Algoritma Minimax untuk A.I. Tic-Tac-Toe

David/13513019
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13513019@std.stei.itb.ac.id

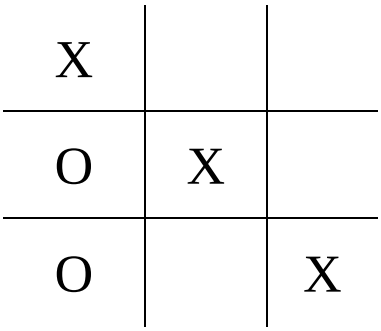
***Abstrak*—Minimax adalah sebuah algoritma yang digunakan untuk memilih langkah sedemikian sehingga didapat langkah yang kemungkinan kalah paling sedikit. Pemilihan langkah identik dengan *Depth Limited Search*, yang setiap simpulnya mempunyai nilai. Algoritma minimax banyak dipakai sebagai A.I. dari sebuah permainan yang berbasis pemilihan keputusan. Permainan Tic-Tac-Toe merupakan contoh yang A.I. nya biasa menggunakan algoritma minimax.**

Kata kunci: *Minimax, A.I., Depth Limited Search, Tic-Tac-Toe*

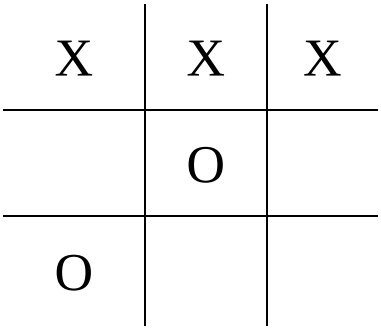
I. PENDAHULUAN: PERMAINAN TIC-TAC-TOE

Permainan tic-tac-toe adalah permainan yang dimainkan oleh dua pemain secara bergantian pada kotak 3x3. Permainan berakhir apabila salah satu pemain menang atau semua kotak telah terisi. Syarat menang yaitu dengan membuat tiga simbol yang sama pada salah satu baris, kolom, ataupun diagonal.

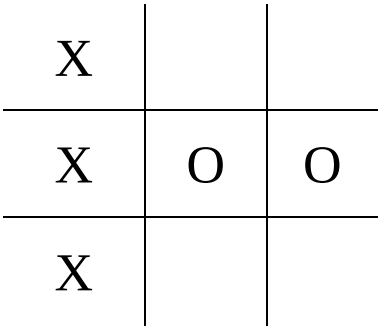
Ilustrasi aturan permainan dapat dilihat pada gambar berikut.



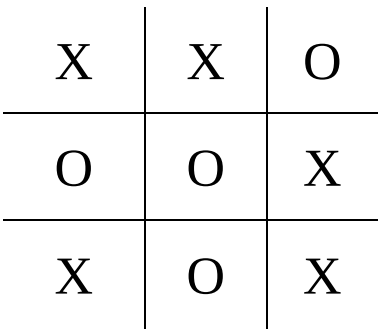
Gambar 1(a). Pemain X menang secara diagonal
(sumber: koleksi penulis)



Gambar 1(b). Pemain X menang secara horizontal
(sumber: koleksi penulis)



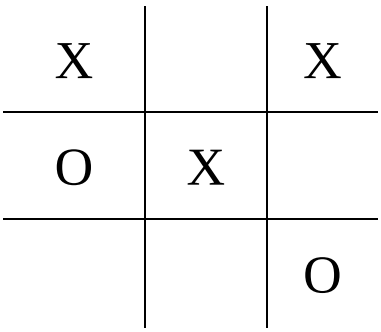
Gambar 1(c). Pemain X menang secara vertikal
(sumber: koleksi penulis)



Gambar 1(d). Permainan berakhir seri

Permainan tic-tac-toe pada awalnya kelihatannya cukup sederhana dan tidak mungkin untuk kalah, akan tetapi, sebenarnya dibutuhkan berbagai strategi untuk menjamin agar kekalahan tidak mungkin terjadi.

