BAB III

ANALISIS DAN PERANCANGAN

3.1 Analisis Sistem

Analisis sistem merupakan penguraian dari suatu sistem yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan dan hambatan-hambatan yang terjadi dan kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikan.

Pembangunan perangkat lunak terdiri dari beberapa tahap, salah satunya adalah tahap menganalisa sistem. Sistem yang akan dianalisis untuk membangun perangkat lunak permainan checkers menggunakan algoritma minimax. Analisis sistem yang akan dibahas mengenai cara kerja algoritma minimax yang diimplementasikan pada permainan checkers

Tahapan analisis harus dilakukan dengan teliti agar diketahui detail yang ada dalam sistem yang berjalan saat ini. Hal-hal yang akan dianalisis terdiri dari:

1. Analisis Masalah
2. Analisis Komponen Permainan
3. Analisis Kebutuhan Non Fungsional
4. Analisis Kebutuhan Fungsional

3.1.1 Analisis Masalah

Strategi di belakang Minimax algoritma adalah komputer berasumsi bahwa kedua pemain akan main sejauh kemampuan mereka. Maka, jika lawan

mempunyai pilihan suatu gerak tidak baik atau suatu yang baik gerak, komputer akan mempunyai lawan memilih yang baik gerak. sungguh mencoba untuk memilih gerak yang mengakibatkan nilai titik yang paling rendah. Konsep relatif nyata dan sederhana ini menjadi rahasia di belakang minimax pohon. Jika komputer di program untuk mencari maksimal, maka rangkaian atau gerakan yang terbaik akan selalu ditemukan yang mengakibatkan nilai titik yang paling tinggi. Demikian juga dengan minimal, maka rangkaian atau gerakan yang terbaik akan selalu ditemukan yang mengakibatkan nilai titik yang paling rendah

3.1.2 Analisis Komponen Permainan

Beberapa komponen dalam permainan *checkers*:

1. Pemain.

   Permainan ini dimainkan oleh dua orang. Berperan sebagai lawan satu sama lain.

2. Papan.

   Papan yang digunakan dalam permainan ini mirip dengan papan catur. Terdiri dari 8 x 8 kotak-kotak kecil warna terang dan gelap. Permukaan yang dimainkan terdiri dari 32 kotak hanya gelap. Sebuah konsekuensi ini adalah bahwa, dari perspektif masing-masing player, kiri dan kanan tikungan mendorong strategi yang berbeda.

3. Koin.

Koin yang digunakan berbentuk silinder datar seperti koin pada umumnya. Koin yang digunakan pada permainan ini berwarna terang dan gelap. Pada

awalnya, warna terang direpresentasikan dengan warna putih dan warna gelap direpresentasikan dengan warna merah. Tapi pada perkembangannya,banyak warna yang digunakan untuk menggantikan warna-warna tersebut.

4. Posisi awal.

Masing-masing pemain mempunyai 12 koin sewarna yang diletakkan pada 3 baris pertama pada bidang yang terdekat dengan pemain dan diletakkan pada bidang yang berwarna gelap. Pemain yang memainkan koin berwarna gelap, memulai langkah awal terlebih dahulu.

5. Cara bergerak.

Koin bergerak secara diagonal mengikuti warna bidang yang gelap, satu di tiap langkahnya. Cara lain adalah dengan melangkahi satu buah koin lawan. Keadaan itu mungkin dapat dilakukan jika pada diagonal setelah koin lawan, merupakan bidang kosong. Jika langkah kedua itu terjadi, koin lawan yang dilangkahi mati, dan harus keluar dari bidang permainan. Koin dengan pangkat “biasa” hanya dapat bergerak maju. Namun, koin dengan pangkat “raja”, dapat bergerak maju maupun mundur.

6. Raja.

Raja merupakan sebutan untuk koin yang mendapat perlakuan istimewa. Koin ini adalah koin yang bertahan dan berhasil masuk ke ujung baris bidang lawan. Koin ini didefinisikan sebagai tumpukan dari dua koin biasa. Keistimewaan koin ini adalah dapat bergerak baik maju maupun mundur.

7. Permainan selesai

   Permainan selesai ditandai oleh habisnya koin lawan pada bidang permainan atau koin sudah tidak dapat bergerak kemanapun.

3.2 Analisis Kebutuhan Non Fungsional

Analisis Kebutuhan Non fungsional adalah sebuah langkah dimana seorang pembangun perangkat lunak menganalisis sumber daya yang akan menggunakan perangkat lunak yang dibangun. Analisis Kebutuhan Non Fungsional untuk penerapan algoritma minimax pada permainan checkers, yaitu :

1. Analisis Pengguna
2. Analisis Perangkat Lunak
3. Analisis Perangkat Keras

3.2.1 Analisis Pengguna

Perangkat keras dan perangkat lunak yang ada tidak akan berguna apabila tidak ada perangkat pikir yang mengoperasikannya. Perangkat pikir yang terlibat dalam sistem adalah pemain/player untuk memainkan permainan checkers.

