# BAB II

# KAJIAN PUSTAKA

## 2.1. *Artificial Intelligence* (AI)

Kecerdasan buatan atau lebih di kenal sebagai *Artificial Intelligence*, memiliki beberapa definisi, antara lain :

a. Menurut Kusumadewi (2003), "Kecerdasan buatan atau *artificial intelligence* merupakan salah satu bagian ilmu komputer yang membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia".

b. Menurut Avron Barr dan Edward E. Feigenbaum (1982), *Artificial Intellegence* adalah sebagian dari komputer sains yang mempelajari (dalam arti merancang) sistem komputer yang berintelegensi, yaitu sistem yang memiliki karakteristik berpikir seperti manusia.

c. Menurut Rich dan Knight (1991) kecerdasan buatan merupakan sebuah studi tentang bagaimana membuat komputer melakukan hal-hal yang pada saat ini dapat dilakukan lebih baik oleh manusia.

Berdasarkan definisi di atas, maka kecerdasan buatan menawarkan media maupun uji teori tentang kecerdasan. Teori-teori ini nantinya dapat dinyatakan dalam bahasa pemrograman dan eksekusinya dapat dibuktikan pada komputer nyata. Layaknya manusia yang memiliki otak, komputer juga dapat memiliki perangkat lunak yang bekerja sebagai otak. Manusia dapat menyelesaikan berbagai masalah bukan hanya karena manusia memiliki otak yang mampu menalar dan menganalisa, tapi manusia juga memiliki basis data, pengetahuan, kumpulan informasi, yang semuanya itu diperoleh dari pengalaman, dan belajar.

Semuanya itu kemudian diproses secara otomatis dan sering tidak disadari oleh manusia itu sendiri. Beberapa lingkup utama dalam kecerdasan buatan adalah :

a. Sistem Pakar (*Expert System*)

Sistem Pakar adalah suatu perangkat lunak yang menyimpan pengetahuan seorang pakar, dengan demikian komputer tersebut akan memiliki keahlian layaknya seorang pakar.

b. Pengolahan Bahasa Alami (*Natural Language Processing*)

Dengan Pengolahan Bahasa Alami ini, diharapkan dikemudian hari manusia dapat berkomunikasi dengan komputer menggunakan bahasa sehari-hari.

c. Pengenalan Ucapan (*Speech Recognition*)

Dengan teknologi ini, diharapkan manusia nantinya dapat berbincang-bincang dengan komputer.

d. *Computer Vision*

Teknologi ini merupakan upaya pengenalan citra atau objek visual pada komputer. Dengan mengunakan alat sensor atau *scanner* sebagai indra, maka komputer dapat mengenali objek apa yang ditangkap oleh indranya.

e. *Game Playing*

Mengembangkan berbagai macam bentuk permainan interaktif yang cerdas. Diasumsikan dengan teknologi tersebut dapat menarik minat para penggemar *game*.

### 2.1.1. Sejarah *Artificial Intelligence* (AI)

Pada awal abad ke 17, Rene Descartes mengemukakan bahwa tubuh hewan bukanlah apa-apa melainkan hanya mesin-mesin yang rumit. Blaise Pascal menciptakan mesin penghitung digital mekanis pertama pada tahun 1642. Pada abad ke 19, Charles Babbage dan Ada Lovelace bekerja pada mesin penghitung mekanis yang dapat diprogram.

Bertrand Russell dan Alfred North Whitehead menerbitkan Principia Mathematica, yang merombak logika formal. Warren McCulloch dan Walter Pitts menerbitkan "Kalkulus Logis" Program AI pertama yang bekerja ditulis pada tahun 1951 untuk menjalankan mesin Ferranti Mark di University of

Manchester (UK), sebuah program permainan naskah yang ditulis oleh Christopher Strachey dan program permainan catur yang ditulis oleh Dietrich Prinz. John McCarthy membuat istilah "kecerdasan buatan" pada konferensi pertama yang disediakan untuk pokok persoalan ini, pada tahun 1956. Dia juga menemukan bahasa pemrograman Lisp. Alan Turing memperkenalkan "*Turing Test*" sebagai sebuah cara untuk mengoperasionalkan test perilaku cerdas.