Pada gambar berikut, pemain O melakukan kesalahan pada langkah pertamanya yang ternyata berakibat pada kekalahan.



Gambar 2. Pemain X melakukan serangan dari dua arah (sumber: koleksi penulis)

Permainan Tic-Tac-Toe pada dasarnya adalah permainan yang dengan langkah optimal dari kedua pihak, hanya akan menghasilkan seri, meskipun jika langkah dilakukan secara acak, maka pemain pertama akan mempunyai peluang untuk menang lebih besar.

Pada makalah ini, akan dibahas A.I. Tic-Tac-Toe yang tidak mungkin kalah, yaitu A.I. yang menggunakan algoritma minimax dalam pemilihan langkah optimal.

II. ALGORITMA MINIMAX

Minimax adalah sebuah algoritma pemilihan sehingga langkah yang dipilih menghasilkan kerugian (*loss*) paling sedikit. Algoritma minimax didasarkan atas permainan dengan jumlah bobot nol, yaitu permainan yang apabila seorang pemain mendapat nilai tambah, maka pemain yang lainnya mendapat nilai kurang. Contohnya ketika pemain A melakukan langkah sehingga dia mendapat nilai +10, maka pemain B akan mendapat nilai -10 dan sebaliknya.

Definisi formal teori Minimax adalah sebagai berikut [1].

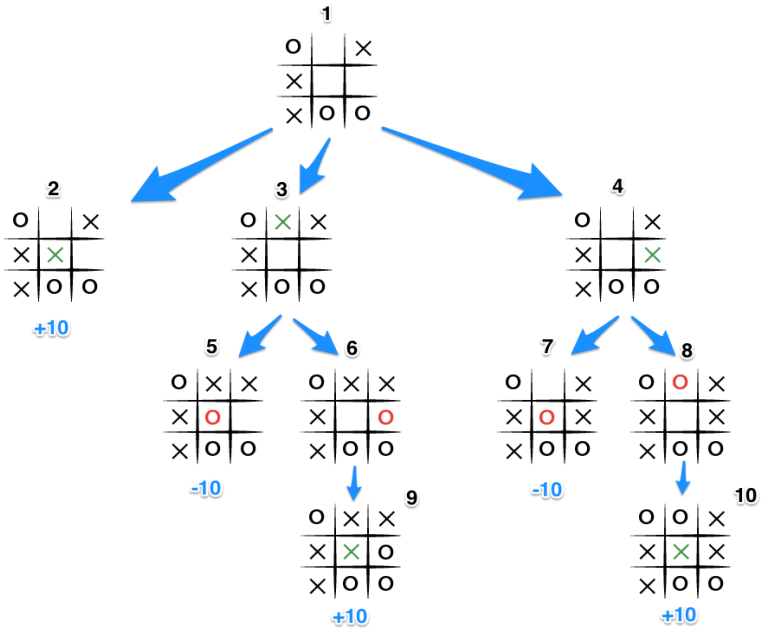
> Untuk setiap permainan dua-orang dengan jumlah bobot nol, dengan strategi yang terhingga banyaknya, terdapat sebuah nilai V dan strategi campuran untuk kedua pemain, sehingga
> (a) Dengan strategi pemain 2, keuntungan maksimum yang mungkin untuk pemain 1 adalah V, dan
> (b) Dengan strategi pemain 1, keuntungan maksimum yang mungkin untuk pemain 2 adalah –V.

Pencarian pada algoritma minimax mirip dengan algoritma pencarian pada *Depth Limited Search*, kemudian melakukan pemilihan keputusan yang mempunyai nilai kerugian terkecil. Dalam program A.I Tic-Tac-Toe ini, diterapkan aturan sebagai berikut.