3.2.2 Analisis Perangkat Lunak

Kebutuhan perangkat lunak yang digunakan untuk bisa mengoperasikan aplikasi yang akan dibangun adalah Windows XP SP I/II

3.2.3 Analisis Perangkat Keras

Perangkat keras yang digunakan untuk memainkan permainan ini antara lain adalah :

a. Procesor dengan kecepatan 1,8 GHz
b. RAM 512 Mb
c. Harddisk 40 Gb
d. Sound Card.
e. CD-Rom 52x
f. Monitor
g. *Keyboard* dan *Mouse*

3.3 Analisis Kebutuhan Fungsional

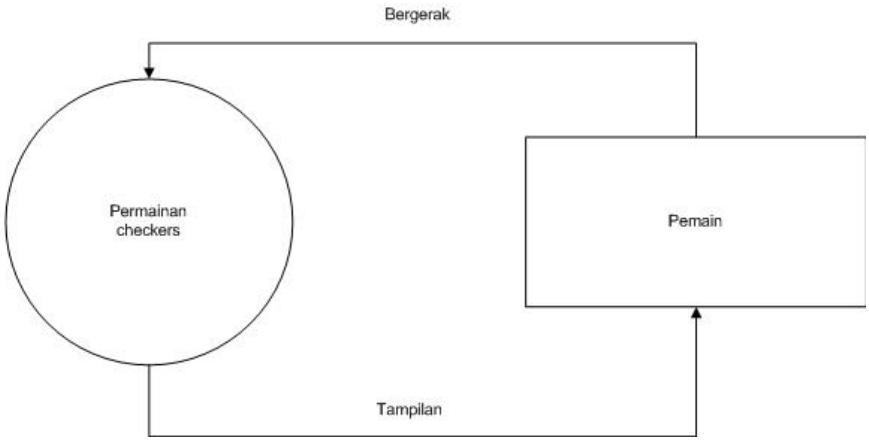
Analisis Kebutuhan Fungsional adalah asumber daynalisis untuk menentukan entitas-entitas baik entitas internal maupun entitas eksternal, data yang mengalir, serta prosedur-prosedur yang bisa dilakukan oleh masing-masing entitas.

Untuk mempermudah menggambarkan hasilnya, maka dibentuklah hasil analisis kebutuhan non fungsional dalam bentuk diagram-diagram serta alat bantu berupa:

1. Diagram konteks
2. Diagram alir data
3. Spesifikasi proses

3.3.1 Diagram Konteks

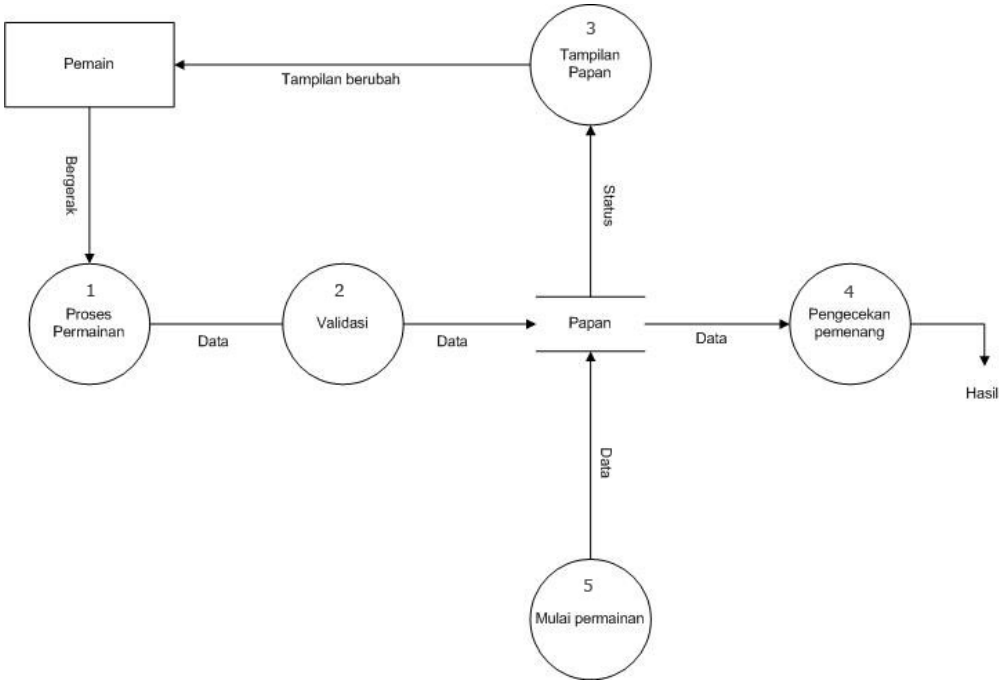
Diagram konteks merupakan salah satu alat bantu dalam melakukan analisis terstruktur. Diagram konteks ini menggambarkan suatu sistem secara garis besarnya atau keseluruhannya saja.



