Michael Jonathan Halim - 13521124 - Tic Tac Toe

1. Jelaskan secara umum algoritma minimax!

   Algoritma minimax merupakan suatu algoritma AI yang digunakan dalam permainan bergiliran yang hanya dimainkan oleh dua orang seperti Tic Tac Toe dengan mencari langkah terbaik yang dapat diambil. Algoritma ini menggunakan konsep backtracking untuk mengambil keputusan. Biasanya dua pemain ini disebut sebagai maximizer dan minimizer. Maximizer akan mencari score setinggi mungkin sedangkan minimizer akan mencari score sekecil mungkin. Algoritma minimax akan membangun suatu pohon permainan dengan node-nodenya sebagai suatu kondisi dari permainan tersebut akibat suatu langkah yang diambil. Oleh karena itu, algoritma akan selalu memilih langkah terbaik yang menghasilkan keuntungan maksimal atau kerugian minimal.

2. Jelaskan bagaimana algoritma minimax mengambil langkah terbaik dalam permainan TicTacToe yang kalian buat!

   Jadi, dalam permainan TicTacToe yang dibuat, algoritma minimax digunakan sebagai bot AI yang akan bertanding dengan pemain (user). Digunakan bahasa pemrograman Python dan library Tkinter untuk aplikasi GUI. Berikut adalah langkah-langkah algoritma minimax yang diimplementasikan.

   1. Buat variabel untuk menyimpan score tertinggi, menyimpan posisi board yang akan dimainkan oleh bot, menyimpan alpha, dan menyimpan beta.
   2. Iterasikan seluruh kotak pada papan. Dalam permainan TicTacToe, ukuran papan adalah 3x3 sehingga iterasikan 9 kotak tersebut.
   3. Periksa apakah kotak tersebut sudah terisi atau belum. Jika sudah terisi, abaikan.
   4. Jika belum terisi, kita akan hitung score yang bisa didapatkan jika bot mengambil kotak tersebut dengan algoritma minimax secara rekursif. Karena kita mencari langkah terbaik untuk bot, maka pemanggilan minimax pertama kali untuk giliran pemain. Jangan lupa untuk menginisiasi alpha dengan bilangan negatif besar dan beta dengan bilangan positif besar.
   5. Periksalah jika bot mengambil kotak tersebut, apakah permainan berakhir atau tidak. Permainan berakhir bisa berarti bahwa bot telah menang, pemain telah menang, atau seri. Jika pemain menang, kembalikan -10. Jika bot menang, kembalikan 10. Jika seri, kembalikan 0.

6. Jika permainan belum berakhir, periksa apakah saat ini merupakan giliran pemain atau bot. Jika sekarang giliran pemain, maka kita akan memanggil fungsi minimax untuk giliran bot lagi. Pemanggilan ini jika dilakukan dengan mengiterasikan seluruh kotak untuk mencari semua kemungkinan langkah. Tujuannya tentu adalah untuk mencari best score. Jika sekarang giliran bot lagi, maka kita akan memanggil fungsi minimax untuk giliran pemain lagi.
7. Namun, perbedaannya adalah untuk pemain, kita akan memegang peran sebagai minimizer sehingga kita akan mencari score sekecil mungkin sedangkan untuk bot memegang peran sebagai maximizer yaitu mencari score terbesar mungkin.
8. Karena mengimplementasikan alpha beta pruning, kita juga perlu mencari alpha terbaik untuk setiap best score pada maximizer dan beta terendah untuk setiap best score pada minimizer.
9. Jika alpha sudah lebih dari beta, maka kita akan cutoff pemanggilan rekursifnya dan mengembalikan best score pada state tersebut. Pemeriksaan ini dilakukan di sisi maximizer juga minimizer.
10. Alpha dan beta disini berperan untuk memotong percabangan pemeriksaan yang tidak relevan atau dalam kata lain tidak perlu diperiksa karena tidak akan menghasilkan best score sehingga mempercepat waktu eksekusi.
11. Jika score akhir lebih besar dari best score, maka kita simpan score tersebut sebagai best score. Simpan juga kotak yang memiliki best score tersebut.
12. Jika sudah didapatkan score terbaik dari seluruh kemungkinan langkah, update langkah bot sesuai dengan kotak yang memiliki best score tersebut.
13. Periksa kembali apakah permainan sudah berakhir atau belum. Jika belum, giliran akan diserahkan ke pemain.