Combinatorial Game Theory, Game Tree, dan Intelegensia Buatan

Ripandy Adha - 13507115
Jurusan Teknik Informatika ITB, Bandung 40116, email: if17115@students.if.itb.ac.id

Abstract *– Makalah ini membahas tentang kompleksitas suatu permainan dan salah satu cara pengukurannya dengan Combinatorial Game Theory. Makalah ini akan melihat sebuah permainan sebagai objek nyata dari matematika yang menggunakan pohon berarah pada perhitungan langkah yang mungkin dilakukan oleh pemain, serta dalam pembuatan game itu sendiri. Banyaknya kemungkinan yang terjadi untuk mencapai suatu akhir permainan dikenali sebagai kompleksitas game. Kompleksitas game pada makalah ini akan diukur dengan menggunakan Game Tree atau pohon permainan. Game Tree memiliki peranan yang cukup penting dalam pembuatan intelegensia buatan.*

Kata Kunci*: Game Tree, Kompleksitas Game, Kombinatorial, Intelegensia Buatan.*

1. PENDAHULUAN

Di dunia modern ini, perkembangan teknologi sudah sangat berpengaruh dalam setiap aspek kehidupan manusia. Salah satu aspek penting dalam kehidupan manusia adalah hiburan. Salah satu bentuk dari hiburan dapat berupa suatu permainan. Dalam perkembangan teknologi yang semakin pesat, dunia permainan atau *game* terus berkembang beserta teori-teori yang muncul untuk mengembangkan pemainan sejalan dengan berkembangnya teknologi.

Salah satu cabang matematika yang digunakan dalam pembuatan atau teori game adalah kombinatorial. Kombinatorial adalah cabang matematika yang mempelajari objek-objek. Solusi yang ingin kita peroleh dengan kombinatorial adalah jumlah cara pengaturan objek-objek tertentu di dalam himpunannya.

Salah satu bagian dari teori kombinatorial adalah *Combinatorial Game Theory*. Teori ini mempelajari tentang permainan dengan 2 pemain yang memiliki posisi dari tiap langkah yang diambil tiap pemain untuk mencapai kondisi kemenangan yang telah didefinisikan pada game tersebut.

Dalam pembuatan suatu permainan digunakan banyak konsep graf berarah. Salah satu bentuk graf yang umum adalah pohon. Dalam makalah ini pohon yang digunakan adalah *Game Tree*, yang merupakan salah satu bagian dari. *Combinatorial Game Theory.*

2. TEORI GAME

Game pada dasarnya dikenali sebagai suatu permainan yang digunakan untuk tujuan hiburan. Akan tetapi, dari sisi yang lain, *game* merupakan penerapan dari ilmu matematika dan merupakan objek matematika dan secara lebih luas digunakan untuk keperluan ilmu sosial seperti ekonomi, biologi, teknik, politik, hubungan internasional, filosopi, dan ilmu komputer (khususnya intelegensia buatan). *Game Theory* bertujuan untuk menangkap perilaku dalam situasi strategis, dimana keputusan seseorang sangat tergantung kepada keputusan orang lain.

Suatu *game* terdiri dari beberapa orang pemain, langkah-langkah atau strategi yang dapat dilakukan oleh tiap pemain dan spesifikasi dari setiap langkah atau urutan permainan.

Representasi dari suatu permainan disesuaikan dengan spesifikasi dari permainan tersebut serta jenis permainannya. Beberapa bentuk representasi *game* diantaranya adalah :

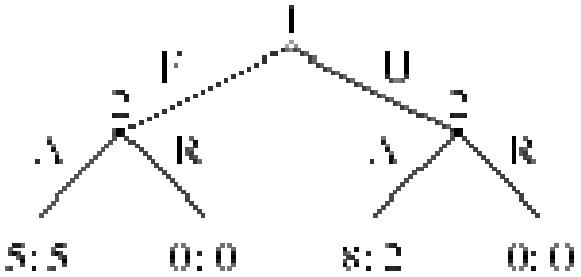
Bentuk Normal, atau bentuk strategi umumnya direpresentasikan sebagai matriks yang menampilkan pemain, strategi, serta dampak (perolehan/ pengurangan poin) pada masing-masing pemain.

