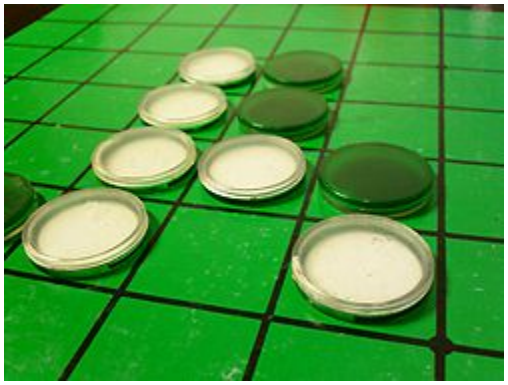
Disusun oleh:

Sanrio Hernanto	/ 13507019
Shauma Hayyu	/ 13507025
Jonathan Marcel	/ 13507072

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung
Jl. Ganesha 10, Bandung 40132

Bab 1
Deskripsi Masalah

Othello atau *Reversi* adalah permainan yang menggunakan papan (*board game*) dan sejumlah koin yang berwarna gelap (misalnya hitam) dan terang (misalnya putih). Ukuran papan biasanya 8 x 8 kotak (grid) dan jumlah koin gelap dan koin terang masing-masing sebanyak 64 buah. Sisi setiap koin memiliki warna yang berbeda (sisi pertama gelap dan sisi kedua terang). Pada permainan ini kita asumsikan warna hitam dan putih. Jumlah pemain 2 orang.



Dalam permainan ini setiap pemain berusaha mengganti warna koin lawan dengan warna koin miliknya (misalnya dengan membalikkan koin lawan) dengan cara "menjepit" atau memblok koin lawan secara vertikal, horizontal, atau diagonal. Barisan koin lawan yang terletak dalam satu garis lurus yang diapit oleh sepasang koin pemain yang *current* diubah (*reverse*) warnanya menjadi warna pemain yang *current*.

Aturan Permainan

1. Pada awal permainan diletakkan dua koin hitam dan dua koin putih di tengah-tengah papan dengan susunan seperti di bawah ini:

2. Pemain dengan koin hitam mulai duluan. Koin hitam hanya boleh ditempatkan pada posisi kotak sedemikian sehingga sepasang koin hitam akan menjepit seluruh koin putih yang ada dalam satu baris (vertikal, horizontal, atau diagonal).
3. Setelah penempatan semua koin hitam, maka semua koin putih sepanjang garis lurus yang berada di antara koin hitam yang baru dan koin hitam di seberangnya dibalikkan menjadi warna hitam.
4. Sekarang giliran pemain dengan koin putih yang bermaian. Pemain ini menggunakan aturan yang sama seperti pada poin 2 di atas. Kemungkinan penempatan koin putih adalah seperti gambar di bawah ini:
5. Setiap pemain bergantian meletakkan koinnya. Jika seorang pemain tidak dapat meletakan koin karena tidak ada posisi yang dibolehkan, permainan kembali ke pemain lainnya. Jika kedua pemain tidak bisa lagi meletakkan koin, maka permainan berakhir. Hal ini terjadi jika seluruh kotak telah terisi, atau ketika seorang pemain tidak memiliki koin lagi, atau ketika kedua pemain tidak dapat melakukan penempatan koin lagi. Pemenangnya adalah pemain yang memiliki koin paling banyak di atas papan.

Bab 2
Dasar Teori

Definisi Algoritma Greedy

Algoritma *greedy* adalah algoritma yang memecahkan masalah langkah per langkah, dimana pada setiap langkah:

- Menggunakan prinsip "take what you can get now", artinya akan diambil langkah yang terbaik yang dapat diperoleh pada saat itu tanpa memikirkan konsekuensi ke depan
- Memiliki harapan bahwa dengan pemilihan terhadap optimum lokal pada setiap langkah akan berakhir dengan terjadinya optimum global

Algoritma *greedy* mengasumsikan bahwa optimum lokal merupakan bagian dari optimum global.

Skema Umum Algoritma Greedy

Persoalan optimasi dalam konteks algoritma *greedy* disusun oleh 5 buah elemen berikut:

1. Himpunan kandidat, C

Merupakan himpunan yang berisi elemen-elemen pembentuk solusi.

2. Himpunan solusi, S

Merupakan himpunan yang berisi kandidat-kandidat yang terpilih sebagai solusi persoalan.

3. Fungsi seleksi – dinyatakan dengan predikat SELEKSI

Merupakan fungsi yang pada setiap langkah memilih kandidat yang paling memungkinkan mencapai solusi optimal.

4. Fungsi kelayakan – dinyatakan dengan predikat LAYAK

Merupakan fungsi yang memeriksa apakah suatu kandidat yang telah dipilih dapat memberikan solusi yang layak.

