Penerapan Algoritma Alpha-Beta Pruning pada Permainan Nine Men's Morris

Kevin Winata - 13510073
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
kevinwinata@students.itb.ac.id

***Abstract*—Permainan papan merupakan salah satu masalah yang sering dipakai sebagai objek pembuatan Artificial Intelligence. Para programmer berlomba-lomba untuk membuat pemain-komputer yang tidak terkalahkan dengan algoritma secepat dan seefisien mungkin. Pada makalah ini akan dibahas mengenai algoritma untuk memecahkan salah satu permainan papan yang mungkin jarang terdengar, yaitu Nine Men's Morris. Algortima yang akan digunakan adalah alpha beta pruning yang merupakan kelanjutan dari algoritma minimax.**

***Index Terms*— alpha beta pruning, DFS, minimax, zero-sum**

I. Permainan Nine Men's Morris

A. Mengenal Nine Men's Morris

Nine Men's Morris adalah sebuah permainan yang sudah sangat tua, bahkan salah satu yang tertua sepanjang sejarah. Permainan ini banyak ditemukan di berbagai negara. Papan tertua yang pernah ditemukan berada di Mesir, diperkirakan berasal dari 1400 SM. Nama Nine Men's Morris sendiri tercipta ketika permainan ini mencapai Inggris, jauh setelah permainan ini tercipta, yaitu pada masa Middle Ages. Pada zaman inilah Nine Men's Morris meraih popularitas di antara kalangan masyarakat.

Permainan ini juga memiliki banyak versi. Nine Men's Morris memiliki 9 bidak untuk masing-masing pemain. Ada empat permainan sejenis yang memiliki aturan mirip tapi jumlah bidaknya berbeda, yaitu Three Men's Morris, Six Men's Morris, Eleven Men's Morris, dan Twelve Men's Morris. Selain itu, ada juga varian dengan aturan Nine Men's Morris yang dimodifikasi, contohnya Misere, Diagonals, dan No Fly.

Ada berbagai nama lain untuk Nine Men's Morris. Contohnya, di Jerman disebut Mühle, di Perancis disebut Jeu De Moulin atau Merelles, di India disebut Sujjua, di Cina disebut Sam K'i, di Mexico disebut Picaria, dan masih banyak lagi. Hal ini menunjukan popularitas Nine Men's Morris yang melintasi banyak benua.

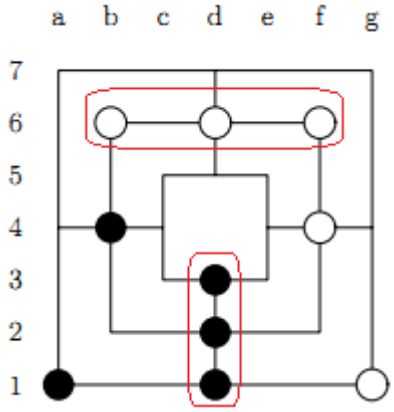
Papan yang digunakan berupa 3 buah persegi berbeda ukuran dengan yang lebih kecil berada di dalam persegi yang lebih besar. Titik tengah sisi sebuah persegi terhubung dengan titik tengah sisi dua persegi lain dengan sebuah garis. Bidak, yang berjumlah 9 dengan warna biasanya masing-masing hitam dan putih, diletakkan pada perpotongan antara dua garis.

Gambar 1Papan Permainan Nine Men's Morris

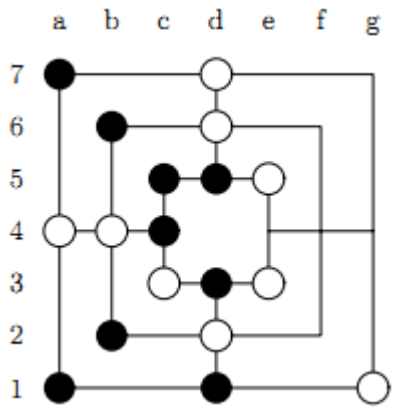
B. Peraturan Permainan

Pada permainan Nine Men's Morris, untuk mengalahkan lawan seorang pemain harus membentuk "mill" (atau disebut juga "morris") yaitu garis lurus dengan tiga bidak berwarna sama. Jika mill berhasil terbentuk, pemain berhak mengambil bidak pemain lain yang tidak berada di dalam garis mill.