	Player 2 chooses *Left*	Player 2 chooses *Right*
Player 1 chooses *Up*	**4**, **3**	**–1**, **–1**
Player 1 chooses *Down*	**0**, **0**	**3**, **4**

Gambar 2.1 *A Normal Form Game*

Bentuk Ekstensif, umumnya digunakan untuk formalisasi permainan yang memiliki urutan-urutan dan langkah-langkah penting. Bentuk representasi ini biasanya digambarkan dengan media pohon yang menyimpan data kejadian pada *game*. Cabang dari pohon menunjukkan langkah yang diambil pemain, sementara simpul menunjukkan pemain yang

mengambil langkah pada saat itu. Perolehan point ditunjukkan pada daun pohon.



Gambar 2.2 *An Extensive Game Form*

Sebagian besar persoalan dalam dunia *game* dapat direpresentasikan sebagai suatu sistem kombinatorial, yang kemudian membentuk suatu pohon. Teori yang lebih mendalam, berkaitan dengan kombinatorial dalam teori game adalah *Combinatorial Game Theory*.

2.1 *Combinatorial Game Theory*

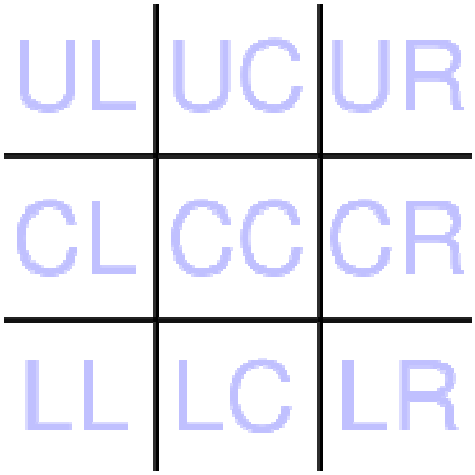
Combinatorial Game Theory adalah teori matematika yang mempelajari permainan dengan 2 orang pemain yang memiliki posisi dari tiap langkah yang diambil tiap pemain dengan langkah yang sudah didefinisikan untuk mencapai kondisi kemenangan yang telah didefinisikan pada suatu game tersebut. *Combinatorial Game Theory* tidak mempelajari tentang permainan berpeluang seperti poker. Teori ini hanya mempelajari permainan yang posisi dan langkah-langkahnya diketahui oleh kedua pemain, begitu juga dengan langkah-langkah permainan yang boleh dilakukan oleh kedua pemain.

Mengaplikasikan *Combinatorial Game Theory* dalam setiap posisi bertujuan untuk mengetahui langkah-langkah optimum untuk kedua pemain hingga permainan berakhir, dan dengan melakukan hal tersebut akan menunjukkan langkah optimum di setiap posisi. Akan tetapi, untuk mengaplikasikannya akan sangat sulit, kecuali jika permainan yang dilakukan sangat sederhana.

Combinatorial Game yang mengaplikasikan *Combinatorial Game Theory* memiliki syarat sebagai berikut :

1. Permainan memiliki tepat 2 pemain.
2. Pada umumnya pemain yang melakukan langkah terakhir menang (beberapa permainan yang disebut misere form, dimana pemain yang melakukan langkah terakhir kalah).
3. Pemain bermain secara bergantian
4. Permainan memiliki akhir, tidak berlangsung terus menerus.
5. Dalam permainan tidak ada seri, hanya ada menang dan kalah.
6. Tidak ada informasi yang disembunyikan dari pemain (misalnya seperti pada poker).
7. Permainan tidak berdasarkan pada keberuntungan.