5. Fungsi objektif

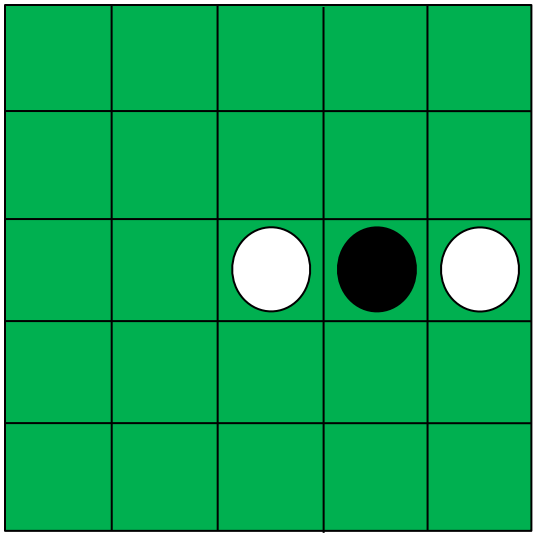
Fungsi yang memaksimumkan atau meminimumkan nilai solusi.

Bab 3
Analisis pemecahan masalah

Algoritma greedy by count

Pertama-tama akan dilakukan iterasi pada matriks berukuran 8 * 8. Kemudian elemen-elemen P (bertipe point) yaitu P.x dan P.y akan diassign dengan nilai 1 sampai 8. Setelah itu akan digunakan fungsi kelayakan, apabila langkah player layak untuk ditempatkan, akan dipanggil fungsi kalkulasi untuk menghitung jumlah piece yang akan dibalikkan. Selanjutnya jumlah tersebut akan diassign ke dalam suatu variable. Karena dilakukan pengulangan, maka algoritma ini akan memperoleh jumlah piece maksimum yang dapat dibalikkan. Inilah mengapa algoritma tersebut dinamakan greedy by count.

Pada fungsi kelayakan, pertama-tama akan dilakukan pengecekan pada seluruh papan permainan, apabila bernilai nilai x atau y berada di luar 1 – 8 (di luar matriks 8*8), maka akan dilakukan pengembalian nilai 0. Selanjutnya akan dilakukan pengecekan pada tiap-tiap kotak ke segala arah (8 arah) pada satu kotak. Misalnya pada pengecekan ke kanan, pertama-tama kotak (x,y) – dalam hal ini merupakan kotak yang paling tengah berisi piece putih, kemudian dicek piece yang berada di kanannya, apabila warna piece di kanannya ialah hitam, akan dilakukan pengecekan terus hingga piece di kanannya berwarna putih. Apabila sudah menemukan piece berwarna putih, maka akan dilakukan pengembalian nilai 1.



Perhatikan gambar di atas, pada langkah berikutnya piece hitam akan diflip menjadi putih, karena penempatan piece putih yang paling kanan dianggap layak.

Sedangkan pada fungsi kalkulasi, pertama-tama akan dilakukan pengecekan pada tiap-tiap kotak ke segala arah (8 arah). Ambil contoh misalnya pada pengecekan ke kanan posisi pada x dan y awal ialah piece yang berwarna putih. Apabila piece di kanan ialah piece hitam, pengecekan akan terus dilakukan hingga ditemukan piece di kanan yang berwarna putih. Hal ini sebenarnya mirip dengan fungsi layak, tetapi pada fungsi kalkulasi akan menghitung jumlah piece yang akan dibalik (dalam kasus ini ialah piece hitam di sebelah kanan).

a) Himpunan Kandidat

Elemen-elemennya adalah koordinat-koordinat pada papan yang valid untuk ditempati. Pemain dapat meletakkan koin di papan setelah setidaknya memakan satu koin lawan.

b) Himpunan Solusi

Kandidat yang terpilih sebagai himpunan solusi dalam algoritma ini adalah koordinat yang dapat memakan koin lawan paling banyak.

c) Fungsi Seleksi

Kandidat yang dipilih adalah kandidat yang menghasilkan nilai maksimum dari daftar koordinat-koordinat yang memenuhi fungsi kelayakan.

d) Fungsi Kelayakan

Setelah mendapatkan kandidat solusi, kandidat tersebut dimasukkan dalam sebuah fungsi untuk memeriksa apakah kandidat tersebut tidak melanggar syarat-syarat solusi yang berlaku.

e) Fungsi Objektif

Fungsi yang memaksimumkan dan meminimumkan solusi pada algoritma ini, adalah fungsi yang membandingkan berapa banyak koin lawan yang dapat dibalik dengan menempatkan koin pada setiap koordinat pada papan.