Gambar 3.1 Diagram Konteks permainan checkers

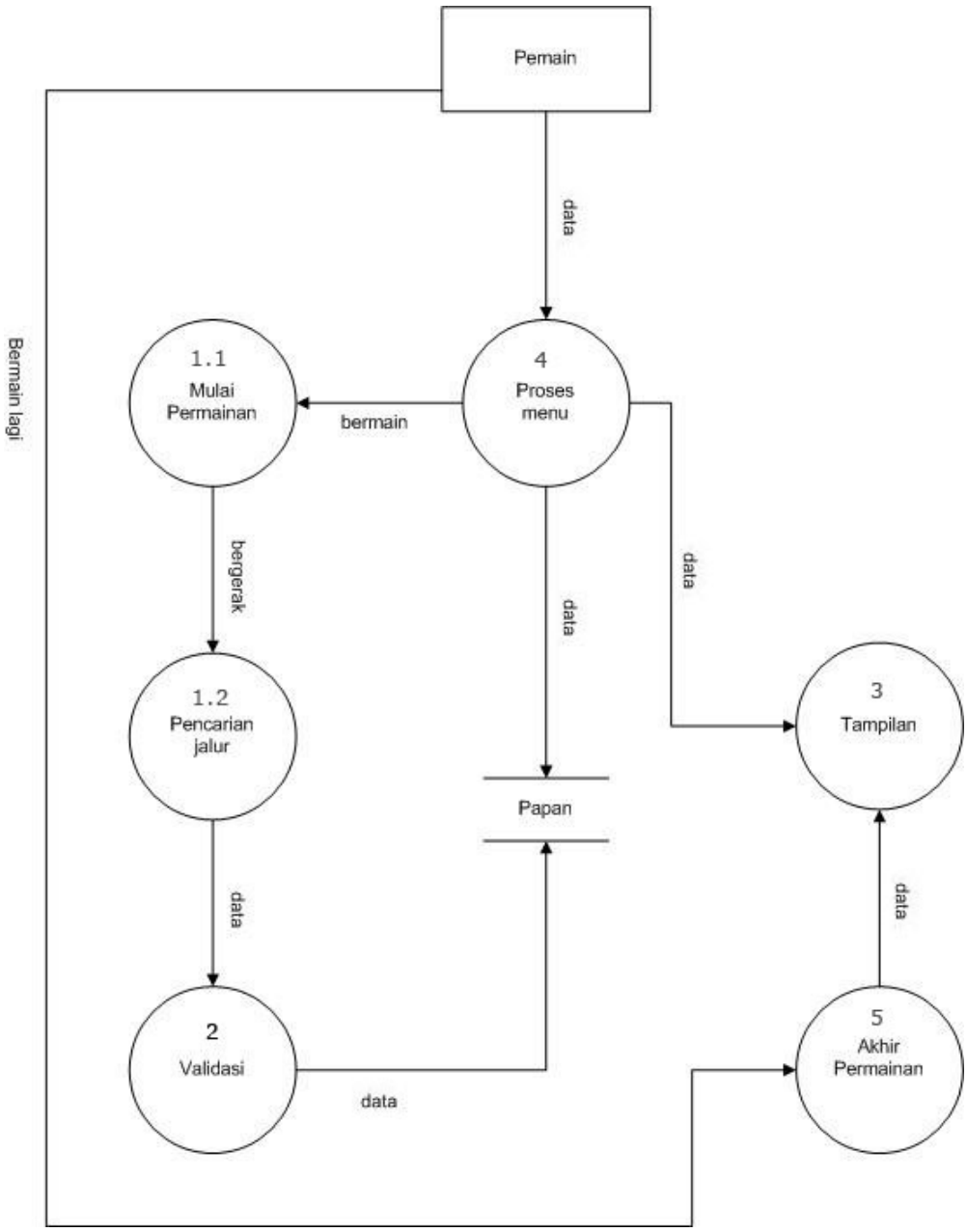
3.3.1.1 Data Flow Diagram (DFD) Level 1

Diagram alir data merupakan sebuah representasi dari suatu sistem yang menggambarkan bagian-bagian dari sistem tersebut beserta keterkaitan antara bagian-bagian yang ada. Adapun diagram alir data level satu sebagai berikut :



Gambar 3.2 DFD Level 1 permainan checkers

3.3.1.2 Data Flow Diagram (DFD) Level 2



Gambar 3.3 DFD Level 2 permainan checkers

3.3.2 Spesifikasi Proses

Spesifikasi proses digunakan untuk menggambarkan proses model aliran yang terdapat pada DFD. Spesifikasi proses dari gambaran DFD di atas akan dijelaskan sebagai berikut :

No.	**Proses**	**Keterangan**
1	No. Proses	1
	Nama Proses	*Mulai permainan*
	Source (sumber)	1. Player 2. Komputer
	Input	1. *Langkah pemain*
	Output	*pencarian*
	Destination (tujuan)	Proses 1.
	Logika Proses	Begin { pemain memulai langkah dalam permainan} End
2	No. Proses	2
	Nama Proses	*validasi*
	Source (sumber)	1. komputer
	Input	*Hasil langkah pemain*
	Output	*Hasil pencarian*
	Destination (tujuan)	Proses 3
	Logika Proses	Begin {komputer melakukan pencarian keputusan} End

Gambar 3.4 spesifikasi proses permainan

3.3.3 Analisis Proses

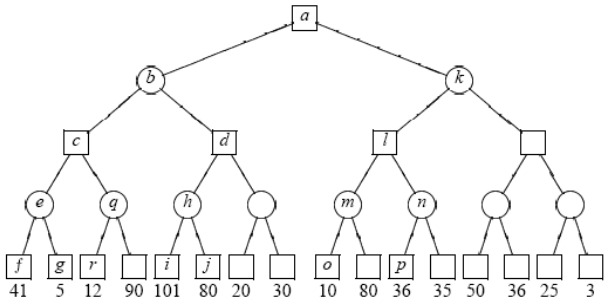
Diasumsikan bahwa permainan memilik dua pemain, disebut Maks dan Min, dan yang Maks bergerak pertama. Max ingin memaksimalkan hasil yang bisa dicapai. Min, di lain pihak, ingin meminimalkan hasil yang diperoleh dapat Max. Permainan teori-nilai yang permainan ketika adalah hasil baik dan Min Maks bermain dengan baik. Nilai ini dapat ditentukan, dalam teori, oleh semua kemungkinan memperluas secara rekursif dari continuations
permainan's mulai posisi semua jalan menuju ke akhir permainan (posisi terminal).

