

KONSEP PERMAINAN TIC-TAC-TOE MENGGUNAKAN ALGORITMA GENETIKA

Irving Vitra Papatungan

Laboratorium Komputasi dan Sistem Cerdas, Jurusan Teknik Informatika, Fakultas Teknologi Industri
Universitas Islam Indonesia, Yogyakarta

E-mail: ipink@engineer.com

ABSTRAKSI

Makalah ini akan mencoba menjelaskan sebuah konsep algoritma genetika dan pembelajaran mesin yang diterapkan pada permainan tic-tac-toe. Setelah pembelajaran dari strategi-strategi untuk sebuah permainan didapat, algoritma genetika diharapkan mampu membantu pemain memainkan tic-tac-toe dengan baik. Hasil dari proses ini diharapkan dapat dibandingkan dengan beberapa metode-metode yang lain dalam lingkup kecerdasan buatan.

Kata kunci: Algoritma Genetika, Pembelajaran Mesin, Permainan, Tic-Tac-Toe.

1. PENDAHULUAN

Memainkan sebuah permainan merupakan sebuah konsep penelitian yang sangat populer dikalangan komunitas AI (kecerdasan buatan). Para peneliti di bidang itu telah mengembangkan beberapa metode pencarian untuk permainan. Metode-metode pencarian yang dikembangkan bisa berupa pencarian secara tradisional maupun secara heuristic yang lebih cerdas. Penelitian ini akan mencoba memfokuskan pada permainan sederhana yang dinamakan Tic-Tac-Toe. Algoritma genetika mempunyai ciri yaitu mampu membangun beberapa strategi pada berbagai situasi, dan melalui aturan 'kemampuan bertahan hidup', beberapa strategi yang tidak baik dapat dieliminasi sehingga hanya strategi baik yang dipakai.

Walaupun Tic-Tac-Toe merupakan sebuah permainan sederhana, penelitian ini akan mencoba menerapkan konsep algoritma genetika dan pembelajaran mesin untuk mencari strategi-strategi yang pas. 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.

2. STUDI PUSTAKA

Ada beberapa penelitian sejenis yang sudah dilakukan. Misalnya seperti yang dilakukan Samuel dalam *Some Studies in Machine Learning Using the game of Checkers* [6]. Pada makalah tersebut, Samuel menganalisa pembelajaran mesin dan permainan checkers. Samuel menggunakan 26 parameter pada fungsi polinomialnya, berikut nilai-nilainya untuk menentukan bagaimana posisi biji pada papan yang tepat. Tujuannya adalah untuk mencari nilai X_n yang paling efektif dalam menerapkan strategi permainan checkers. Setelah beberapa kali percobaan dilakukan, diperoleh sebuah strategi yang mampu terbaik sehingga mampu bertahan sampai akhir permainan. Sayangnya, proses

ini memerlukan waktu yang panjang. Akan tetapi, Samuel menemukan sebuah strategi yang bagus sehingga mampu mengalahkan manusia. Penelitian ini diharapkan mampu menemukan strategi yang serupa terkait dengan permainan Tic-Tac-Toe, dan lebih efektif dengan waktu yang tidak terlalu lama.

Penelitian lain yang sejenis juga telah dilakukan oleh Michie[5]. Michie menganalisa proses pembelajaran mesin dan permainan Tic-Tac-Toe dan menemukan sekitar 300 posisi yang berbeda untuk pemain melakukan langkah awal. Percobaan dilakukan pada semua langkah tersebut dengan memberikan nilai pada tiap langkah, nilai yang tertinggi, akan memulai langkah terlebih dahulu. Pada bagian seleksi menggunakan rumus, jika langkah tersebut menghasilkan kemenangan, maka nilai langkah dinaikkan, dan sebaliknya, namun untuk langkah pertama, dilakukan secara acak. Setelah dilakukan beberapa kali permainan, mesin akhirnya mampu mempelajari permainan tic-tac-toe tersebut. Ini menunjukkan bahwa, algoritma genetika mampu mendapatkan strategi yang baru.

