IMPLEMENTASI ALGORITMA *ALPHA – BETA PRUNING* PADA GAME *QUATROPOLLY* DENGAN MENGGUNAKAN JAVA

SKRIPSI

Ditujukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
Jurusan Teknik Informatika

Disusun Oleh :
Gede Putra Kusuma
015314056

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS SANATA DHARMA
YOGYAKARTA
2007

ALPHA - BETA PRUNING ALGORITHM IMPLEMENTATION IN *QUATROPOLLY* GAME USING JAVA

SKRIPSI

Purposed to Fulfill One of The Requirements
To Obtain Bachelor Degree
In Information Technology

By:
Gede Putra Kusuma
015314056

INFORMATION TECHNOLOGY
FACULTY OF TECHNIQUE
SANATA DHARMA UNIVERSITY
YOGYAKARTA
2007

IMPLEMENTASI ALGORITMA *ALPHA – BETA PRUNING*
PADA GAME *QUATROPOLLY*
DENGAN MENGGUNAKAN JAVA

SKRIPSI

Oleh :
Gede Putra Kusuma
NIM : 015314056

Telah disetujui oleh :

Pembimbing,

(Drs. Jong Jek Siang, M.Sc.) Tanggal Agustus 2007

SKRIPSI

IMPLEMENTASI ALGORITMA *ALPHA – BETA PRUNING*
PADA GAME *QUATROPOLLY*
DENGAN MENGGUNAKAN JAVA

Dipersiapkan dan ditulis oleh :
Gede Putra Kusuma
NIM : 015314056

Telah dipertahankan di depan Panitia Penguji
pada tanggal 8 Agustus 2007 dan
dinyatakan memenuhi syarat

Susunan Panitia Penguji

	Nama Lengkap	Tanda Tangan
Ketua	: Drs. Jong Jek Siang, M.Sc.	
Sekretaris	: St. Wisnu Wijaya, S.T.,M.T.	
Anggota	: St. Wisnu Wijaya, S.T.,M.T.	
	Ridowati Gunawan, S.Kom., M.T.	
	H. Agung Hernawan, S.T.	

Yogyakarta, Agustus 2007
Dekan Fakultas Teknik
Universitas Sanata Dharma

(Ir.Greg. Heliarko S.J.,S.S.,B.S.T.,M.A.,M.Sc.)

PERNYATAAN KEASLIAN KARYA

Saya menyatakan dengan sesungguhnya bahwa skripsi yang saya tulis ini tidak memuat atau karya orang lain, kecuali yang telah disebutkan dalam kutipan dan daftar pustaka sebagaimana layaknya karya ilmiah.

Yogyakarta, Agustus 2007

Penulis

Gede Putra Kusuma

ABSTRAK

Quatropolly merupakan *game* papan yang dimainkan oleh dua pemain, dengan cara menyusun bola-bola pada kotak yang disediakan secara bergantian. Pemain yang dapat menyusun empat bola atau lebih secara vertikal, horisontal maupun diagonal terlebih dahulu, menjadi pemenangnya.

Dalam skripsi ini dibuat program bantu game *Quatropolly* yang menggunakan algoritma *alpha-beta pruning*. Penggunaan algoritma *alpha-beta pruning* untuk langkah komputer, akan memudahkan komputer untuk menentukan posisi terbaik dan mempercepat perhitungan.

Program terdiri dari tiga jenis ukuran papan, yaitu 6x5, 6x6 dan 6x7. Permainan dirancang untuk kompoter melangkah terlebih dahulu. Pemain dapat memilih tiga *level* yang disediakan, yaitu *easy*, *normal* atau *hard*. Pada *level easy*, pemilihan posisi dilakukan secara *random*. Pada *level normal*, pemilihan posisi akan dilakukan dengan mencari nilai terbaik untuk satu langkah kedepan. Pada *level hard*, pemilihan posisi dilakukan dengan mencari nilai terbaik untuk dua langkah kedepan.

ABSTRACT

Quatropolly is a board game which played by two players, by arrange the balls in the prepared boxes which there are two players taking turns to play moves. The player which can arrange four balls or more vertically, horisontally and also diagonally in the first, is the winner.

In this minithesis is made a assist program *Quatropolly* games which using *alpha-beta pruning* algorithm. Usage *alpha-beta pruning* algorithm for the move of komputer, will facilitate computer to determine best position and quicken calculation.

Program consist of three kind measure board, that is 6x5, 6x6 and 6x7. Game designed for the computer of step beforehand. Player can chosen three provided level, that is easy, normal or hard. At easy level, election of position is conducted by random. At normal level, election of position will be conducted with searching best value to one step to the fore. At hard level, election of position is conducted with searching best value to two step to the fore.

HALAMAN PERSEMBAHAN

Kupersembahkan karya sederhana ini untuk :

...

Ida Sang Hyang Widhi Wasa yang selalu memberikan rahmatnya, petunjuk dan senantiasa menemaniku terutama dikala merasa putus asa.

...

Bapak, Ibu, dan kakak-kakakku , dan keluarga besarku terima kasih atas segala dorongan, semangat dan kasih saying yang telah diberikan.

...

dan seluruh teman-temanku yang telah memberikan warna dalam hari hariku.

HALAMAN MOTTO

Lakukanlah apa yang engkau anggap benar

dan tidak membuat orang menderita.

Jangan melakukan sesuatu

yang orang lain tidak ingin melakukannya kepadamu.

Aku memang tidak selalu mendapat keberungan yang inginkan, namun

kuharap aku kan selalu mendapat keberuntungan yang kuperlukan.

KATA PENGANTAR

Puji dan syukur kepada Ida Sang Hyang Widhi Wasa, Tuhan Yang Maha Esa atas segala berkat dan rahmat-Nya sehingga penyusunan skripsi yang berjudul **"Implementasi Algoritma *Alpha-Beta Pruning* pada Game Quatropolly dengan Menggunakan Java"** ini telah dapat diselesaikan dengan baik. Skripsi ini disusun dalam rangka memenuhi salah satu syarat untuk memperoleh gelar Sarjana Teknik Program Studi Teknik Informatika, di Fakultas Teknik Universitas Sanata Dharma..

Selama penyusunan skripsi ini, penulis menerima banyak bantuan dari berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Bapak Drs. Jong Jek Siang, M.Sc. selaku dosen pembimbing yang dengan sabar telah memberikan pengarahan dan bimbingan selama proses penyusunan skripsi ini.
2. Bapak St. Wisnu Wijaya, S.T.,M.T., Ibu Ridowati Gunawan, S.Kom., M.T. dan Bapak H. Agung Hernawan, S.T selaku dosen penguji yang telah banyak memberikan masukkan kepada penulis.
3. Bapak Ir.Greg. Heliarko S.J.,S.S.,B.S.T.,M.A.,M.Sc. selaku Dekan Fakultas Teknik Universitas Sanata Dharma, Yogyakarta.
4. Ibu Agnes Maria Polina, S.Kom., M.Sc. selaku Kaprodi Fakultas Teknik.
5. Bapak Alb. Agung Hadhiatma, S.T., M.T. selaku dosen pembimbing akademik, terima kasih atas bimbingannya selama ini.

6. Seluruh dosen Fakultas Teknik Universitas Sanata Dharma dan Staff Sekretariat, terima kasih atas semua bimbingan dan bantuannya.
7. Seluruh staf Jaringan Komputer, Basis data, Komputer dasar, terima kasih semuanya.
8. Bapak, Ibu, Kakak-kakakku tercinta di Bali yang selalu mendoakan dan memberi dukungan selama ini.
9. Elisabeth Ni Luh Yunita Erlinda, S.Farm., Apt. yang selalu mendampingiku, memberi dukungan serta kasih sayang dan perhatian selama ini.
10. Anak-anak White house thanks for everything.
11. Teman-teman angkatan 2001.
12. Teman teman dweber, klik-kanan terima kasih atas masukan masukannya.

Semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu dalam penyusunan skripsi ini.

Semoga Tuhan Yang Maha Kasih membalas semua perhatian, kasih sayang dan kebaikan yang telah diberikan.

No body's perfect, penulis menyadari bahwa tulisan ini masih jauh dari sempurna, oleh karena itu, penulis menerima segala bentuk kritik dan saran demi kesempurnaan tulisan ini. Penulis berharap semoga tulisan ini bermanfaat bagi pembaca.

Yogyakarta, Agustus 2007
Penulis

Gede Putra Kusuma

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Games adalah fasilitas yang sangat menarik dalam komputer. Ide *games* pertama kali dikemukakan oleh Claude Shanon (1950) yang menuliskan paper tentang mekanisme pembuatan program permainan catur. Beberapa tahun kemudian, Alan Turing mendeskripsikan program permainan catur namun ia sendiri belum pernah membuat rancangan program. Baru kemudian pada awal tahun 1960-an Arthur Samuel mencoba untuk membuat program catur tersebut.

Agar komputer dapat bertindak seperti dan sebaik manusia, komputer diberikan beberapa metode dan dibekali komponen komponen yang nantinya dapat membuat komputer menjadi mesin yang pandai. Hal ini dapat pula dilakukan dengan menerapkan teknik kecerdasan buatan pada program komputer. Salah satu algoritma yang digunakan untuk membuat komputer lebih pintar adalah *alpha-beta pruning*.

Quatropolly merupakan salah satu *game* papan yang dimainkan dua pemain yang kemenangannya ditentukan dari kecepatan pemain untuk membariskan bola. Siapa yang lebih dulu membariskan bola baik secara vertikal, horisontal maupun diagonal terlebih dahulu, maka pemain tersebut yang memeroleh kemenangan.

1.2. Rumusan Masalah

Masalah yang akan diselesaikan adalah bagaimana mengimplementasikan algoritma *alpha-beta pruning* pada game *quatropolly*.

1.3. Tujuan Penulisan

Adapun tujuan dari penulisan skripsi ini adalah mengimplementasikan algoritma *alpha-beta pruning* pada *game quatropolly*.

1.4. Batasan Masalah

a. Papan permainan menggunakan ukuran 6x5, 6x6 dan 6x7.

b. Level permainan terdiri dari *easy, normal* dan *hard*.

1.5. Sistematika Penulisan

BAB I. PENDAHULUAN

Berisi latar belakang, rumusan masalah, tujuan, batasan masalah, dan sistematika penulisan.

BAB II. LANDASAN TEORI

Berisi landasan teori yang dipakai sebagai dasar dalam pembuatan analisis, perancangan, dan implemetasi program. Dalam bab ini berisi sedikit penjelasan mengenai algoritma minmax, algortiman alpha-beta pruning serta dasar dari game quatropolly.

BAB III PERANCANGAN SISTEM

Pada bab ini berisikan perancangan dari game yang akan dibuat.

BAB IV IMPLEMENTASI

Berisi *coding* program beserta penjelasan singkat dari program yang dibuat.

BAB V KESIMPULAN DAN SARAN

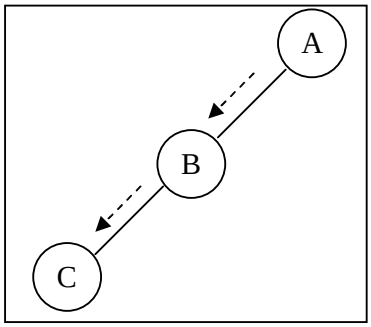
Berisi jawaban secara singkat tentang rumusan masalah yang dikemukakan pada BAB I. Pendahuluan. Juga berisi saran yang nantinya dapat digunakan untuk pengembangan sistem lebih lanjut.

BAB II

LANDASAN TEORI

2.1. MINIMAX

Salah satu teknik *game* yang terkenal adalah *Minimax*. *Minimax* menggunakan *depth-first search* dengan kedalaman terbatas. Pada *depth-first search* proses pencarian akan dilakukan pada semua anaknya sebelum dilakukan pencarian ke simpul-simpul yang selevel. Pencarian dimulai dari simpul akar ke level yang lebih tinggi. Proses ini diulangi terus hingga ditemukannya solusi. (Gambar 2.1).



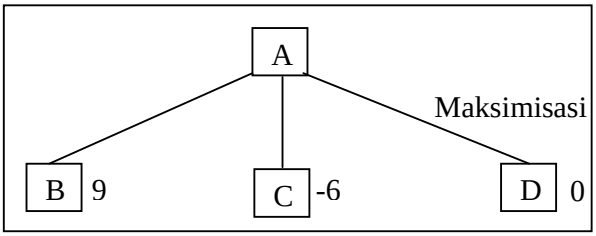
Gambar 2.1 *Depth- First Search*

Fungsi evaluasi yang digunakan adalah fungsi evaluasi statis (mengestimasi seberapa baik suatu simpul dari sudut pandang komputer, dipakai sudut pandang komputer karena komputerlah nantinya yang akan menggunakannya), dengan mengasumsikan bahwa lawan akan membuat langkah terbaik yang mungkin.

