Penerapan Algoritma Greedy dalam Permainan *Connect 4*

Muhammad Hasby (13509054)[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]mhmmd.hsby@gmail.com

***Abstrak*—*Connect4* adalah sebuah permainan papan yang menggunakan papan berlubang dan dimainkan dengan menggunakan sejumlah koin dalam dua warna yang berbeda. Permainan ini dilakukan dengan cara menyusun koin-koin yang dimiliki hingga terdapat 4 koin yang letaknya berurutan. Terdapat dua strategi dalam permainan ini, yaitu bertahan dan permainan jangka panjang. Penerapan algoritma *greedy* ini mengadopsi strategi bertahan dengan beberapa penyempurnaan. Penyempurnaan tersebut berupa pemberian prioritas pada setiap jenis langkah. Algoritma *greedy* berdasarkan prioritas ini mengambil langkah dengan nilai tertinggi dengan harapan menghasilkan hasil yang optimal dan memenangkan permainan. Pada makalah ini akan dibahas lebih lanjut mengenai penggunaan prioritas dalam algorita greedy. Pada akhir makalah terdapathasil pengujian dengan melawan tiga level komputer yan berbeda. Dua level terbawah dapat dimenangkan namun algoritma ini kalah melawan level tersulit**

***Index Terms*—algoritma, *greedy*, *connect4*, prioritas.**

I. PENDAHULUAN

Connect 4 atau biasa juga disebut '*four in a line'* adalah sebuah permainan papan (*board game*) yang menggunakan papan berlubang dan sejumlah koin dalam dua warna yang berbeda. Permainan ini memiliki beberapa versi yang dibedakan dengan ukuran papan yang digunakan. Papan yang paling umum digunakan berukuran 6x7, namun permainan ini dapat juga dimainkan dengan papan berukuran 8x7, 9x7, atau 10x7.

Gambar 1 Papan permainan *connect 4*

Permainan ini dirancang untuk dimainkan oleh dua orang. Masing-masing pemain akan diberi 21 koin untuk dimainkan secara bergiliran. *Connect4* dimainkan dengan cara menjatuhkan koin ke dalam salah satu kolom pada papan sehingga koin tersebut menempati posisi terbawah yang belum terisi pada papan. Kedua pemain berlomba untuk menempatkan empat buah koin secara berurutan dalam posisi tegak, mendatar, atau diagonal.

Permainan dimenangkan oleh pemain pertama yang berhasil memperoleh konfigurasi empat koin yang bersebelahan. Jika seluruh lubang pada papan telah terisi penuh dan belum ada pemain yang memenangkan permainan, maka kedua pemain dinyatakan seri.

Algoritma *greedy* dapat diterapkan pada permainan ini untuk mencari langkah yang paling optimal dalam setiap giliran. Dengan demikian, diharapkan bahwa kemungkinan untuk memenangkan permainan dapat diperbesar.

II. ALGORITMA GREEDY

Algoritma *greedy* merupakan algoritma yang sering digunakan dalam pemecahan masalah optimasi. Algoritma ini mencari solusi paling optimal dalam setiap langkah dengan harapan solusi tersebut adalah solusi paling optimal secara keseluruhan. Dalam pemilihan solusi setiap langkah tersebut, algoritma ini tidak memperhatikan konsekuensi untuk langkah-langkah berikutnya. Karena itu algoritma *greedy* dikenal dengan prinsipnya yaitu "*take what you can get now*".

Algoritma *greedy* terdiri dari beberapa elemen sebagaio berikut:

1. Himpunan Kandidat
   Himpunan yang berisi elemen-elemen yang bisa menjadi solusi dari permasalahan.
2. Himpunan Solusi
   Himpunan yang berisi kandidat-kandidat yang telah terpilih sebagai solusi persoalan.
3. Fungsi Seleksi
   Fungsi yang digunakan dalam setiap langkah untuk memilih kandidat yang paling memungkinkan menjadi solusi optimal.
4. Fungsi Kelayakan
   Fungsi yang digunakan untuk memastikan bahwa kandidat dan himpunan solusi yang sudah terbentuk memenuhi syarat yang ada.