Aturan yang digunakan untuk menyebarkan hasil dikenal (dari pandangan Max)dari terminal kembali ke posisi awal. Apabila itu adalah Max's memindahkan, dia selalu memilih beraktivitas yang mengarah ke nilai maksimum. Ketika itu Min's memindahkan, dia akan mencoba untuk meminimalkan Max's keuntungan dengan memilih selalu berpindah tempat dengan nilai minimum.

Berasumsi bahwa mungkin hasil dari permainan adalah nilai bulat di rentang -9-9 (dimana angka positif menunjukkan hasil yang menguntungkan untuk Maks). Itu posisi awal adalah untuk mencari kedalaman 2 lapis (satu lapis merujuk dibuat untuk memindahkan satu per satu player). Setiap node di pohon mewakili posisi permainan dan ujung-ujungnya antara node mewakili bergerak. Square node menunjukkan posisi dimana Maks telah berpindah tempat, dan circled node Min adalah di mana telah beraktivitas

Algoritma Minimax yang meluaskan pohon secara rekursif dalam ke-kiri-kanan. Pencarian mulai dalam posisi A. Pertama yang bergerak terkemuka a1 adalah diperluas untuk permainan posisi B, kemudian bergerak menuju ke posisi b1 C, yang merupakan posisi terminal (dalam hal ini contoh sederhana) dan memiliki hasil +5. Algoritma yang sekarang backtracks, kembali +5 atas ke tingkat sebelumnya (node B), di mana bergerak B2 dan b3 yang diperluas dengan gaya yang sama; mereka mendapatkan nilai +9 dan +8, masing-masing. Pada B Min pemain yang memiliki pilihan yang pindah ke memilih, dan dia memilih memindahkan b1 karena mengurangi Max's keuntungan. Dengan demikian +5 (nilai pindah dari b1) adalah kembali kembali ke A. Dengan demikian, memindahkan a1 menjamin Maks nilai minimal +5. Max, namun harus terus menjelajahi tersisa dua bergerak, karena mereka dapat mengakibatkan sebuah hasil yang lebih menguntungkan. Dalam contoh ini yang tidak kasus, dan pencarian nilai yang akan ditentukan +5

Untuk memberikan gambaran bagaimana Alpha-Beta bekerja, akan diperlihatkan bagaimana traverses pohon secara rinci dan konsep.



Gambar 3.7 pohon pencarian alpha beta

Untuk lebih detail , berikut kode Alpha Beta :

function alphabeta(*n*, ,) *g*;

if *n* = leaf **then return** eval(*n*);

else if *n* = max **then**

g := ;

c := firstchild(*n*);

while *g* < **and** *c* = **do**

g := max *g*, alphabeta(*c*, ,) ;

:= max(, *g*);

c := nextbrother(*c*);

else /* *n* is a min node */

g := + ;

c := firstchild(*n*);

while *g* > **and** *c* = **do**

g := min *g*, alphabeta(*c*, ,) ;

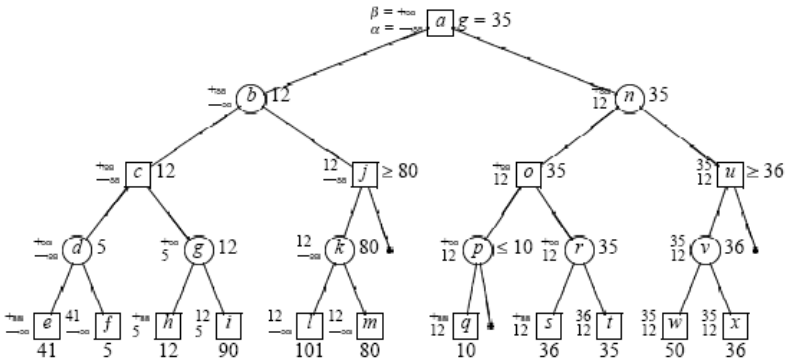
:= min(, *g*);

c := nextbrother(*c*);

return *g*;

Kedua fungsi yang paling menarik adalah bagaimana dan mengapa cutoffs dilakukan, dan melihat bagaimana Alpha-Beta konstruksi minimal pohon . Dalam contoh ini Alpha-Beta akan menemukan beberapa cutoffs, akan tetapi lebih dari yang melintasi minimal pohon, anak-anak sejak beberapa node tidak

memerintahkan terbaik pertama. (Misalnya, max di node yang paling kiri-anak tidak b tertinggi dan di menit node k kiri-paling anak tidak terendah.) Nilai-nilai untuk, dan g sebagai Alfa-Beta traverses mereka adalah ditampilkan di samping node dalam gambar 3.7. Anak-anak yang putus ditampilkan sebagai kecil titik hitam.