Hanya ada dua kemungkinan bentuk mill yang dapat diraih. Kemungkinan pertama yaitu tiga bidak yang berada pada sisi yang sama dari persegi ketiga atau persegi kedua. Kemungkinan lain adalah tiga bidak yang berada pada garis pemisah dengan arah mata angin yang sama.



Gambar 2 Contoh terbentuknya mill



Gambar 3 Pemain hitam tidak bisa bergerak

Permainan ini terdiri dari tiga fase, dan pada ketiganya pemain dapat membentuk mill dan mengambil bidak pemain lawan.

1) Fase Pertama : Meletakkan Bidak

Pada fase ini, masing-masing pemain secara bergantian meletakkan bidaknya pada papan. Jika seorang pemain berhasil membentuk mill, pemain tersebut boleh mengambil bidak pemain lawan di papan. Fase ini berakhir ketika seluruh bidak telah diletakkan pada papan permainan atau diambil pemain lain.

2) Fase Kedua : Menggerakkan Bidak

Pada fase ini, pemain dapat menggeser bidaknya secara bergiliran sesuai dengan garis yang ada. Kedua pemain dapat saling berlomba untuk membentuk mill atau menghalangi lawan membentuk mill.

3) Fase Ketiga : Flying

Fase ketiga dimulai ketika seorang pemain hanya tinggal memiliki tiga bidak. Dalam fase ini pemain tersebut dapat memindahkan bidaknya ke posisi manapun di dalam papan.

Permainan berakhir berdasarkan tiga kondisi.

1) Kondisi pertama adalah salah satu pemain tidak bisa menggerakkan bidaknya kemanapun, maka pemain itu dinyatakan kalah.
2) Kondisi kedua adalah jika seorang pemain memiliki kurang dari tiga bidak, maka pemain itu dinyatakan kalah.
3) Kondisi ketiga adalah ketika posisi seluruh bidak terulang, maka permainan dinyatakan seri.

Sebenarnya aturan ini sendiri masih menjadi perdebatan antara para ahli. Ada beberapa yang berpendapat bahwa pada fase pertama jika seorang pemain berhasil membentuk dua mill sekaligus pada satu giliran maka pemain itu bisa mengambil dua bidak pemain lawan sekaligus. Ada juga pendapat lain dimana ketika seorang pemain berhasil membentuk mill, pemain tersebut tidak boleh mengambil bidak pemain lawan yang berada dalam millnya sendiri.

C. Strategi untuk Memenangkan Permainan

Ada beberapa strategi yang umum digunakan oleh para pemain dalam memenangkan permainan Nine Men's Morris :

1. Lokasi yang strategis

   Pada fase pertama, sangat penting untuk meletakkan para bidak di posisi yang strategis (mudah untuk digerakkan, tidak terhalang), dan tidak berfokus untuk membuat mill. Contoh posisi yang strategis adalah titik tengah di kotak terdalam, dimana bidak dapat bergerak pada tiga arah. Namun meletakkan bidak condong ke salah satu kotak juga tidak direkomendasi, distribusi bidak harus merata.

2. Menghalangi pemain lain

   Keunikan permainan ini adalah mill yang dibentuk dapat digunakan berkali-kali. Ketika seorang pemain sudah membentuk sebuah mill, dia bisa memindahkan bidaknya ke tempat lain pada giliran berikutnya, dan mengembalikannya ke tempat semula pada giliran selanjutnya, dan mill akan terbentuk kembali. Maka dia berhak mengambil dua bidak pada tiga giliran tersebut. Kondisi seperti ini harus dicegah oleh pemain lawannya, dengan cara meletakkan bidaknya sendiri di posisi yang ditinggalkan pada giliran kedua. Taktik ini disebut dengan blocking. Atau jika blocking tidak memungkinkan, pemain tersebut juga bisa mencoba membentuk mill sendiri, sehingga pemain lawan harus memilih antara melanjutkan membentuk mill atau melakukan blocking.

3. Double mill

   Double mill dibentuk dengan dua mill yang bersisian sedemikian rupa sehingga pemain dapat menggerakkan bidak dari sebuah mill yang sudah terbentuk ke posisi yang menyebabkan mill lain terbentuk. Double mill adalah kondisi yang sangat menguntungkan bagi seorang pemain, karena dia bisa

mengambil bidak lawan pada setiap giliran. Double mill harus menjadi prioritas blocking pemain lawan.

Algoritma yang nantinya dibuat akan memanfaatkan strategi yang sudah disebutkan di atas.