Pada *minimax* ada 2 prosedur yang dijalankan, yaitu maksimasi dan minimisasi. Dalam hal ini diperlukan suatu fungsi evaluasi statis yang menyatakan nilai yang mungkin didapat oleh pemain (misalkan nilai tersebut antara -10

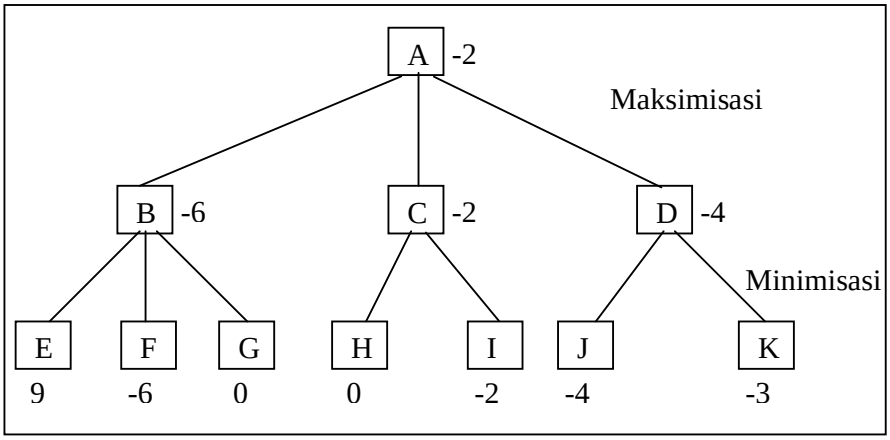
sampai +10. Nilai -10 untuk mengindikasikan kemenangan lawan sedangkan untuk nilai +10 mengindikasikan kemenangan pemain).

Untuk pencarian satu lapis akan dilakukan proses maksimisasi. Hal ini dilakukan agar nantinya akan mendapatkan nilai tertinggi. Untuk proses pencarian satu lapis seperti gambar 2.2. maka akan dilakukan proses maksimisasi yakni mencari nilai tertinggi, sehingga nantinya simpul yang dipilih tentunya B karena simpul ini memiliki nilai tertinggi.



Gambar 2.2. Pencarian satu lapis

Untuk pencarian dua lapis seperti pada gambar 2.3, pada level pertama akan dilakukan proses maksimisasi. Sedangkan pada level kedua akan dilakukan proses minimisasi. Jika pada proses maksimisasi dipilih simpul B maka saat lawan melakukan proses minimisasi tentu akan memilih simpul F yang memiliki nilai -6 sehingga akan membuat mendekati kemenangan. Hal ini tentu tidak diinginkan. Maka pada proses maksimisasi akan dipilih simpul C yang nantinya akan menyebabkan lawan hanya bisa memilih simpul H atau I saja. Proses ini dinamakan minimisasi karena pada tahap ini pemain berusaha untuk meminimalkan peluang lawan untuk memperoleh kemenangan.



Gambar 2.3. Pencarian dua lapis

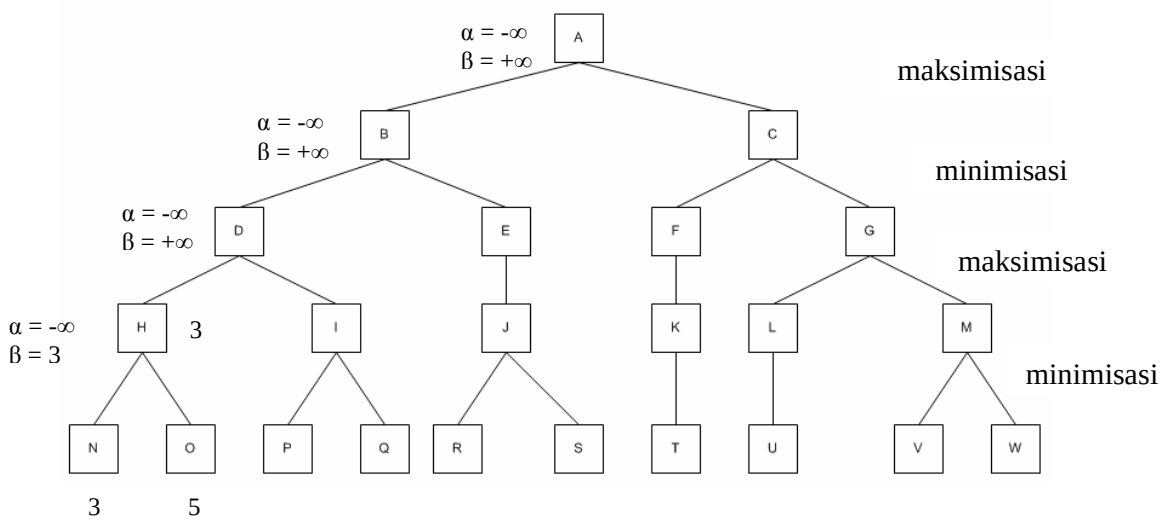
2.2. αβ-Prunning

Alpha-Beta pruning merupakan kelanjutan dari *minimax*. *Alpha beta pruning* merupakan salah satu metode untuk memodifikasi algoritma *minimax*. Variabel *alpha* (α) digunakan sebagai batas bawah simpul yang akan melakukan maksimisasi. Sedangkan variabal *beta* (β) digunakan sebagai batas atas bagi simpul yang akan melakukan proses minimisasi. Pada simpul-simpul yang melakukan proses minimisasi, evaluasi akan dihentikan jika sudah didapat simpul anak yang memiliki nilai lebih kecil dibandingkan dengan batas bawah (α). Sedangkan pada simpul-simpul yang melakukan proses maksimisasi, evaluasi akan dihentikan jika sudah didapat simpul anak yang memiliki nilai lebih besar dibandingkan batas atas (β).

Pada akar pohon pencarian, mula mula nilai α diset sama dengan -∞ sedangkan untuk nilai *beta* akan diset sama dengan +∞. Simpul-simpul yang melakukan proses maksimisasi akan memberikan nilai *alpha* dari nilai anak anaknya sedangkan simpul-simpul yang melakukan proses minimisasi akan

memberikan nilai *beta* dari nilai anak-anaknya. Jika *alpha* > *beta*, maka evaluasi dihentikan.

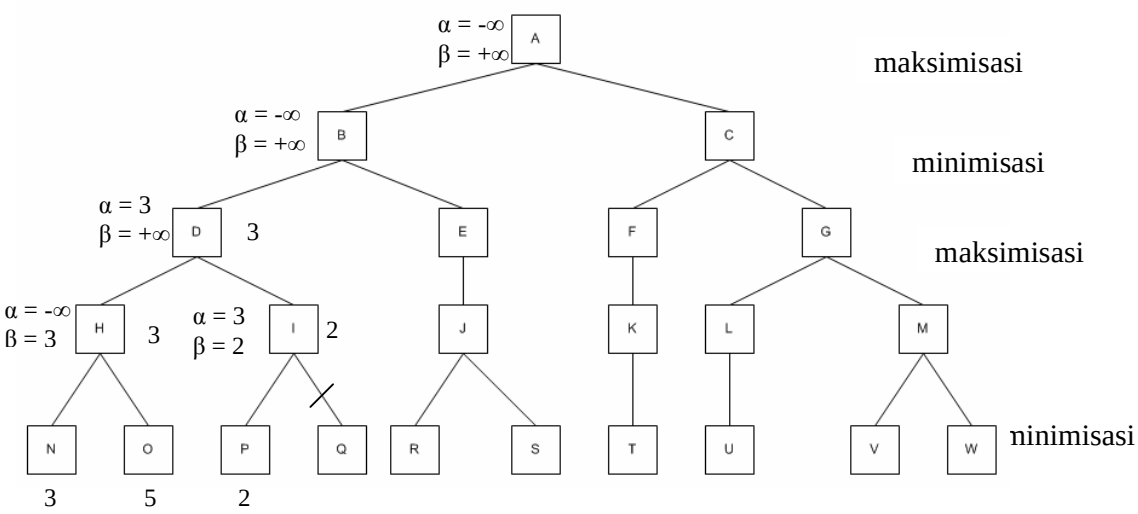
Tiap tiap simpul yang melalui nilai *alpha* dan *beta* akan dievaluasi anak-anaknya. Simpul simpul anak akan memperbaiki nilai *alpha* dan *beta*. Simpul yang melakukan minimisasi akan memberikan nilai *beta* sesuai dengan nilai simpulnya. Sedangkan simpul yang melakukan maksimisasi akan memberikan nilai *alpha* sesuai dengan nilai simpulnya.



Gambar 2.4a. Penerapan αβ pruning

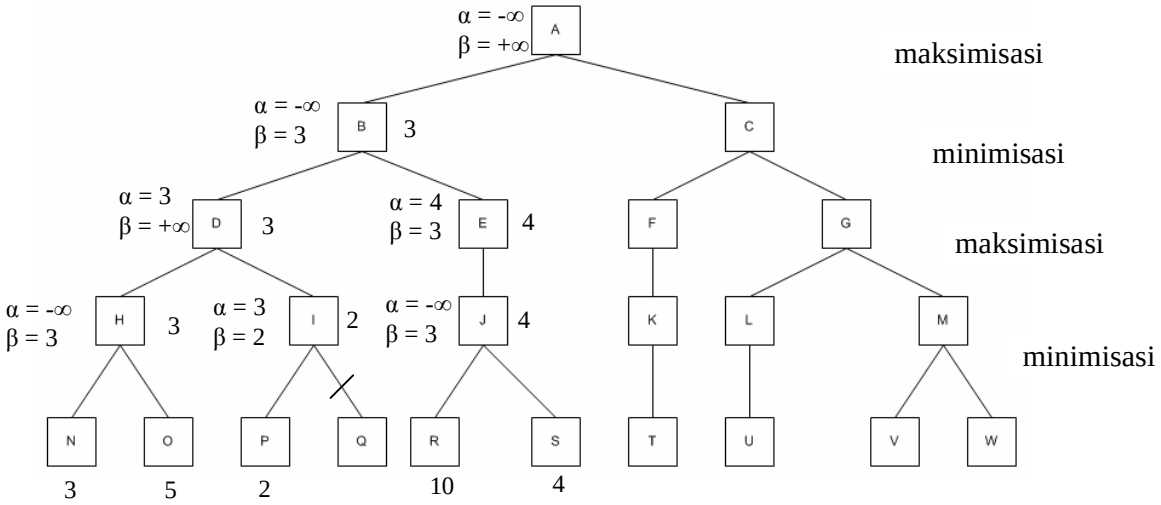
Seperti gambar 2.4a pertama kali akan dilakukan penelusuran ke simpul B. penelusuran akan dilakukan untuk anak-anak B terlebih dahulu. Jika N bernilai tiga maka nilai tiga ini akan memperbaiki nilai *beta*. Sehingga nilai *beta* dari *+inf* menjadi tiga. Kemudian nilai ini dibandingkan dengan nilai dari O. Karena nilai O lebih besar dari N maka nilai O tidak memperbaiki nilai *beta* sehingga nilai *beta* akan tetap menjadi tiga dan nilai H menjadi tiga.

Dari nilai *beta* yang diperoleh untuk simpul N dan O akan memperbaiki nilai H sehingga H menjadi bernilai tiga dan *alpha* menjadi tiga.



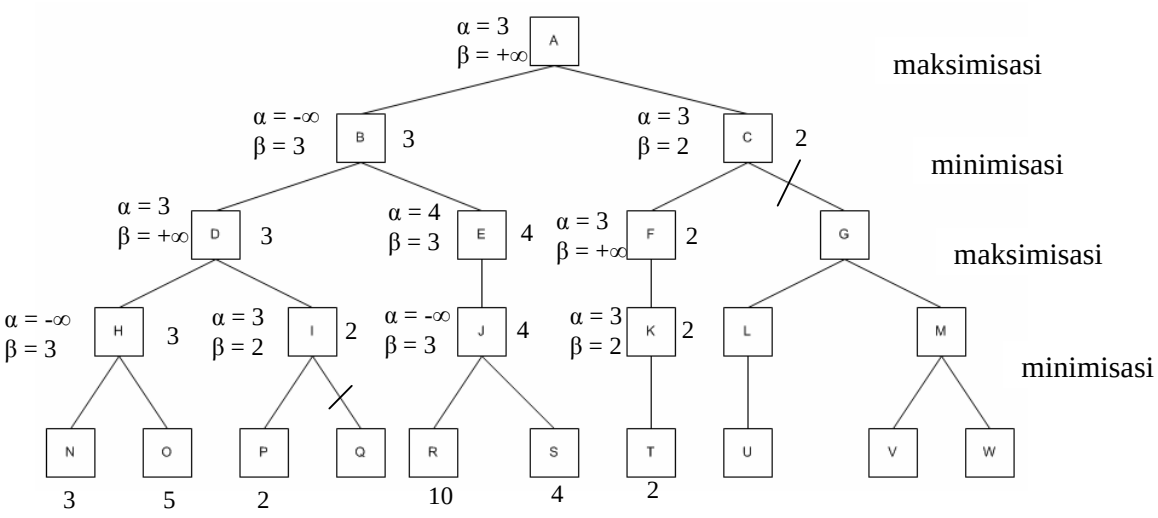
Gambar 2.4b. Penerapan αβ pruning

Jika P bernilai dua maka nilai ini akan dibandingkan dengan nilai *beta*. Maka nilai P ini akan memperbaiki nilai beta sehingga nilai *beta* sama dengan 2. Karena nilai *alpha* lebih besar dari *beta* maka proses dihentikan dan cabang yang lain untuk simpul yang sama dipotong. Ini dikarenakan pada saat D melakukan proses maksimisasi maka akan dipilih simpul H karena simpul H bernilai lebih besar dibandingkan dengan simpul I. Jika dimisalkan simpul Q bernilai 4, maka nilai dari simpul I tetap dua. Jika simpul Q bernilai satu maka nilai I malah menurun menjadi satu. Berapapun nilai dari Q tidak akan bepengaruh. Sehingga nilai D menjadi tiga.