Gambar 3.8 pohon minimax alpha beta

Angka dalam gambar 3.8 memberikan langkah-oleh-langkah Alpha-Beta Cutoffs ditunjukkan dalam tabel, dan dijelaskan di bawah ini. Sejak alpha-beta bekerja menuju maksimum dan mencari solusi yang min pohon dari nilai sama, akan ditunjukkan dalam tabel ini juga. Kondisi dari Alpha-Beta jika node kembali nilai *g* yang terletak dalam pencarian jendela, kemudian kedua yang T + dan T memiliki telah ditransfer. Hal ini berlaku untuk sebagian besar node. Hanya node j, p u kembali dan sebuah nilai mereka di luar jendela. Nilainya ditentukan oleh satu solusi pohon. Atas batas untuk node ditandai sebagai | +, lebih rendah sebagai batas |. Max adalahsolusi pohon ditampilkan sebagai + T, min solusi sebagai pohon T.

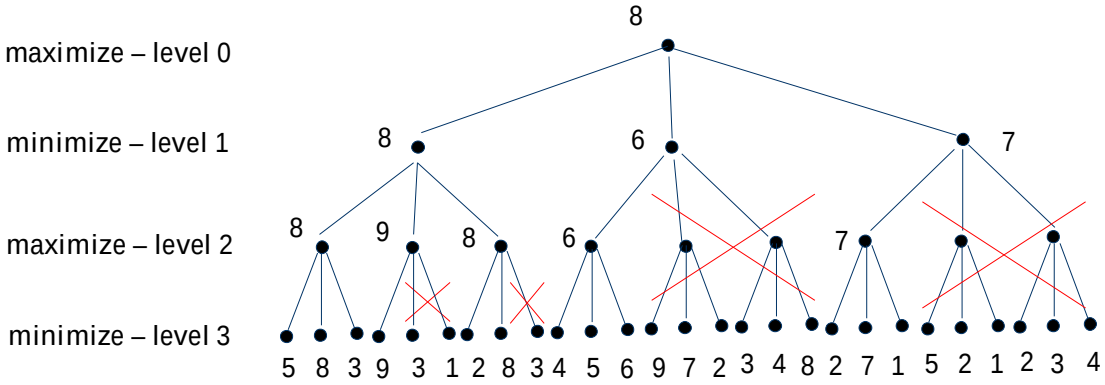
Root disebut dengan α = - α dan β = +∞. B anak pertama adalah dengan memperluas sama parameter. Hal yang sama berlaku untuk node *c, d* dan *e.* Node e adalah daun, yang panggilan yang evaluasi fungsi. Itu kembali minimax dengan nilai 41 untuk induk. Di sini, di node *d*, nilai-nilai *g* dan diperbarui sampai dengan 41. Node *d* memeriksa *g*> α (karena 41 > -∞) pencarian terus ke anak berikutnya, |. Karena telah diupdate dalam parent, node ini adalah dengan mencari sebuah jendela (-∞,41). Node *f* kembali dengan minimax nilai. Induk *d* 5 kembali, minimum 41 dan 5. parent *c* pembaruan *g* dan ke 5. Node *c* meneruskan untuk mencari node *g*, sejak 5 <+. Semakin rendah terikat digunakan untuk pencarian dan node g anak-anak dengan pencarian jendela kecil dari 5, +. Node *g* kembali minimum 12 dan 90 untuk *c*, yang memberikan hasil yang maksimal 5 dan 12 untuk *b.*Dalam node b pencarian dilanjutkan untuk memperluas *j*. B node adalah node menit.Yang g-12 merupakan nilai atas terikat, substantiated oleh maks solusi pohon yang berisi node *c, d,* |, *g dan h.* Lihat di baris 16 meja. Jendela pencarian untuk node j berkurang ke 12, menunjukkan bahwa orang tua b sudah memiliki terikat atas 12, sehingga jika di atas anak-anak lebih rendah dari *b* terikat 12 muncul, pencarian dapat dihentikan. Node *j* meluaskan sub-pohon berakar dalam anak k, yang kembali 80. Iniyang menyebabkan romulos dari saudara dalam node *j*, sejak 80 <12. Itu pencarian dari node *j*,yang nilai, menjadi max node, adalah lebih rendah terikat 80 yang hanya dapat peningkatan, tidak lagi berguna. Nilai *b* induk sudah serendah-rendahnya 12, dan sejak itu adalah min node, ia tidak akan naik. Karena *b* tidak lain untuk anak-anak itu lebihrendah nilai lebih lanjut,

hasilnya dia nilai terlalu. Nilai hasil dari b didefinisikan oleh min solusi pohon *b, c, g, h, i, j, k, l, m* dan maks solusi pohon *b, c, d, |, g, h.*

Berikut contoh pohon pencarian dari langkah diatas :