II. Algoritma Alpha Beta Pruning

A. Zero-Sum Game

Setiap permainan (dalam hal ini yang dibahas adalah board game) pasti memiliki solusi. Suatu permainan disebut telah dipecahkan jika sudah ada yang menemukan solusi yang paling optimal untuk tiap kombinasi langkah yang ada. Salah satu cara untuk menemukan solusi ini adalah dengan cara menggunakan algoritma. Algoritma yang paling sederhana adalah sebagai berikut : Dengan BFS atau DFS, buat simpul untuk tiap langkah yang mungkin, lalu berhenti jika sudah menemukan kondisi menang. Algoritma ini dapat dipakai pada permainan sederhana seperti Tic-Tac-Toe. Akan tetapi, untuk permainan-permainan yang lebih rumit seperti Nine Men's Morris, pohon yang harus dibangun untuk menemukan solusi akan sangat besar.

Oleh karena itu, dibutuhkan suatu algoritma yang dapat membantu menentukan pilihan langkah mana yang lebih baik diambil. Bagaimana cara menentukan pilihan ini dipelajari dalam sebuah studi yang disebut teori permainan (game theory). Dalam teori permainan sendiri sebuah permainan dapat diklasifikasi sebagai zero-sum game, yaitu permainan dimana keuntungan satu pihak akan mengakibatkan kerugian yang setara pada pihak yang lain, dan bila keuntungan atau kerugian tersebut direpresentasikan sebagai sebuah nilai maka jumlah keduanya adalah nol.

Permainan Nine Men's Morris merupakan permainan dengan kriteria zero-sum game. Ada berbagai metode dalam pengambilan keputusan pada zero-sum game, tetapi yang paling sering digunakan dan juga efisien adalah minimax.

B. Algoritma Minimax

Minimax merupakan sebuah aturan pengambilan keputusan yang meminimalkan kemungkinan mendapat kerugian maksimum. Algoritma minimax adalah algoritma yang memakai aturan ini dalam menentukan langkah. Pada algoritma ini, akan dibangun sebuah pohon yang simpul-simpulnya merepresentasikan sebuah langkah, dan memakai basis DFS dalam traversal pembangunan pohonnya.

Dalam pengambilan keputusan, algoritma minimax memerlukan sebuah fungsi penghitung skor yang mengembalikan suatu nilai yang menunjukkan tingkat keuntungan pada sebuah simpul. Tiap langkah diambil berdasarkan fungsi skor tersebut. Keuntungan direpresentasikan sebagai nilai positif, sedangkan kerugian direpresentasikan sebagai nilai negatif. Dalam hal ini, algoritma minimax menjadi mirip seperti algoritma Branch&Bound namun tidak memakai BFS, melainkan DFS.

Dalam algoritma ini disimulasikan dua pemain, yaitu pemain Max dan pemain Min. Pemain Max akan mengambil langkah dengan skor paling maksimum, yaitu dengan keuntungan terbesar. Sedangkan pemain Min akan mengambil langkah dengan skor paling minimum, yaitu kerugian terbesar bagi pemain Max. Sesuai teori zero-sum game, keuntungan masing-masing pemain akan setara dengan kerugian pemain yang lain, sehingga sebenarnya kedua pemain ini memiliki cara pengambilan keputusan yang sama, namun berbeda tanda (positif-negatif) pada perhitungan skornya.

Algoritma ini juga memiliki parameter kedalaman, yang disebut juga ply, untuk menentukan seberapa dalam algoritma ini akan mencari keputusan langkah terbaik yang akan diambil. Kedalaman ini harus ditentukan oleh programmer berdasarkan performansi dan waktu yang diinginkan (berapa lama komputer "berpikir"), karena jika tidak ditentukan algoritma ini akan mencari hingga kedalaman tidak terbatas.