Selama tahun 1960-an dan 1970-an, Joel Moses mendemonstrasikan kekuatan pertimbangan simbolis untuk mengintegrasikan masalah di dalam program Macsyma, program berbasis pengetahuan yang sukses pertama kali dalam bidang matematika. Marvin Minsky dan Seymour Papert menerbitkan Perceptrons, yang mendemostrasikan batas jaringan syaraf sederhana dan Alain Colmerauer mengembangkan bahasa komputer Prolog. Ted Shortliffe mendemonstrasikan kekuatan sistem berbasis aturan untuk representasi pengetahuan dan inferensi dalam diagnosa dan terapi medis yang kadangkala disebut sebagai sistem pakar pertama. Hans Moravec mengembangkan kendaraan terkendali komputer pertama untuk mengatasi jalan berintang yang kusut secara mandiri.

Pada tahun 1980-an, jaringan syaraf digunakan secara meluas dengan algoritma perambatan balik, pertama kali diterangkan oleh Paul John Werbos pada 1974. Tahun 1990-an ditandai perolehan besar dalam berbagai bidang AI dan demonstrasi berbagai macam aplikasi. Lebih khusus Deep Blue, sebuah komputer permainan catur, mengalahkan Garry Kasparov dalam sebuah pertandingan 6 game yang terkenal pada tahun 1997. DARPA menyatakan

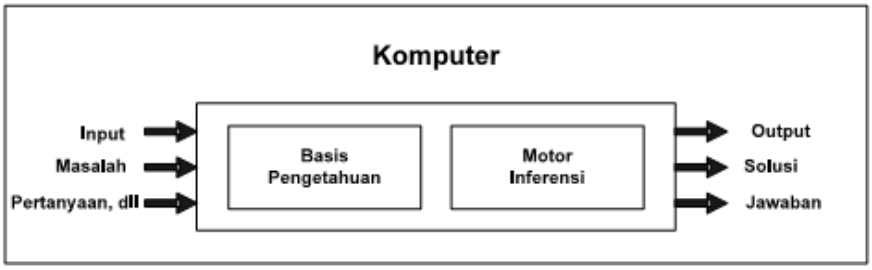
bahwa biaya yang disimpan melalui penerapan metode AI untuk unit penjadwalan dalam Perang Teluk pertama telah mengganti seluruh investasi dalam penelitian AI sejak tahun 1950 pada pemerintah AS.

Tantangan Hebat DARPA, yang dimulai pada 2004 dan berlanjut hingga hari ini, adalah sebuah pacuan untuk hadiah $2 juta dimana kendaraan dikemudikan sendiri tanpa komunikasi dengan manusia, menggunakan GPS, komputer dan susunan sensor yang canggih, melintasi beberapa ratus mil daerah gurun yang menantang.

### 2.1.2. Komponen *Artificial Intelligence* (AI)

Untuk dapat membangun aplikasi kecerdasan buatan ada 2 bagian utama yang sangat dibutuhkan, yaitu :

a. Basis Pengetahuan (*Knowledge Based*), berisi fakta-fakta, teori, pikiran, dan hubungan antara satu dan yang lainnya.
b. Motor Inferensi (*Inferensi Engine*), yaitu kemampuan menarik kesimpulan berdasarkan pengalaman.




**Gambar 2.1. Penerapan Konsep Kecerdasan Buatan di Komputer**

**(**Sumber : Kusumadewi, 2003)

### 2.1.3. Keuntungan *Artificial Intelligence* (AI)

Jika dibandingkan dengan kecerdasan alami (kecerdasan yang dimiliki oleh manusia), kecerdasan buatan memiliki beberapa keuntungan secara komersial antara lain :

a. Kecerdasan buatan lebih bersifat permanen. Kecerdasan alami akan cepat mengalami perubahan. Hal ini dimungkinkan karena sifat manusia yang pelupa. Kecerdasan buatan tidak akan berubah sepanjang sistem komputer dan program tidak mengubahnya.

b. Kecerdasan buatan lebih mudah diduplikasi dan disebarkan. Mentransfer pengetahuan manusia dari satu orang ke orang lain membutuhkan proses yang sangat lama, dan juga suatu keahlian itu tidak akan pernah dapat diduplikasi dengan lengkap. Oleh karena itu, jika pengetahuan terletak pada suatu sistem komputer, pengetahuan tersebut dapat disalin dari komputer tersebut dan dapat dipindahkan dengan mudah ke komputer lain.