2.1.1 Gambaran Singkat mengenai *Combinatorial Game Theory*



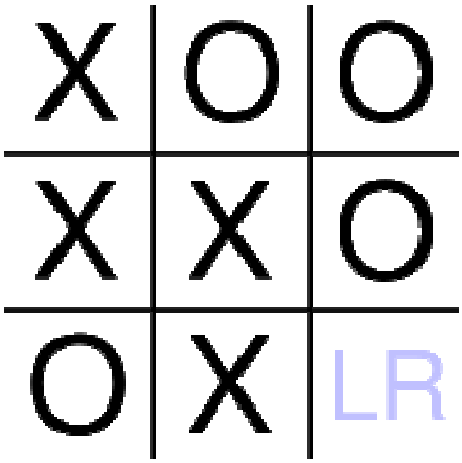
Gambar 2.3 Papan permainan *tic-tac-toe*

Untuk memberikan gambaran singkat tentang teori ini, kita gunakan sebuah permainan sederhana yang disebut Tic-Tac-Toe. Masing-masing pemain kita sebut Left dan Right. L merupakan langkah yang bisa diambil oleh pemain Left, dan R merupakan langkah yang dapat digunakan pemain Right. Jika kita melabeli masing-masing dari 9 kotak di atas dengan UL untuk Upper Left(kiri atas), CC untuk Cener Center (tengah tengah), LR untuk Lower Right (kanan bawah), dan seterusnya, dan memungkinkan untuk menempatkan X atau O pada setiap kotak, awal dari permainan Tic-Tac-Toe dapat direpresentasikan sebagai

$$XUL = \{XUL\_XUC, XUL\_XUR, XUL\_XCL, \ldots | XUL\_OUC, \ldots\}$$

Seiring dengan berjalannya permainan, kemudian permainan mungkin akan sampai pada keadaan seperti berikut :

XUL_OUR_XCC_OCR_XLC_OLL_XCL_OUC = {{ | } | { | }}



Gambar 2.4 Keadaan aneh tapi valid dalam *tic-tac-toe*

Keadaan ini menggambarkan keadaan yang hanya menyisakan satu langkah untuk kedua pemain, yaitu pada kotak sebelah kanan bawah atau kotak LR, dan jika pemain mengisi kotak tersebut, maka pemain tersebut akan memenangkan permainan. Lambang {|} pada setiap urutan dari langkah pemain disebut *zero game*, yang dapat dinyatakan dengan lambang 0. Dalam *zero game*, tidak satupun dari pemain memiliki langkah yang valid, sehingga siapapun yang mengambil langkah saat *zero game* muncul akan kalah secara otomatis.

Selain itu, permainan dengan keadaan yang cukup kompleks dengan label "XUL_OUR_XCC_OCR_ XLC_OLL_XCL_OUC" seperti diatas juga memiliki notasi yang lebih sederhana, dan disebut sebagai *star game*, yang dapat dilambangkan dengan *. Dalam *star game*, satu-satunya langkah yang valid mengarah pada *zero game*, yang artinya siapapun yang mendapat giliran berikutnya dipastikan memenangkan permainan.

Tipe permainan lain yang tidak terdapat dalam *tic-tac-toe*, adalah *loopy game*, yang mengarahkan permainan kepada awal permainan ketika salah satu pemain mengambil langkah yang valid. Permainan yang tidak memiliki langkah seperti itu disebut *nonloopy.*

Combinatorial Game Theory yang menerapkan teori kombinatorial, menerapkan langkah-langkahnya dengan representasi graf berarah, yaitu pohon, yang menunjukkan setiap kemungkinan posisi dan langkah yang dapat diambil setiap pemain, dimulai dari akar yang berupa posisi awal permainan, hingga setiap kondisi akhir permainan. Pohon tersebut kemudian dikenal dengan *Game Tree*.

2.2. *Game Tree*

Game Tree dalam *Combinatorial Game Theory* adalah suatu pohon dengan simpul menunjukkan posisi dalam permainan dan sisi menunjukkan langkah-langkah yang diambil. Suatu *Game Tree* yang lengkap untuk suatu permainan adalah sebuah pohon yang memiliki akar berupa posisi atau keadaan awal, dan memiliki setiap kemungkinan langkah untuk setiap posisi.