Beberapa tulisan juga telah dibuat untuk membahas permasalahan sejenis, misalnya : Ginsberg (1993)[2], Haykin (1994)[4], Goldberg (1989)[3] dan Bagley(1967)[1]. Tic-tac-toe, merupakan permasalahan permainan yang kompleks dibandingkan permainan lain sejenis, misalnya 'Hexapawn game'. Perbandingan konfigurasinya mencapai 52:612. ini menyebabkan permainan Tic-Tac-Toe sangat sulit dipecahkan.

3. THE GAME

Permainan Tic-Tac-Toe sangatlah sederhana; diilhami oleh permainan anak-anak. Dimainkan pada papan berukuran 3 x 3. Pada awal permainan, papan dikosongkan. Kedua pemain, X dan O, akan menempatkan biji-bijinya keatas papan, sekali pasang satu biji. Pemain yang mampu menempatkan tiga bijinya dalam satu garis (vertikal, horizontal, diagonal) itulah yang menang. Dan dikatakan seri

apabila papan telah penuh maupun tidak ada yang menang.

3.1 Metode Sederhana AI

Umumnya, ada dua metode untuk memecahkan masalah semacam Tic-Tac-Toe. (1) metode tradisional dan (2) metode pengurutan dan pencarian Heuristic.

Untuk menyelesaikan permasalahan pencarian, penggunaan Pohon Keputusan merupakan langkah pertama yang bisa dilakukan, memperluas pohon permainan seluas mungkin, dan menganalisa tiap kemungkinan langkah dan hasilnya. Tiap percabangan juga dapat dianalisa menggunakan Algoritma Minimax yang akan membentuk sebuah fungsi untuk mengevaluasi tiap kemungkinan solusi dan memberikan nilai untuk langkah yang berpeluang memenangkan permainan. Kemungkinan terbesar untuk memenangkan permainan merupakan hasil dari penentuan langkah, dan langkah tersebut ditentukan oleh hasil evaluasi fungsinya.

Sebagai alternatif penyelesaian yang lain, beberapa aturan pencarian heuristic dapat digunakan untuk mengurangi beberapa cabang pohon dari pohon keputusan yang didapat sehingga terlihat lebih pendek. Contohnya menggunakan *Depth First Search*.

4. ALGORITMA GENETIKA

Algoritma Genetika pada dasarnya adalah program komputer yang mensimulasikan proses evolusi. Dalam hal ini populasi dari kromosom dihasilkan secara random dan memungkinkan untuk berkembang biak sesuai dengan hukum-hukum evolusi dengan harapan akan menghasilkan individu kromosom yang prima. Kromosom ini pada kenyataannya adalah kandidat penyelesaian dari masalah, sehingga bila kromosom yang baik berkembang, solusi yang baik terhadap masalah diharapkan akan dihasilkan [7].

Algoritma Genetika ini banyak dipakai pada aplikasi bisnis, teknik maupun pada bidang keilmuan. Algoritma ini dapat dipakai untuk mendapatkan solusi yang tepat untuk masalah optimal dari satu variabel atau multi variabel. Sebelum algoritma ini dijalankan, masalah apa yang ingin dioptimalkan itu harus dinyatakan dalam fungsi tujuan, yang dikenal dengan fungsi fitness. Jika nilai fitness semakin besar, maka sistem yang dihasilkan semakin baik. Walaupun pada awalnya semua nilai fitness kemungkinan sangat kecil (karena algoritma ini menghasilkannya secara random), sebagian akan lebih tinggi dari yang lain. Kromosom dengan nilai fitness yang tinggi ini akan memberikan probabilitas yang tinggi untuk bereproduksi pada generasi selanjutnya. Sehingga untuk setiap generasi pada proses evolusi, fungsi fitness yang mensimulasikan seleksi alam, akan menekan populasi kearah fitness yang meningkat.