5. Fungsi Obyektif
   Memilih solusi yang paling optimal dari himpunan solusi.

Pada umumnya penggunaan algoritma *greedy* memiliki skema yang sama, yaitu sebagai berikut:

1. Inisialisasi himpunan solusi dengan himpunan kosong.
2. Pilih sebuah kandidat yang paling memungkinkan mencapai solusi optimal dengan menggunakan fungsi seleksi dari himpunan kandidat.
3. Periksa apakah kandidat yang dipilih dan himpunan solusi yang telah terbentuk telah layak dengan menggunakan fungsi kelayakan. Jika ya, masukkan kandidat tersebut ke dalam himpunan solusi.
4. Periksa apakah himpunan solusi yang didapat telah memberikan solusi yang lengkap dengan menggunakan fungsi solusi.
5. Jika ya, maka proses telah selesai, namun jika tidak, ulangi dari langkah 2.

Penerapan algoritma *greedy* dalam permainan ini menggunakan elemen-elemen dan skema umum yang disebutkan di atas. Solusi yang dihasilkan dari skema tersebut diharapkan merupakan solusi paling optimal untuk memenangkan permainan.

III. Analisis Permasalahan

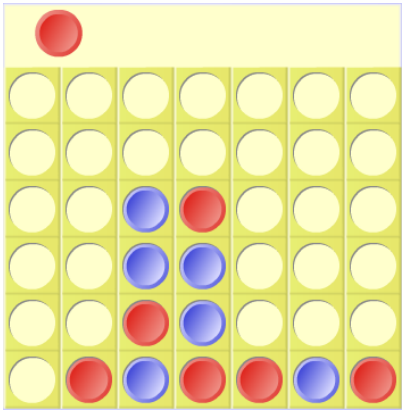
Secara umum terdapat dua jenis strategi dalam memainkan permainan *connect 4.* Strategi pertama adalah stategi bertahan. Strategi ini dilakukan dengan cara melihat langkah lawan dan berusaha untuk menghalangi lawan memenangkan permainan. Hal itu dilakukan sambil mencoba untuk menempatkan empat koin yang dimiliki secara berurutan. Sementara strategi kedua adalah mencari kemenangan dalam jumlah langkah yang banyak. Strategi ini efektif untuk permainan yang panjang namun terdapat banyak perhitungan yang harus dilakukan.

Dari kedua strategi yang disebutkan sebelumnya, strategi yang paling umum digunakan adalah strategi pertama. Hal itu dikarenakan seorang pemain akan lebih berkonsentrasi untuk betahan dan menghalangi kemenangan lawan.

Dalam strategi permainan *connect4,* terdapat beberapa istilah yang perlu diketahui. Istilah pertama adalah *group* atau kelompok. *Group* adalah satu set lubang pada papan yang terhubung secara horizontal, vertikal, atau diagonal. Pemain pertama yang berhasil mengisi sebuah grup dengan koin-koinnya akan memenangkan permainan. Istilah kedua adalah *threat* yaitu sebuah *group* yang sudah terisi dengan tiga buah koin berwarna sama, dengan lubang keempat dan lubang yang berada tepat di bawahnya masih kosong.

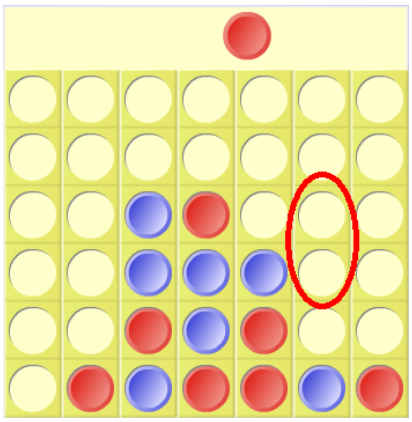
Sebelum melanjutkan pembahasan mengenai strategi dalam permainan, terdapat dua istilah yang perlu diketahui. Pertama adalah *group,* yaitu sebuah kumpulan koin yang terhubung atau ancaman. *Threat* adalah suatu kondisi dimana