## 2.2. *Game Playing*

### 2.2.1. Definisi *Game Playing*

*Game* diwakili oleh pohon pencarian di mana *node-node* menunjukan semua kemungkinan keadaan *game* dan sisi-sisi (*edges*) mewakili langkah antara kedua pemain. *Game playing* merupakan problem pencarian yang di definisikan oleh beberapa komponen berikut :

1. Keadaan Awal (*initial state*), yaitu keadaan yang mendifinisikan konfigurasi awal permainan dan mengidentifikasi pemain pertama yang bergerak.
2. Fungsi Penerus (*successor function*), yang bertugas mengidentifikasi kemungkinan-kemungkinan yang dapat dicapai dari keadaan saat ini. Fungsi ini membuat sebuah daftar pasangan gerak dan keadaan, yang masing-masing pasangan menunjukan langkah legal dan keadaan yang dihasilkan.
3. *Goal Test*, yang bertugas untuk memeriksa apakah suatu keadaan tertentu adalah keadaan tujuan atau bukan. Keadaan-keadaan di mana permainan berakhir disebut sebagai keadaan terminal.
4. *Path Cost* / *Utility* / *Payoff Function*, yang memberikan nilai numerik untuk keadaan-keadaan terminal. Dalam catur misalnya, hasilnya adalah menang, kalah, atau seri, dengan nilai 1, -1, atau 0. Beberapa *game* memiliki rentang yang lebih luas dari hasil yang mungkin.

### *2.2.2. Type Game*

Berdasatkan *type*-nya, *game* dibagi menjadi dua, yaitu *game* dengan informasi lengkap dan *game* dengan informasi tak lengkap.

#### 2.2.2.1. *Perfect Information Game*

*Game* dengan informasi lengkap adalah suatu *game* di mana pemain mengetahui semua langkah yang mungkin terjadi dari dirinya sendiri dan dari lawan

main, dan hasil akhir dari permainan mereka. Contoh *game* yang termasuk dalam type ini adalah catur, reversi (othello), dan *tic-tac-toe*.

#### 2.2.2.2. *Imperfect Information Game*

*Game* dengan *type* ini adalah *game* di mana pemain tidak tahu semua kemungkinan langkah lawan. Contoh yang termasuk *game* ini adalah permainan kartu Poker dan Bridge, karena tidak semua kartu diketahui oleh para pemain.

## 2.3. Pohon Permainan

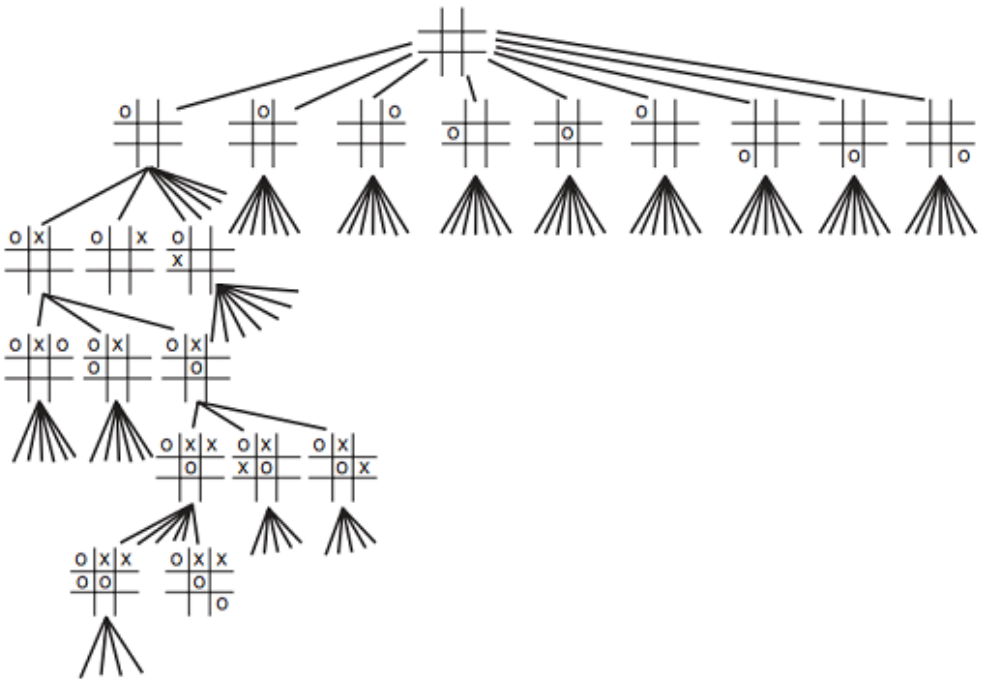
Sebuah pohon permainan dapat merepresentasikan kondisi-kondisi yang mungkin dihadapi pada permainan dimulai dari kondisi yang sedang dihadapi sekarang hingga beberapa kondisi ke depan. Sebuah pohon permainan merupakan representasi grafis dari contoh permainan. Pohon permainan menyediakan informasi akan pemain, hasil, strategi, dan pilihan langkah.

