

Implementasi Algoritma Minimax Pada Permainan Checker Berbasis *Artificial Intelligence*

Mahrizal Masri¹⁾, Iliyen Uari²⁾, Hermansyah Alam³⁾, M. Ilmi Zikri Firdaus⁴⁾

¹⁾Dosen Tetap Prodi Teknik Elektro Fakultas Teknik UISU

²⁾Alumni Teknik Informatika Institut Teknologi Medan

³⁾Dosen Tetap Teknik Elektro UNPAB

⁴⁾Alumni Teknik Elektro ITM

Abstrak

Permainan (*game*) sering memanfaatkan kecerdasan buatan (*Artificial Intelligence*) untuk memberikan tantangan berupa kesempatan untuk melawan mesin yang memiliki tingkat intelektualitas tertentu dalam berpikir. Algoritma Minimax merupakan salah satu dalam bidang kecerdasan buatan yang dapat melakukan kalkulasi langkah dengan pertimbangan yang logis, berdasarkan aturan permainan yang digunakan, khususnya untuk permainan yang melibatkan dua pemain seperti permainan checker. Dalam penelitian ini, algoritma minimax akan diimplementasikan ke dalam permainan checkers, dalam bentuk sebuah computer AI yang dapat mengambil keputusan langkah secara otomatis pada saat permainan berlangsung. Hasil implementasi algoritma minimax kedalam permainan checker yang diteliti pada penelitian ini mampu menunjukkan bagaimana proses berfikir (*self thinking*) dari computer AI berdasarkan konfigurasi tingkat kesulitan yang dipilih oleh pengguna.

Kata Kunci : Kecerdasan Buatan, Algoritma Minimax, Checker

I. PENDAHULUAN

Permainan (*game*) sering memanfaatkan kecerdasan buatan (*Artificial Intelligence*) untuk memberikan tantangan di dalam permainan tersebut. Tantangan ini dapat berupa kesempatan untuk melawan mesin yang memiliki tingkat intelektualitas tertentu dalam berpikir, sehingga pengguna tidak harus mencari lawan tanding yang lain jika ingin bermain. Kemampuan berpikir yang diterapkan di dalam suatu permainan didukung oleh algoritma yang tentunya harus memiliki efisiensi dan efektifitas sesuai dengan peraturan permainan yang digunakan. Salah satu algoritma yang dapat diterapkan di dalam permainan adalah algoritma *minimax*. *Minimax* dapat melakukan kalkulasi langkah dengan pertimbangan yang logis, berdasarkan aturan permainan yang digunakan. Secara khusus, algoritma ini sering digunakan untuk permainan yang melibatkan dua pemain, dimana untuk setiap langkah (*turn*), akan dilakukan kalkulasi untuk mencari langkah yang paling optimal dengan mempertimbangkan kemungkinan langkah yang akan dilakukan oleh lawan. Dalam algoritma *minimax*, seluruh kemungkinan langkah akan dievaluasi dalam sebuah pohon permainan, dengan menggunakan fungsi heuristik untuk menentukan simpul *max* dan simpul *min* yang digunakan sebagai dasar untuk menentukan langkah di dalam sebuah permainan. Simpul *max* dan simpul *min* di dalam pohon kemungkinan ini digunakan untuk melihat tingkat keuntungan atau kerugian dari setiap kemungkinan langkah, sehingga dapat diambil satu langkah yang paling efisien dan efektif untuk mengalahkan lawan. Dengan melihat deskripsi dari algoritma ini, *minimax* sangat cocok untuk diterapkan ke dalam

permainan seperti *othello*, *checkers*, *bridge*, *tic-tac-toe* atau *backgammon* yang hanya melibatkan dua pemain, dimana setiap pemain akan berusaha untuk mengalahkan pemain lawan dengan mempertimbangkan langkah yang paling efisien dan paling efektif.

Dalam penelitian ini, algoritma *minimax* akan diimplementasikan ke dalam permainan *checkers*, dimana pengguna dapat memilih satu opsi untuk melawan komputer yang telah diberikan kemampuan berpikir berdasarkan cara kerja algoritma *minimax*.