Algoritma

1. Hampiri node pertama pada leaf dengan nilai 5, naik ke parent pada level 2 masukkan nilai 5, hampiri 8, karena 8 > 5 maka ganti parent dengan 8, hampiri 3.
2. Setelah ketiga leaf pertama terhampiri, naik lebih tinggi lagi ke level 1 masukkan nilai 8.
3. Hampiri node keempat pada leaf dengan nilai 9, naik ke parent pada level 2 masukkan nilai 9. Jika kita menghampiri leaf berikutnya, kita mencari nilai yang lebih tinggi dari 9, sementara pada level 1 kita mencari yang lebih kecil dari 8, maka leaf 3 dan 1 kita potong (tidak kita hampiri).
4. Begitu seterusnya hingga kita dapatkan hasil akhir dari tree di atas adalah seperti gambar 3.10



Gambar 3.11 Tree untuk Hasil Metode Alpha Beta Prunning

3.4 Perancangan

Perancangan merupakan bagian dari metodologi pembangunan suatu perangkat lunak yang harus dilakukan setelah melalui tahapan analisis.

3.4.1 Perancangan Komponen Permainan

1. Pemain.

Permainan ini dimainkan oleh dua orang. Berperan sebagai lawan satu sama lain.di dalam perancangan ini pemain dilambangkan dengan koin. Masing-masing pemain mempunyai 12 koin sewarna yang diletakkan pada 3 baris pertama pada bidang yang terdekat dengan pemain dan diletakkan pada bidang yang berwarna gelap. Pemain yang memainkan koin berwarna gelap, memulai langkah awal terlebih dahulu.

2. Koin.

Koin yang digunakan berbentuk silinder datar seperti koin pada umumnya. Koin yang digunakan pada permainan ini berwarna terang dan gelap. Pada awalnya, warna terang direpresentasikan dengan warna putih dan warna gelap direpresentasikan dengan warna putih

Di dalam perancangan permainan koin yg dibuat diwakili dengan warna,warna tersebut adalah merah dan biru..



= Mewakili koin yang dimainkan oleh computer/koin pemain bila bermain dalam modus 2 pemain (pemain versus pemain)

 = Mewakili koin yang dimainkan oleh pemain/pemain bila bermain dalam modus 2 pemain (pemain versus pemain)

Koin bergerak secara diagonal mengikuti warna bidang yang gelap, satu di tiap langkahnya. Cara lain adalah dengan melangkahi satu buah koin lawan. Keadaan itu mungkin dapat dilakukan jika pada diagonal setelah koin lawan, merupakan bidang kosong. Jika langkah kedua itu terjadi, koin lawan yang dilangkahi mati, dan harus keluar dari bidang permainan. Koin dengan pangkat "biasa" hanya dapat bergerak maju. Namun, koin dengan pangkat "raja", dapat bergerak maju maupun mundur.

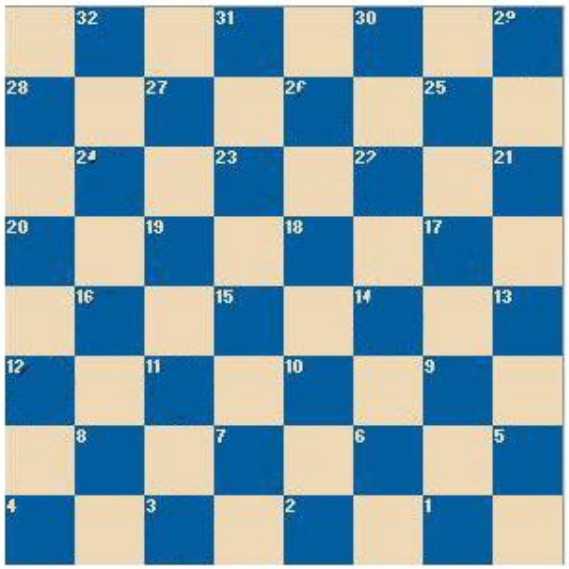
 = Koin dengan Pangkat raja

Raja merupakan sebutan untuk koin yang mendapat perlakuan istimewa. Koin ini adalah koin yang bertahan dan berhasil masuk ke ujung baris bidang lawan. Koin ini didefinisikan sebagai tumpukan dari dua koin biasa. Keistimewaan koin ini adalah dapat bergerak baik maju maupun mundur

3. Papan.

Papan yang digunakan dalam permainan ini mirip dengan papan catur. Terdiri dari 8 x 8 kotak-kotak kecil warna terang dan gelap. Permukaan yang dimainkan terdiri dari 32 kotak hanya gelap. Sebuah konsekuensi ini adalah bahwa, dari perspektif masing-masing player, kiri dan kanan tikungan mendorong strategi yang berbeda.