Pohon permainan dapat direpresentasikan dengan sangat baik untuk permainan yang dimainkan oleh dua pemain. Pohon permainan memiliki *root* yang merupakan representasi dari kondisi langkah terakhir atau kondisi di mana langkah belum diambil, *nodes* pada pohon yang merepresentasikan keadaan-keadaan yang mungkin diambil pada permainan, dan *arcs* yang merepresentasikan langkah.

Penggunaan pohon permainan pada permainan yang dimainkan oleh dua pemain direpresentasikan dengan cara berselingan. Untuk *edges* dari tingkat pertama ke tingkat kedua merepresentasikan langkah-langkah yang dapat diambil oleh pemain pertama, sedangkan untuk *edges* dari tingkat kedua ke tingkat ketiga

merepresentasikan langkah-langkah yang dapat diambil oleh pemain kedua, dan begitu seterusnya.

*Leaf nodes* pada pohon permainan merepresentasikan keadaan akhir pada permainan, di mana permainan tersebut dimenangkan, dikalahkan ataupun seri. Pada permainan yang sederhana, untuk mencapai *leaf nodes* mungkin dapat direpresentasikan, tetapi untuk permainan yang rumit seperti Catur atau Reversi, pencapaian *leaf nodes* sangat tidak dimungkinkan karena percabangan pada pohon permainan yang sangat besar sehingga hanya akan dibatasi dengan beberapa kedalaman saja.



**Gambar 2.2. Contoh Pohon Permainan pada *Tic-Tac-Toe***

(Sumber : Ben Coppin, 2004)

Berikut adalah penjelasan pohon permainan *tic-tac-toe* pada Gambar 2.2 :

1. Terdapat *root* yang merupakan keadaan awal di mana permainan belum dimulai dan langkah belum diambil.
2. *Edges* yang menghubungkan tingkat pertama (*root*) dengan tingkat kedua merupakan langkah pemain pertama dan begitu seterusnya. Sehingga pohon permainan tersebut merepresentasikan langkah kedua pemain secara berselingan.
3. Untuk *nodes* pada pohon tersebut merepresentasikan keadaan-keadaan yang dapat diambil oleh pemain yang akan melangkah.
4. Percabangan pertama yang dihasilkan adalah 9, kemudian untuk percabangan berikutnya adalah 8, dan begitu seterusnya hingga mencapai keadaan akhir (*leaf nodes*).

Pada aplikasi permainan Reversi yang akan dirancang, permainan tersebut tidak memiliki unsur kemungkinan atau kesempatan pada permainan dan semua pengetahuan akan keadaan permainan tersebut diketahui secara keseluruhan, artinya pemain tidak dapat menyimpan suatu informasi agar pemain lawan tidak mengetahuinya (terkecuali untuk strategi).

