ALGORITMA MINIMAX SEBAGAI PENGAMBIL KEPUTUSAN DALAM *GAME TIC-TAC-TOE*

Muhammad Kurniawan[1)], Afib Pamungkas[2)], Salman Hadi[3)]

[1), 2), 3)] Teknik Informatika STMIK AMIKOM Yogyakarta
Jl Ring road Utara, Condongcatur, Sleman, Yogyakarta 55281
Email : muhammad.7071@students.amikom.ac.id [1)], afib.p@students.amikom.ac.id [2)], salman.h@students.amikom.ac.id [3)]

Abstrak

Pada saat ini perkembangan teknologi semakin maju. Banyak sekali mesin–mesin canggih yang bermunculan. Mesin mesin itu sudah layaknya seseorang yang bisa menggantikan pekerjaan orang. Bukan hanya itu, perkembangan dalam dunia game semakin berkembang. Dimana banyak game yang telah menanamkan Artificial Intelligence (AI) kedalam game. Sehingga pemainan game semakin menarik. Game yang sederhana menjadi menarik dengan ditambahkan AI kedalam program game. Setiap game memiliki tingkat kesulitan mulai dari yang mudah, menengah dan susah. Banyak algoritma yang dapat diterapkan dalam sebuah game, salah satu contohnya adalah algoritma minimax.

Algoritma minimax merupakan metode yang sangat terkenal dalam pengambilan keputusan untuk meminimalisir peluang kalah atau rugi. Algoritma sederhana namun cukup terkenal untuk pembuatan game sederhana. Dimana algoritma ini bekerja dengan mengukur seberapa baik posisi pada saat itu. Algoritma ini menggunakan heuristic f(n)=1 jika komputer menang, f(n)= 0 jika permainan seimbang, f(n)= -1 jika komputer kalah. Dengan algoritma minimax komputer akan langsung mencari hingga state terakhir dari setiap kemungkinan yang ada. Sehingga komputer dengan cepat memiliki solusi terbaik sehingga dapat memenangkan permainan.

Dengan menanambahkan algoritma minimax pada permainan tic-tac-toe yang bertindak sebagai AI, membuat komputer dapat leluasa mencari solusi terbaik dan membuat komputer memenangkan permainan. Adapun hasil terbaik yang dapat diperoleh oleh pemain adalah hasil imbang. Algoritma minimax juga tidak terlalu sulit untuk diterapkan pada permainan tic-tac-toe.Selain itu algoritma ini sangat mudah untuk dipahami.

Kata kunci: *tic-tac-toe, Artificial Intelligence , minimax, resource, heuristic, state.*

1. Pendahuluan

Saat ini, *video game* tidak seperti yang kita kenal dulu. Jika dulu *game* biasa dimainkan oleh dua orang dan hanya bisa bermain ditempat yang sama, sekarang dengan kemajuan teknologi terutama pada jaringan internet, *video game* beralih dari yang dahulu dimaikan secara *offline,* sekarang mulai berganti menjadi *game online*. *Game online* dapat dimainkan kapanpun dan dimana saja asalkan kita memiliki jaringan internet. *Game online* tidak hanya dimainkan oleh satu orang atau dua orang, tetapi dapat dimainkan lebih dari 100 orang atau lebih pada waktu yang bersamaan. Tidak hanya di satu daerah saja namun dapat terhubung ke seluruh penjuru dunia asalkan ada jaringan internet.

Permainan-permainan berbasis komputer ini juga beraneka ragam. Salah satu kelebihannya adalah kita tidak harus mencari orang untuk menjadi lawan tanding. Dengan adanya *AI,* kita dapat bermain sendiri melawan komputer yang dibuat untuk dapat bermain seperti manusia. Dalam dunia *game* biasanya kita tidak hanya bisa melawan pemain manusia lainnya, tetapi kita juga dapat melawan mesin atau bisa kita sebut *bot* dimana setiap *bot* memiliki tingkat kesulitan yang dapat diatur mulai dari yang mudah, sedang, dan susah. Untuk membuat pemain merasa seperti melawan pemain manusia lainnya, maka diperlukan suatu algoritma yang dapat membuat *AI* ini mampu mengambil keputusan terbaik agar dapat mengalahkan pemain atau setidaknya menghalau pemain menang. Ada banyak algoritma untuk membuat *AI* bekerja layaknya seperti manusia, salah satunya adalah algoritma *minimax*. Banyak *game* sederhana yang dibuat menggunakan algoritma *minimax*, salah satu contohnya adalah *tic-tac-toe*. Algoritma *minimax* ini merupakan algoritma yang sering sekali dipakai untuk membuat *AI* dapat memenangkan semua permainan. Dan permainan *tic-tac-toe* merupakan salah satu contoh yang cukup sederhana untuk kita pelajari bagaimana cara kerja dan efeknya.