1. Apabila permainan seri, nilai dari simpul adalah 0.
2. Apabila permainan dimenangkan pemain, nilai dari simpul adalah -10
3. Apabila permainan dimenangkan A.I., nilai dari simpul adalah 10.

Berdasarkan aturan tersebut, maka dipilih langkah yang nilai simpulnya terbesar.

Berikut adalah contoh gambar hasil pohon pencarian yang dibentuk dengan menggunakan algoritma minimax.



Gambar 3. Pohon pencarian algoritma minimax (sumber: http://neverstopbuilding.com/minimax)

Pseudocode dari algoritma minimax adalah sebagai berikut: [2]

```
function minimax(depth, player)
    if (gameover || level == 0)
        return score
    children = all legal moves for this player
    if (player is computer, i.e., maximizing player)
        // find max
        bestScore = -inf
        for each child
            score = minimax(level - 1, opponent)
            if (score > bestScore) bestScore = score
                return bestScore
    else (player is opponent, i.e., minimizing player)
        // find min
        bestScore = +inf
        for each child
            score = minimax(level - 1, computer)
            if (score < bestScore) bestScore = score
              return bestScore
```

Algoritma minimax memakai ruang yang cukup besar karena merupakan algoritma yang rekursif. Selain itu, algoritma minimax biasa dipakai pada permainan yang ketika pemilihan langkahnya akan membangkitkan banyak simpul. Oleh karena itu, biasanya algoritma ini dilengkapi juga dengan evaluasi heuristik agar pengambilan keputusan tidaklah terlalu lama.

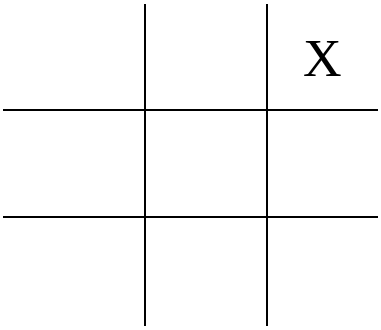
III. IMPLEMENTASI

Sebelum memulai mengimplementasi A.I., diimplementasikan dulu permainan Tic-Tac-Toe seperti pemilihan siapa yang jalan duluan dan kapan suatu permainan berakhir. Ini bagian yang sangat penting agar dapat mengimplementasi fungsi minimax yang mengisi nilai dari suatu simpul berdasarkan siapa yang memenangkan permainan.

Pada implementasi A.I., digunakan juga fungsi heuristik sederhana untuk melakukan pemilihan langkah awal dari A.I. yang dibuat. Apabila langkah pertama dilakukan oleh A.I. maka bebas memilih salah satu dari keempat sudut, karena menurut teori, ini adalah langkah yang menghasilkan peluang menang terbesar. Apabila langkah kedua dilakukan oleh A.I. maka A.I akan memilih tengah jika tengah belum diambil (menjamin tidak mungkin kalah), atau memilih salah satu sudut jika petak tengah telah diambil.

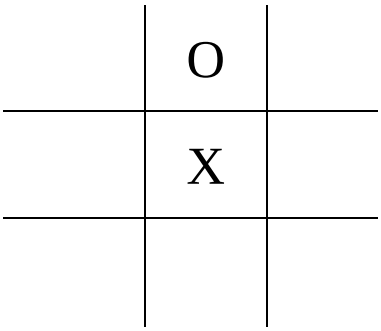
Beberapa gambar berikut akan menunjukkan langkah awal yang dipilih oleh A.I. (simbol X)

1. A.I. melangkah terlebih dahulu



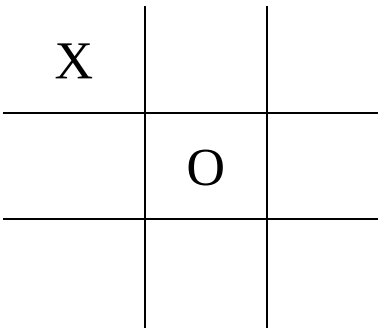
Gambar 4. A.I memilih salah satu sudut
(sumber: koleksi penulis)