Aturan permainan *checkers* yang digunakan adalah aturan umum, dimana setiap pemain hanya dapat menjalankan koin yang dimilikinya secara diagonal dengan jumlah maksimal langkah sebanyak dua kotak. Setiap koin yang berhasil mencapai batas akhir dari kotak lawan, maka akan diberikan promosi, dimana koin tersebut dapat bergerak maju atau mundur. Permainan dinyatakan berakhir jika salah satu pemain berhasil menghabiskan seluruh koin lawan atau berhasil seluruh koin yang dimilikinya berhasil mencapai batas akhir dari kotak lawan.

II. METODE PENELITIAN

2.1 Pengumpulan Data

Dalam melakukan penelitian ini, metode pengumpulan data yang dilakukan adalah sebagai berikut :

1. Penelitian Kepustakaan

Data yang dikumpulkan dari hasil penelitian kepustakaan adalah berupa teori mengenai aturan permainan checker, kecerdasan buatan (*artificial intelligence*) dan cara kerja algoritma *minimax*

yang dikumpulkan dari sumber buku bacaan serta jurnal penelitian.

2. Penelitian Lapangan

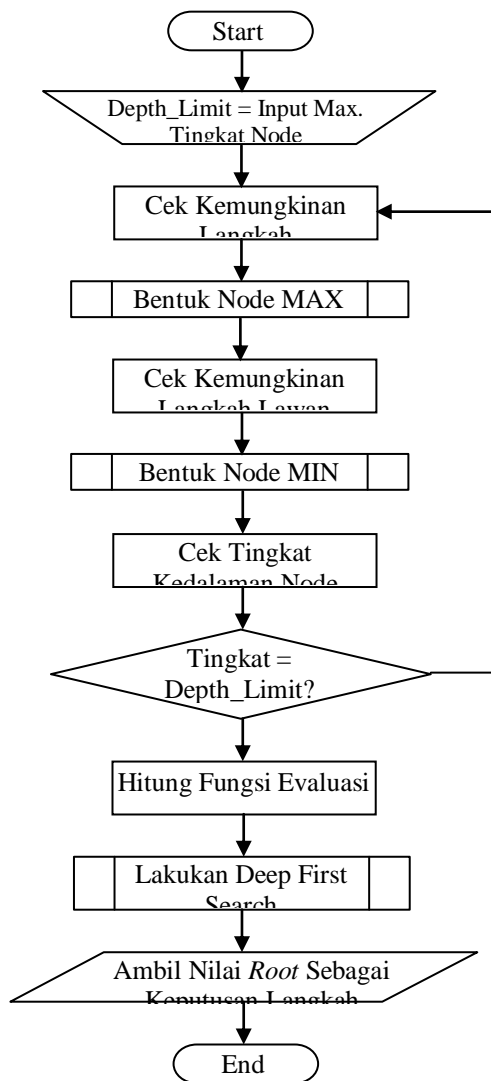
Data yang dikumpulkan dari hasil penelitian lapangan adalah berupa pengamatan langsung dari orang yang sedang bermain *checker*, untuk melihat bagaimana aturan permainan *checker* yang sebenarnya.

2.2 Analisa Sistem

Dalam perancangan aplikasi permainan *checker* yang menggunakan optimasi algoritma *minimax* ini, dilakukan beberapa analisa terhadap sistem yang akan dirancang. Adapun analisa tersebut terbagi menjadi dua bagian, yaitu analisa aturan permainan *checker* serta analisa algoritma *minimax*

2.3 Rancangan Penelitian

Dalam perancangan aplikasi permainan *checkers* dengan menggunakan optimasi algoritma *minimax* ini, tahapan perancangan yang dilakukan terbagi menjadi dua bagian, yaitu rancangan proses serta rancangan antarmuka program.