Jumlah simpul daun dari suatu *Game Tree* yang lengkap merepresentasikan jumlah dari setiap kemungkinan langkah-langkah dalam permainan yang dapat dimainkan. Sebagai contoh, pada permainan *tic-tac-toe* memiliki 26.830 simpul daun, yang juga berarti jumlah kemungkinan dari akhir permainan.

Game tree memiliki peranan penting dalam pembuatan intelegensia buatan, karena salah satu cara yang paling baik untuk menentukan langkah yang tepat dalam suatu permainan adalah dengan menelusuri pohon permainan dengan algoritma *minimax* atau variasinya. *Game Tree* pada permainan *tic-tac-toe* dapat ditelusuri dengan mudah, sedangkan *game tree* yang lengkap pada permainan catur akan terlalu besar dan luas untuk ditelusuri. Untuk hal ini, sebuah program dalam permainan catur menelusuri hanya sebagian dari *game tree*, yaitu menelusuri sebanyak mungkin langkah dari posisi sekarang hingga menemukan langkah yang sesuai dalam waktu yang tersedia.

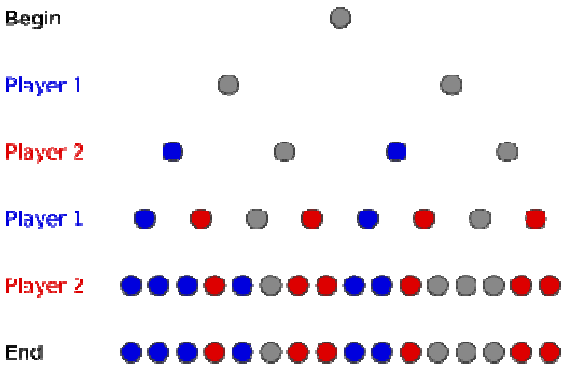
Dengan sebuah *Game Tree* yang lengkap, sangatlah mungkin untuk menemukan penyelesaian untuk suatu permainan, yaitu dengan mencari langkah-langkah yang mengarahkan salah satu pemain kepada akhir pemainan dengan akhir yang telah diketahui, baik menang maupun seri. Untuk melakukannya dapat digunakan algoritma sebagai berikut :

1. Tentukan warna untuk node yang dimenangkan pemain A dengan warna 1, pemain B dengan warna 2, dan seri dengan warna 3. Letakkan pada level akhir, jadi pada awalnya semua node final akan berwarna 3.

2. Lalu lihat pada node di level atasnya, jika ada warna pemain A warnai dengan warna 1, jika ada node pemain B warnai dengan warna 2, jika tidak ada keduanya warnai dengan warna 3.

3. Lanjutkan algoritma di atas secara rekursif hingga akhir permainan.

Umumnya, untuk menemukan penyelesaian suatu permainan dapat digunakan hanya sebagian saja dari *game tree* permainan tersebut, karena sebagian besar permainan tidak perlu memperhatikan suatu langkah, jika ada langkah yang lebih baik untuk kedua pemain. Bagian pohon yang digunakan untuk penyelesaian suatu permainan ini disebut dengan *decision tree*.



Gambar 2.2.1 Contoh pohon yang telah sepenuhnya berwarna

3. INTELEGENSIA BUATAN

Di dunia dengan teknologi yang semakin maju ini, *game* banyak dikenali sebagai permainan yang terdapat pada komputer. *Game* pada komputer, tentunya memiliki bagian tidak terpisahkan yaitu Intelegensia Buatan. Setiap permainan dalam komputer memiliki program untuk menjalankan permainan tersebut, dan mempunyai "otak" untuk berpikirnya. Otak ini dibuat oleh manusia agar komputer dapat berpikir seperti manusia dalam menjalankan suatu permainan. Otak inilah yang disebut dengan *Artificial Intelligence (AI)* atau Intelegensia Buatan.

Contoh sederhana permainan pada komputer yang menggunakan intelegensia buatan adalah permainan catur. Ketika hanya ada satu orang pemain, dan ingin memainkan catur, maka lawan bermainnya adalah otak buatan manusia, yaitu komputer dengan program intelegensia buatan.