Algoritma greedy by Position

Strategi lain dalam memenangkan othello adalah dengan menempatkan koin di koordinat yang strategis, yaitu paling pinggir papan adalah yang paling strategis, dan tepat sebelum paling pinggir papan adalah tempat yang paling tidak strategis.

a) Himpunan Kandidat

Elemen-elemennya adalah koordinat-koordinat pada papan yang valid untuk ditempati. Pemain dapat meletakkan koin di papan setelah setidaknya memakan satu koin lawan.

b) Himpunan Solusi

Kandidat yang terpilih sebagai himpunan solusi dalam algoritma ini adalah koordinat yang letaknya paling strategis, yaitu baris ke-1 dari tepi paling strategis, setelah baris ke-3, baris ke-4, dan terakhir baris ke-2.

c) Fungsi Seleksi

Posisi x dan y kandidat diperiksa, jika koordinat tersebut terletak paling strategis, maka posisi tersebut yang dipilih.

d) Fungsi Kelayakan

Setelah mendapatkan kandidat solusi, kandidat tersebut dimasukkan dalam sebuah fungsi untuk memeriksa apakah kandidat tersebut tidak melanggar syarat-syarat solusi yang berlaku.

e) Fungsi Objektif

Fungsi yang memaksimumkan dan meminimumkan solusi pada algoritma ini adalah fungsi yang membandingkan jarak koordinat penempatan dengan tepi papan.

Bab 4
Implementasi dan pengujian

Spesifikasi Teknis Program

Struktur Data

Senarai (array)

Terdapat beberapa senarai yang digunakan dalam program ini, antara lain:

1. Senarai move[61] digunakan untuk menyimpan langkah-langkah yang telah dijalankan sebelumnya
2. Senarai color[3] digunakan untuk menyimpan nilai warna player

Matriks

Matriks yang digunakan dalam program ini ialah senarai 2 dimensi board[10][10] digunakan untuk menyimpan state board.

Fungsi dan Prosedur

Terdapat beberapa fungsi maupun prosedur yang digunakan di dalam program ini antara lain:

```
void start_board()
//prosedur untuk mengembalikan board ke state awal
void Taruh(Point LP,int player)
//prosedur untuk menaruh piece player ke dalam LP kemudian
membalikkannya sesuai aturan permainan othello
void changeplayer(int *player)
//prosedur untuk mengganti giliran pemain
void changestate(int t)
//prosedur untuk mengubah-ubah state permainan
void endturn(int *player,int *state)
//prosedur untuk menyelesaikan giliran player
int Layak(Point LP,int player)
//fungsi untuk mengecek kelayakan peletakkan piece 'player' di
posisi LP, akan mengembalikan 0 jika tidak layak dan 1 jika
layak
```

```
int count(int player)
//fungsi untuk menghitung jumlah piece ‘player’ yang berada di
board
int Kalkulasi(Point LP,int player)
//fungsi untuk mengkalkulasi penaruhan piece 'player' ke posisi
LP, akan mengembalikan jumlah piece yang akan didapatkan player
Point GreedyTurnAI(int player)
//fungsi untuk melakukan pergerakan AI dengan algoritma greedy,
akan mengembalikan posisi dengan piece maksimum
```

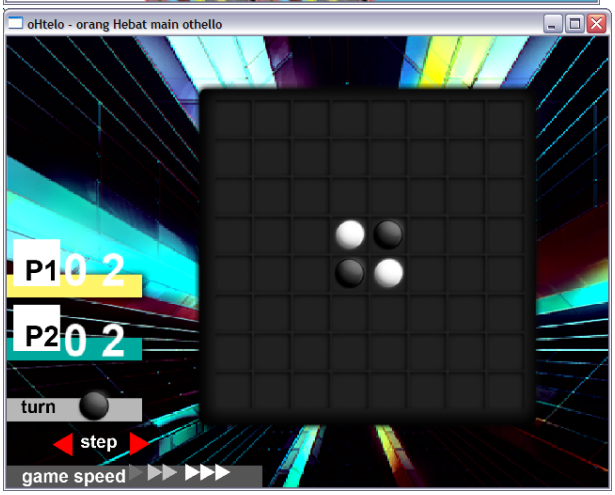
Antarmuka

Pembuatan antarmuka program ini menggunakan kakas Game Engine DarkGDK, yang selanjutnya di-compile dengan compiler Microsoft Visual C++ 2008.