Dalam perancangan permainan ini papan diwakili dengan gambar kotak berwarna hitam dan putih yang disusun sehingga membentuk arena permainan



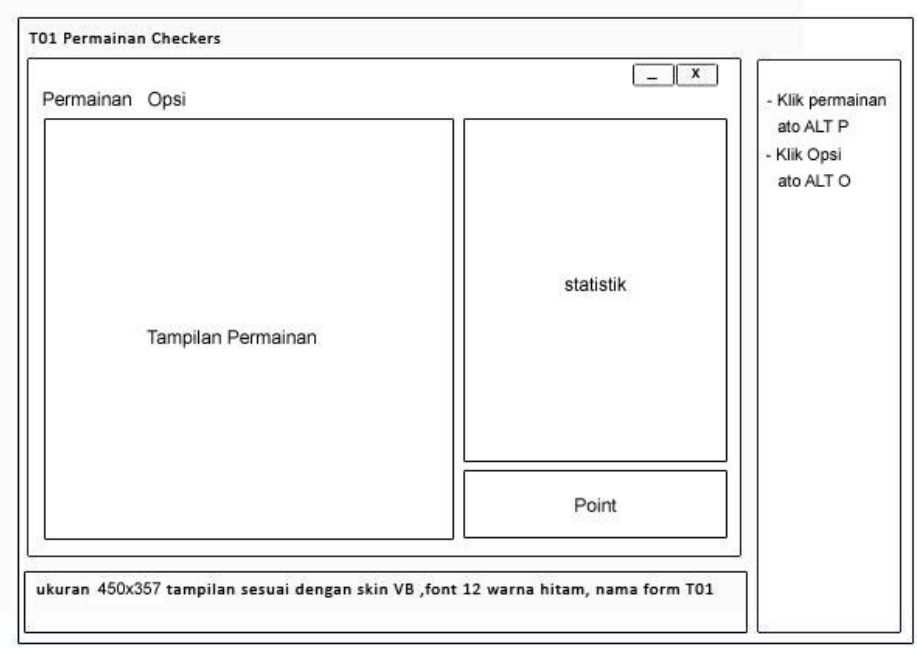
Gambar 3.11 Perancangan papan permainan

Permainan selesai ditandai oleh habisnya koin lawan pada bidang permainan atau koin sudah tidak dapat bergerak kemanapun.

1.4.2 Perancangan Antar Muka

Perancangan antar muka dilakukan untuk mempermudah dalam mengimplementasikan sistem yang akan dibangun.

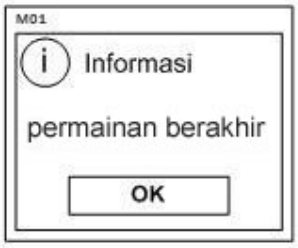
3.4.4.2 Form Utama



Gambar 3.12 Rancangan Form utama

3.4.4.3 Perancangan Antar Muka Pesan

Perancangan Antar Muka ini adalah tampilan pesan yang munccul pada aplikasi ini seperti ditunjukan pada Gambar