Gambar 2.4c. Penerapan αβ pruning

Saat simpul R bernilai 10 maka nilai ini tidak akan memperbaiki nilai *beta*. Begitu pula dengan nilai S yang bernilai empat tidak akan memperbaiki nilai *beta* sehingga *beta* tetap bernilai tiga. Nilai R dan S akan di bandingkan kemudian akan dicari yang terkecil yang digunakan untuk memperbaiki nilai J sehingga nilai J menjadi empat. Nilai J akan memperbaiki nilai *alpha* dari *–inf* menjadi empat. Sehingga nilai E menjadi empat dan nilai B sama dengan tiga.

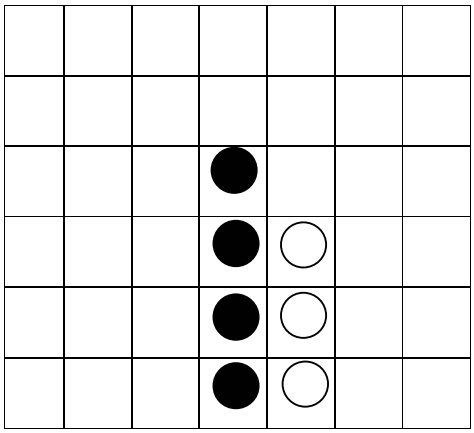


Gambar 2.4d. Penerapan αβ pruning

Nilai T akan memperbaiki nilai nilai *beta* yang tadinya bernilai tiga menjadi dua. Nilai ini akan memperbaiki nilai K sehingga K akan bernilai dua dan nilai F menjadi dua juga. Kemudian nilai ini diteruskan ke simpul C sehingga C bernilai dua. Simpul G akan dipotong karena nilai *alpha* > *beta*.

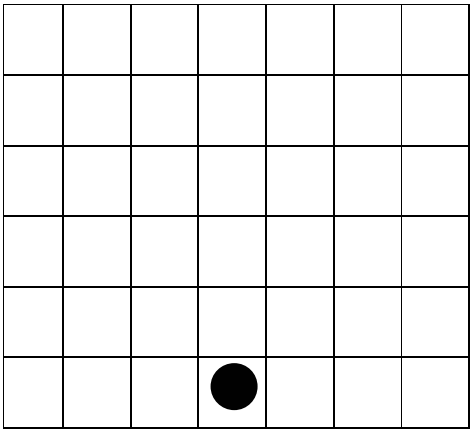
2.3. Quatropolly

Quatropolly merupakan game papan yang dimainkan oleh dua pemain. Pada game ini pemain akan membariskan bola secara vertikal, horisontal maupun diagonal sebanyak 4 bola. Pemain yang terlebih dahulu membariskan sebanyak 4 akan keluar sebagai pemenang (seperti pada gambar 2.5.).

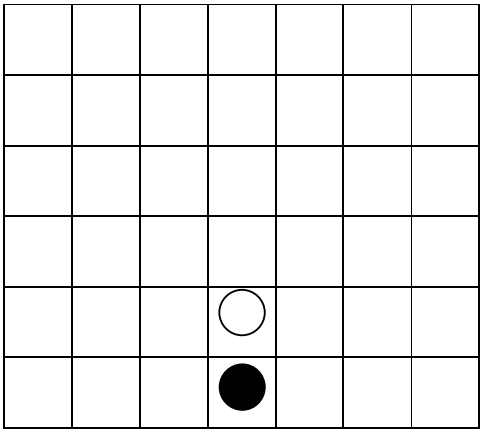


Gambar 2.5. Pemain dengan Bola hitam menang

Permainan juga akan berakhir seri jika tidak ada satupun pemain yang bisa membariskannya sampai papan penuh berisi bola. Pengisian bola pada papan dilakukan bergantian. Pada permainan ini pemain akan mengisikan dari baris terbawah seperti pada gambar 2.6a. Pemain hanya menentukan kolom mana yang akan diisi. Kemudian bola akan jatuh ke baris terbawah (pemain tidak bisa meletakkan bola secara menggantung). Jika bagian terbawah sudah terisikan bola, maka pengisiannya akan dilakukan diatas bola yang telah ada pada kolom yang ditunjuk (seperti pada gambar 2.6b).



Gambar 2.6a. Pengisian jika kosong



Gambar 2.6b. Pengisian jika sudah terisi

BAB III

PERANCANGAN SISTEM

3.1. Gambaran Umum Sistem

Quatropolly merupakan salah satu game papan. Pertama kali pemain diharapkan untuk memilih ukuran papan yang telah disediakan. Setelah ukuran papan dipilih maka dapat memilih level permainan yang diinginkan. Game ini dirancang untuk melawan komputer. Perhitungan komputer menggunakan algoritma *alpha-beta pruning*. Perhitungan ini dipergunakan agar komputer lebih pintar.

3.2. Perancangan Umum

Pertama kali pemain akan memilih ukuran papan permainan yang akan dimainkannya. Untuk ukuran papan akan disediakan tiga ukuran papan yakni:

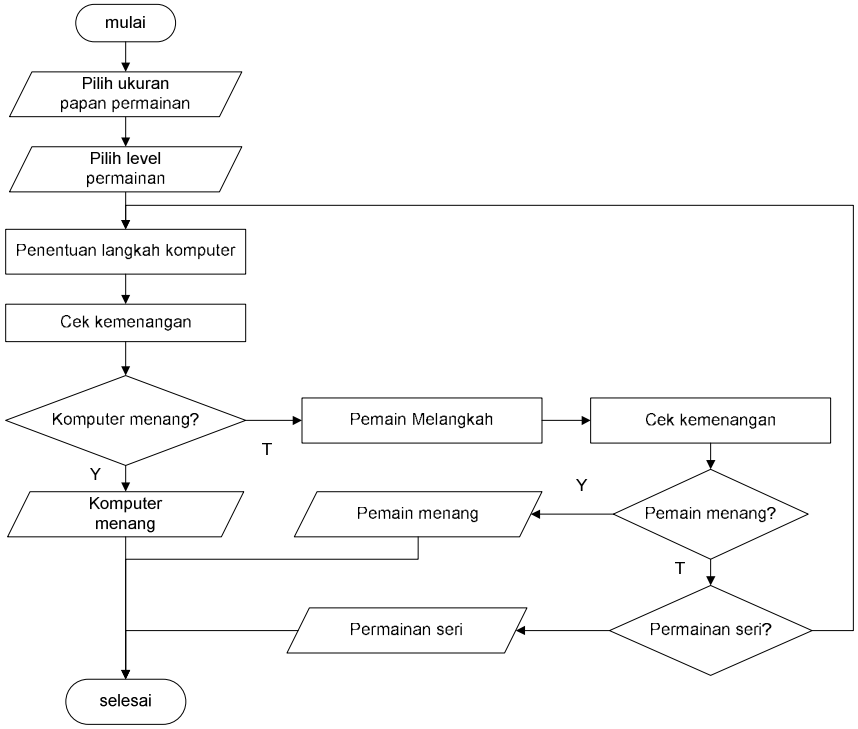
- 6X5 (6 baris dan 5 kolom)
- 6X6 (6 baris dan 6 kolom)
- 6X7 (6 baris dan 7 kolom)

Setelah memilih ukuran papan, user akan memilih level permainan yang diinginkan. Untuk level permainan disediakan 3 level permainan, yakni:

- Easy
- Normal
- Hard

Karena permainan ini dirancang untuk melawan komputer, setelah pemain memilih level permainan akan dimulai. Penentuan langkah komputer akan

ditentukan berdasarkan level permainan yang dipilih pemain sebelumnya. Cek kemenangan akan selalu dilakukan agar mengetahui siapa yang terlebih dahulu memperoleh kemenangan. Jika pada saat dicek ternyata komputer terlebih dahulu memperoleh kemenangan maka sistem akan menampilkan pemberitahuan bahwa komputer menang. Jika tidak maka langkah selanjutnya akan dilakukan oleh pemain. Kemudian dicek lagi jika pemain menang maka sistem akan menampilkan bahwa pemain yang memperoleh kemenangan. Kemudian di cek lagi. Jika permainan seri maka sistem akan memberi tahukan bahwa permainan berakhir seri. Jika tidak maka langkah selanjutnya dilakukan oleh komputer. Demikian seterusnya. Jika salah satu pemain berhasil menang maka pemainan berakhir.

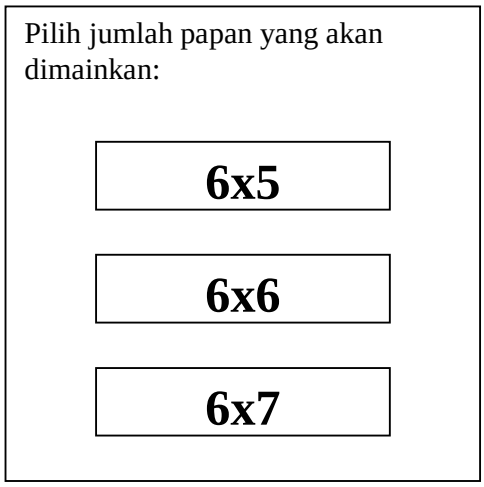


Gambar 3.1. Flow chart untuk perancangan umum

3.3. Perancangan Antarmuka

3.3.1. Desain Input

Untuk inputan, pemain akan memilih ukuran papan dan memilih level permainan yang akan dimainkan sesuai dengan ukuran papan dan level permainan yang disediakan.

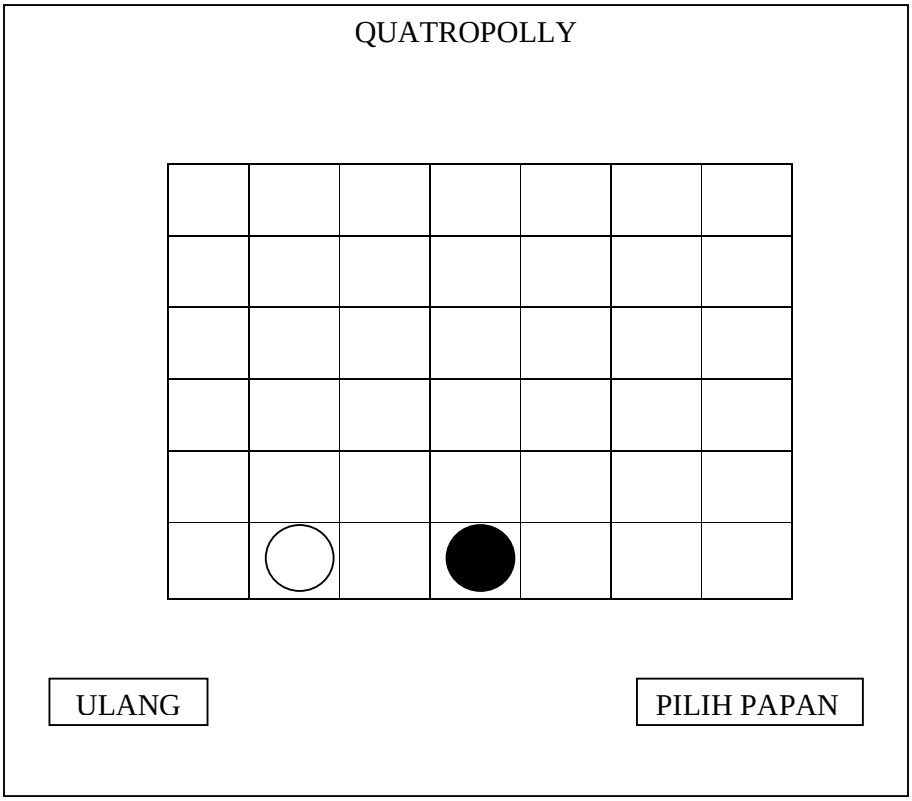


Gambar 3.2. Pemilihan ukuran papan

Gambar 3.3. Pemilihan level

Permainan akan memerlukan :

- papan permainan
- bola



Gambar 3.4.
Perancangan papan permainan

3.3.2. Desain Output

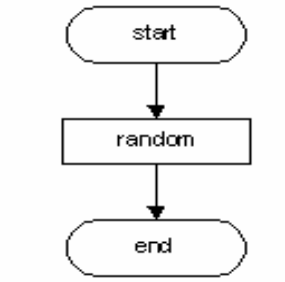
Output akan tampil jika komputer atau pemain lebih dulu memperoleh kemenangan. Jika komputer menang, maka akan ditampilkan pesan ‘Komputer menang’. Jika pemain menang maka pesan yang ditampilkan ‘SELAMAT!!, anda menang...’. Jika permainan berakhir seri maka akan ditampilkan pesan ‘wah... seri...’.

3.4. Perancangan Proses

Langkah komputer dibagi menjadi 3 level permainan.