2. Pemain melangkah lebih dahulu (bukan tengah)



Gambar 5. A.I. mengambil petak tengah
(sumber: koleksi penulis)

3. Pemain melangkah lebih dahulu (tengah)



Gambar 6. A.I. mengambil salah satu sudut
(sumber: koleksi penulis)

Pengambilan keputusan untuk langkah ketiga sampai selesai akan menggunakan fungsi minimax yang diimplementasi seperti pseudocode pada Bagian II makalah ini, dengan menggunakan depth pencarian = 2.

Pencarian lebih dari 2 boleh dilakukan, tetapi melalui pengamatan, tidak terdapat perbedaan hasil antara pencarian dengan depth=2 dan pencarian dengan depth>2. Ini disebabkan oleh permainan tic-tac-toe yang hanya cukup melihat 2 langkah ke depan supaya mencegah kekalahan terjadi maupun mendapatkan strategi kemenangan setelah dua langkah awal dilakukan. Meskipun demikian, tidak terasa perbedaan waktu ketika

A.I. melakukan langkah dalam *depth* berapapun.

IV. Analisis hasil implementasi

Implementasi program dan A.I. Tic-Tac-Toe ini menggunakan bahasa C++ sehingga komputasi langkah sangatlah cepat. Jumlah kombinasi pada Tic-Tac-Toe hanyalah 9! = 362.880 dan akan dapat diselesaikan dalam waktu kurang dari 1 detik pada komputer sekarang, meskipun implementasi hanyalah menggunakan depth 2 yang menghasilkan sekitar 40 sampai 50 kombinasi per langkah.

Pada semua percobaan yang dilakukan, A.I. selalu melakukan langkah yang optimal sehingga A.I. yang dibuat belum pernah mengalami kekalahan. A.I belum dicoba secara utuh 362.880 kemungkinan, sehingga tidak dapat diklaim bahwa A.I. ini tidak akan pernah mengalami kekalahan, karena mungkin terdapat *bug* pada program disebabkan oleh keterbatasan waktu, sehingga implementasi cukup dilakukan dengan terburu-buru.

Bagian selanjutnya akan diisi dengan gambar hasil implementasi (Player menggunakan simbol O, dan A.I. menggunakan simbol X)

V. Gambar Hasil Implementasi

Berikut adalah gambar-gambar hasil running dengan 4 kasus, yaitu A.I. menang dua kali dan A.I. seri dua kali. (Tidak ditemukan kasus di mana A.I. kalah)

O	X	X
X	X	O
O	O	X

Gambar 7. Permainan berakhir seri (A.I. melangkah duluan)

O	O	X
	X	O
X		

Gambar 8. Permainan dimenangkan A.I. (Player melangkah duluan)

O	X	O
X	O	
O		X

Gambar 9. Permainan dimenangkan A.I. (A.I melangkah duluan)