Pseudo-code algoritma minimax

```
function MINIMAX(N)
if N cukup dalam then
    return skor(N)
else
   Misalkan N1, N2, ..,Nm adalah anak dari
simpul N
   if N adalah simpul pemain Min then
      return min(MINIMAX(N1), ..,
MINIMAX(Nm))
   else
      return max(MINIMAX(N1), ..,
MINIMAX(Nm))
```

C. Alpha Beta Pruning

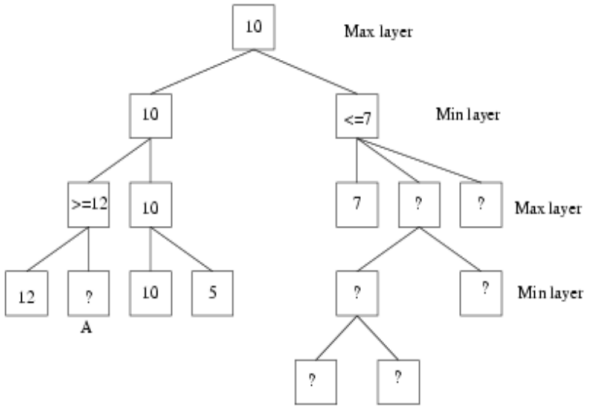
Dalam algoritma minimax, banyak terjadi kejadian dimana terjadi pengecekan sebuah simpul yang seharusnya tidak dicek karena tidak akan mempengaruhi hasil akhir. Untuk menghindari hal yang demikian, telah dibuat suatu algoritma minimax yang lebih optimal, yaitu algortima alpha beta pruning.

Hasil algoritma alpha beta pruning sendiri tidak berubah dari algoritma minimax, yang berubah hanya traversalnya yang lebih sedikit. Cara kerja algoritma ini adalah dengan mengecek sebuah simpul n, dan jika pemain memiliki pilihan yang lebih baik pada akar n atau simpul-simpul selanjutnya, maka sebenarnya n tidak pernah dicapai pada waktu permainan. Maka pada n akan dilakukan pruning, yaitu simpul n tidak akan dikembangkan lagi pada pohon.

Algoritma alpha beta pruning memanfaatkan dua nilai, yaitu alpha dan beta. Nilai alpha menunjukkan skor pilihan terbaik yang bisa diambil pemain Max, dan nilai beta menunjukan skor pilihan terbaik yang bisa diambil pemain Min. Pruning dilakukan ketika simpul yang sedang ditinjau pada pemain Max, yaitu n, memiliki skor

yang lebih rendah daripada alpha, maka n tidak perlu ditinjau lagi berikutnya. Hal yang sama dilakukan pada pemain Min memakai nilai beta.

Contoh alpha beta pruning adalah sebagai berikut :



Seperti terlihat, nilai skor yang tidak ditinjau (dengan '?') tidak mempengaruhi nilai skor pada akar pohon keputusan. Coba lihat node A. Node ini tidak perlu dievaluasi karena pemain Min sudah mengambil keputusan di simpul bernilai 10. Ini disebut dengan beta-cutoff. Kemudian coba lihat node yang belum ditinjau di subpohon bagian kanan. Node-node ini juga tidak perlu dievaluasi karena pemain Max sudah mengambil keputusan di simpul bernilai 10 untuk mencegah pemain Min mengambil keputusan di simpul bernilai 7. Ini disebut dengan alpha-cutoff. Dengan kedua cutoff tadi, terlihat bahwa simpul yang perlu ditinjau dengan alpha beta pruning jauh lebih sedikit daripada jika kita memakai algoritma minimax biasa.

Pseudo-code algoritma alpha beta pruning

```
function MINIMAX-AB(N, a, b)
if N cukup dalam then
   return skor(N)
else
   alpha <- a
   beta <- b
if N adalah simpul pemain Min then
   for each anak(N) Ni
      beta <- min(beta, MINIMAX-AB(Ni,
alpha, beta))
      if alpha >= beta then
         return beta
   end for
   return beta
else
   for each anak(N) Ni
      alpha <- max(alpha,MINIMAX-AB(Ni,
alpha,beta))
      if alpha >= beta then
         return alpha
   end for
   return alpha
```

III. Pemecahan Permainan Nine Men's Morris

Pada bagian ini akan dibahas tentang pemecahan permainan Nine Men's Morris menggunakan algoritma alpha beta pruning. Seperti yang sudah dijelaskan di atas, alpha beta pruning memerlukan sebuah fungsi penghitung skor untuk mengukur tingkat keuntungan suatu simpul.
Karena permainan ini terdiri dari tiga fase yang berbeda total dalam cara bermain, maka fungsi penghitung skor yang digunakan pada tiap fase juga haruslah terpisah. Masing-masing fase akan memiliki tingkat kedalaman pencarian yang berbeda pula.
Pada pembuatan fungsi penghitung skor, mula-mula kita definisikan hal-hal yang akan mempengaruhi pengambilan keputusan. Hal ini berupa relasi (atau keterhubungan) antar bidak di dalam papan, dan ditentukan melalui observasi dalam permainan yang sesungguhnya (antar manusia), sesuai dengan strategi umum Nine Men's Morris yang telah disebutkan di bagian sebelumnya.