Gambar 1. Algoritma *Minimax* Pada Permainan *Checker*

2.4 Rancangan Proses

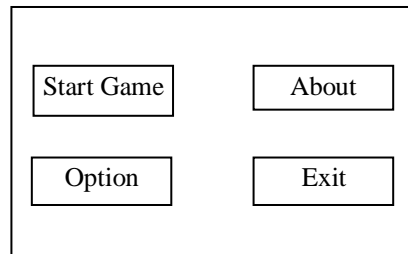
Berdasarkan hasil analisa sistem yang dilakukan, dirancang sebuah alur proses yang akan digunakan dalam perancangan aplikasi. Adapun bentuk rancangan proses algoritma *minimax* yang diimplementasikan ke dalam aplikasi permainan *checker* ini sebagaimana terlihat pada *flowchart* yang ditunjukkan oleh Gambar 1 diatas

2.5 Rancangan Antarmuka

Dalam rancangan antarmuka ini, penulis merancang beberapa *form* yang akan digunakan sebagai media interaksi antara sistem dengan pengguna. Ada empat *form* yang dirancang pada aplikasi permainan *checker* dengan optimasi algoritma *minimax* ini, yaitu *form* Main, *form* Game, *form* Option dan *form* About.

1. Rancangan *Form* Main

Form Main dirancang sebagai *form* induk dari aplikasi ini, *form* ini dirancang untuk tampil pertama kali dan sebagai media untuk memilih opsi menu seperti memulai permainan, memilih opsi tingkat kesulitan AI serta tampilan info permainan. Adapun bentuk rancangan *form* Main sebagaimana terlihat pada Gambar 2



Gambar 2. Rancangan *Form* Main

2. Rancangan *Form* Game

Form Game dirancang sebagai *form* yang menampilkan permainan *checker*. *Form* ini juga berfungsi untuk menampilkan kalkulasi langkah yang diambil oleh *computer* AI dengan menggunakan optimasi algoritma *minimax*. Adapun bentuk rancangan *form* Game sebagaimana terlihat pada Gambar 3.

3. Rancangan *Form* Option

Form Option dirancang untuk menampilkan konfigurasi *computer* AI yang digunakan di dalam penentuan langkah pada permainan *checker*. Pada *form* ini terdapat beberapa pilihan yang dapat diubah oleh pengguna seperti batas waktu pencarian alternatif langkah, jumlah maksimum alternatif langkah yang ditelusuri, jumlah maksimum kedalaman *node* yang diperiksa, serta jumlah maksimum nilai probabilitas (nilai MAX atau MIN) yang digunakan di dalam pencarian langkah yang paling optimum. Adapun bentuk rancangan *form* Option sebagaimana terlihat pada Gambar 4.

Checkers	X																
GameMovementsSetup Versus PlayerUndoSetup Mode Versus AIRedoClear Board																	
Remaining Pieces Whit <input type="text"/> Black <input type="text"/>																	
Minimax Calculation For Computer <table border="1"> <thead> <tr> <th>Evaluation</th> <th>Results</th> </tr> </thead> <tbody> <tr> <td>Node's Denth</td> <td><input type="text"/></td> </tr> <tr> <td>Total Movements</td> <td><input type="text"/></td> </tr> <tr> <td>Total Time</td> <td><input type="text"/></td> </tr> <tr> <td>Average Speed</td> <td><input type="text"/></td> </tr> <tr> <td colspan="2">Final</td> </tr> <tr> <td>From</td> <td><input type="text"/></td> </tr> <tr> <td>To</td> <td><input type="text"/></td> </tr> </tbody> </table>		Evaluation	Results	Node's Denth	<input type="text"/>	Total Movements	<input type="text"/>	Total Time	<input type="text"/>	Average Speed	<input type="text"/>	Final		From	<input type="text"/>	To	<input type="text"/>
Evaluation	Results																
Node's Denth	<input type="text"/>																
Total Movements	<input type="text"/>																
Total Time	<input type="text"/>																
Average Speed	<input type="text"/>																
Final																	
From	<input type="text"/>																
To	<input type="text"/>																