Gambar 3.13 Rancangan Antar Muka Kotak pesan

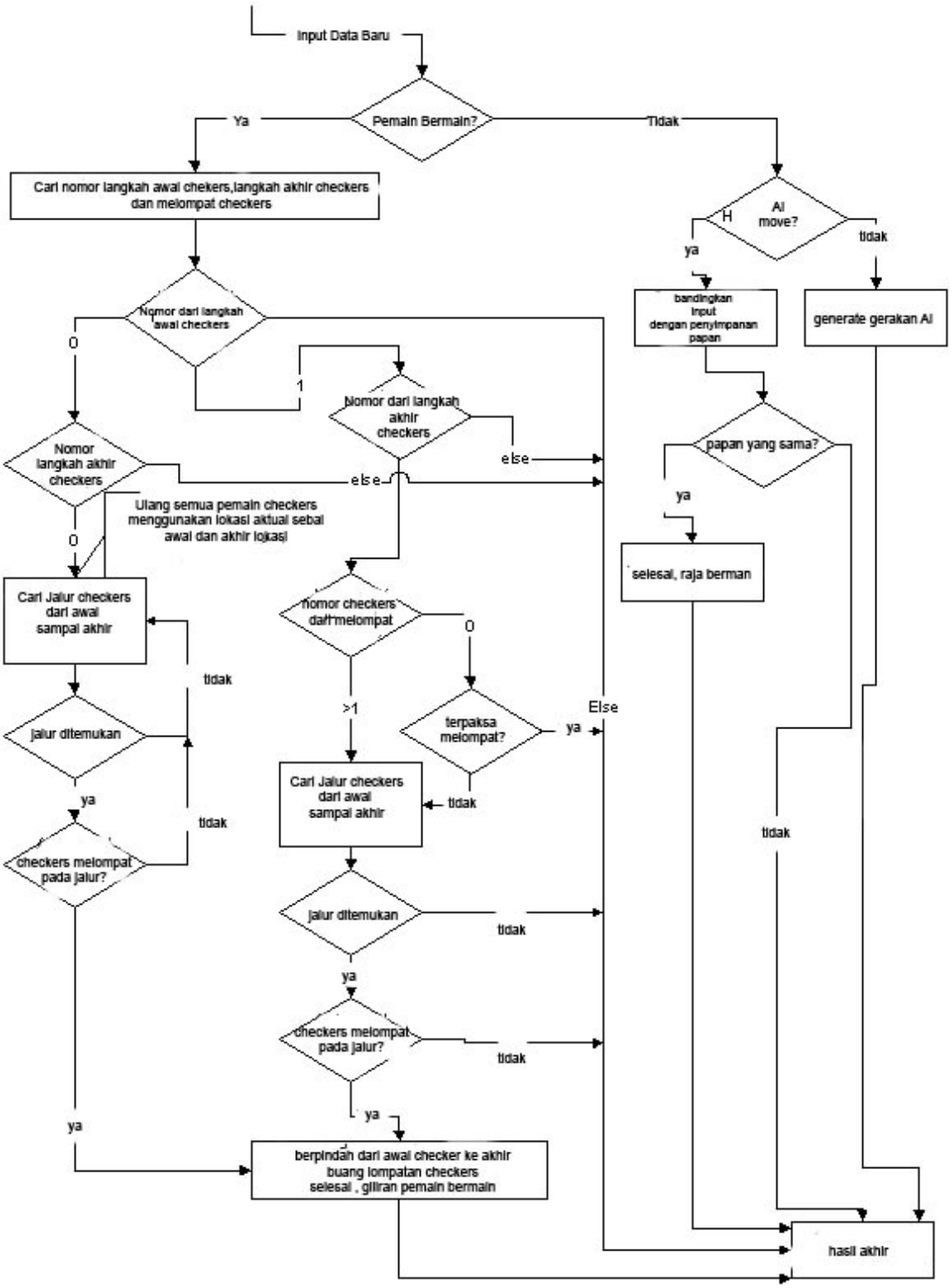
3.5 Perancangan Prosedural

Sebagai langkah terakhir dalam perancangan yaitu merancang prosedural yang akan diimplementasikan ke dalam sistem. Prosedural ini akan digunakan sebagai algoritma dasar dalam mengkodekan prosedur yang ada.

Perancangan prosedural permainan checkers adalah sebagai berikut:

1. Prosedur *permainan*

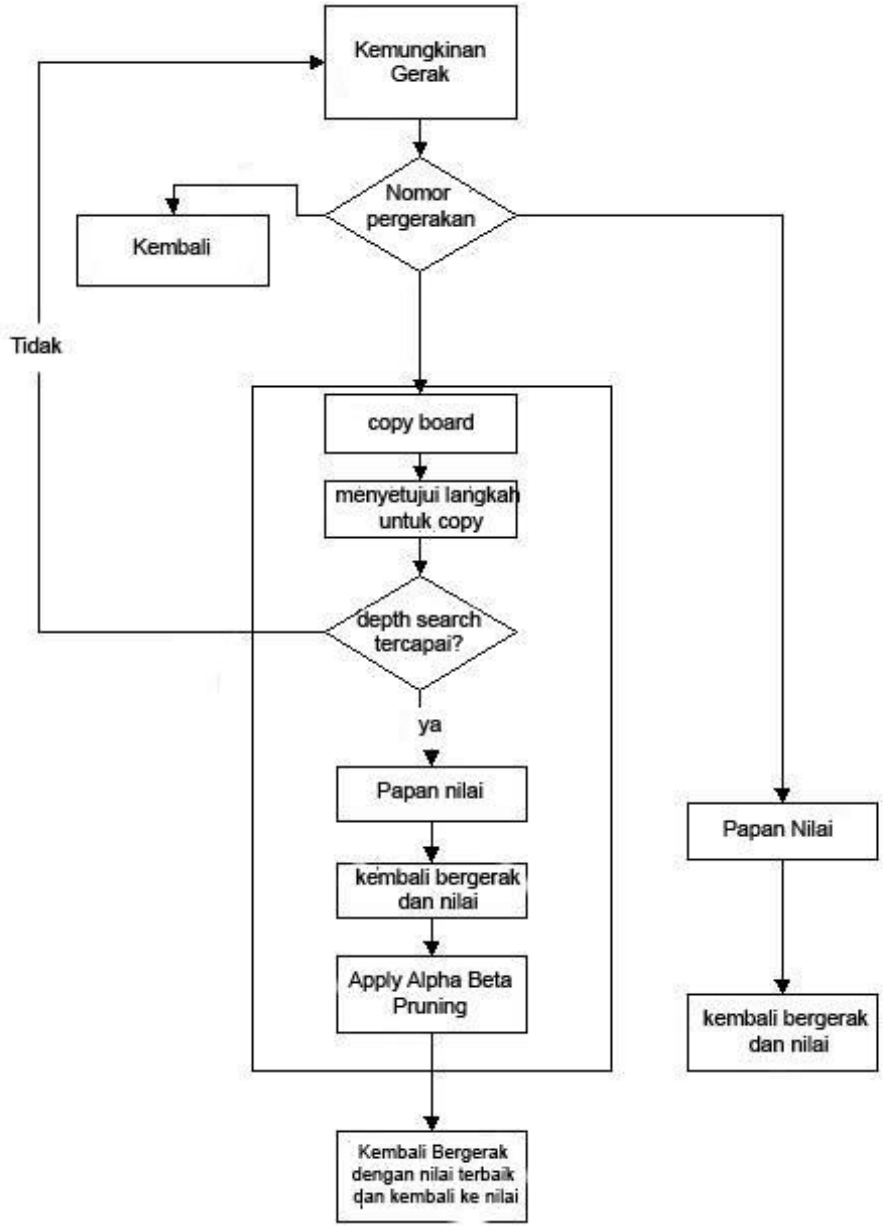
   Diagram alir pada prosedur *permainan* adalah sebagai berikut :



Gambar 3.15 Prosedur permainan checkers

2. Prosedur algoritma alpha beta

Diagram alir prosedur algoritma minimax alpha beta adalah sebagai berikut :



Gambar 3.16 Prosedur minimax alpha beta pada permainan checkers