Relasi yang diperhitungkan pada fase pertama adalah :
R1. Jumlah mill yang baru terbentuk
R2. Total mill yang ada di papan
R3. Jumlah bidak lawan yang terblokir
R4. Jumlah 2-pieces configuration, yaitu konfigurasi dua bidak berurutan yang memungkinkan terbentuk mill pada giliran berikutnya
R5. Jumlah 3-pieces configuration, yaitu konfigurasi dimana dapat terbentuk mill di dua tempat sehingga lawan tidak bisa melakukan blocking

Relasi yang diperhitungkan pada fase kedua adalah :
R1. Jumlah mill yang baru terbentuk
R2. Total mill yang ada di papan
R3. Jumlah bidak lawan yang terblokir
R4. Jumlah bidak di papan
R5. Jumlah mill yang terbuka, yaitu pada mill yang sudah terbentuk bidaknya dipindahkan ke tempat lain
R6. Jumlah Double Morris
R7. Konfigurasi unggul, yaitu konfigurasi-konfigurasi tertentu yang berdasarkan pengalaman dapat membawa kemungkinan lebih besar untuk menang.

Relasi yang diperhitungkan pada fase ketiga adalah :
R1. Jumlah 2-pieces configuration
R2. Jumlah 3-pieces configuration
R3. Jumlah mill yang baru terbentuk
R4. Konfigurasi unggul

Semua relasi ini akan dihitung kembali pada tiap simpul untuk menghitung skor.
Setelah relasi-relasi ini didefinisikan, muncul pertanyaan baru : seberapa besar masing-masing relasi ini berpengaruh pada fungsi penghitung skor pada algoritma? Atau dengan kata lain, jika masing-masing relasi

dipandang sebagai variabel penentu total skor, perlu ditentukan suatu koefisien untuk masing-masing relasi agar total skor yang dihitung menjadi benar-benar optimal. Karena Nine Men's Morris bukanlah suatu permainan yang trivial, koefisien tersebut perlu ditentukan melalui percobaan. Petcu dan Holban (2008) telah melakukan percobaan ini dengan sistem turnamen. Percobaan dilakukan dengan membuat beberapa program AI Nine Men's Morris menggunakan algoritma yang sama tapi dengan fungsi penghitung skor yang berbeda, yaitu ke-17 koefisiennya diambil secara random (namun pada domain yang masih relevan), kemudian masing-masing program akan 'ditandingkan' dengan skema yang teratur sehingga muncul suatu program yang unggul.

Berikut adalah tahap percobaan yang dilakukan :

1. Bangkitkan himpunan ke-17 koefisien secara random sebanyak 50 buah, dan masukkan ke program
2. Turnamenkan 50 program tersebut, masing-masing mendapat giliran pertama secara bergantian
3. Hitung hasil dan ambil 10 program yang paling unggul
4. Buat 40 program baru berdasarkan kombinasi 10 program unggul tersebut
5. Turnamenkan kembali 10 + 40 program yang baru
6. Ambil sebuah program yang paling unggul, maka koefisien program tersebutlah yang paling optimal

Berikut adalah hasil dari percobaan tersebut.
Koefisien untuk fase 1 :
18 26 1 6 12 7
Koefisien untuk fase 2 :
14 43 10 8 7 42 1086
Koefisien untuk fase 3 :
10 1 16 1190

Walaupun koefisien ini juga bukan merupakan angka yang pasti, karena berdasarkan hasil percobaan yang terbatas, namun koefisien ini sudah dites di berbagai macam turnamen dan berhasil memenangkan banyak pertandingan.

IV. Algoritma AI Nine Men's Morris

Algoritma ini merupakan pseudo-code yang dimodifikasi dari pseudo-code alpha beta pruning. Fungsi penghitung koefisien diasumsikan sudah diimplementasi terlebih dahulu.