Berdasarkan latar belakang di atas, maka dapat dirumuskan sebagai berikut:

1. Bagaimana menerapkan algoritma *minimax* sebagai pengambil keputusan dalam permainan *tic-tac-toe*

Tujuan dari penelitian ini adalah untuk menganalisis algoritma *minimax* sebagai pengambil keputusan dalam permainan *tic-tac-toe*. Peneliti mencoba menerapkan algoritma *minimax* dalam permainan *tic-tac-toe*.

Berdasarkan penelitian yang telah dilakukan sebelumnya, telah dirancang sebuah permainan *tic-tac-toe* dengan algoritma *genetika* oleh Irving Vitra Paputungan[1]. Dalam penelitian tersebut dijelaskan bahwa Irving menerapkan konsep algoritma *genetika* dan pembelajaran mesin untuk mencari strategi-strategi yang pas dalam permainan. Dikarenakan kemampuan algoritma *genetika* yang bagus, diharapkan parameter-parameter dan *operator-operator* yang dipilih mampu menyelesaikan masalah ini, juga dapat dipakai untuk menyelesaikan permasalahan bidang yang hampir sama terkait dengan permainan.

1..1 Kecerdasan Buatan

Kecerdasan Buatan *(Artificial Intelligence)* merupakan salah satu bagian dari ilmu komputer yang mempelajari bagaimana membuat mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia bahkan bisa lebih baik daripada yang dilakukan manusia.

Menurut John McCarthy, 1956, *AI*: untuk mengetahui dan memodelkan proses-proses berpikir manusia dan mendesain mesin agar dapat menirukan perilaku manusia. Cerdas, berarti memiliki pengetahuan ditambah pengalaman, penalaran (bagaimana membuat keputusan dan mengambil tindakan), moral yang baik.

Untuk membuat aplikasi kecerdasan buatan ada 2 bagian utama yang sangat dibutuhkan:

1. Basis pengetahuan (*knowledge base*), bersifat fakta-fakta, teori, pemikiran dan hubungan antar satu dengan yang lainnya.
2. Motor inferensi (*inference engine*), kemampuan menarik kesimpulan berdasarkan pengetahuan dan pengalaman.[2]

Kecerdasan Buatan (*Artificial Intelligence* atau *AI*) didefinisikan sebagai kecerdasan yang ditunjukkan oleh suatu entitas buatan. Sistem seperti ini umumnya dianggap komputer. Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin (komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (*games*), logika *fuzzy*, jaringan syaraf tiruan dan robotika. [3]

Sistem memperlihatkan sifat-sifat khas yang dihubungkan dengan kecerdasan dalam kelakuan atau tindak-tanduk yang sepenuhnya bisa menirukan beberapa fungsi otak manusia, seperti pengertian bahasa, pengetahuan, pemikiran, pemecahan masalah, dan lain sebagainya. [4]

1..2 Algoritma *Minimax*

Algortima *Minimax* merupakan algoritma yang digunakan untuk menentukan pilihan agar memperkecil kemungkinan kehilangan nilai maksimal. Algoritma ini diterapkan dalam permainan yang melibatkan dua pemain seperti *tic-tac-toe*, *checkers*, *go* dan permainan yang menggunakan strategi atau logika lainnya. Hal ini berarti permainan-permainan tersebut dapat dijelaskan sebagai suatu rangkaian aturan atau premis. [5]

Algoritma *minimax* merupakan algoritma yang sering digunakan dalam permainan berbasis *AI* seperti permainan catur. *AI* yang ditanamkan dalam permainan catur sudah sangat terkenal dimana *AI* tersebut dapat mengalahkan juara dunia. Pada algoritma *minimax*, pengecekan akan dilakukan menurut *rule* yang ada sehingga didapat seluruh kemungkinan yang ada sampai akhir permainan dilakukan. Pengecekan tersebut akan menghasilkan pohon permainan yang berisi semua kemungkinan dari permainan tersebut. Tentunya dibutuhkan *resource* yang berskala besar untuk menangani komputasi pencarian pohon solusi tersebut berhubung kombinasi kemungkinan untuk sebuah permainan catur pada setiap geraknya sangat banyak sekali.