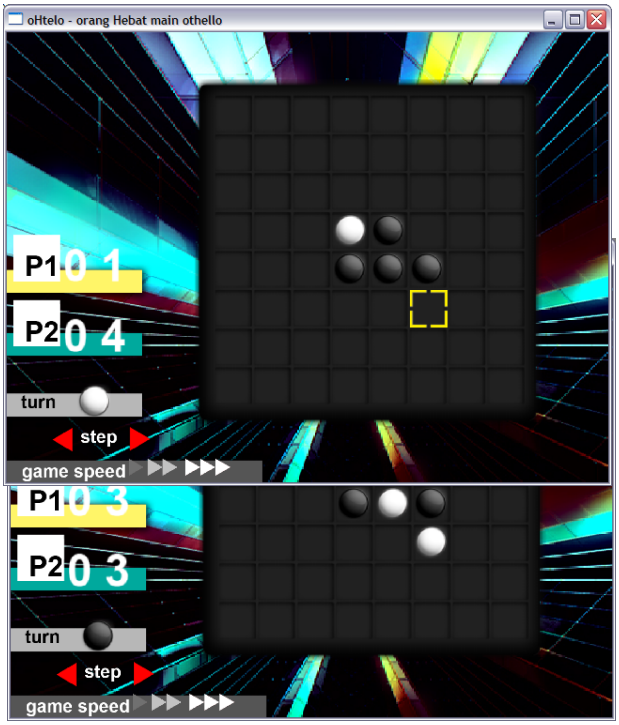
Screenshot Program

Pada saat pertama kali menjalankan game, anda akan menemui menu utama seperti gambar disamping kiri. Untuk memulai permainan, silahkan klik tombol start dan untuk mengakhiri permainan, silahkan klik tombol quit.

Ketika anda menekan tombol start, anda akan menemui menu pilihan seperti gambar disamping kiri. Anda dapat mengkustomisasi pemain othello (bisa human ataupun berbagai macam AI). Selain itu anda juga dapat memilih warna piece untuk setiap pemain. Setelah selesai, klik tombol VS untuk memulai permainan.



Gambar disamping kiri merupakan menu game. Papan berukuran 8*8 tersebut merupakan papan permainannya, dan gambar-gambar bola tersebut merepresentasikan piece.
Angka-angka di bagian kiri ialah jumlah piece tiap player. Anda juga dapat mengubah-ubah kecepatan (dengan mengeklik tombol pada game speed) dan melihat langkah sebelumnya dengan menekan tombol pada panel step.



Gambar di samping menunjukkan keadaan permainan sebelum di-klik pada kursor kuning tersebut.

Gambar di samping menunjukkan keadaan permainan setelah di-klik pada kursor kuning tersebut**.** Perhatikan bahwa piece hitam di-flip menjadi putih seperti peraturan permainan othello.

Analisis Hasil Pengujian

Algoritma greedy saja sebenarnya kurang optimum untuk memecahkan permasalahan othello, karena selain langkahnya bisa dibaca oleh lawan karena tidak adanya fungsi random, dan membutuhkan lebih banyak poin-poin greedly lainnya, tidak hanya jumlah dan posisi.
Untuk algoritma greedy berdasarkan jumlah, memiliki kelemahan karena kondisi game pada saat turn algoritma greedy tersebut belum tentu sama apabila sudah berganti dengan turn lawan. Karena bisa saja pada saat turn lawan tiba, posisi peletakkan poin oleh algoritma greedy sebelumnya, menjadi posisi yang menguntungkan bagi lawan untuk menghasilkan koin terbanyak.
Untuk algoritma greedy berdasarkan jarak dengan tepi, kelemahannya adalah di othello terdapat jauh lebih banyak lagi strategi-strategi yang bisa digunakan seperti menempatkan di antara koin lawan sehingga koin tidak dapat direbut dan banyak lagi yang algoritma greedy ini belum perhitungkan

Bab 5
Kesimpulan dan Saran

Kesimpulan

Kesimpulan dari tugas ini ialah:

Program berhasil mensimulasikan board game Othello, dimana user dapat memilih yang bermain adalah human atau computer. Apabila komputer yang bermain, AI akan mengimplementasikan algoritma Greedy yang dibuat.

Algoritma Greedy yang dibuat ialah greedy by count dan greedy by position. Masing-masing algoritma greedy ini memiliki elemen-elemen greedy yang berbeda-beda.

Saran

Beberapa saran yang dapat diberikan antara lain:

Untuk berikutnya dapat dikembangkan tugas yang serupa dengan tipe algoritma yang berbeda-beda, bukan hanya algoritma Greedy saja.

Selain itu algoritma Greedy yang dibuat juga dapat diimplementasikan ke dalam board game yang lain, misalnya dalam permainan *chekers* ataupun *stratego*.

Referensi

Munir, Rinaldi. 2009. Diktat Kuliah IF3051 Strategi Algoritma.

Dark Game Development Kit Documentation.