Untuk menjalankan strategi yang disebutkan sebelumnya, ada beberapa trik yang dapat digunakan. Salah satu trik yang paling mudah adalah membuat dua buah *threat* sekaligus yang letaknya berdekatan atas dan bawah. Contoh kondisinya dapat dilihat pada gambar berikut:



Gambar 2 penggambaran kondisi

Pada gambar di atas dapat dilihat bahwa koin merah harus ditempatkan pada kotak F2 (F menunjukkan urutan kolom dimulai dari A di paling kiri, dan 2 menunjukkan baris tempat kotak itu berada) agar pemain dengan koin biru tidak menang. Namun setelah itu pemain biru akan menempatkan koin pada kotak F3. Jika keadaan tersebut terjadi maka pemain dengan koin biru akan mendapatkan dua buah *threat* pada posisi G3 dan G4 seperti gambar berikut:



Gambar 3 penggambaran kondisi *double threat*

Setelah mendapatkan posisi seperti itu pemain dengan koin biru akan mendapatkan kemenangan.

Selain memang sengaja melakukan trik tersebut, mungkin juga kondisi seperti itu didapat karena kesalahan lawan dalam menempatkan koin. Pada intinya strategi

yang dapat digunakan dalam permainan ini adalah menghalangi pemain lawan untuk memenangkan permainan sekaligus memikirkan cara untuk menempatkan koin sendiri dalam *group*.

IV. METODOLOGI PEMACAHAN MASALAH

Algoritma optimasi kemenangan yang dibahas dalam makalah ini pada dasarnya merupakan pengembangan dari strategi bertahan yang telah dibahas sebelumnya. Peningkatan terhadap strategi tersebut didasarkan pada beberapa poin penting dalam permainan *connect4*. Poin-poin tersebut diperoleh dari analisis yang dapat dijelaskan sebagai berikut:

1. Pemain harus berlomba untuk menempatkan empat buah koin secara berurutan agar dapat memenangkan permainan.
2. Pemain juga harus memerhatikan gerakan lawan, jangan sampai lawan berhasil menempatkan empat koin secara berurutan lebih dahulu.
3. Hal lain yang harus diperhatikan adalah *threat* yang dilakukan oleh pemain lawan. Hal ini dapat dilakukan dengan menghalangi lawan untuk menempatkan tiga koin secara berurutan sehingga kemungkinan *threat* menjadi lebih kecil.

Berdasarkan analisis yang telah dijelaskan sebelumnya, dapat disusun sejumlah prioritas langkah yang perlu diambil pemain guna memenangkan permainan. Prioritas tersebut adalah sebagai berikut:

1. Langkah dengan prioritas tertinggi yaitu menempatkan koin keempat yang dapat melengkapi sebuah *group.*
2. Prioritas kedua adalah langkah yang diambil jika lawan telah memiliki tiga koin dalam satu *group*. Dapat dilakukan dengan menempatkan koin pada lubang keempat dalam *group* tersebut sehingga menghalangi lawan memenangkan permainan.
3. Langkah dengan prioritas ketiga dilakukan jika terdapat sebuah *group* yang sudah berisi dua koin. Langkah ini dilakukan dengan cara menempatkan koin pada salah satu lubang hingga membuat tiga koin terhubung.
4. Langkah prioritas keempat yaitu langkah yang diambil untuk menghalangi pemain lawan menempatkan tiga koin secara berurutan.
5. Prioritas terakhir yaitu langkah untuk mendapatkan dua buah koin yang terhubung baik secara mendatar, tegak, atau diagonal.

Prioritas-prioritas tersebut kemudian dapat dimanfaatkan dalam mengimplementasikan sebuah algoritma *greedy* berdasarkan prioritas langkah. Dengan melakukan optimasi setiap langkah, diharapkan juga dapat mengoptimasi peluang kemenangan pemain.