Pembuatan intelegensia buatan pada suatu permainan didasari dengan menggunakan *game tree* lengkap dari permainan tersebut yang berisi tentang informasi mengenai setiap kemungkinan langkah yang dapat diambil, beserta prediksi langkah yang akan diambil oleh pemain. Program akan terus mengeksekusi setiap langkah yang merupakan langkah terbaik sebisa mungkin. Inti dari program intelegensia buatan ini adalah pencarian. Pembuatan intelegensia buatan juga perlu memperhatikan juga kompleksitas *game*.

3.1 Kompleksitas *Game*

Suatu permainan memiliki kompleksitas *game*. Pada *Combinatorial Game Theory*, kompleksitas *game* dapat dihitung dengan menggunakan *game tree* yang mendasari permainan tersebut. Beberapa cara mengukur kompleksitas *game* antara lain dengan menggunakan ukuran *game tree*, kompleksitas keputusan, serta kompleksitas *game tree*.

Ukuran *Game Tree*, adalah jumlah keseluruhan kemungkinan permainan dapat dimainkan. Pada *game tree* adalah jumlah daun dari pohon dengan akar pada posisi awal dari permainan. Akan tetapi, untuk permainan dengan jumlah langkah yang tidak terbatas, *game tree* dari permainan tersebut juga tidak terbatas.

Pengukuran juga dapat dilakukan dengan menggunakan *decision tree*, yaitu hanya memperhitungkan sebagian saja dari *game tree* suatu permainan, dimana masing-masing posisi langkah telah ditandai dengan "kemenangan untuk pemain A", "kemenangan untuk pemain B", atau "seri", jika posisi-posisi tersebut dapat dibuktikan memiliki nilai tersebut.

Kompleksitas Keputusan (*Decision Complexity)* dalam suatu permainan adalah jumlah daun dari *decision tree* terkecil yang menunjukkan nilai dari posisi awal permainan. Pada pohon seperti ini menyediakan semua kemungkinan untuk setiap keputusan bagi pemain kedua, tetapi hanya menyediakan satu kemungkinan untuk setiap keputusan bagi pemain pertama (pemain yang memulai permainan).

Kompleksitas *game tree* dalam suatu permainan adalah jumlah daun di dalam sebuah pohon penuh terkecil dari sebuah *decision tree*, yang menunjukkan nilai dari awal permainan. Pohon penuh ini juga termasuk setiap simpul pada setiap tingkat. Nilai ini adalah perkiraan jumlah posisi yang harus di evaluasi dalam pencarian *minimax* untuk menentukan nilai dari posisi awal.

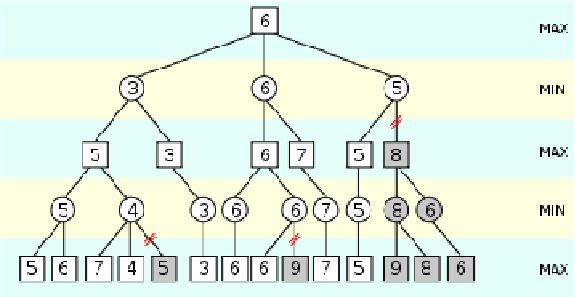
Bilangan Shannon adalah bilangan yang memperkirakan nilai batas bawah dari kompleksitas permainan catur. Shannon juga mengatakan bahwa untuk permainan catur, pada prinsipnya memungkinkan untuk melakukan *perfect play* atau membuat mesin melakukannya. Alasannya, akhir permainan harus ada, baik berakhir dengan kemenangan, kekalahan, atau berakhir seri. Hal ini dipertimbangkan dengan memperhatikan aturan 50 langkah seri. Shannon juga telah memperkirakan jumlah setiap kemungkinan posisi bidak catur pada papan catur sebesar $64! / 32!(8!)^2(2!)^6$, atau sekitar 10^{43}. Angka ini termasuk posisi bidak catur yang tidak mungkin terjadi.