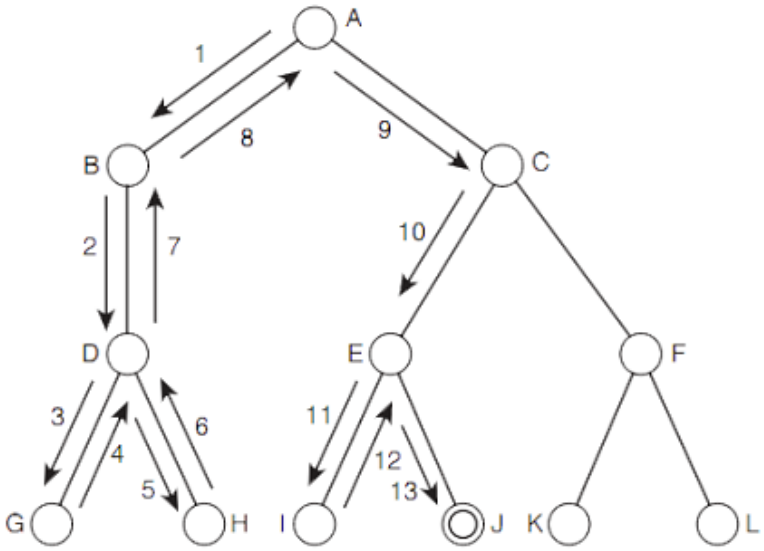
Aplikasi permainan Reversi yang akan dirancang berbasis *zero-sum*, artinya pendapatan poin untuk pemain yang satu, merupakan kehilangan poin untuk pemain yang satunya lagi. Apabila pemain yang satu menang, maka pemain yang satunya lagi kalah. Kemungkinan lainnya hanya seri.

## 2.4. Metode Pencarian (*Searching Method*)

Penulis telah mengetahui apa itu pohon permainan, namun penulis belum mengetahui bagaimana cara yang baik untuk menelusuri pohon permainan tersebut sehingga mendapatkan nilai-nilai dari pohon permainan tersebut. Di sini penulis akan menganalisa sebuah algoritma pencarian pada pohon permainan yaitu *Depth-First Search* (DFS).

### 2.4.1. *Depth-First Search* (DFS)

*Depth-First Search* (DFS) merupakan algoritma pencarian yang paling umum digunakan. DFS akan melakukan pencarian pada pohon dengan cara menelusuri suatu jalur tertentu hingga mencapai kedalaman yang paling akhir (*leaf node*) terlebih dahulu sebelum melanjutkan ke jalur selanjutnya. Jika DFS telah mencapai *leaf node* tetapi belum menemukan *goal state* maka DFS akan melakukan proses *backtrack* ke *node* teratas yang belum pernah ditelusuri. Penelusuran pada pohon permainan akan dilakukan hingga DFS menemukan *goal state* atau telah selesai menelusuri keseluruhan pohon permainan. Pada Gambar 2.3 mengilustrasikan tentang bagaimana cara kerja algoritma DFS.




**Gambar 2.3. Penelusuran Pohon Permainan dengan Metode DFS**

(Sumber : Ben Coppin, 2004)

Pada Gambar 2.3 dapat diketahui bahwa proses penelusuran DFS dimulai dari A-B-D-G-D-H-D-B-A-C-E-I-J. A merupakan *root* yang menandakan keadaan belum diambil, lalu ditelusuri hingga kedalaman yang paling dalam sebelah kiri yaitu G, lalu melakukan proses *backtrack* ke D lalu lanjut lagi ke H. Proses tersebut berhenti karena telah mencapai *goal state* yaitu *leaf node* J.

Langkah-langkah cara kerja algoritma DFS adalah sebagai berikut :

1. Masukkan *root* ke dalam struktur data tumpukan (*stack*).
2. Ambil simpul dari tumpukan teratas, dan diperiksa apakah simpul merupakan solusi.

3. Jika simpul merupakan solusi, maka pencarian selesai dan hasil dikembalikan.
4. Jika simpul bukan solusi, masukkan seluruh simpul yang bertetangga dengan simpul tersebut ke dalam tumpukan.
5. Jika tumpukan kosong dan setiap simpul sudah ditelusuri, pencarian selesai dan solusi tidak ditemukan.
6. Ulangi pencarian dari poin kedua.

Untuk pencarian pada pohon yang memiliki kedalaman yang sangat dalam ataupun hampir tak terhingga, algoritma DFS tidak bekerja dengan baik karena solusi bisa jadi tidak ditemukan, dan apabila ditemukan pasti membutuhkan waktu operasi yang sangat lama. Namun algoritma DFS, bisa lebih cepat menemukan solusi secara kebetulan apabila jalur yang ditelusuri pertama kali merupakan jalur menuju solusi pada pohon tersebut.