Gambar 3. Rancangan Form Game

Game Option	X										
<table border="1"> <tr> <td>Max. Thinking Time</td> <td><input type="text"/></td> </tr> <tr> <td>Max. Movement Search</td> <td><input type="text"/></td> </tr> <tr> <td>Node's Depth Limit</td> <td><input type="text"/></td> </tr> <tr> <td>Max. Probability Search</td> <td><input type="text"/></td> </tr> <tr> <td>Max. Total Score</td> <td><input type="text"/></td> </tr> </table>	Max. Thinking Time	<input type="text"/>	Max. Movement Search	<input type="text"/>	Node's Depth Limit	<input type="text"/>	Max. Probability Search	<input type="text"/>	Max. Total Score	<input type="text"/>	
Max. Thinking Time	<input type="text"/>										
Max. Movement Search	<input type="text"/>										
Node's Depth Limit	<input type="text"/>										
Max. Probability Search	<input type="text"/>										
Max. Total Score	<input type="text"/>										
Close											

Gambar 4. Rancangan Form Option

4. Rancangan Form About

Form About dirancang untuk menampilkan informasi mengenai permainan checker dengan menggunakan optimasi algoritma minimax ini. Adapun bentuk rancangan form About sebagaimana terlihat pada Gambar 5.

Game Info	X
Checker With Minimax	
In this game, you can play versus The AI Computer movements will You can set the difficulty level by increasing or decreasing the Have fun playing this game. Hope Copyright 2014 by Iliyen Uari	
Close	

Gambar 5. Rancangan Form About

III. HASIL DAN PEMBAHASAN

3.1 Pengujian Aplikasi

Hasil dari perancangan aplikasi permainan checker dengan optimasi algoritma minimax ini diuji untuk melihat apakah hasilnya telah sesuai dengan rencana awal penelitian. Adapun tahapan pengujian yang dilakukan adalah sebagai berikut :

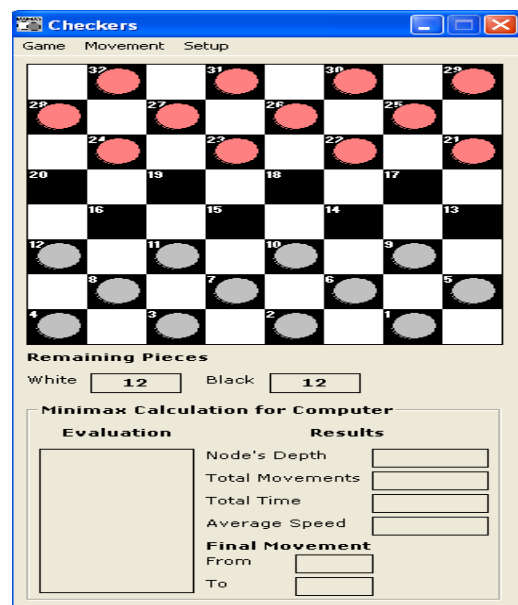
1. Pengujian Form Main

Pengujian form Main dilakukan untuk menguji apakah seluruh opsi yang ditampilkan dapat diakses dengan baik. Untuk itu, dilakukan eksekusi terhadap aplikasi permainan ini sehingga muncul tampilan form Main sebagaimana terlihat pada Gambar 6.



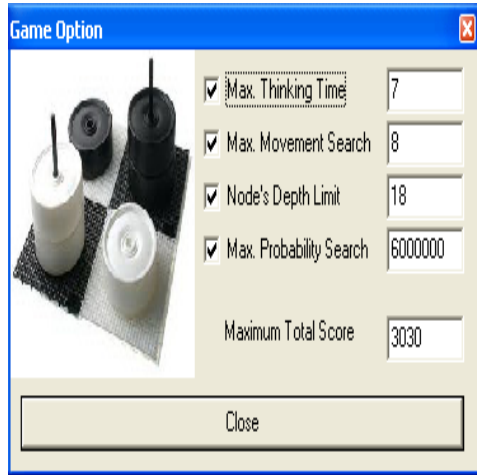
Gambar 6. Tampilan Form Main

Selanjutnya, dilakukan pemilihan menu untuk masing-masing pilihan yang ada seperti Start Game, Option dan About. Untuk menu Start Game, pada saat dilakukan penekanan tombol Start Game pada form Main, hasil yang diperoleh adalah munculnya form Game sebagaimana terlihat pada Gambar 7.