Secara umum algoritma *greedy* berdasarkan prioritas ini dilakukan dengan cara memeriksa setiap kemungkinan langkah dan memberi nilai pada setiap langkah-langkah tersebut. Nilai yang diberikan adalah 0 sampai 5. Nilai 0 diberikan jika langkah tersebut tidak termasuk ke dalam lima prioritas yang telah dijelaskan. Selanjutnya nilai 1 sampai 5 diberikan sesuai dengan prioritas dari paling rendah hingga prioritas tertinggi. Langkah yang diambil adalah langkah yang memiliki nilai prioritas tertinggi. Untuk lebih menjelaskan penerapannya dalam permainan ini, berikut akan dijelaskan mengenai elemen-elemen penyusun algoritma *greedy* yang digunakan.

Elemen pertama adalah himpunan kandidat, dimisalkan dengan C. Himpunan kandidat dari persoalan ini adalah seluruh lubang yang belum terisi oleh koin.

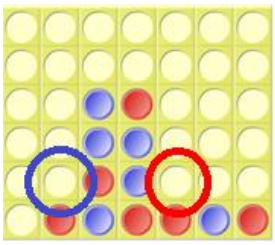
Elemen kedua adalah himpunan solusi yang dimisalkan dengan S. Isi dari himpunan ini adalah posisi lubang-lubang yang menjadi solusi optimal.

Selanjutnya adalah fungsi seleksi yang dinyatakan dengan predikat SELEKSI. Fungsi ini akan memilih posisi lubang yang memiliki nilai prioritas yang paling tinggi.

Elemen keempat adalah fungsi kelayakan yang dinyatakan dengan predikat LAYAK. Fungsi ini akan memeriksa apakah posisi lubang yang dihasilkan dari fungsi seleksi telah memenuhi syarat atau tidak. Dalam hal ini syarat yang berlaku adalah posisi lubang harus merupakan posisi terbawah dari lubang yang kosong dalam suatu kolom.

Elemen terakhir adalah fungsi obyektif yang dinyatakan sebagai SOLUSI. Fungsi ini akan memeriksa apakah langkah yang dilakukan memiliki nilai prioritas optimum dan berhasil menempatkan empat koin di dalam satu *group*.

Selain elemen-elemen di atas terdapat juga fungsi untuk menghitung nilai prioritas (0 sampai 5) seperti yang telah dijelaskan sebelumnya. Perlu diperhatikan bahwa nilai untuk suatu lubang atau langkah merupakan penjumlahan dari seluruh prioritas di tempat tersebut. Karena mungkin saja dalam satu lubang terdapat lebih dari satu langkah prioritas yang mungkin. Untuk lebih jelasnya dapat dilihat pada contoh berikut:



Gambar 4 Contoh perhitungan prioritas

Dari gambar di atas dapat dilihat terdapat dua buah lingkaran dengan warna yang berbeda. Lingkaran biru menyatakan bahwa biru sedang mendapat giliran main dan lingkaran merah menyatakan bahwa merah sedang

mendapat giliran main. Berikut adalah penjelasan penghitungan nilai prioritas:

1. Untuk lingkaran biru, jika koin biru ditempatkan disitu maka koin tersebut akan terhubung dengan koin yang berada di kanan-bawahnya dan sekaligus juga terhubung dengan koin di kanan-atasnya. Oleh karena itu posisi itu mendapat nilai 2 yang didapat dari 2 kali langkah dengan prioritas terkecil yaitu 1.

2. Untuk lingkaran merah, jika koin merah ditempatkan disitu maka koin tersebut akan menghalangi kemenangan pemain biru sehingga mendapat nilai 4. Namun selain itu koin itu juga akan terhubung dengan koin dibawahnya sehingga mendapat nilai 1. Jadi nilai untuk posisi itu adalah 5.