Berbeda dengan permainan catur yang memerlukan *rule* yang banyak serta *knowledge base* yang lengkap, Pada permainan *tic-tac-toe* ini mempunyai lebih sedikit *rule* dan *knowledge base* yang cenderung lebih sedikit. Dari *rule* yang ada mampu menghasilkan solusi-solusi yang dapat digunakan untuk memenangkan permainan. Dari *rule* yang sedikit, komputer

mampu melakukan komputasi untuk memainkan setiap kombinasi langkah dari setiap posisi dan kondisi. Namun hal ini dapat dihindari dengan membatasi sejauh mana komputer akan menganalisis hasil dari langkah langkah yang mungkin (menentukan kedalaman pohon). Tetapi dengan hal ini, kita harus menambah kedalaman pohon tersebut setiap langkahnya agar kedalaman pohon pada *state* tersebut sama dengan *state* sebelumnya

Algoritma *minimax* ini bekerja secara rekursif dengan mencari langkah yang akan membuat lawan mengalami keuntungan *minimum*. Semua strategi lawan akan dihitung dengan algoritma yang sama dan seterusnya. Ini berarti, pada langkah pertama komputer akan menganalisis seluruh pohon permainan. Dan untuk setiap langkahnya, komputer akan memilih langkah yang membuat lawan mendapatkan keuntungan *minimum*, dan membuat komputer itu sendiri mendapatkan keuntungan maksimum.

Dalam penentuan keputusan tersebut dibutuhkan suatu nilai yang merepresentasikan kerugian atau keuntungan yang akan diperoleh jika langkah tersebut dipilih. Untuk itulah disini digunakan sebuah fungsi *heurisitic* untuk mengevaluasi nilai sebagai nilai yang merepresentasikan hasil permainan yang akan terjadi jika langkah tersebut dipilih. Pada umumnya pada permainan *tic-tac-toe* ini digunakan nilai 1, 0, -1 untuk mewakilkan hasil akhir permainan berupa menang, seri, dan kalah. Dari nilai-nilai *heuristic* inilah komputer akan menentukan simpul mana dari pohon permainan yang akan dipilih, tentunya simpul yang akan dipilih tersebut adalah simpul dengan nilai *heuristic* yang akan menuntun permainan ke hasil akhir yang menguntungkan bagi komputer.

1..3 HTML

Menurut Anhar (2010:40) menjelaskan bahwa "Hypertext Markup Language (HTML) adalah sekumpulan simbol-simbol atau tag-tag yang dituliskan dalam sebuah file yang digunakan untuk menampilkan halaman pada web browser".

Bermula dari sebuah bahasa yang sebelumnya banyak digunakan di dunia penerbitan dan percetakan yang disebut dengan SGML (Standard Generalized Markup Language), HTML adalah sebuah standar yang digunakan secara luas untuk menampilkan halaman web. HTML saat ini merupakan standar internet yang didefinisikan dan dikendalikan penggunaannya oleh World Wide Web Consortium (W3C). HTML pertama kali dibuat oleh kolaborasi Caillau TIM dengan Berners-Lee Robert ketika mereka bekerja di CERN, sebuah lembaga penelitian fisika energi tinggi di Jenewa-Swiss pada tahun 1989.[6]

Berikut garis besar algoritma minimax secara umum :

```
Cari langkah yang dengan nilai maksimum
IF langkah tersebut merupakan langkah kemenangan THEN pilih lagkah tersebut.
ELSE
  FOR EACH kemungkinan langkah yang ada
    Cari langkah lawan yang bernilai minimum.
    RETURN nilai dari langkah tersebut.
  Pilih langkah yang bernilai maksimum dari langkah-langkah tersebut.
```

Gambar 1. Contoh kondisi awal algoritma

Pemakaian algoritma umum diatas untuk permainan *tic tac-toe* adalah sebagai berikut :

```
IF ada langkah kemenangan THEN pilih langkah tersebut.
ELSE IF lawan mempunyai 2 spot terisi dalam satu garis dengan spot ketiga masih kosong THEN tutup langkah tersebut (isi spot kosong ketiga tersebut).
ELSE melangkah ke state yang mempunyai kemungkinan menang tertinggi (berdasarkan nilai heuristic yang dibangkitkan).
```

Gambar 2. Contoh algoritma dalam permainan

Dilihat dari algoritma diatas, nilai *heuristic* yang dibangkitkan akan sangat mempengaruhi analisis dari *AI* tersebut. Oleh karena itu, semakin bagus fungsi *heuristic* maka semakin bagus pula analisis *AI* tersebut, dan semakin sulit pula *AI* untuk dikalahkan