Kompleksitas dalam kondisi terburuk pada algoritma DFS adalah O(bm) dengan keterangan b merupakan unsur percabangan pada pohon dan m merupakan tingkat kedalaman yang ada pada pohon permainan. Sedangkan kompleksitas untuk kondisi terbaik pada algoritma DFS adalah O(1), dengan kondisi pencarian simpul pertama langsung menemukan solusi.

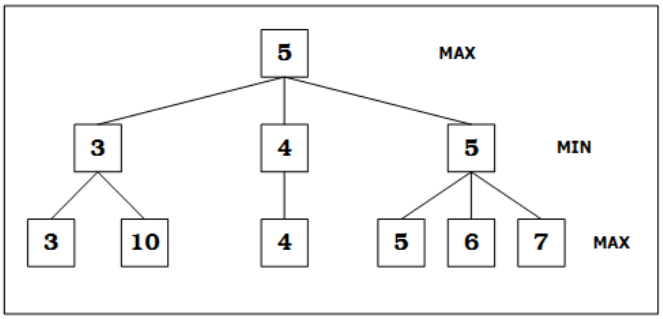
## 2.5. Algoritma Minimax

Algoritma Minimax dikembangkan oleh John von Neuman pada tahun 1928 dan sekaligus merupakan salah satu prinsip algoritma yang paling dikenal digunakan dalam *game*. Prinsip ini hanya berlaku pada *game* yang memiliki dua

pemain secara bergantian, yaitu manusia melawan komputer seperti dalam permainan catur, reversi (othello), dan *tic-tac-toe*. Algoritma Minimax juga merupakan salah satu algoritma yang berbasis *zero-sum* yang artinya keuntungan untuk pemain pertama berarti kerugian untuk pemain kedua.

Minimax bekerja sangat baik bila seluruh keadaan langkah yang mungkin terjadi dari pemain dapat diketahui seperti permainan catur, reversi (othello), atau *tic-tac-toe*. Algoritma Minimax tidak bisa digunakan pada permainan seperti poker karena komputer tidak bisa melihat semua langkah pemain lawan yang mungkin terjadi.

Untuk mengimplementasikan algoritma minimax, diperlukan sebuah metode untuk mengukur seberapa baik sebuah posisi saat itu. Metode ini disebut sebagai fungsi evaluasi. Fungsi evaluasi merupakan sebuah *heuristik*, yang kenyataannya sangat sulit untuk menentukan berapa nilai presisi pada sebuah posisi saat itu. Fungsi evaluasi yang baik akan memperhitungkan dari banyak aspek yang berbeda, seperti posisi papan, jumlah *pieces relative* terhadap lawan, kontrol papan, kontrol kunci tertentu pada papan, dan mobilitas.




**Gambar 2.4. Sebuah Pohon Pencarian untuk Sebuah Permainan Logika**
**(**Sumber : Ben Coppin, 2004)

Dalam Minimax, MAX adalah sebutan bagi pemain (komputer) yang bertujuan untuk mendapatkan nilai maksimum dan MIN adalah sebutan bagi lawan (*human*) yang bertujuan untuk mendapatkan nilai minimal. Berikut prosedur minimax :

1. Tandai masing-masing *level* dari ruang pencarian sesuai dengan langkahnya pada *level* tersebut.
2. Mulai pada *node* daun (*node* paling bawah), dengan menggunakan fungsi evaluasi, berikan nilai pada masing-masing *node*.
3. Arah menjalar ke atas jika *node* orangtua adalah MAX, pilihlah nilai terbesar yang terdapat pada *node* anak dan berikan nilai ini pada MAX (*node* orangtua).
4. Arah menjalar ke atas jika *node* orangtua adalah MIN, pilihlah nilai terkecil yang terdapat pada *node* anak dan berikan nilai ini pada MIN (*node* orangtua).