3.4.1. Level easy

Untuk level *easy* (seperti gambar 3.5.), langkah yang diambil komputer nantinya bersifat *random.*

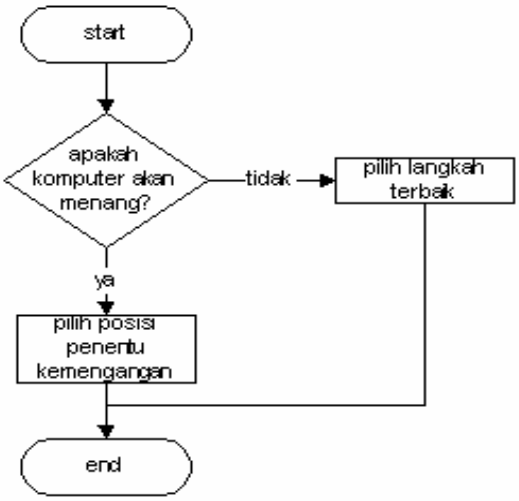


Gambar 3.5. langkah komputer level easy

3.4.2. Level normal

Untuk level *normal* (seperti gambar 3.6.), komputer akan melakukan beberapa pengecekan:

- Komputer akan mengecek apakah dengan meletakkan di posisi tertentu komputer akan menang? Jika jawabannya ya maka komputer akan memilih untuk meletakkannya di posisi tersebut. Namun apabila tidak, maka komputer akan memilih langkah terbaik.

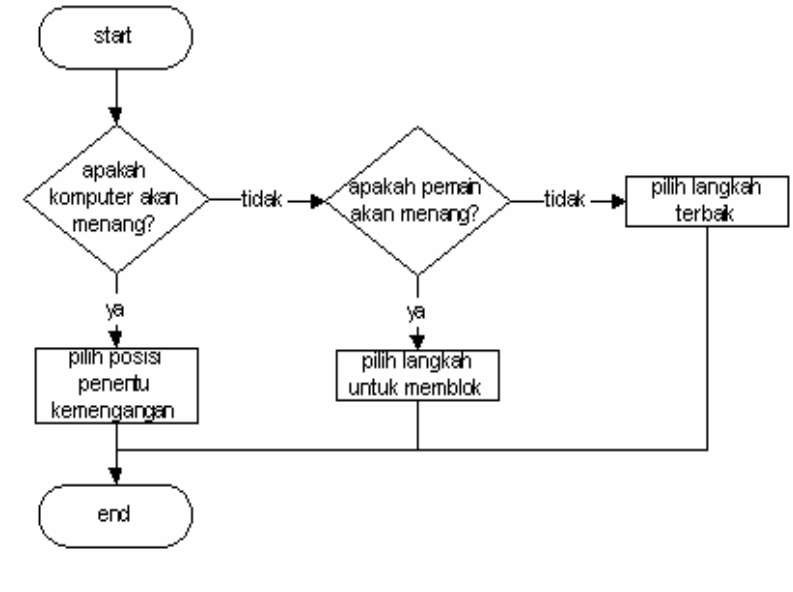


Gambar 3.6. langkah komputer untuk level normal

3.4.3. Level hard

Untuk level hard (seperti gambar 3.7.), komputer akan melakukan beberapa pengecekan:

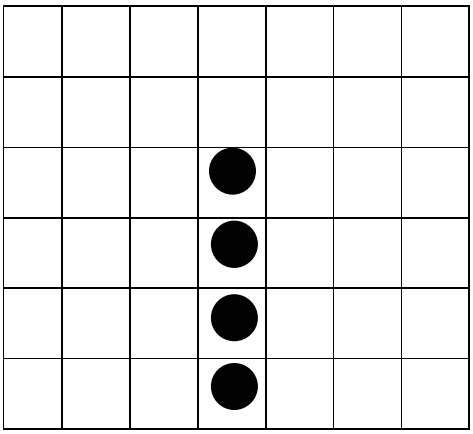
- Komputer akan mengecek apakah dengan meletakkan di posisi tertentu komputer akan menang? Jika jawabannya ya maka komputer akan memilih untuk meletakkannya di posisi tersebut. Namun apabila tidak, komputer akan mengecek apakah pemain akan menang? Jika ya maka komputer akan memilih langkah untuk memblok. Jika tidak komputer akan memilih langkah terbaik.



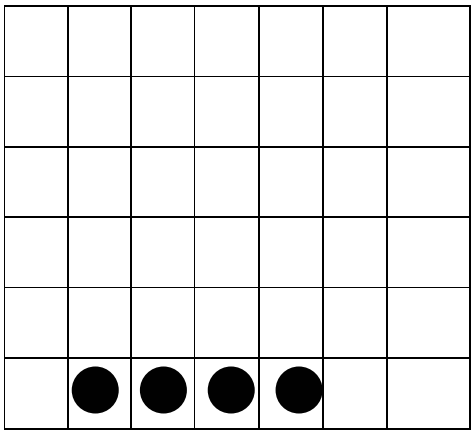
Gambar 3.7. Langkah komputer untuk level hard

3.4.4. Pengecekan kondisi menang

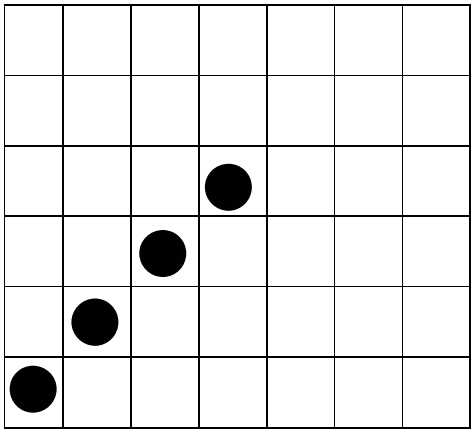
Untuk permainan ini, kemenangan akan diperoleh apabila pemain ataupun komputer terlebih dahulu berhasil membariskan bola sebanyak empat bola baik secara vertikal, horisontal maupun diagonal.



Gambar 3.8. kemenangan vertikan untuk papan 7x6

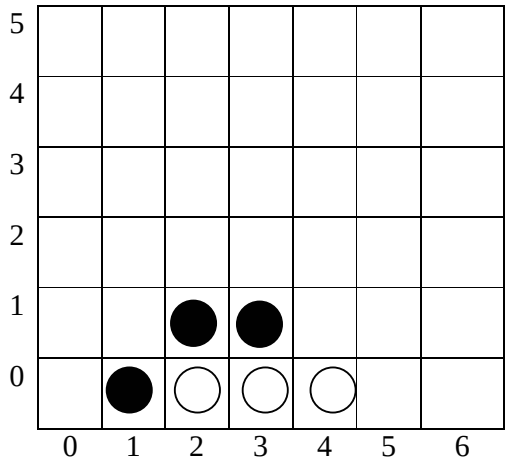


Gambar 3.9. kemenangan horisontal untuk papan 7x6



Gambar 3.10. kemenangan diagonal untuk papan 7x6

Komputer akan memilih langkah terbaik yang berarti juga memilih langkah yang memblok langkah pemain jika akan memperoleh kemenangan. Komputer akan mengecek apakah bola pemain ada yang berbaris sebanyak tiga, baik secara vertikal, horisontal maupun diagonal. Jika ada maka pengeblokan akan dilakukan.



Gambar 3.11. Pengeblokan yang akan dilakukan oleh bola hitam

Seperti gambar 3.11. apabila komputer akan mendapat giliran selanjutnya, maka apabila level permainan yang dipilih adalah *hard* atau normal, maka komputer akan melakukan pemblokan dengan meletakkan bola hitam pada baris ke nol dan kolom ke lima. Ini dikarenakan akan terdapat kombinasi kemenangan untuk pemain yang berjumlah 16 atau dengan kata lain jumlah bola akan lebih dari 3 jika pemain meletakkan bola pada baris ke nol dan kolom ke lima.

3.4.5. Kondisi awal

Untuk menentukan langkah terbaik komputer maka komputer akan melakukan perhitungan terlebih dahulu. Untuk nilai awal masing masing posisi akan diberi nilai 0. Nilai 0 ini dimaksudkan untuk mengetahui bahwa sel masing masing papan permainan masih kosong.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Sel papan masih kosong

Untuk mengetahui pemilik bola maka sel papan memerlukan satu atribut lagi untuk menyimpan kode pemain. Jika bernilai -1 maka sel tersebut telah diisikan oleh pemain. Namun jika bernilai 1 maka papan tersebut telah diisikan oleh komputer.

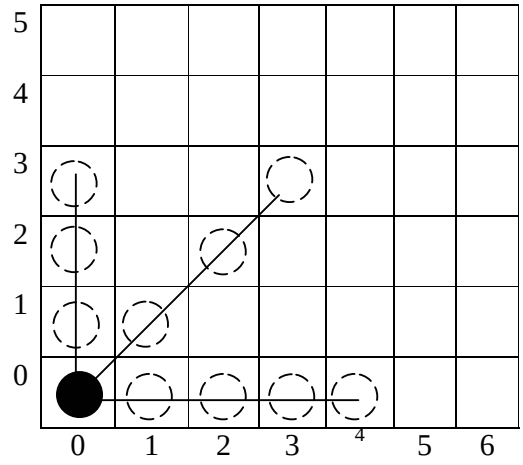
Untuk papan yang masih kosong, diset nilainya menjadi 0 dan jika telah terisikan bola, akan diberi nilai sesuai dengan nilai pemiliknya (1 atau -1). Nilai ini digunakan untuk mengetahui sel mana yang telah berisi bola. Jika pemain atau komputer akan meletakkan dikolom yang telah berisikan bola maka pengisian bola nantinya akan diatas dari papan yang telah berisikan bola

3.4.6. Skor pemain

Skor pemain untuk kombinsi kemenangan diset dengan nilai 2^b. Dengan b = jumlah bola yang telah dibentuk dalam suatu kombinasi kemenangan. Jika pemain telah berhasil menyusun 4 bola dalam suatu kombinasi kemenangan (baris atau kolom ataupun diagonal) maka nilai yang akan didapat adalah 2^4=16. Jika komputer atau pemain terlebih dahulu memperoleh nilai 16 untuk satu kombinasi kemenangan maka akan keluar sebagai pemenang. Jika suatu kombinasi kemenangan tertentu sudah tidak memungkinkan lagi (telah di blok atau dihalangi pemain lain), maka nilainya di set menjadi nol.

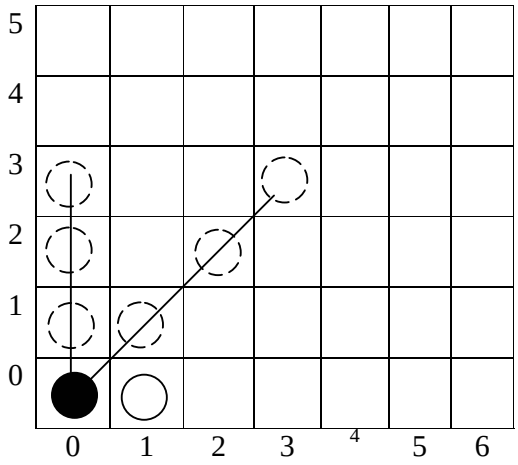
3.4.7. Bobot masing masing sel papan

Untuk masing masing sel pada papan permainan mempunyai bobot tertentu. Seperti pada gambar 3.12. Untuk sel yang telah berisikan bola akan mempunyai bobot 6. Kombinasi kemenangan yang mungkin didapat: baris, kolom maupun diagonal. Karena untuk masing masing kombinasi kemenangan hanya mempunyai bola 1 dan untuk satu kombinasi kemenangan dengan satu bola akan mempunyai nilai 2^1 maka untuk ketiga kombinasi tersebut akan mempunyai nilai 6.



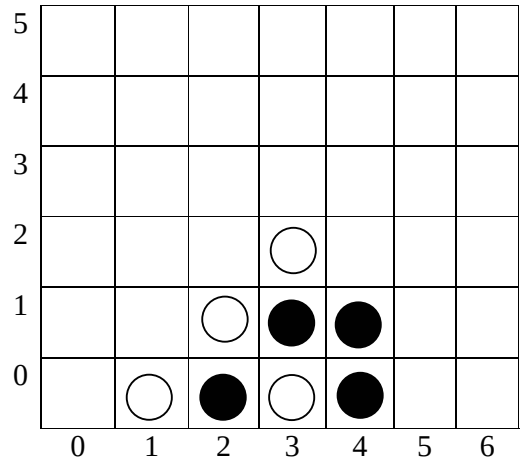
Gambar 3.12. Kombinsai kemenangan yang mungkin

Untuk gambar 3.13. nilai dari sel (0,0) akan bernilai 4. karena bola putih sudah menghalangi untuk kombinasi kemenangan baris, maka kombinasi kemeangan bola hitam tidak memungkinkan lagi sehingga nilainya di set menjadi 0.



Gambar 3.13. Kombinsai kemenangan yang terhambat

Langkah terbaik yang dipilih komputer nantinya akan memberikan nilai terbaik bagi komputer. Untuk pemilihan nilai terbaik, komputer akan memilih nilai bukan hanya dari besarnya tetapi juga menghitung nilai yang akan didapat lawannya (pemain) sehingga nantinya nilai yang didapat lawan menjadi lebih kecil (minimisasi) dan nilai yang didapat komputer lebih besar (maksimisasi). Untuk papan yang telah memiliki beberapa bola (seperti gambar 3.14), maka perhitungan untuk masing masing kombinasi dapat dilakukan dengan menghitung jumlah bola yang ada untuk setiap kombinasi kemenangan yang masih mungkin. Untuk mempermudahnya setiap kombinasi kemenangan dapat digunakan tabel 3.1. Skor total yang diperoleh oleh pemain A (bola putih) yakni 34. Sedangkan skor total yang diperoleh pemain B yakni 24. Dari tabel 3.1. dapat diketahui pula kombinasi yang mana saja sudah tidak mungkin mencapai kemenangan (nilainya sudah menjadi 0). Jika sudah terdapat suatu kombinasi yang bernilai 16 maka pemain yang memiliki nilai kombinasi tersebut sudah memperoleh kemenangan.