3.2 Algoritma Alpha-Beta

Dalam pembuatan intelegensia buatan pada suatu permainan, program bekerja dengan menggunakan algoritma pencarian, yang mencari langkah terbaik dengan memperhatikan *game tree* dari permainan

tersebut. Pencarian tersebut menggunakan algoritma pencarian *minimax* (algoritma yang mencari keputusan terbaik), yang salah satu pemanfaatannya dengan menggunakan algoritma alpha-beta.

Algoritma alpha-beta merupakan suatu algoritma pencarian pada pohon n-*ary*. Algoritma ini bekerja dengan mengurangi jumlah simpul yang telah dievaluasi dari suatu pohon dengan dengan memanfaatkan algoritma *minimax*. Algoritma ini digunakan dalam permainan dengan dua pemain. Algoritma ini akan berhenti menilai suatu langkah ketika menemukan paling sedikit satu kemungkinan langkah yang telah terbukti lebih buruk dari langkah yang dievaluasi sebelumnya.



Gambar 3.1 Pohon algoritma alpha-beta

Algoritma ini memiliki dua nilai, alpha dan beta. Algoritma ini menggunakan nilai alpha adalah negatif tak terhingga, dan nilai beta adalah positif tak terhingga, kemudian secara rekursif mengganti nilai alpha dan beta sesuai langkah yang dilakukan pada setiap simpul secara terus menerus. Dengan algoritma rekursif, simpul yang akan dievaluasi semakin mengecil. Setelah terdapat nilai beta lebih kecil dari nilai alpha, artinya posisi tersebut tidak dapat dijadikan langkah terbaik untuk setiap pemain, dan tidak lagi diperhitungkan. Berikut adalah *pseudocode* dari algoritma alpha-beta :

```
function alphabeta(node, depth, α, β)
    (* β represents previous player best choice - doesn't want it if α would worsen it *)
    if node is a terminal node or depth = 0
        return the heuristic value of node
    foreach child of node
        α := max(α, -alphabeta(child, depth-1, -β, -α))
        (* use symmetry, -β becomes subsequently pruned α *)
        if β≤α
            break
(* Beta cut-off *)
    return α

(* Initial call *)
alphabeta(origin, depth, -inf, +inf)
```

4. KESIMPULAN

1. Teori *game* merupakan salah satu penerapan ilmu matematika.
2. Teori *game* dapat direpresentasikan dengan bentuk normal, maupun bentuk ekstensif.
3. *Combinatorial Game Theory* adalah teori *game* yang memanfaatkan cabang ilmu kombinatorial.
4. *Game tree* adalah representasi *combinatorial game* yang memanfaatkan teori graf, yaitu dalam bentuk pohon.
5. *Game tree* memiliki peranan yang sangat penting dalam pembuatan intelegensia buatan.
6. Kompleksitas *game* perlu diperhitungkan dalam pembuatan intelegensia buatan.
7. Intelegensia buatan pada beberapa *game* menggunakan algoritma pencarian.

DAFTAR REFERENSI

[1] Munir, Rinaldi. *Diktak Kuliah IF2091 Struktur Diskrit*. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. 2008.

[2] http://en.wikipedia.org/wiki/Game_theory
Tanggal Akses 04 Januari 2008.

[3] http://en.wikipedia.org/wiki/Combinatorial_game_theory
Tanggal Akses 04 Januari 2008.

[4] http://en.wikipedia.org/wiki/Game_complexity
Tanggal Akses 04 Januari 2008.

[5] http://en.wikipedia.org/wiki/Game_tree
Tanggal Akses 04 Januari 2008.

[6] http://nerdwisdom.com/2007/08/29/combinatorial-game-theory/
Tanggal Akses 06 Januari 2008.

[7] http://www.ics.uci.edu/~eppstein/cgt/
Tanggal Akses 05 Januari 2008.

[8] http://en.wikipedia.org/wiki/Alpha-beta_pruning
Tanggal Akses 06 Januari 2008.