Berikut ini adalah *pseudocode* algoritma minimax :

```
MinMax (GamePosition game) {
  return MaxMove (game);
}
MaxMove (GamePosition game) {
  if (GameEnded(game)) {
    return EvalGameState(game);
  }
  else {
    best_move <- {};
    moves <- GenerateMoves(game);
    ForEach moves {
       move <- MinMove(ApplyMove(game));
       if (Value(move) > Value(best_move)) {
          best_move <- move;
       }
    }
    return best_move;
```

```
  }
}
MinMove (GamePosition game) {
  best_move <- {};
  moves <- GenerateMoves(game);
  ForEach moves {
     move <- MaxMove(ApplyMove(game));
     if (Value(move) > Value(best_move)) {
        best_move <- move;
     }
  }
  return best_move;
}
```

### 2.5.1. Karakteristik Algoritma Minimax

Terdapat karakteristik dari algoritma minimax menurut Edy Mulyanto (2010 : 442) yaitu sebagai berikut :

a. *Completeness*

Minimax dikatakan lengkap jika pohon pencariannya berhingga. Sebagai contoh, permainan catur memiliki pohon pencarian yang sangat besar, tetapi berhingga. Jadi, minimax lengkap dalam kasus permainan catur.

b. Optimalitas

Algoritma minimax akan optimal jika melawan pemain yang langkahnya juga optimal. Jika lawan tidak optimal maka minimax akan lebih optimal lagi.

c. Kompleksitas Waktu

Algoritma minimax melakukan eksplorasi pencarian lengkap kedalaman pertama, kompleksitas waktu adalah $O(b^m)$, di mana b adalah faktor percabangan dan m adalah kedalaman pohon. Untuk permainan catur, b kira-kira sama dengan 35 dan m kira-kira sama dengan 100. Oleh karena itu, kompleksitas waktunya kira-

kira sekitar $O(35^{100})$, apalagi untuk permainan-permainan yang lebih besar dalam permainan catur. Oleh karena itu, menerapkan algoritma minimax pada permainan catur dan permainan yang lebih besar dari permainan catur tentu saja tidak efisien. Solusi yang tepat harus dicari agar kompleksitas waktu tidak bersifat eksponensial. Salah satu pendekatan standar yang bisa diterapkan adalah pencarian batas kedalaman, di mana pencarian dibatasi pada kedalaman beberapa node dari node root (node saat ini). Node pada kedalaman tertentu itu diasumsikan sebagai node daun dan nilai fungsi evaluasinya diperkirakan.

d. Kompleksitas Ruang

Kompleksitas ruang dari algoritma minimax adalah $O(b^m)$.

## 2.6. Windows Phone

Berdasarkan penelusuran pada wikipedia (http://id.wikipedia.org/ wiki/Windows_Phone) yang diakses pada tanggal 3 Januari 2013, ensiklopedia bahasa Indonesia dan berdasarkan situs resmi dari Windows Phone, Windows Phone merupakan salah satu sistem operasi dari keluarga Microsoft yang berjalan pada perangkat *mobile* yang sekaligus merupakan pengganti dari sistem operasi Windows Mobile. Windows Phone ditujukan pada pasar konsumen perusahaan seperti Nokia, dll.

### 2.6.1. Sejarah Windows Phone

Pada tanggal 11 Februari 2011 di London, CEO Microsoft Steve Ballmer dan CEO Nokia Stephen Elop mengumumkan kerja sama di antara kedua perusahaan, di mana Windows Phone akan ditetapkan menjadi sistem operasi utama

pada telepon genggam buatan Nokia di masa mendatang yang mana sebelumnya Nokia menggunakan sistem operasi Symbian.

Menurut Puja Pramudya, (2011:6) Windows Phone (sebelumnya dikenal sebagai Windows Phone 7) adalah sistem operasi genggam yang dikembangkan oleh Microsoft, dan merupakan pengganti dari Windows Mobile. Microsoft memperkenalkan Windows Phone pada tgl 15 Februari 2010 di pameran MWC, Barcelona. Sistem operasi ini dirilis di Amerika Serikat pada tgl 8 november 2010. Windows Phone mendukung sampai 25 bahasa. Sampai sekarang ini, Marketplace yang merupakan tempat untuk membeli aplikasi, telah dapat diakses di 35 negara.

### 2.6.2. Perkembangan Windows Phone

Dengan Windows Phone, Microsoft menciptakan *user interface* yang baru menggunakan *design language* yang di beri nama Modern Style UI. Selain itu, perangkat lunak ini terintegrasi dengan aplikasi pihak ketiga dan berbagai layanan Microsoft serta menerapkan persyaratan minimal untuk perangkat keras yang memakai sistem operasi ini.