Gambar 3.14 papan berisi beberapa bola

Pada papan gambar 3.14, jika yang mendapat giliran selanjutnya adalah komputer (menggunakan bola putih), maka komputer akan memiliki tujuh kemungkinan langkah (0,0), (1,1), (2,2), (3,3), (4,2), (5,0), dan (6,0)

Bobot untuk satu sel papan untuk kemungkinan langkah selanjutnya untuk satu kotak adalah:

- Untuk sel papan 0,0 akan memiliki nilai 4
- Untuk sel papan 1,1 akan memiliki nilai 4
- Untuk sel papan 2,2 akan memiliki nilai 20
- Untuk sel papan 3,3 akan memiliki nilai 12
- Untuk sel papan 4,2 akan memiliki nilai 16
- Untuk sel papan 5,0 akan memiliki nilai 2
- Untuk sel papan 6,0 akan memiliki nilai 4

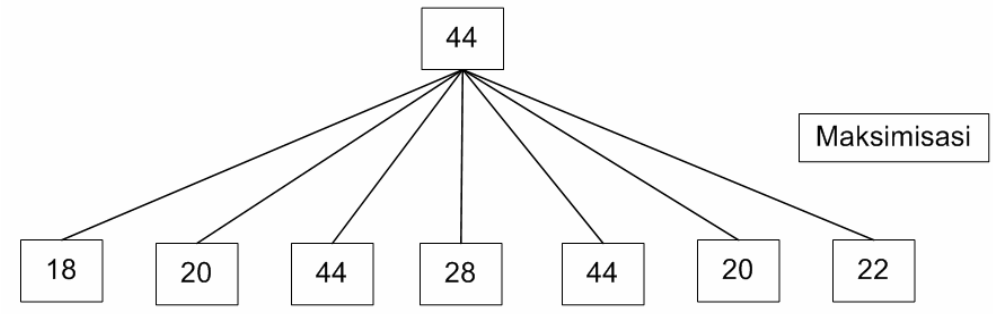
Perhitungan bobot total diperoleh dari pengurangan nilai total (Nilai total diperoleh dengan menjumlahkan nilai untuk semua kemungkinan) untuk pemain

yang akan meletakkan bola (yang mendapat giliran) dengan pemain lawan. Maka nilai total masing masing sel papan akan diperoleh:

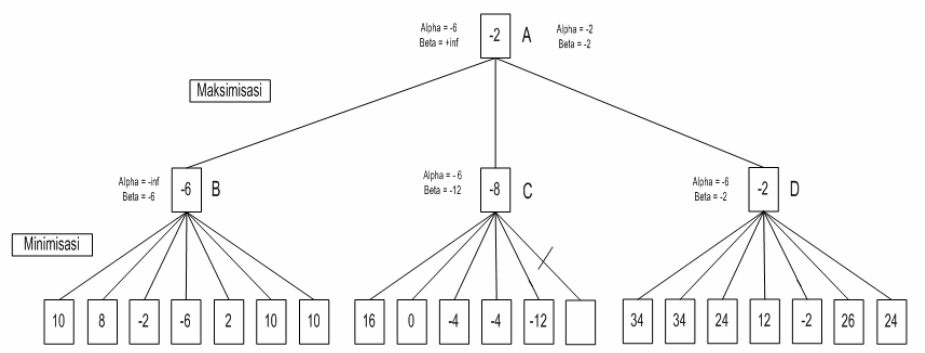
- Untuk sel papan 0,0 akan memiliki bobot 18
- Untuk sel papan 1,1 akan memiliki bobot 20
- Untuk sel papan 2,2 akan memiliki bobot 44
- Untuk sel papan 3,3 akan memiliki bobot 28
- Untuk sel papan 4,2 akan memiliki bobot 44
- Untuk sel papan 5,0 akan memiliki bobot 20
- Untuk sel papan 6,0 akan memiliki bobot 22

Jika untuk perhitungan satu langkah kedepan maka komputer akan memilih sel papan yang bernilai terbesar. Ini dikarenakan untuk satu langkah selanjutnya komputer melakukan prose maksimisasi.

Jika digambarkan dengan tree akan menampilkan seperti gambar Gambar 3.15.



Gambar 3.15 Tree untuk satu langkah kedepan



Gambar 3.16 Tiga kemungkinan awal

Pada gambar 3.16. merupakan tiga kemungkinan awal langkah 2 lapis yang dilakukan komputer. Untuk anak anak dari simpul B akan dilakukan proses minimisasi sehingga didapat *beta* sama dengan minus enam. Nilai *beta* ini akan memperbaiki nilai simpul B menjadi minus enam. Anak dari simpul A akan dilakukan proses maksimisasi, sehingga nilai *alpha* yang diperoleh dari simpul B adalah minus enam. Anak dari simpul C paling kanan dipotong dikarenakan nilai *beta* yang diperoleh sebelumnya bernilai minus duabelas. Nilai ini jauh lebih kecil dari *alpha* yang bernilai minus enam. Sehingga simpul selanjutnya dipotong dan tidak usah dihitung. Dari simpul D diperoleh nilai *beta* sama dengan minus dua. Nilai ini kemudian memperbaiki nilai D sehingga bernilai minus dua. Pada saat simpul A melakukan proses maksimisasi maka simpul A akan memilih simpul D karena simpul ini memiliki nilai paling besar, sehingga nilai ini mengubah *alpha* dari minus enam menjadi minus dua dan mengubah nilai simpul A menjadi dua.

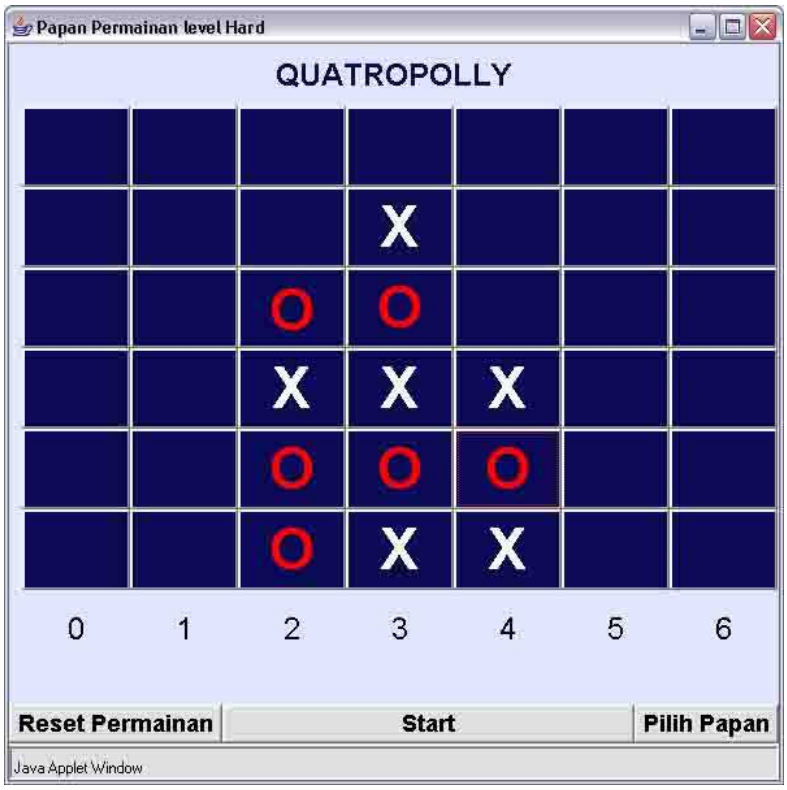
Tabel 3.1.

Tabel Kombinasi kemenangan untuk gambar 3.14.

NO	Kombinasi	A	B	Keterangan
1	{(0,0),(1,0),(2,0),(3,0)}	0	0	Baris 0
2	{(1,0),(2,0),(3,0),(4,0)}	0	0	Baris 0
3	{(2,0),(3,0),(4,0),(5,0)}	0	0	Baris 0
4	{(3,0),(4,0),(5,0),(6,0)}	0	0	Baris 0
5	{(0,1),(1,1),(2,1),(3,1)}	0	2	Baris 1
6	{(1,1),(2,1),(3,1),(4,1)}	0	2	Baris 1
7	{(2,1),(3,1),(4,1),(5,1)}	0	2	Baris 1
8	{(3,1),(4,1),(5,1),(6,1)}	0	2	Baris 1
9	{(0,2),(1,2),(2,2),(3,2)}	2	0	Baris 2
10	{(1,2),(2,2),(3,2),(4,2)}	2	0	Baris 2
11	{(2,2),(3,2),(4,2),(5,2)}	2	0	Baris 2
12	{(3,2),(4,2),(5,2),(6,2)}	2	0	Baris 2
13	{(0,2),(0,3),(0,4),(0,5)}	0	0	Kolom 0
14	{(1,0),(1,1),(1,2),(1,3)}	2	0	Kolom 1
15	{(2,1),(2,2),(2,3),(2,4)}	2	0	Kolom 2
16	{(3,0),(3,1),(3,2),(3,3)}	0	0	Kolom 3
17	{(3,1),(3,2),(3,3),(3,4)}	0	0	Kolom 3
18	{(3,2),(3,3),(3,4),(3,5)}	2	0	Kolom 3
19	{(4,0),(4,1),(4,2),(4,3)}	0	4	Kolom 4
20	{(1,0),(2,1),(3,2),(4,3)}	8	0	Diagonal kedepan baris 0
21	{(2,0),(3,1),(4,2),(5,3)}	0	4	Diagonal kedepan baris 0
22	{(3,0),(4,1),(5,2),(6,3)}	0	0	Diagonal kedepan baris 0
23	{(2,1),(3,2),(4,3),(5,4)}	4	0	Diagonal kedepan baris 1
24	{(3,1),(4,2),(5,3),(6,4)}	0	2	Diagonal kedepan baris 1
25	{(3,2),(4,3),(5,4),(6,5)}	2	0	Diagonal kedepan baris 2
26	{(3,0),(2,1),(1,2),(0,3)}	4	0	Diagonal Kebelakang baris 0
27	{(4,0),(3,1),(2,2),(1,3)}	0	4	Diagonal Kebelakang baris 0
28	{(3,1),(2,2),(1,3),(0,4)}	0	2	Diagonal Kebelakang baris 1
29	{(3,2),(2,3),(1,4),(0,5)}	2	0	Diagonal Kebelakang baris 2
	Jumlah	34	24	

3.5. Contoh kasus

Pada gambar 3.17. komputer menggunakan X dan pemain menggunakan O. langkah selanjutnya akan dilakukan oleh komputer.



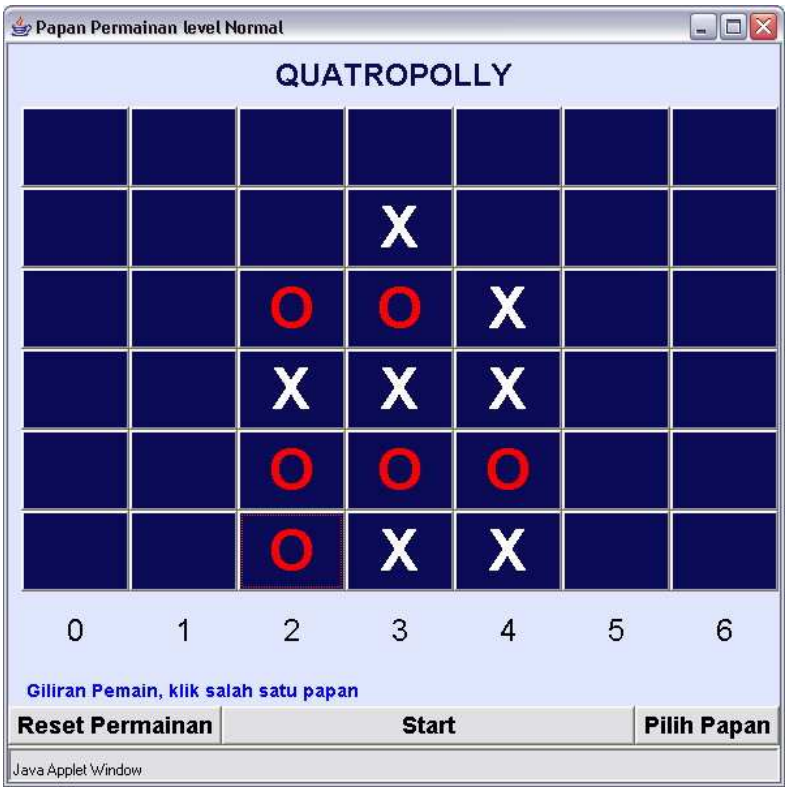
Gambar 3.17.
Posisi awal

Untuk langkah komputer (X) pertama akan dihitung masing masing bobot. Nilai sell papan :

- Untuk sell papan 0,0 akan memiliki bobot 5.
- Untuk sell papan 3,2 akan memiliki bobot 5.
- Untuk sell papan 2,4 akan memiliki bobot 4.
- Untuk sell papan 3,5 akan memiliki bobot 13
- Untuk sell papan 4,3 akan memiliki bobot 17
- Untuk sell papan 5,0 akan memiliki bobot 12

- Untuk sell papan 6,0 akan memiliki bobot 12

Dengan melihat bobot tersebut, untuk proses maksimisasi pada satu langkah kedepan otomatis komputer akan memilih sell papan yang mempunyai bobot terbesar. Sehingga komputer akan meletakkan pada sell papan 4,3 (Gambar 3.18.).