Selain itu, dari gambar 4 dapat dilihat bahwa langkah yang layak hanya terdiri dari 7 yaitu A1, B2, C5, D5, E2, F2, dan G2. Dimana setiap posisi tersebut adalah posisi lubang yang kosong terbawah dalam setiap kolom.

Setelah mengetahui elemen-elemen penyusun algoritma ini, maka dapat dirumuskan skema algoritma *greedy* yang digunakan. Skema ini pada umumnya hampir sama dengan skema umum yang telah dijelaskan sebelumnya. Skema dari algoritma yang digunakan untuk memainkan permainan ini adalah sebagai berikut:

1. Inisialisasi S dengan kosong.
2. Iterasi semua lubang kosong yang ada pada papan.
3. Hitung nilai untuk setiap lubang tersebut dengan menggunakan fungsi penghitung prioritas.
4. Pilih lubang yang memenuhi fungsi seleksi, yaitu lubang yang memiliki nilai prioritas paling tinggi dari seluruh lubang kosong yang ada,
5. Periksa apakah solusi yang dihasilkan tidak melanggar aturan yang ada pada fungsi layak, yaitu merupakan baris terbawah dalam kolom.
6. Jika solusi itu tidak melanggar maka dimasukkan ke dalam himpunan solusi.
7. Jika permainan belum mendapatkan pemenang maka kembali ke langkah 2.

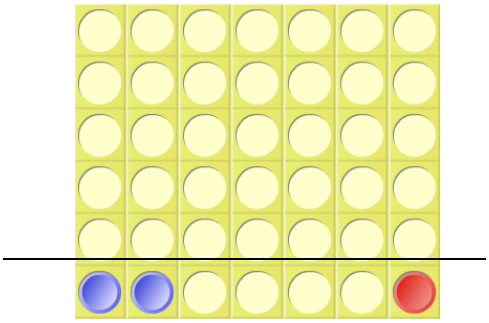
Jika hendak diterapkan ke dalam bahasa pemrograman, algoritma ini juga dapat dianyatakan dengan notasi pseudocode sebagai berikut:

```
Int Temp ← 0
S ← () {inisialisasi S dengan kosong}
Int n ← 0

while (not SOLUSI) do
   {iterasi untuk setiap lubang}
   for i ← 0 to jumlah lubang kosong do
      {menghitung prioritas untuk lubang ini}
      hitungPrioritas
      {SELEKSI}
      if (prioritas > temp) then
         {jika memenuhi syarat}
         if (LAYAK) then
            temp ← prioritas
            n ← nomor lubang
         endif
      endif
   endfor
   S ← n
   lakukan langkah dari elemen terakhir pada S
endwhile
return S
```

V. PENGUJIAN ALGORITMA

Dalam melakukan pengujian algoritma ini, penulis menggunakan salah satu permainan *connect4* di internet dengan alamat mathisfun.com. Pengujian dilakukan sebanyak 3 kali melawan 3 level computer. Untuk setiap pengujian hanya akan diperlihatkan urutan langkah dan nilai dari langkah yang layak saja. Untuk lebih jelasnya akan diberikan contoh seperti berikut:



Gambar 5 Contoh Pengujian

Untuk gambar di atas urutan langkahnya ditunjukkan pada tabel di bawah

Biru (komputer)	Merah(pemain)
A1	G1
B1	

Tabel 1 tabel urutan gerakan – contoh

Pada tabel di atas dapat dilihat bahwa biru yang dijalankan oleh komputer mendapat giliran pertama (yang berada di kolom kiri mendapat giliran pertama). Untuk setiap lubang dinamai dengan 2 karakter. Karakter pertama menunjukkan kolom dari A (paling kiri) sampai G(paling kanan). Karakter kedua menunjukkan baris dari 1 sampai 6 dari baris paling bawah. Pertama-tama koin biru ditempatkan di kiri-bawah yaitu A1. Giliran berikutnya koin merah ditempatkan kanan-bawah yaitu G1. Untuk selanjutnya tidak akan dijelaskan.