Microsoft mengumumkan beberapa persyaratan minimal telepon genggam agar bisa menggunakan Windows Phone yaitu dapat dilihat pada tabel berikut :

**Tabel 2.1. Persyaratan Minimal Windows Phone**

| **Persyaratan Minimal Windows Phone** |
|---|
| *Capacitive*, Layar 4-point *multi-touch* WVGA dengan resolusi (480x800) |
| ARM v7 "Cortex/Scorpion" – Snapdragon QSD8X50, MSM7X30, dan MSM8X55 |
| DirectX9 *rendering-capable* GPU |
| 256 MB of RAM dengan sedikitnya 8 GB Flash memory |
| Accelerometer dengan kompas, *ambient light* sensor, *proximity* sensor, Assisted GPS, dan Gyroscope |
| 5-megapixel kamera dengan flash |
| FM radio tuner |
| Six (6) *dedicated hardware buttons – back*, *start*, *search*, 2-*stage camera*, *power/sleep* dan Volume *Up* - *Down*. |

Sumber : http://id.wikipedia.org/wiki/Windows_Phone

Pada tanggal 29 Oktober 2012 kemarin Microsoft merilis versi terbaru yaitu Windows Phone 8.0. Perbedaan yang mencolok dari versi sebelumnya yaitu pada Windows Phone 8.0 Microsoft menggunakan kernel Windows NT dari versi sebelumnya yang menggunakan kernel Windows CE. Kernel Windows NT ini yaitu kernel yang diterapkan juga pada sistem operasi Windows 8 pada PC dan Tablet. Jadi untuk pengembangan aplikasi pada Windows Phone 8.0, maka akan kompatibel juga pada Windows 8 karena memiliki kernel yang sama.

### 2.6.3. Versi Windows Phone

Sistem operasi Windows Phone terdapat beberapa versi, diantaranya :

1. Windows Phone 7.5 (Mango)

Diumumkan pada Februari 2011 di Mobile World Congress. Steve Ballmer mengumumkan *update* besar untuk Windows Phone 7 karena menjelang akhir tahun, dan meluncurkan fitur termasuk versi *mobile* Internet Explorer 9 yang mendukung standar web yang sama dan kemampuan grafis dengan versi desktop, Twitter integrasi untuk People Hub, multi-tasking dari aplikasi pihak ketiga dengan menunda tugas aktif saat beralih ke tugas aktif dalam pandangan, dan Windows Live SkyDrive.

2. Windows Phone 7.6 (Tango)

Tango akan minor *update*, mirip dengan *update* "NoDo". Ini membantu Telepon Windows untuk berjalan di perangkat-murah dengan 256 MB RAM. Ini juga mendukung Skype dan integrasi Google+ melalui hub teman, pesan, dan *chatting*. Selain itu, Tango mengubah ukuran kontrol media untuk tampilan lebih nyaman, mendapatkan dukungan untuk 120 bahasa (menambahkan dukungan untuk 85 bahasa tambahan), membuat perangkat lebih mudah diakses secara internasional, tambahkan C++ dukungan aplikasi, dan memungkinkan pengelompokan ubin layar awal ke dalam folder.

3. Windows Phone 8 (Apollo)

Apollo adalah nama kode untuk generasi berikutnya dari Windows Phone, seperti ditegaskan dalam sebuah seminar MSDN pada Agustus 2011. Engadget percaya bahwa Apollo sebenarnya mengacu pada Windows Phone 8. Hal diyakini *update* akan menambahkan teknologi NFC, dual-core dukungan chipset, lebih tinggi resolusi layar ke panggung, dan akhirnya mengarah pada konvergensi dari sistem Microsoft operasi untuk PC, ponsel, tablet, konsol video dan permainan melihat sebagai Xbox 360 juga menggunakan antarmuka Metro. Hal ini berspekulasi bahwa Microsoft memindahkan kernel Windows Phone dari Windows CE ke dalam Windows inti yang terisolasi secara informal disebut bahwa MinWin adalah dasar dari Windows 7 dan Windows 8, pengaturan ulang API untuk Windows dan Windows Phone.