Gambar 3.18.
Langkah komputer normal

Untuk langkah komputer hard, komputer akan melakukan perhitungan untuk dua langkah selanjutnya. Ini berarti komputer akan menghitung kemungkinan langkah pemain jika komputer meletakkan di salah satu sell papan.

Kemungkinan pertama (gambar 3.19.), jika komputer meletakkan pada sell papan 0,0 maka pemain memiliki tujuh kemungkinan langkah, yakni (0,1), (1,0), (2,4), (3,5), (4,3), (5,0) dan (6,0).

Pada saat penelusuran simpul A akan terjadi minimisasi yang nantinya akan memperbaiki nilai *beta*. Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus dua belas. Kemudian dibandingkan dengan cabang yang kedua. Karena cabang kedua lebih besar maka nilai *beta* tetap menjadi minus dua belas. Saat dibandingkan dengan cabang ke lima, karena cabang ke lima nilainya lebih kecil dari cabang sebelumnya maka nilai *beta* akan berubah menjadi minus dua puluh tiga. Kemudian cabang A akan bernilai dengan minus dua puluh tiga. Dari nilai *beta* yang diperoleh untuk simpul A akan memperbaiki nilai simpul A menjadi minus dua puluh tiga dan *alpha* menjadi minus dua puluh tiga.

Kemungkinan kedua (gambar 3.20.), jika komputer meletakkan pada sell papan 1,0 maka pemain memiliki tujuh kemungkinan langkah, yakni (0,0), (1,1), (2,4), (3,5), (4,3), (5,0) dan (6,0). Pada saat penelusuran simpul B akan terjadi minimisasi yang nantinya akan memperbaiki nilai *beta*. Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus lima. Kemudian dibandingkan dengan cabang yang kedua. Karena cabang kedua lebih kecil maka nilai *beta* akan menjadi minus dua puluh delapan. Karena nilai *alpha* sama dengan dua puluh tiga dan *beta* sama dengan dua puluh delapan maka *alpha* lebih besar dari *beta* sehingga proses pencarian dihentikan. Dan untuk simpul yang lain dipotong. Kemudian cabang B akan bernilai dengan minus dua puluh delapan dan *alpha* bernilai dua puluh tiga.

Kemungkinan ketiga (gambar 3.21.), jika komputer meletakkan pada sell papan 2,4 maka pemain memiliki tujuh kemungkinan langkah, yakni (0,0), (1,0), (2,5), (3,5), (4,3), (5,0) dan (6,0). Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus lima. Kemudian dibandingkan dengan cabang yang lain.

Karena cabang kelima lebih kecil maka nilai *beta* akan menjadi minus tiga puluh tiga. Pencarian untuk cabang yang lain dihentikan dan dipotong ini karena nilai *alpha* lebih besar dari nilai *beta*. Kemudian cabang C akan bernilai dengan minus tiga puluh tiga dan *alpha* bernilai dua puluh tiga.

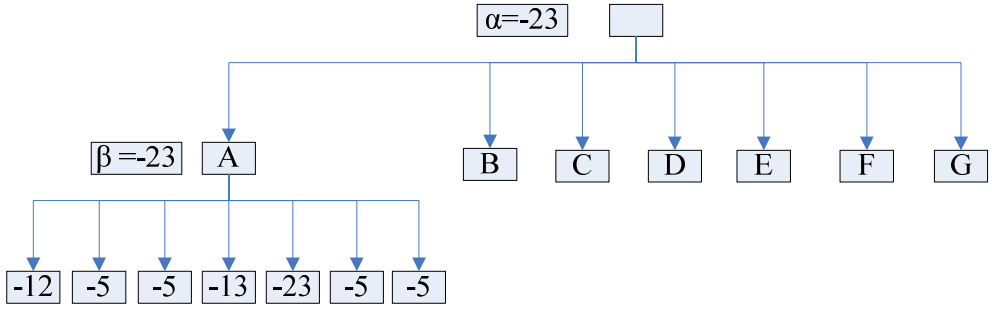
Kemungkinan keempat (gambar 3.22.), jika komputer meletakkan pada sell papan 3,5 maka pemain memiliki enam kemungkinan langkah, yakni (0,0), (1,0), (2,4), (4,3), (5,0) dan (6,0). Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus lima. Kemudian dibandingkan dengan cabang yang lain. Karena cabang keempat lebih kecil maka nilai *beta* akan menjadi minus tiga puluh tiga. Pencarian untuk cabang yang lain dihentikan dan dipotong ini karena nilai *alpha* lebih besar dari nilai *beta*. Kemudian cabang D akan bernilai dengan minus tiga puluh tiga dan *alpha* bernilai dua puluh tiga.

Kemungkinan kelima (gambar 3.23.), jika komputer meletakkan pada sell papan 4,3 maka pemain memiliki tujuh kemungkinan langkah, yakni (0,0), (1,0), (2,4),(3,5), (4,4), (5,0) dan (6,0). Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus lima. Kemudian dibandingkan dengan cabang yang lain. Setelah penyelekasian sampai akhir maka didapat nilai *beta* menjadi minus tiga belas. Dari nilai *beta* yang diperoleh untuk simpul E akan memperbaiki nilai simpul E menjadi minus tiga belas dan *alpha* menjadi minus tiga belas.

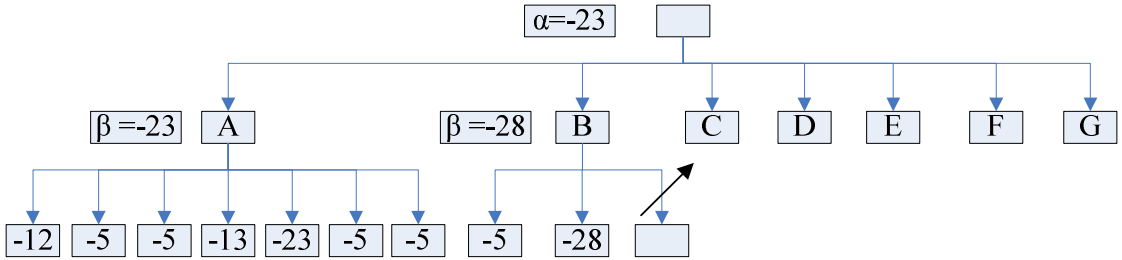
Kemungkinan keenam (gambar 3.24.), jika komputer meletakkan pada sell papan 5,0 maka pemain memiliki tujuh kemungkinan langkah, yakni (0,0), (1,0), (2,4),(3,5), (4,3), (5,1) dan (6,0). Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus lima. Kemudian dibandingkan dengan cabang yang lain. Karena cabang kelima lebih kecil maka nilai *beta* akan menjadi minus tiga

puluh tiga. Pencarian untuk cabang yang lain dihentikan dan dipotong ini karena nilai *alpha* lebih besar dari nilai *beta*. Dari nilai *beta* yang diperoleh untuk simpul F akan memperbaiki nilai simpul F menjadi minus tiga puluh tiga dan *alpha* menjadi minus tiga belas.

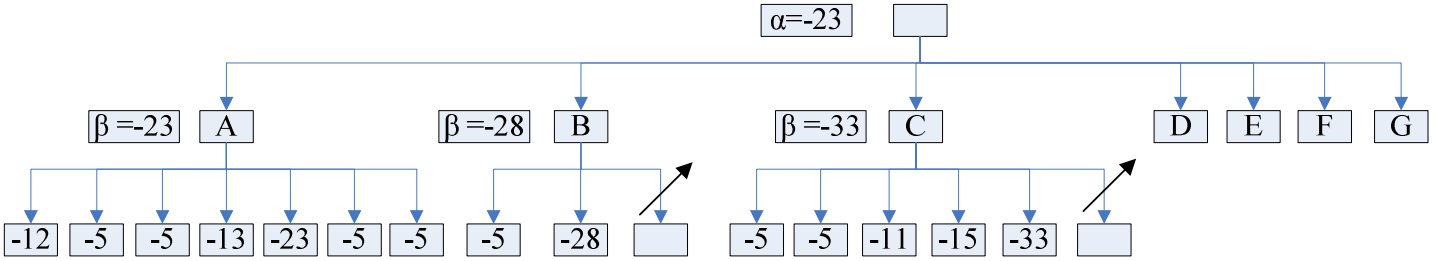
Kemungkinan ketujuh (gambar 3.25.), jika komputer meletakkan pada sell papan 6,0 maka pemain memiliki tujuh kemungkinan langkah, yakni (0,0), (1,0), (2,4),(3,5), (4,3), (5,0) dan (6,1). Untuk cabang pertama akan menghasilkan nilai *beta* sama dengan minus lima. Kemudian dibandingkan dengan cabang yang lain. Karena cabang kelima lebih kecil maka nilai *beta* akan menjadi minus tiga puluh tiga. Pencarian untuk cabang yang lain dihentikan dan dipotong ini karena nilai *alpha* lebih besar dari nilai *beta*. Dari nilai *beta* yang diperoleh untuk simpul G akan memperbaiki nilai simpul G menjadi minus tiga puluh tiga dan *alpha* menjadi minus tiga belas.



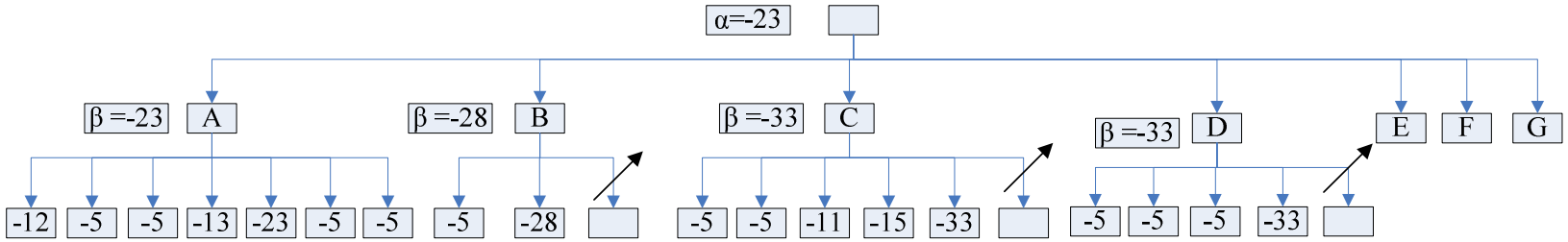
Gambar 3.19.
Kemungkinan pertama langkah hard



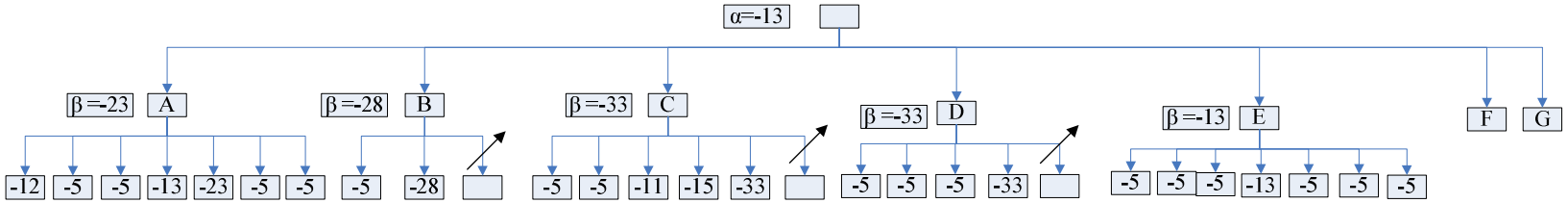
Gambar 3.20.
Kemungkinan kedua langkah hard



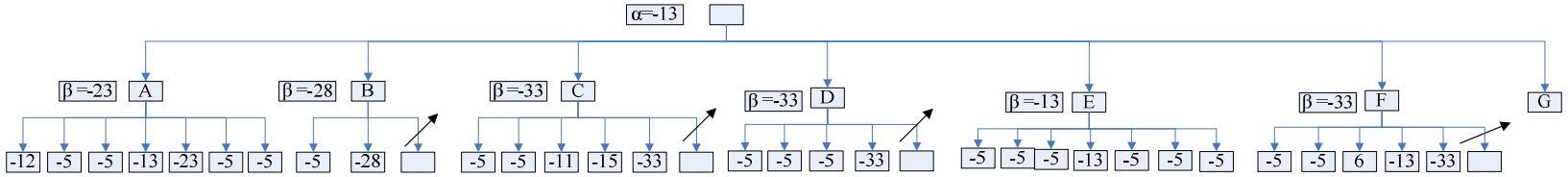
Gambar 3.21.
Kemungkinan ketiga langkah hard