Sedangkan untuk nilai setiap lubang dalam setiap langkah dapat dilhat pada tabel berikut:

	A	B	C	D	E	F	G
Langkah 1	0	0	2	0	0	1	1

Tabel 2 tabel nilai prioritas – contoh

Baris pada tabel di atas menunjukkan nilai untuk setiap langkah dan kolom menunjukkan nilai untuk tiap kolom papan dari A hingga G. Kolom C pada langkah ke-1

mendapat nilai 2 karena langkah itu memiliki prioritas 2 yaitu mengahalangi lawan menempatkan 3 koin berurutan. Sedangkan kolom F dan G mendapat nilai 1 karena berhasil menempatkan 2 buah koin secara berurutan. Untuk selanjutnya tidak akan dijelaskan nilai untuk untuk setiap kolom dan langkah.

A. Pengujian melawan komputer (*beginner*)

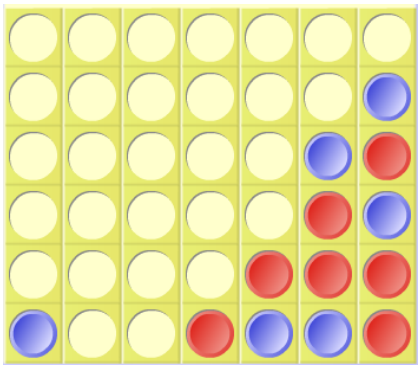
Pengujian ini dimenangkan oleh pemain dengan menggunakan algoritma *greedy*. Total langkah yang dilakukan adalah 7 langkah dengan penjelasan lebih lanjut dapat dilihat pada tabel-tabel berikut:

Merah (pemain)	Biru (komputer)
G1	F1
G2	G3
F2	A1
F3	F4
G4	E1
E2	G5
D1 (Menang)	

Tabel 3 urutan gerakan – uji1

	A	B	C	D	E	F	G
Langkah1	0	0	0	0	0	0	0
Langkah2	0	0	0	0	0	0	1
Langkah3	0	0	0	0	0	0	2
Langkah4	0	0	0	0	1	2	0
Langkah5	0	0	0	0	1	0	1
Langkah6	0	0	0	2	6	1	1
Langkah7	0	0	0	7	6	1	0

Tabel 4 nilai prioritas – uji1



Gambar 6 hasil akhir - uji1

Pengujian ini berjalan lancar dan setiap langkah merupakan langkah yang optimal. Algoritma ini baik digunakan untuk lawan yang tidak terlalu pandai bermain.

B. Pengujian melawan komputer (*easy*)

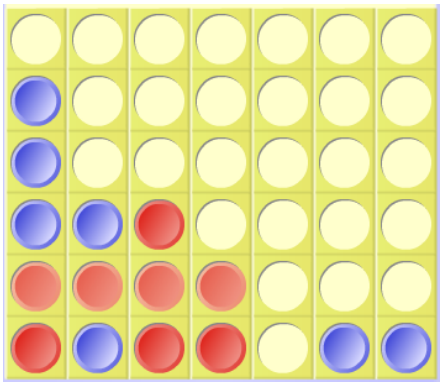
Pengujian ini dimenangkan oleh pemain dengan menggunakan algoritma *greedy*. Total langkah yang dilakukan adalah 8 langkah dengan penjelasan lebih lanjut dapat dilihat pada tabel-tabel berikut:

Merah (pemain)	Biru (komputer)
A1	B1
A2	A3
B2	B3
C1	A4
D1	G1
C2	F1
C3	A1
D2 (Menang)	

Tabel 5 urutan gerakan – uji2

	A	B	C	D	E	F	G
Langkah1	0	0	0	0	0	0	0
Langkah2	1	1	0	0	0	0	0
Langkah3	0	2	0	0	0	0	0
Langkah4	0	0	1	0	0	0	0
Langkah5	2	0	6	1	0	0	0
Langkah6	2	0	7	2	1	0	0
Langkah7	2	0	8	7	5	0	0
Langkah8	4	0	5	8	5	0	0

Tabel 6 nilai prioritas – uji2



Gambar 7 hasil akhir – uji2

Pengujian ini tidak berjalan lancar karena ada beberapa langkah yang bukan merupakan langkah yang optimal. Namun algorita in tetap dapat memenangkan permainan. Algoritma ini bisa digunakan untuk lawan yang biasa saja.