O	O	X
X	X	O
O	X	O

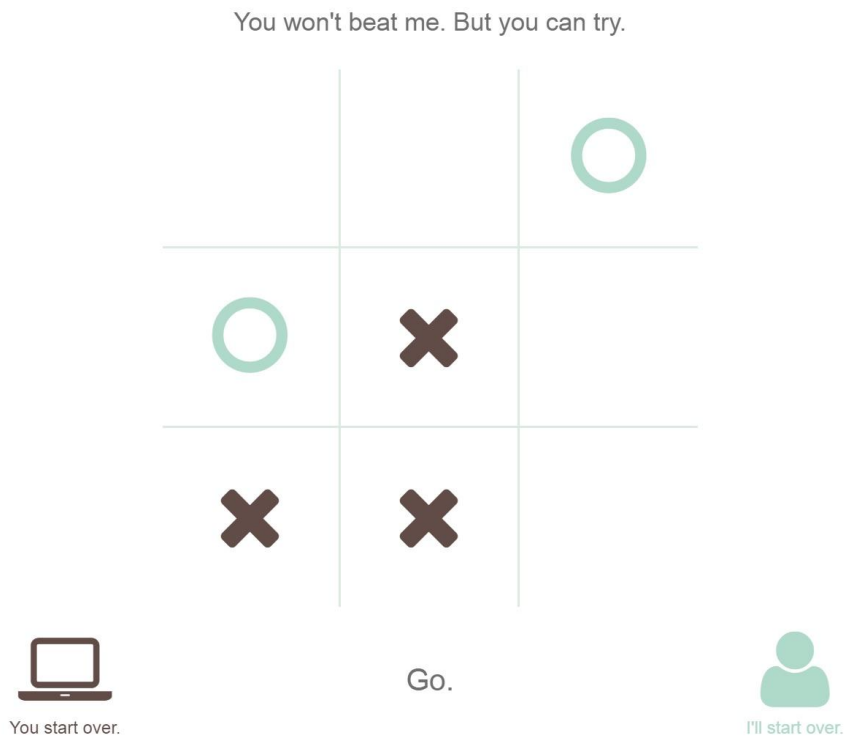
Gambar 10. Permainan seri (A.I. melangkah duluan)

VI. Penutup

Algoritma minimax merupakan algoritma pemilihan sehingga kerugian yang dialami minimal. Masih terdapat banyak algoritma lain yang dapat diimplementasi pada A.I. Tic-Tac-Toe seperti Rule-based Strategy (dapat dibaca lebih lanjut pada pranala https://www3.ntu.edu.sg/home/ehchua/programming/java/JavaGame_TicTacToe_AI.html). Meskipun demikian, algoritma minimax inilah yang menurut penulis paling gampang untuk diimplementasikan untuk permainan Tic-Tac-Toe.

Algoritma minimax ini mungkin akan kurang efektif jika diimplementasi pada A.I. di permainan lain. Ini disebabkan oleh basis pencarian algoritma yang seperti Depth Limited Search, sehingga apabila permainan dengan langkah yang memerlukan depth yang besar akan memerlukan waktu yang lama.

Untuk contoh A.I. tic-tac-toe yang tidak dapat dikalahkan, dapat mengunjungi pranala http://perfecttictactoe.herokuapp.com/. Aplikasi tersebut menggunakan bahasa ruby, sehingga sedikit lebih lambat dibandingkan dengan implementasi penulis. Akan tetapi tampilan yang digunakan jauh lebih menarik karena tampilan penulis menggunakan *command line*.



Gambar11. Hasil screenshot http://perfecttictactoe.herokuapp.com/, A.I. bersimbol X

VII. UCAPAN TERIMA KASIH

Pertama-tama penulis ingin bersyukur kepada Tuhan Yang Maha Esa, karena hanya oleh rahmat-Nya penulis dapat menyelesaikan tulisan ini. Penulis juga berterima kasih kepada dosen Dr. Ir. Rinaldi Munir beserta Dr. Nur Ulfa Maulidevi atas bimbingan mereka penulis dibekali dengan pengetahuan untuk membuat tulisan ini. Tak lupa pula penulis berterima kasih atas rekan-rekan yang telah membantu dalam pembuatan tulisan ini dalam bentuk dorongan dan semangat.

REFERENSI

[1] Osborne, Martin J., and Ariel Rubinstein. A Course in Game Theory. Cambridge, MA: MIT, 1994. Print.

[2] Case Study on Tic-Tac-Toe Part 2: With AI, https://www3.ntu.edu.sg/home/ehchua/programming/java/JavaGame_TicTacToe_AI.html, diakses 3 Mei 2015

Tic Tac Toe: Understanding The Minimax Algorithm, http://neverstopbuilding.com/minimax, diakses 3 Mei 2015

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 4 Mei 2015

David / 13513019