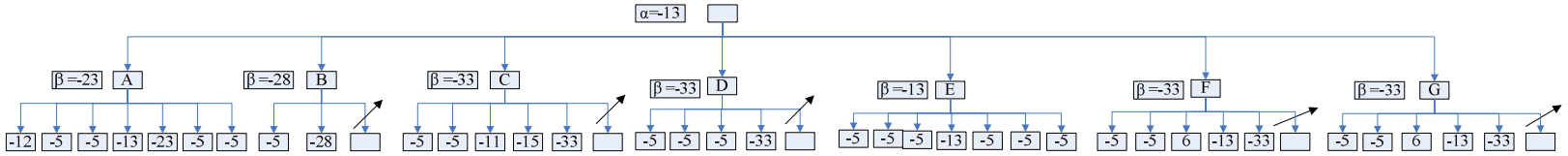
Gambar 3.22.
Kemungkinan keempat langkah hard



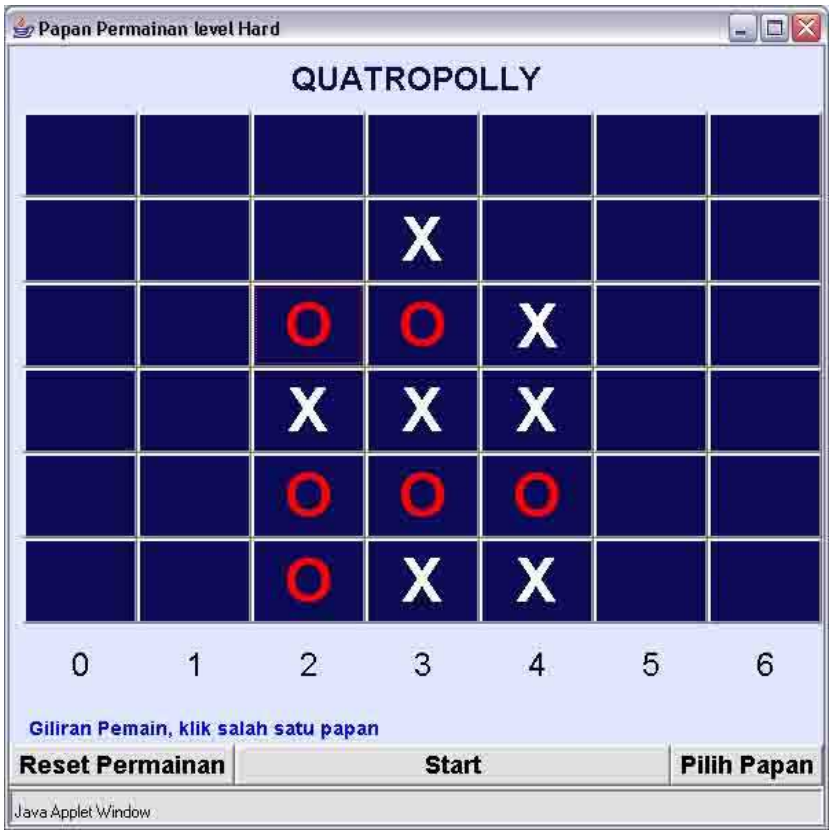
Gambar 3.23.
Kemungkinan kelima langkah hard



Gambar 3.24.
Kemungkinan keenam langkah hard



Gambar 3.25.
Kemungkinan ketujuh langkah hard



Gambar 3.26.
Langkah komputer Hard

Dengan melihat kemungkinan kemungkinan yang ada, maka komputer akan memilih langkah terbaik yakni komputer akan meletakkan pada sell papan 4,3 (gambar 3.26.).

BAB IV

IMPLEMENTASI

Permainan Quatropolly ini dikembangkan dengan menggunkan bahasa pemrograman java. Bahasa pemrograman java merupakan bahasa pemrograman yang tidak bergantung pada platform yang artinya java dapat dijalankan pada sembarang komputer bahkan pada sembarang sitem operasi. Permainan quatropolly ini dirancang untuk melawan komputer.

4.1. Implementasi

4.1.1. Struktur data

```
sellPapan[kol][bar] = new Button();
sellPapan[kol][bar].addActionListener(new sellPapanListener(kol));
```

Papan permainan direpresentasikan dalam bentuk *array* sellPapan[kol][bar] dengan tipe button dua dimensi dengan ukuran tuju kali enam atau enam kali enam atau lima kali enam sesuai dengan pilihan pemain pada saat pemilihan ukuran papan. Masing masing sel papan dapat bernilai nol, satu atau minus satu. Jika suatu sell papan masih kosong atau belum terisi maka sel papan tersebut akan bernilai nol. Jika bernilai satu maka sell papan tersebut telah berisikan bola dan bola tersebut adalah milik komputer. Namun jika sell papan bernilai minus satu maka sell papan tersebut dimiliki oleh pemain. Jadi disamping dapat mengetahui apakah sel papan telah terisi atau belum, dapat juga diketaui siapa pemiliknya.

4.1.2. Penanganan kejadian (*Event Handler*)

```
class utamatombolListener implements ActionListener{
      public void actionPerformed(ActionEvent e) {
         if (e.getSource() == start) {
            new ukuranPapan().setVisible(true);
         }
         else
         {
            new bantuan().setVisible(true);
         }
      }
   }
```

Pada saat user meng-klik suatu objek dari button, maka event akan tercipta. Event ini kemudian akan “ditangkap” *event listener* karena setiap komponen sudah ada identitasnya masing masing, seperti start atau help (pada halaman utama) dan masing masing memiliki *event listener* sehingga Java akan mengenali komponen mana yang menstimulasi *event*. Kemudian diperlukan *even handler* yaitu blok yang akan memproses bila terjadi suatu *event*. Seperti pada saat tombol ‘Mulai Permainan’ atau ‘Bantuan’ pada menu utama di tekan, maka *event* (kejadian) ini akan di *handler* (tangani) oleh ‘utamatombolListener’. Di utamatombolListener akan di bandingkan apakah sourcenya sama dengan start atau tidak. Jika ya maka akan menampilkan ukuran papan, namun jika tidak akan menampilkan layar bantuan.

4.1.3. Langkah pemain dan langkah komputer

4.1.3.1. Langkah pemain

```
public void langkahPemain(int kol)
   {
      if (pemenang!=0)
      {
         return;
      }

      if (Play(interf.PEMAIN,kol))
      {
         interf.setPapan(kol,tinggi[kol]-1,interf.PEMAIN);
         if (cekMenang(kol))
         {
            pemenang = interf.PEMAIN;
            interf.setPesan(pemainMenang);
         }

         if (pemenang==0)
         {
            langkahKomp();
         }

         if (pemenang!=0)
         {
            menampilkanPemenang();
         }
      }
      else interf.setPesan(kotakPenuh);
   }
```

Pada saat pemain mengklik sell papan maka kejadian ini akan ditangani oleh fungsi sellpapanListener. Kemudian fungsi ini akan memanggil fungsi langkahPemain. Pertama kali akan dilakukan pengecekan awal, apakah permainan sudah ada pemenangnya atau belum. Jika tidak maka pemain akan mendapat giliran untuk melangkah. Pada saat pemain melangkah, pemain dapat meletakkan bola berdasarkan kolomnya. Dibaris manapun pemain mengklik tidak akan berpengaruh karena bola akan tampil di baris terbawah sesuai dengan kolom yang di pilih. Jika

kolom telah penuh dan pemain memilih untuk meletakkan bola pada kolom tersebut maka akan keluar peringatan yang menyatakan kolom tersebut telah penuh. Saat pemain telah meletakkan bola maka kolom tersebut akan di set dengan nilai pemain. Pengesetan ini dimaksudkan untuk mengetahui pemain mana yang meletakkan bola di papan (jika bernilai minus satu maka ini berarti sellpapan tersebut telah diisi bola oleh pemain. Namun jika bernilai satu maka diisi oleh komputer. Jika kosong akan bernilai nol). Setelah pemain melangkah, akan dilakukan pengecekan kembali, apakah pemain menang atau tidak. Jika iya maka akan di tampilkan pesan bahwa pemain menang, namun jika tidak maka permainan di lanjutkan dengan langkah komputer.

4.1.3.2. Langkah komputer

```
public void langkahKomp()
   {
      langkahPilihan move=cari(level);
      Play(interf.KOMP,kol);
      interf.setPapan(kol,tinggi[kol]-1,interf.KOMP);
      if (cekMenang(kol))
      {
         pemenang = interf.KOMP;
         interf.setPesan(komputerMenang);
      }
   }

langkahPilihan langkahPilihanCari(int pemainn, int level, int alpha, int beta)
   {
      langkahPilihan[] langkahpilihanM = new langkahPilihan[ambilKolom];
      int kompMoves, kol, kekuatan, i, j;
      langkahPilihan langkahterbaik = null;
      if (level<1)
      {
        return new langkahPilihan(-1,evaluasi());
      }
```

```
    kompMoves = 0;
    for (kol=0;kol<ambilKolom;kol++)
    {
       if (Play(pemainn,kol))
       {
          int skor = 2;
          papan[kol][--tinggi[kol]] = 0;
          i = 0;
          for (j=kompMoves;j>i;j--)
          {
             langkahpilihanM[j] = langkahpilihanM[j-1];
          }
          langkahpilihanM[i] = new langkahPilihan(kol,skor);
          kompMoves++;
       }
    }

    for (i=0;i<kompMoves && alpha<beta;i++)
    {
       langkahPilihan langkEvalu;
       kol = langkahpilihanM[i].getKolomn();
       Play(pemainn,kol);
       langkEvalu = langkahPilihanCari(-pemainn,level-1,-beta,-alpha);
       langkEvalu.kebalikanKekuatan();
       papan[kol][--tinggi[kol]] = 0;
       kekuatan = langkEvalu.getKekuatan();
       if (kekuatan>alpha)
       {
          alpha = kekuatan;
          langkahterbaik = new langkahPilihan(kol,kekuatan);
       }
    }

    if(langkahterbaik !=null)
    {
       return langkahterbaik;
    }
    else
    {
       return new langkahPilihan(-1,-kedalaman);
    }
  }
```

Langkah komputer akan ditangani oleh fungsi langkahkomp. Pada fungsi ini, akan memanggil fungsi langkahpilihanCari yang bertugas untuk mencari langkah

terbaik untuk komputer. Pada fungsi langkahpilihanCari diterapkan metode *alpha-beta pruning*. Dengan menggunakan metode ini pemotongan terhadap cabang cabang yang dianggap menguntungkan lawan akan ditangani oleh *alpha* dan *beta*. Pada saat menemukan posisi yang lebih baik maka nilai tersebut akan di simpan di *alpha*. Saat pencarian jika suatu cabang memiliki nilai yang malah lebih menguntungkan lawan atau dengan kata lain *alpha* nilainya lebih besar dari *beta*, maka proses pada cabang tersebut dapat di hentikan.

Perbedaan dari langkah komputer untuk *normal* dan *hard* hanya terletak pada kedalaman pencarian. Untuk langkah *normal* pencarian langkah terbaik dilakukan hanya untuk satu langkah kedepan. Sedangkan untuk langkah komputer *hard* pencarian untuk langkah terbaik dilakukan untuk dua langkah kedepan. Penggunaan algoritma *alpha-beta pruning* pada langkah komputer dapat membantu komputer untuk memilih langkah terbaik dan untuk memotong cabang cabang sehingga untuk pencarian tiga langkah kedepan dapat dilakukan lebih cepat.

```
public void langkahKomp()
   {
     double d = Math.random()*7;
     int randm = (int)d;
     interf.setPesan(wait);
     interf.setPesan(giliranPemain);
      if(Play(interf.KOMP,randm))
      {
         interf.setPapan(randm,tinggi[randm]-1,interf.KOMP);
         if (cekMenang(randm))
         {
            pemenang = interf.KOMP;
            interf.setPesan(komputerMenang);
         }
      }
      else
      {
         langkahKomp();
```

```
    }
  }
```

Untuk komputer *easy* langkah yang dipilih adalah *random*. Jadi pada saat melangkah komputer tidak akan melakukan perhitungan. Komputer akan meletakkan di sembarang tempat.

4.1.4. Pengecekan Kemenangan

```
boolean cekMenang(int kol)
  {
    int max[], pluscol, plusrow, temp, pemainn, bar;
    max=new int[1];
    bar=tinggi[kol]-1;
    pemainn=papan[kol][bar];
    for (temp=0;temp<4;temp++)
    {
      pluscol=coln[temp];
      plusrow=rown[temp];
      max[0]=1;
      maxline(pemainn,kol,bar, pluscol, plusrow,max);
      maxline(pemainn,kol,bar,-pluscol,-plusrow,max);
      if (max[0]>3)
      {
        return true;
      }
    }
    return false;
  }
```

Pengecekan kemenangan untuk pemain ataupun komputer akan ditangani oleh fungsi cekMenang. Masing masing sell papan akan diseleksi. Penyeleksian akan dilakukan berdasarkan baris, kolom diagonal depan dan diagonal belakang. Pada saat penyeleksian akan dilihat siapa dari pemilik masing masing sel papan. Jika ada sel papan yang bernilai sama, maka dipastikan pemilik dari sell papan tersebut adalah sama. Nilai dari max akan terus bertambah sesuai dengan penemuan bola yang

pemiliknya sama dan berdekatan (baik berdasarkan baris, kolom maupun diagonal). Pada saat max bernilai lebih dari tiga maka dapat dipastikan bahwa pemilik dari bola menang. Namun jika pada saat penyeleksian bola nilai bola yang disebelahnya berbeda maka kombinasi kemenangannya tidak memungkinkan kembali sehingga nilai dari max di kembalikan seperti semula.