2. Pembahasan

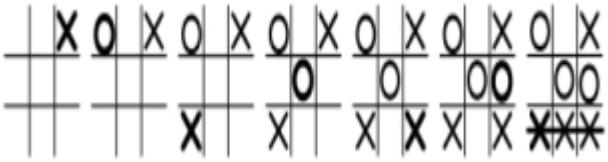
Tic tac toe merupakan salah satu game klasik yang hanya bisa dimainkan oleh dua orang pemain. Pemain biasanya bergiliran mengisi kotak dengan simbol (biasanya lingkaran dan silang) kedalam kotak berukuran 3x3. Pemain akan dianggap menang apabila berhasil membuat simbol menjadi horizontal, vertikal, atau diagonal

Permainan akan berakhir seri apabila pemain tidak dapat membuat lingkaran atau silang apabila kotak sudah penuh. Karena itu, apabila pemain sadar bahwa dirinya tidak bias menang maka ia dapat mencegah lawannya menang dengan menjadikan game berakhir seri. Oleh sebab itu salah satu strategi pemain di atas adalah berusaha bertahan (defense) dengan cara menghalangi

pemain lainnya agar ia tidak dapat membentuk sebuah garis lurus.

Disini, program dibuat dengan menggunakan bahasa HTML. Tujuannya karena HTML mudah dibuat, ringan, serta tidak memerlukan compiler untuk dijalankan sehingga cukup dibuka dengan menggunakan browser.

Berikut contoh permainan *tic-tac-toe* :

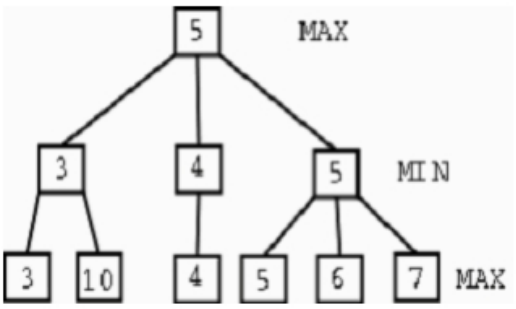


Gambar 3. Contoh kondisi menang permainan *tic-tac-toe*

Gambar 4. Contoh kondisi seri permainan *tic-tac-toe*.

2.1 Penerapan Algoritma *Minimax*

Sebagai contoh, ada 1 orang pemain yang mencoba bermain melawan komputer (pemain = A dan komputer = B). Apabila pemain A dapat memenangkan permainan dalam 1 langkah, maka pemain A akan menang. Pemain B yang mengetahui bahwa langkah tersebut akan membuat pemain A memenangkan permainan, dan di sisi lain ada langkah lain yang akan membuat permainan menjadi seri, maka pilihan paling logis untuk pemain B adalah langkah yang akan membuat permainan menjadi seri. Pada tahapan algoritma ini kita mengasumsikan bahwa pemain A mencoba untuk memaksimalisasi peluang permainan agar ia dapat menang. Di sisi lain, pada giliran berikutnya pemain B juga berusaha mencoba untuk meminimalisir peluang kemungkinan menang pemain A. Oleh sebab itu, pemain A dapat disebut sebagai *maximizing player* (*MAX*) dan B dapat disebut sebagai *minimizing player* (*MIN*). Sehingga pembentukan pohon pencarian solusi yang digunakan akan memakai konsep *depth-first*, yang akan dimulai dari awal permainan hingga akhir permainan. Setelah itu, posisi terakhir permainan akan dievaluasi melalui sudut pandang *MAX* seperti gambar dibawah ini:



Gambar 5 Representasi pohon pencarian pada algoritma *minimax*.

Setelah itu nilai-nilai dari setiap simpul diisi dari bawah keatas dengan nilai yang telah dievaluasi dengan fungsi *heuristic*. Simpul milik pemain A (*MAX*) menerima nilai maksimum dari simpul-simpul anaknya. Sedangkan simpul pemain B (*MIN*) akan memilih nilai *minimum* dari simpul anak-anaknya.