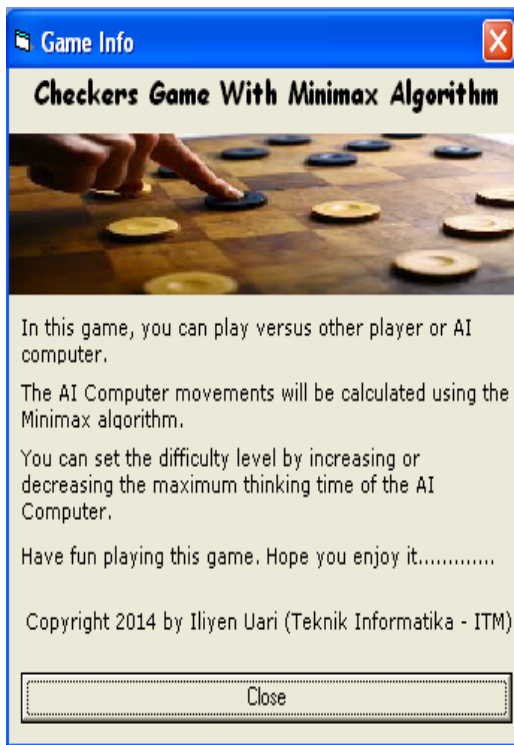
Gambar 7. Tampilan Form Game

Selanjutnya dilakukan penekanan tombol Option pada *form* Main. Hasil yang diperoleh adalah munculnya *form* Option, sebagaimana terlihat pada Gambar 8.



Gambar 8. Tampilan *Form* Option

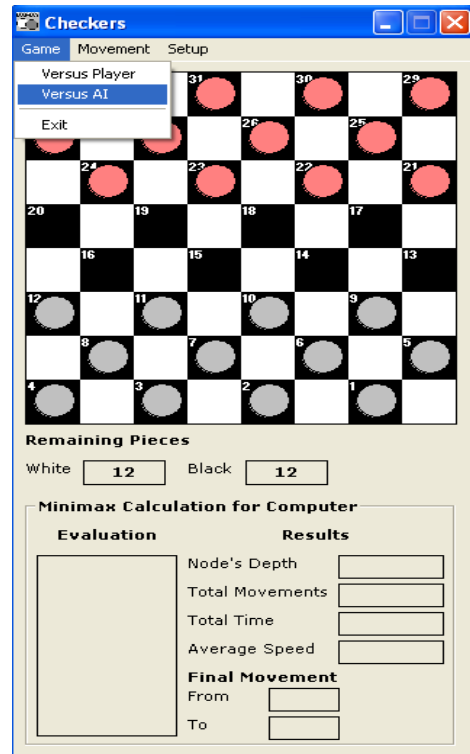
Tahap akhir dari pengujian ini adalah melakukan penekanan terhadap tombol About. Hasil yang diperoleh adalah munculnya *form* About sebagaimana terlihat pada Gambar 9.