4.5. Menampilkan Pemenang

```
void menampilkanPemenang()
   {
      int max[] = new int[1];
      int col, row, temp;
      for (col=0;col<ambilKolom;col++)
      {
         for (row=0;row<ambilBaris;row++)
         {
            int pemainn=papan[col][row];
            if (pemainn!=0)
            {
               for (temp=0;temp<4;temp++)
               {
                  int pluscol=coln[temp];
                  int plusrow=rown[temp];
                  max[0]=1;
                  maxline(pemainn,col,row, pluscol, plusrow,max);
                  maxline(pemainn,col,row,-pluscol,-plusrow,max);
                  if (max[0]>3)
                  {
                     tandai(pemainn,col,row, pluscol, plusrow);
                     tandai(pemainn,col,row,-pluscol,-plusrow);
                  }
               }
            }
         }
      }
   }
```

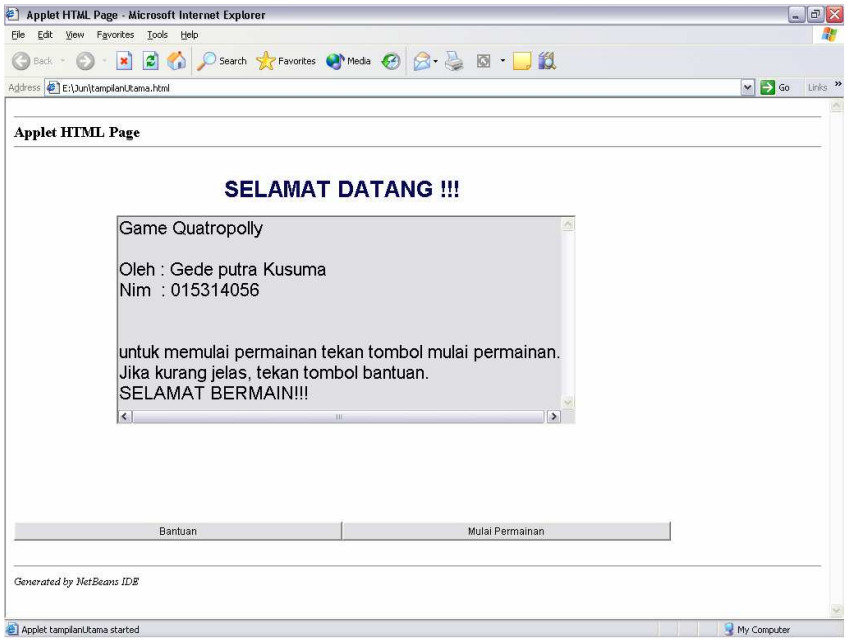
Untuk menampilkan pemenang digunakan fungsi menampilkanPemenang. Pada fungsi menampilkan pemenang akan mengecek untuk semua baris dan kolom

yang tidak bernilai nol. Ini dikarenakan jika sell papan bernilai nol berarti sell papan tersebut masih kosong. Pada saat masuk ke fungsi maxline akan didapatkan nilai dari max. Jika max bernilai lebih dari tiga maka akan masuk ke fungsi tandai. Pada fungsi tandai, sell papan yang memiliki nilai max lebih dari tiga akan menampilkan pemenangnya di papan permainan.

4.2. Tampilan program

4.2.1. Halaman Utama.

Tampilan utamanya seperti pada gambar 4.1



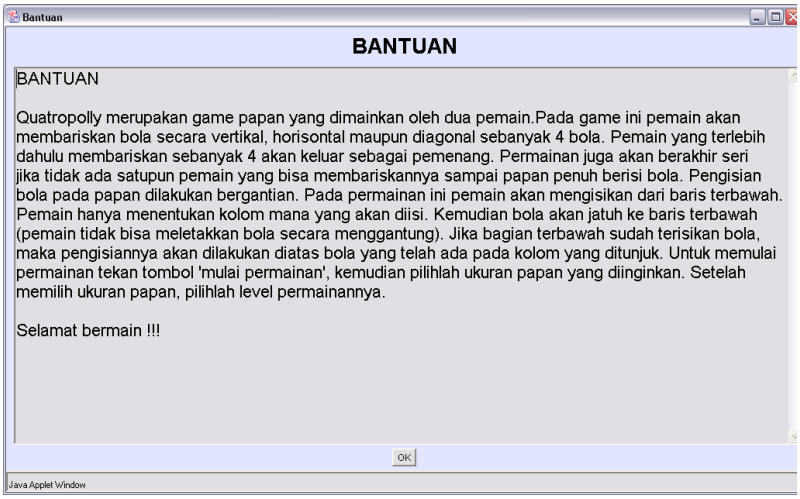
Gambar 4.1.
Halaman Utama

Pada halaman utama (gambar 4.3), terdapat 2 tombol yang dapat di tekan. Jika menekan tombol bantuan maka akan menampilkan halaman bantuan (gambar4.4.).

Setelah mengklik tombol “mulai permainan” pada halaman utama, maka pemain akan diberi tampilan ukuran papan.

4.2.2. Bantuan

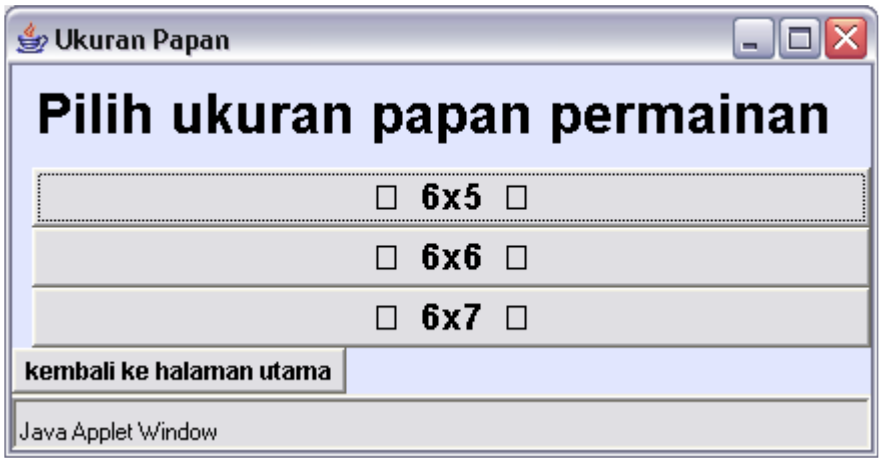
Pada halaman bantuan (gambar 4.2.) pemain dapat mengetahui aturan main dari permainan quatropolly. Dengan mengklik “ok” pada halaman bantuan, maka halaman bantuan akan di tutup dan kembali ke halaman utama.



Gambar 4.2.
Bantuan

4.2.3. Memilih ukuran papan

Pada halaman ukuran papan, pemain dapat memilih ukuran papan yang disediakan. Disediakan tiga jenis ukuran papan, yakni enam kali lima, enam kali enam dan enam kali tujuh. (gambar 4.3.). Disediakan pula tombol untuk kembali ke halaman utama jika pemain ingin kembali ke halaman utama.



Gambar 4.3.
Ukuran papan

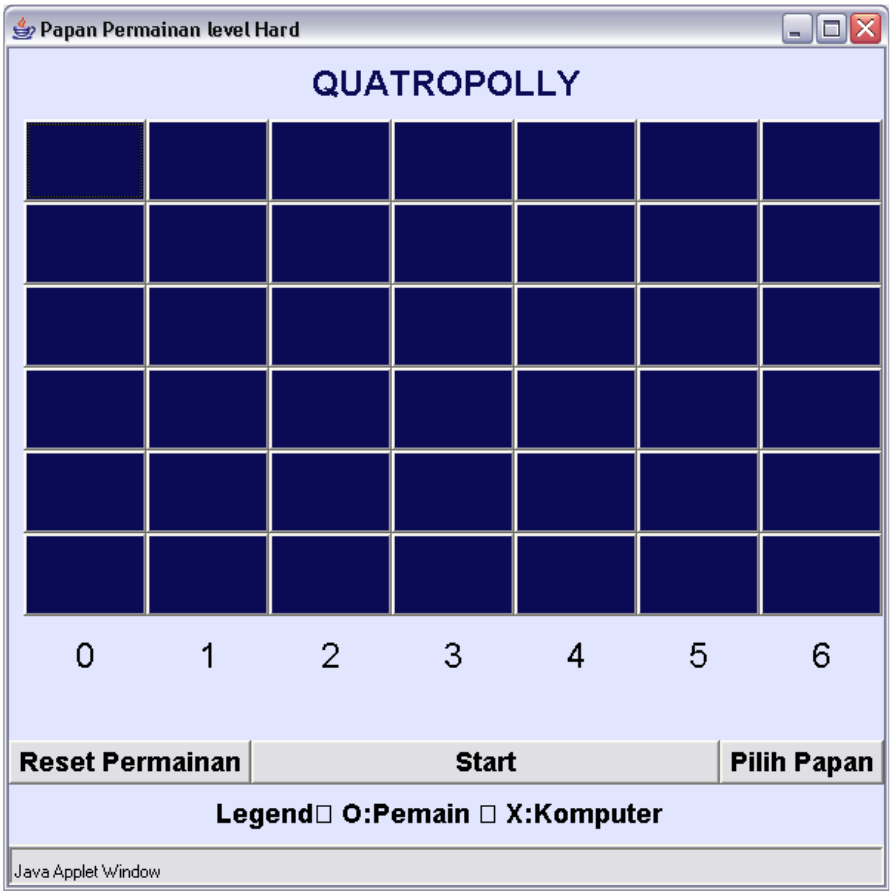
4.2.4. Memilih level permainan

Setelah memilih ukuran papan pemain diharapkan untuk memilih salah satu level permainan yang disediakan yakni *level easy*, *level normal* dan *level hard* (gambar 4.4.). Pemain dapat menekan tombol 'kembali halaman utama' untuk kembali ke halaman utama. Disediakan pula tombol untuk kembali ke menu pilih papan jika ingin bermain dengan ukuran papan yang lain.

Gambar 4.4
Level permainan

4.2.5. Papan permainan

Setelah memilih ukuran papan permainan dan level permainan, pemain akan di hadapkan ke papan permainan (gambar 4.5.). Pada papan permainan terdapat 3 tombol yang masing masing sebagai reset permainan, start, dan pilih papan. Tombol start digunakan untuk memulai permainan. Untuk mengulangi permainan digunakan tombol ‘reset permainan’. Pada saat melakukan reset permainan otomatis papan permainan dibersihkan. Pemain juga dapat mengganti ukuran papan maupun level permainan dengan menekan tombol ‘kembali pilih papan’.



Gambar 4.5.
Papan permainan

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

1. Langkah komputer dengan menggunakan metode *alpha-beta pruning* dapat memperkecil jumlah cabang yang akan dievaluasi oleh komputer.
2. Pemotongan jumlah cabang yang dievaluasi oleh komputer dapat mempersingkat waktu pencarian.
3. Kedalaman pencarian juga berpengaruh pada kecerdasan komputer, semakin dalam percarian yang dilakukan komputer, semakin pintar langkah yang di ambil oleh komputer namun waktu yang diperlukan juga bertambah.

5.2. Saran

1. Karena langkah komputer berdasarkan perhitungan jadi langkah komputer tetap sesuai dengan perhitungan. Dengan bermain beberapa kali kita dapat menebak langkah komputer selanjutnya, jadi diharapkan untuk membuat beberapa algoritma perhitungan yang lain untuk di tambahkan agar komputer lebih pintar.

DAFTAR PUSTAKA

- , 2001, Alpha-Beta, http://www.seanet.com/~brucemo/topics/alphabeta.htm, 2007.

- , 2005, *Alpha Beta Pruning*, http://www.cs.berkeley.edu/~milch/cs188/ *Alpha-Beta*.html, 2007

- , 2005, *Alpha-Beta Pruning*, www.modruby.com/articles/*Alpha-Beta_pruning*.htm, 2007

- , 2005, *Alpha-Beta Pruning*, http://www.seanet.com/~brucemo/topics/ minmax.htm, 2007

- , 2005, *Alpha-Beta pruning*, http://netlib.org/utk/ki/pcwLS1/text/node5351.html, 2007

- , 2005, Analysis *Alpha-Beta pruning*, http://www.netlib.org/utk/lsi/pcwLSI/text/node351.html, 2007

- , 2005, *Alpha Beta pruning*, http://sern.ucalgary.ca/courses/CPSC/533a/w99/presentation/L2_5B_lima_Neitz/main.html, 2007

- , 2005, *Alpha-Beta pruning*, http://en.wikipedia.org/wiki/*Alpha-Beta_pruning*, 2007.

- , 2007, Java training, http://www.diskusiweb.com/forumdisplay.php?fid=73, 2007

Hermawan , B., 2004, Menguasai Java 2 dan Object Oriented Programing, ANDI, Yogyakarta.

Kadir, A., 200 , Dasar Pemrograman Java 2, ANDI, Yogyakarta.

Kusumadewi, S., 2003, *Artificial Intelligence (Teknik dan Aplikasinya)*, Graha Ilmu, Yogyakarta.

Rich, E., 1983, *Artificial Intelligence*, Mc Graw-Hill.Inc, Texas