Algoritma genetika sangat tepat digunakan untuk penyelesaian masalah optimasi yang kompleks dan sukar diselesaikan dengan menggunakan metode yang konvensional. Sebagaimana halnya proses evolusi di alam, suatu algoritma genetika yang sederhana umumnya terdiri dari tiga operator yaitu: operator reproduksi, operator *crossover* (persilangan) dan operator mutasi. Struktur umum dari suatu algoritma genetika dapat didefinisikan dengan langkah-langkah sebagai berikut:

- Membangkitkan populasi awal, Populasi awal ini dibangkitkan secara random sehingga didapatkan solusi awal. Populasi itu sendiri terdiri dari sejumlah kromosom yang merepresentasikan solusi yang diinginkan.
- Membentuk generasi baru, Dalam membentuk digunakan tiga operator yang telah disebut di atas yaitu operator reproduksi/seleksi, *crossover* dan mutasi. Proses ini dilakukan berulang-ulang sehingga didapatkan jumlah kromosom yang cukup untuk membentuk generasi baru dimana generasi baru ini merupakan representasi dari solusi baru.
- Evaluasi solusi, Proses ini akan mengevaluasi setiap populasi dengan menghitung nilai fitness setiap kromosom dan mengevaluasinya sampai terpenuhi kriteria berhenti. Bila kriteria berhenti belum terpenuhi maka akan dibentuk lagi generasi baru dengan mengulangi langkah 2. Beberapa kriteria berhenti yang sering digunakan antara lain:
 - Berhenti pada generasi tertentu.
 - Berhenti setelah dalam beberapa generasi berturut-turut didapatkan nilai fitness tertinggi tidak berubah.
 - Berhenti bila dalam n generasi berikut tidak didapatkan nilai fitness yang lebih tinggi.

5. IMPLEMENTASI

5.1 Konfigurasi Pengkodean

Untuk membangun strategi dalam permainan Tic-tac-toe, ada sebuah langkah jitu di tiap konfigurasi papan. Tic-tac-toe dimainkan pada papan dengan sembilan kotak. Tiap kotak dapat terisi oleh 'X' atau 'O' maupun kosong. Total jumlah kemungkinan konfigurasi adalah 3 pangkat 9 atau 19683. Jika diasumsikan bahwa akan dibentuk sebuah susunan angka bayangan, maka '0' akan merepresentasikan kosong, 1 untuk 'X' dan 2 untuk 'O'. Misalnya:

	X	O

Gambar 1. contoh konfigurasi

Angka bayangan: 000000012

Angka desimal: 5

5.2 Inisialisasi

Pertama, semua konfigurasi yang valid harus diperkirakan seluruhnya dengan baik. Untuk pemain 'X', yang akan melangkah terlebih dahulu, jumlah X dan O pada papan dan pada satu waktu, haruslah sama. Sebaliknya, untuk pemain 'O', harus ada X terlebih dahulu sebelum melangkah. Inilah yang menyebabkan banyaknya konfigurasi yang akan terjadi. Contoh konfigurasi untuk pemain 'X' :

Tabel 1. Konfigurasi pemain X

Konfigurasi	0	5	7	11	15	19	...
Index	1	2	3	4	2	4	...
Variasi	0	0	0	0	1	2	...

Index adalah posisi nantinya didalam string gen. Pastikan bahwa konfigurasi 1-4 tidak masuk kedalam tabel, karena keempatnya invalid untuk pemain 'X'. Konfigurasi 5 dan 15 memuat index yang sama, artinya 5 dan 15 merupakan varian untuk konfigurasi yang sama.

	X	O

Gambar 2. Konfigurasi 5

O	X	

Gambar 3. Konfigurasi 15

Seperti yang terlihat diatas, keduanya hanya dicerminkan. Sehingga tidak diperlukan tambahan string untuk merepresentasikan 15. Pada algoritma genetika, tabel akan dijabarkan untuk melihat apakah ada kesamaan dengan konfigurasi 2. Situasi yang sama juga muncul pada konfigurasi 9 dan 11. Sehingga, keduanya mempunyai index yang sama. Tabel diatas juga memiliki sebuah bidang variasi untuk menggambarkan bagaimana konfigurasi tersebut terkait dengan aslinya. Misalnya, '0' untuk menunjukkan bahwa konfigurasi itu original, '1' menunjukkan cerminan secara horizontal, dan '2' cerminan secara vertikal, dan sebagainya.