Gambar 9 Tampilan *Form* About

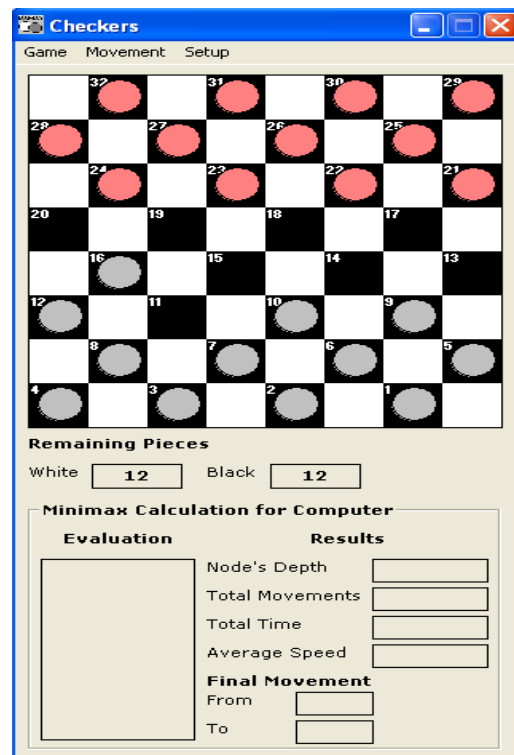
2. Pengujian Permainan *Checker*

Dalam pengujian ini, akan diuji apakah aplikasi permainan telah dapat melakukan *self thinking* untuk mencari langkah optimum pada permainan *checker*. Langkah pertama dalam pengujian ini adalah mengakses *form* Game dan memilih sub menu Versus AI dari menu Game, sebagaimana terlihat pada Gambar 10.



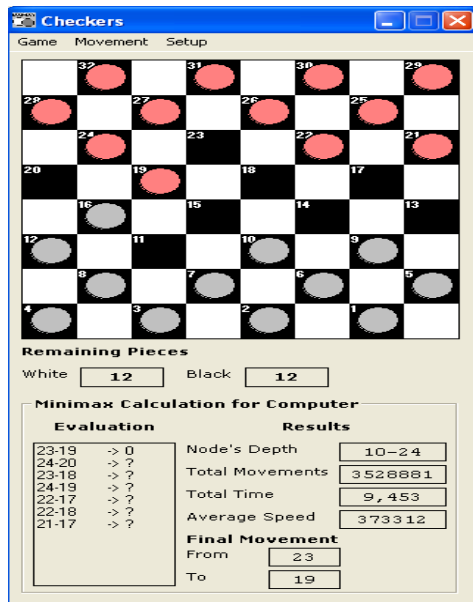
Gambar 10. Memilih Sub Menu Versus AI

Selanjutnya, dilakukan pemindahan koin hitam (*black*) dari posisi 11 menuju posisi 16, sebagaimana terlihat pada Gambar 11.



Gambar 11. Memindahkan Koin Hitam (*Black*)

Setelah langkah pemindahan dilakukan, terlihat bahwa *computer* AI melakukan langkah antisipasi dengan menggerakkan koin putih (*white*) dari posisi 23 menuju posisi 19, sebagaimana terlihat pada Gambar 12.



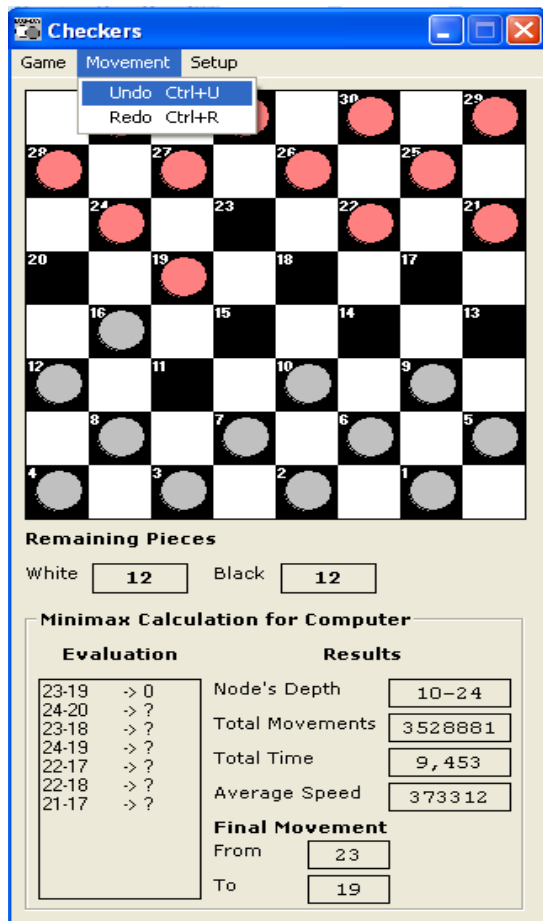
Gambar 12. Pergerakan Computer AI

merupakan nilai yang paling optimum dengan nilai 0.