C. Pengujian melawan komputer (*medium*)

Pengujian ini dimenangkan oleh komputer dengan dengan level *medium*. Total langkah yang dilakukan adalah 10 langkah dengan penjelasan lebih lanjut dapat dilihat pada tabel-tabel berikut:

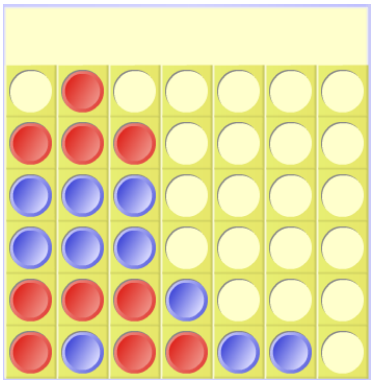
Merah (pemain)	Biru (komputer)
A1	B1
A2	A3
B2	B3
C1	A4
C2	C3
A5	B4
B5	C4
C5	E1

B6 F1
D1 D2 (Menang)

Tabel 7 urutan gerakan – uji3

	A	B	C	D	E	F	G
Langkah1	0	0	0	0	0	0	0
Langkah2	1	1	0	0	0	0	0
Langkah3	0	2	0	0	0	0	0
Langkah4	0	0	1	0	0	0	0
Langkah5	2	0	6	1	0	0	0
Langkah6	2	0	0	2	0	0	0
Langkah7	1	3	3	2	0	0	0
Langkah8	2	2	7	2	0	0	0
Langkah9	2	3	2	2	0	0	0
Langkah10	3	0	3	4	0	0	2

Tabel 8 nilai prioritas – uji3



Gambar 8 hasil akhir – uji3

Pengujian ini tidakberjalan lancar dan ada beberapa langkah yang merupakan langkah yang kurang optimal. Algoritma ini kurang baik digunakan untuk lawan yang pandai bermain.

VI. KESIMPULAN

1. Algoritma *greedy* digunakan untuk mencari solusi optimal lokal dengan harapan solusi tersebut dapat juga menjadi solusi optimal secara global.
2. Jika membandingkan hasil permainan dengan beberapa level, algoritma *greedy* berdasarkan prioritas ini cukup baik. Algoritma ini bias mengalahkan pemain dengan strategi bertahan. Namun jika lawannya lebih pintar dengan menggunakan trik, algoritma ini tidak terlalu baik.
3. Penerapan algoritma *greedy* yang telah dibahas masih membutuhkan penyempurnaan sebab penilian setiap dengan prioritas dengan interval 1 nilai masih kurang tepat. Masih ada kondisi-kondisi yang tidak sesuai dengan prioritas tersebut.

REFERENSI

[1] Munir, Rinaldi. "Diktat Kuliah IF3051 Strategi Algoritma", Program Studi Teknik Informatika ITB, 2009.

[2] http://zingmagic.com/home/2011/04/four-in-a-line/, 8 Desember 2011

[3] http://www.yourturnmyturn.com/rules/connectfour.php, 8 Desember 2011

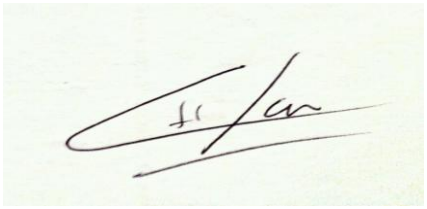
[4] http://www.mathsisfun.com/games/four-in-a-line-drop.html, 8 Desember 2011

[5] http://boardgaminginfo.com/connect-four-rules.php, 8 Desember 2011

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, 8 Desember 2011



Muhammad Hasby / 13509054