Setelah apa yang dijelaskan diatas dieksekusi, maka akan dihasilkan 612 index yang berarti terdapat 612 konfigurasi yang berbeda untuk pemain 'X'. Hal yang sama juga dilakukan pada pemain 'O'.

5.3 Representasi String

Ketika ditemukan 612 konfigurasi yang berbeda dan tiap konfigurasi mempunyai 9 reaksi (jumlah abjadnya 9), sebuah string dengan panjang 612 dibutuhkan. Tiap bit akan merepresentasikan sebuah langkah yang terkait dengan index konfigurasi. Sebagai alternatif, abjad pengganti 0,1, dan 2 dapat dipakai, tetapi akan menyebabkan panjang string menjadi 1224 (dua kalinya).

Tabel 2. Penggunaan Cara Kedua

Index	1	2	3	4	5	6	...
Isi	5	0	4	6	3	8	...

Tabel 3. Penggunaan Cara Kedua

Index	1	3	5	7	9	11	...
Isi	12	0	10	20	10	21	...

Kedua *string* merepresentasikan strategi yang sama. Secara umum, algoritma genetika akan bekerja lebih baik bila menggunakan jumlah abjad yang kecil. Oleh karena itu, metode kedua akan dipakai.

5.4 Pengkodean String

Untuk konfigurasi 5 (**Gambar 2.**), papan akan dikodekan 000000012 (3) = 5 (10). Sedangkan pada tabel sebelumnya, konfigurasi 5 menunjukkan indeks 2, kemudian string pada indeks 2 digunakan (jika menggunakan metode pertama) atau indeks 3 (metode kedua). Dikarenakan menggunakan metode 2, maka hasilnya adalah 0. Sehingga, pemain akan meletakkan 'X' pada papan posisi 0. Terlihat pada gambar :

X		
	X	O

Gambar 4. Langkah Selanjutnya

Apabila langkah selanjutnya akan memilih posisi yang sudah ditempati, misalnya pada posisi ke 8, maka akan diacak angka 0-8 untuk menentukan langkah tersebut. Tentunya, apabila pengacakan tersebut mendapatkan posisi yang tepat, maka kemampuan hidup string tersebut akan lebih lama, dan sebaliknya.

5.5 Fungsi Fitnes

Apabila kedua pemain bermain secara bergantian, maka akan terdapat dua populasi yang akan dihasilkan sebagai inisialisasi. Satu untuk 'X', sisanya untuk 'O'. Dalam masalah ini, apabila sejumlah 30 string dihasilkan pada tiap populasi,

dan tiap populasi memainkan 5 permainan, maka akan ada 30*30*5 permainan di tiap generasi. Fungsi fitness akan tergantung dari catatan menang-kalah-seri string tersebut. Dalam hal ini, apabila 100 string digunakan, dan tiap string memainkan 50 permainan, maka akan terdapat 5000 permainan yang akan dimainkan di setiap generasi.

Tabel 4. Pengaturan Skor

Pemain	Menang	Kalah	Seri
Pemain X	+4	+1	0
Pemain O	+5	+2	0

Perbedaan antara pemain 'X' dan 'O' adalah pada pemain 'X' akan bertujuan untuk memenangkan permainan, karena memulai langkah terlebih dahulu. Dengan memulai terlebih dahulu, peluang untuk mendapatkan skor seri, adalah kecil. Sebaliknya, pemain 'O' akan berusaha untuk menahan pemain 'X' agar tidak memenangkan permainan, sehingga jika permainan berakhir seri, pemain 'O' menganggap sukses, apalagi jika berhasil menangkannya. Berdasarkan tabel diatas, sebagai contoh, apabila pemain 'X' memiliki catatan 11-5-4 setelah 30 kali permainan, maka fitness yang didapat adalah 49, sedangkan untuk 'O' adalah 65.