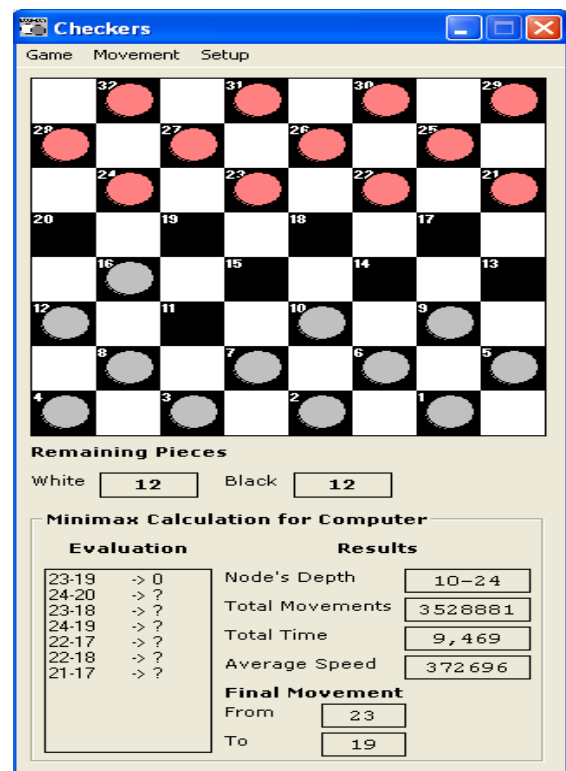
Selain itu, sistem juga dapat menunjukkan berapa tingkat kedalaman *node* yang diperiksa, jumlah total langkah yang diperiksa serta waktu yang dibutuhkan untuk menghasilkan nilai akhirnya.

Pengujian berikutnya adalah menguji apakah aplikasi permainan telah dapat melakukan proses *undo* dan *redo*. Proses *undo* adalah opsi dimana pemain dapat membatalkan langkah terakhir yang diambil oleh *computer* AI maupun langkah dari pemain itu sendiri. Pada pengujian proses *undo* ini, dilakukan pemilihan sub menu Undo dari menu Movements, sebagaimana terlihat pada Gambar 13.

Hasil dari pengujian ini adalah kembalinya koin putih dari posisi 19 menuju posisi 23, sebagaimana terlihat pada Gambar 14.



Gambar 13. Memilih Sub Menu Undo



Gambar 14. Hasil Proses Undo

Langkah yang sama dilakukan untuk proses *redo*, namun yang dipilih adalah sub menu Redo dari menu Movement. Hasil yang diperoleh dari proses pengujian ini adalah posisi koin putih dari posisi 23 kembali dipindahkan ke posisi 19.

Berdasarkan hasil pengujian aplikasi yang dilakukan, diperoleh hasil pembahasan mengenai kinerja algoritma *minimax* pada aplikasi permainan *checker* yang dihasilkan. Adapun hasil pembahasan yang penulis peroleh adalah sebagai berikut :

1. Dalam permainan *checker*, posisi koin pada papan sangat mempengaruhi pengambilan keputusan langkah yang akan diambil pada algoritma *minimax*. Semakin dekat posisi koin

Dari Gambar 12, terlihat bahwa aplikasi permainan telah mampu untuk menunjukkan hasil kalkulasi nilai MIN atau MAX pada kotak informasi Evaluation. Dari beberapa alternatif kemungkinan langkah yang dapat diambil oleh *computer* AI, terlihat bahwa memindahkan posisi koin putih dari posisi 23 menuju posisi 19

pada pinggir papan, akan semakin tinggi nilai dari kemungkinan langkah tersebut, yang selanjutnya akan berpengaruh pada nilai MAX pada algoritma *minimax*. Sebaliknya, semakin jauh posisi koin dari pinggi papan, maka nilai yang dipengaruhinya adalah nilai MIN pada algoritma *minimax*.