Berikut ini adalah gambar *pseudo code* dari algoritma yang digunakan:

```
199 function AITurn(){
200     if(giliran != AI) return;
201     if(hitungKosong == 0) return;
202
203     copyBoard();
204
205     giliranAI = AI;
206
207     var res,ci,cj,choose = -1000;
208
209     for(var i = 0 ; i < kotakAI.length ; i++)
210     {
211         for(var j = 0 ; j < kotakAI[i].length ; j++)
212         {
213             if(kotakAI[i][j] == kosong)
214             {
215                 kotakAI[i][j] = giliranAI;
216                 hitungKosong--;
217                 cekGiliran();
218                 res = search(1);
219                 kotakAI[i][j] = kosong;
220                 hitungKosong++;
221                 if(choose == -1000)
222                 {
223                     choose = res;
224                     ci = i;
225                     cj = j;
226                 }
227                 else if(res > choose)
228                 {
229                     choose = res;
230                     ci = i;
231                     cj = j;
```

```
                }
                cekGiliran();
            }
        }
    }

    currentGame[ci][cj] = AI;

    hitungKosong--;

    setSign(ci,cj,AI,COLOR);

    swapTurn();
}


function search(level)
{
    var res = determine(kotakAI);
    if(res == AI)
    {
        return  100-level;
    }
    else if(res == player)
    {
        return level-100;
    }
    else if(res == "draw")/
    {
        return 0;
    }

    var choose = -1000;
```

Gambar 6. Script perpindahan posisi *AI*

Gambar diatas menunjukkan bagaimana pergerakan suatu langkah dalam permainan. Jadi, pemain A akan memilih langkah yang memiliki nilai paling tinggi di akhir. Sedangkan pemain B akan menghalangi kemenangan pemain A dengan memilih langkah yang terbaik untuknya, yaitu dengan cara meminimalisir hasil dari langkah yang dipilih pemain A.

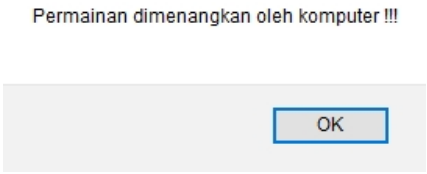
Dari *psudo code* diatas, *AI* akan selalu memilih langkah yang meminimalisir kemungkinan pemain (manusia) untuk menang dengan menghalangi semua langkah yang hasilnya menguntungkan pemain. Oleh sebab itu, permainan akan selalu berakhir seri jika pemain cukup teliti dalam menentukan langkah. Namun apabila pemain mengambil langkah yang salah, *AI* akan langsung menggunakan kesempatan tersebut untuk mengalahkan player.

2.2 Antarmuka Aplikasi

Berikut adalah antarmuka dari aplikasi yang telah dibuat

Gambar 7. Antarmuka aplikasi tic-tac-toe

Jika permainan telah selesai dan pengguna kalah, maka akan muncul pemberitahuan seperti pada gambar di bawah



Gambar 8. Pemberitahuan dari aplikasi tic-tac-toe

3. Kesimpulan

Dari hasil penelitian yang telah dilakukan, terdapat beberapa hal yang dapat diambil kesimpulannya, antara lain:

1. Algoritma *minimax* merupakan algoritma yang sangat bagus dan cocok untuk pengambilan keputusan oleh *AI*, terutama dalam permainan n *player* (n>=2).
2. Pohon solusi dibentuk dari awal permainan sampai akhir permainan
3. Algoritma ini menggunakan DFS dalam pembentukan pohon solusi
4. Pohon solusi akan susah terbentuk pada permainan yang memiliki kemungkinan sangat besar seperti catur.

Dengan menggunakan algoritma *minimax* untuk *AI* dalam permainan *tic-tac-toe*, pemain (manusia) tidak akan dapat mengalahkan *AI*.

Daftar Pustaka

[1] Paputungan, I. V., "*Konsep Permainan Tic-Tac-Toe Menggunakan Algoritma Genetika*," Juni 17, 2006.
[2] Dahria, M, "Kecerdasan Buatan (Artificial Intelligence)", *SAINTIKOM,* Vol.V, pp. 185-196, Agustus 2, 2008.
[3] Nasution, H, "Implementasi Logika Fuzzy pada Sistem Kecerdasan Buatan", *ELKHA,* Vol.IV, pp. 4-7, 2012.
[4] Kristanto, A. 2004. *Kecerdasan Buatan. Yogyakarta*: Graha Ilmu.
[5] Ayuningtyas, N, "*Algoritma Minimax Dalam Permainan Checkers*," 2006.
[6] Anhar. 2010. Panduan Menguasai PHP dan Mysql. Jakarta: Media Kita.

Biodata Penulis

Muhammad Kurniawan, saat ini sedang menempuh pendidikan Sarjana pada Jurusan Teknik Informatika di STMIK AMIKOM Yogyakarta.

Afib Pamungkas, saat ini sedang menempuh pendidikan Sarjana pada Jurusan Teknik Informatika di STMIK AMIKOM Yogyakarta.

Salman Hadi, saat ini sedang menempuh pendidikan Sarjana pada Jurusan Teknik Informatika di STMIK AMIKOM Yogyakarta.

ISSN : 2302-3805