Pseudo-code algoritma AI Nine Men's Morris

```
function NineMenMorris(N, a, b, depth)
if depth(N) = depth then
   return skor(N)
else
   alpha <- a
   beta <- b
if N adalah simpul pemain Min then
   for each bidak
      getArah(arah)
      movePieces(bidak,arah,Ni)
      if (phase(Ni) != phase(N)) then
         newdepth <- getReqDepth(phase(Ni))
      else
         newdepth <- depth + 1
      beta <- min(beta, NineMenMorris (Ni,
alpha, beta, newdepth))
      if alpha >= beta then
         return beta
   end for
   return beta
else
   for each bidak
      getArah(bidak,arah)
      movePieces(bidak,arah,Ni)
      if (phase(Ni) != phase(N)) then
         newdepth <- getReqDepth(phase(Ni))
      else
         newdepth <- depth + 1
      alpha <- max(alpha, NineMenMorris
(Ni, alpha, beta, newdepth))
      if alpha >= beta then
         return alpha
   end for
   return alpha
```

Pseudo-code fungsi penghitung skor simpul

```
function skor(N)
countAllRelationsNeeded(N, phase(N), *R)
switch (phase(N))
   case 1 : return (18*R[1])+ (26*R[2])+
(R[3])+ (6*R[4]) + (12*R[5]) + (7*R[6])
   case 2 : return (14*R[1])+ (43*R[2])+
(10*R[3])+ (8*R[4])+ (7*R[5])+ (42*R[6])+
(1086*R[7])
   case 3 : return (10*R[1])+ (R[2])+
(16*R[3])+ (1190*R[4])
end switch
```

- getArah() mengembalikan arah gerakan yang valid pada bidak tertentu yang belum pernah dicoba.
- movePieces() mencoba menggerakkan bidak ke suatu arah dan membuat sebuah simpul anak baru.
- getReqDepth() mengembalikan kedalaman berapa yang diperlukan berdasarkan fase simpul atau spesifikasi program maupun user.
- countAllRelationsNeeded() menghitung semua relasi yang dibutuhkan pada fase tertentu dan memasukkannya ke array integer R.

Kedalaman (depth) dapat ditentukan oleh user atau programmer. Makin besar kedalamannya, maka AI akan makin kuat dengan kemungkinan menang makin besar, tetapi menambah waktu 'berpikir' AI dalam menentukan keputusan.

Gasser (1993) sudah membuktikan bahwa Nine Men's Morris merupakan solved game dengan hasil draw. Solved game adalah permainan yang bisa ditentukan hasilnya, menang, kalah, atau draw berdasarkan suatu konfigurasi dalam permainan. Level penyelesaian permainan ini masuk pada level lemah, yaitu adanya algoritma yang dapat

memastikan kemenangan (atau draw) berdasarkan semua kemungkinan gerakan yang mungkin oleh pemain lawan.
Kesimpulan ini diambil berdasarkan analisis algoritma retrograde yang dikombinasikan dengan algoritma alpha beta pruning untuk fase pertama, sehingga seluruh state permainan Nine Men's Morris di-solve dengan menggunakan komputer.

IV. KESIMPULAN

Permainan papan umumnya dapat ditentukan hasilnya melalui konfigurasi tertentu. Akan tetapi, dalam mengembangkan Artificial Intelligence untuk sebuah permainan, dibutuhkan algoritma yang cukup efisien dengan kompleksitas waktu yang dapat dikendalikan.
Permainan Nine Men's Morris merupakan contoh yang sesuai. Walaupun permainan ini sudah dibuktikan sebagai solved game pada tahun 1993, pencarian yang dilakukan untuk membuktikannya tidaklah cocok bila dipakai sebagai program AI yang dapat bermain secara kompetitif karena sangat lama dan memerlukan memori yang sangat besar. Perlu dirancang suatu algoritma lain yang dapat bermain dengan waktu yang dibatasi. Solusinya adalah dengan menggunakan algoritma alpha beta pruning dengan kedalaman terbatas. Hanya saja, perlu dilakukan penelitian dan percobaan berulang-ulang untuk dapat mencapai hasil yang paling optimal.

REFERENCES

[1] Petcu, Simona Alexandra dan Stefan Holban, *Nine Men's Morris : Evaluation Function*, Computer Science and Engineering, Politehnica University of Timisoara, 2008.
[2] Gasser, Ralph, *Solving Nine Men's Morris*, MSRI Publications, 1996.
[3] Strong, Glenn, *The Minimax Algorithm*, 2011.
[4] http://gamescrafters.berkeley.edu/games.php?game=ninemensmorris last access : 12/11/2012
[5] http://www.cs.utah.edu/~hal/courses/2009S_AI/Walkthrough/AlphaBeta/ last access : 12/11/2012

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,

Kevin Winata / 13510073