2. Kedalaman *node* yang diperiksa oleh algoritma *minimax* mempengaruhi tingkat akurasi pengambilan keputusan langkah. Semakin dalam *node* yang diperiksa, setiap kemungkinan langkah akan dikalkulasi dan dapat diambil nilai yang lebih akurat.
3. Aplikasi yang dihasilkan telah mampu untuk menunjukkan bagaimana optimasi algoritma *minimax* mempengaruhi kinerja sebuah aplikasi permainan *checker*. Dengan menggunakan algoritma *minimax* ini, permainan *checker* dapat dimainkan tanpa harus tersedianya dua orang pemain. Dengan hanya satu orang pemain saja, permainan tetap dapat berlangsung, dengan *computer AI* menggantikan pemain yang lainnya.

IV. KESIMPULAN

Dari hasil pembahasan dapat disimpulkan sebagai berikut :

1. Algoritma *minimax* yang diaplikasikan ke dalam permainan *checker* ini menyebabkan permainan dapat dilakukan dengan hanya satu pemain saja. Dengan *computer AI* yang menggunakan algoritma *minimax* untuk menentukan langkah koin, pemain yang lain dapat digantikan.
2. Hasil akurasi penentuan langkah pada permainan *checker* dengan optimasi algoritma *minimax* ini dipengaruhi oleh beberapa hal, seperti kedalaman *node* yang diperiksa, posisi koin pada papan permainan, jumlah langkah yang diperiksa serta waktu berfikir yang diberikan. Keseluruhan parameter ini akan mempengaruhi nilai MAX atau MIN dari *leaf node* akhir yang dihasilkan, yang selanjutnya akan mempengaruhi nilai akhir dari langkah tersebut.

DAFTAR PUSTAKA

- [1]. Akbar, M.A (2011) *Analisis Dan Implementasi Kecerdasan Buatan Pada Permainan Checker Menggunakan Algoritma Minimax Dengan Negascout*, Tugas Akhir, Universitas Sumatera Utara.
- [2]. Ayuni, Wulan, 2011, *Implementasi Konsep Kecerdasan Buatan Dalam Rancang Bangun Game Brick Breaker*, Tugas Akhir, Universitas Sumatera Utara.
- [3]. Beny, 2010, *Pengembangan Software Game Menggunakan RPG Maker VX*, Tugas Akhir, Universitas Sumatera Utara.
- [4]. Nuzulla, B. & Solichin, A., 2012. *Implementasi Algoritma Steepest Ascent Hill Climbing Dengan Optimasi Minimax Pada Permainan Tic Tac Toe Berbasis Android*, Prosiding Seminar Nasional Multi Disiplin Ilmu, 65-73.
- [5]. Prasetyo, R.M.K. 2012, *Penyelesaian Permainan Checkers Pada Mobile Device Berbasis Android Menggunakan Algoritma Iterative Deepening Search*, Tugas Akhir, Universitas Sumatera Utara.
- [6]. Ramadhan, A. 2007, *Visual Basic 6.0*, PT. Elex Media Komputindo, Jakarta.
- [7]. Rizky, S. 2011, *Konsep Dasar Rekayasa Perangkat Lunak*, Prestasi Pustaka Publisher, Jakarta.
- [8]. Setiadi, D.R.I.M. 2012, *Implementasi Algoritma Minimax Untuk Artificial Intelligence Pada Permainan Catur Sederhana*, *Techno.COM*, Vol. 11 No.2, 99-107.
- [9]. Simamarmata, J. (2010) *Rekayasa Perangkat Lunak*, Penerbit Andi, Yogyakarta.
- [10]. Sutojo, T., Mulyanto, E., Suhartono, V., 2011, *Kecerdasan Buatan*, Penerbit Andi, Yogyakarta.
- [11]. Toba, H. dan Kurniawan, B. 2008, *Aplikasi Permainan Capsah Banting dengan Penerapan Algoritma Minimax*, *Jurnal Informatika*, Vol. 4 No. 1, 12-27
- [12]. Zarlis, M & Handrizal, 2008, *Algoritma dan Pemrograman : Teori dan Praktik Dalam Pascal*, Edisi Kedua, USU Press, Medan