5.6 Rekombinasi dan Mutasi

Setelah melewati generasi pertama, terdapat 30 string yang terkait dengan skor dan nilai fitness. Selanjutnya, dengan menggunakan seleksi model roda rolet, akan dihasilkan 30 pasang string. Misalnya, apabila inisialisasi diberikan seperti tabel dibawah, maka string pertama akan memiliki probabilitas sebesar 34/430 atau 8% untuk lolos ke generasi selanjutnya. Lebih dari 30 seleksi, akan didapat probabilitas sebesar 91% untuk sebuah kromosom (string) mempunyai susunan yang sama pada generasi selanjutnya. begitu juga pada string yang kedua, akan mempunyai probabilitas 0,9% setelah satu kali seleksi dan 24% setelah 30 seleksi.

Tabel 5. Skor kombinasi dan mutasi

String	1	2	3	4	5	...	Total
Skor	34	4	23	12	2	...	430

Intinya, bagaimana supaya tiap string mampu memiliki nilai fitness yang besar sehingga mampu bertahan pada generasi selanjutnya. Untuk permasalahan ini akan dicoba persilangan model PMX dan OX, sedangkan untuk mutasinya akan dicoba model *Insertion* dan *Reciprocal Mutation*.

5.7 Pemilihan Parameter Algoritma Genetika

Awalnya, proses algoritma genetika digunakan untuk mengoptimasi pemain 'X'. Abjad bayangan yang digunakan memuat 0,1, dan 2, sehingga tiap langkah akan mengkombinasikan dua dari yang ada dan menghasilkan angka antara 0 – 8.

Pertama, lakukan inisialisasi pada level papan, dan konversikan ini kedalam sebuah file sehingga algoritma genetika tidak harus melakukan komputasi setiap mengevaluasi tiap string kromosom. Data valid disimpan, dan abaikan yang invalid. Abaikan juga langkah dari 'X' yang mungkin langsung menang di langkah berikutnya. Selanjutnya, eksekusi algoritma genetika dapat dilakukan. Jumlah individu yang dipakai sejumlah 100 populasi. Lawan akan diberikan strategi:

- Menangkan permainan: lewati semua kotak kosong dan jika memungkinkan untuk menaruh biji di langkah tersebut, lakukan, dan seterusnya.
- Blok lawan : lewati semua kotak kosong, jika lawan memungkinkan menang pada langkah tersebut, maka taruh biji di langkah itu untuk menahan.
- Lakukan langkah acak.

Langkah-langkah tersebut bukanlah yang terbaik, akan tetapi, dengan langkah ini, pemain mampu mengetahui kapan melakukan blok ataupun memenangkan permainan. Jika ini mampu dilakukan, maka strategi algoritma genetika sudah dianggap cukup untuk menjalankan Tic-Tac-Toe.

6. KESIMPULAN

- Dengan algoritma genetika terkait dengan pembelajaran mesin, diharapkan dapat dihasilkan sebuah strategi yang optimal untuk menjalani sebuah permainan sederhana
- Pemilihan operator-operator genetika yang kompleks, diharapkan mampu menghasilkan strategi yang lebih baik.

DAFTAR PUSTAKA

- [1] Bagley, J.D. (1967). *The behaviour of adaptive systems which employ genetic and correlation algorithms*. Doctoral dissertation, Ann arbor: The University of Michigan.
- [2] Ginsberg, M. (1993). *Essentials of artificial intelligence*. New york: McGraw Hill. Pp 49-99
- [3] Goldberg, D.E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: addison-wesley.
- [4] Haykin, S. (1994). *Neural-networks – a comprehensive foundation*. Pp. 397-434. new york: McGraw Hill.
- [5] Michie, D. (1962). *Trial and Error*. new york: McGraw Hill.
- [6] Samuel, A.L. (1959). *Some studies in machine learning using the game of checkers*. new york: McGraw Hill.
- [7] Sukmawan, Budi. (2004). *Sekilas tentang Algoritma Genetika dan Aplikasinya pada optimasi jaringan pipa bersih*.