimpulkan bahwa pendekatan greedy memungkinkan pengambilan keputusan yang cepat dan efisien. Algoritma ini mempertimbangkan keadaan lokal seperti lokasi berlian, lokasi bot lawan, dan ruang inventori untuk menentukan tindakan optimal pada saat ini tanpa melakukan perhitungan yang rumit. Ini sangat berguna dalam permainan yang dibatasi waktu, di mana tindakan cepat dapat memberikan keunggulan kompetitif. Namun, penerapan strategi greedy dalam bentuk murni tidak terlalu luas, terutama dalam lingkungan permainan yang kompetitif dan padat.

Sebagai rekomendasi untuk pengembangan selanjutnya, kami menyarankan untuk mengimplementasikan mekanisme penilaian jangka panjang (lookahead) agar bot dapat mempertimbangkan konsekuensi dari beberapa langkah ke depan, bukan hanya langkah segera. Dengan cara ini, bot dapat menghindari kesalahan taktis dan merencanakan gerakan yang lebih strategis.

LAMPIRAN

Tautan GitHub: https://github.com/SmollJesteR/StimaBOT

DAFTAR PUSTAKA

Institut Teknologi Sumatera. (2024). *Modul Strategi Algoritma greedy bagian 1 IF2211* [PDF]. https://kuliah.itera.ac.id/mod/resource/view.php?id=32758

Institut Teknologi Sumatera. (2024). *Modul Strategi Algoritma greedy bagian 2&3 IF2211* [PDF]. https://kuliah.itera.ac.id/mod/resource/view.php?id=33150

Haziq Amrullah. (2024). tubes1-IF2211-game-engine (v1.1.0). GitHub. https://github.com/haziqam/tubes1-IF2211-game-engine/releases/tag/v1.1.0

Haziq Amrullah. (2024). tubes1-IF2211-bot-starter-pack (v1.0.1). GitHub. https://github.com/haziqam/tubes1-IF2211-bot-starter-pack/releases/tag/v1.0.1

Institut Teknologi Sumatera. (2024). *Implementasi Algoritma Greedy dalam Permainan Diamonds* [PDF]. https://drive.google.com/file/d/17_d7sRWhr0TspjS0ZqIIQCnQnElPaeDR/view

IF2211 Strategi Algoritma – Institut Teknologi Bandung. (2024). *Tugas Besar 1 dan 2: Permainan Diamonds*. Panduan dan dokumentasi teknis. https://docs.google.com/document/d/1L92Axb89yIkom0b24D350Z1QAr8rujvHof7-kXRAp7c