Penerapan Algoritma Branch and Bound pada Computer Go

Tony / 13512018
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13512018@students.itb.ac.id

***Abstract*— Dalam pengembangan *Artificial Intelligence* (AI) sebuah engine, banyak dibutuhkan penerapan beberapa algoritma yang berkaitan dengan pencarian solusi terbaik yang dicari berdasarkan analisa kemungkinan kondisi yang terjadi. Oleh karena banyak sekali kemungkinan kondisi yang akan dianalisa, penganalisaan kondisi tersebut distrukturisasi sebagai sebuah bentuk pohon yang nantinya akan merujuk ke solusi terbaik yang akan diambil oleh komputer tersebut.**

Dalam makalah ini, akan lebih dibahas khusus mengenai penerapan sebuah algoritma pencarian solusi terbaik pada sebuah AI Computer Go yaitu algoritma Branch and Bound. Pada algoritma ini, akan ada sebuah fungsi pembatas yang bersifat heuristik yang merupakan analisa dari keadaan di papan Go tersebut.

***Index Terms*— Go, pencarian solusi, branch and bound, intelegensia buatan**

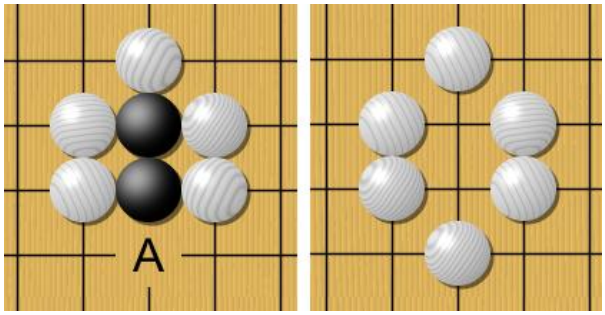
I. PENDAHULUAN

Go atau biasa disebut catur Igo adalah sebuah permainan tradisional yang berasal dari negara China yang ditemukan lebih dari 2.500 tahun yang lalu. Permainan Go adalah sebuah permainan berbasis papan (*board game*) yang dimainkan oleh dua orang yang memegang buah hitam dan putih. Papan tersebut terdiri dari 19 x 19 garis-garis yang melambangkan tempat menaruh buah tersebut (ada juga papan yang 9x9 atau 13x13 untuk pemula).

Permainan ini tidak seperti permainan catur (*chess*) dimana buah yang diletakkan adalah tepat pada kotak-kotak yang ada. Pada permainan ini, buah tersebut diletakkan pada perpotongan antara garis-garis yang terdapat pada papan. Ukuran papannya yang sangat besar pun ini membuat kompleksitas permainan ini sangatlah tinggi. Strategi yang dapat diterapkan dalam permainan ini sangatlah bervariasi walaupun metode dan peraturan dari permainan ini cukup sederhana. Dalam permainan ini, seorang dikatakan memenangkan permainan jika buahnya menguasai total area yang lebih besar daripada area yang dikuasai oleh lawan.

Ilustrasi Papan Go
Sumber Gambar:
http://en.wikipedia.org/wiki/File:FloorGoban.JPG

Peraturan dari permainan ini, buah yang sudah diletakkan di area tertentu tidak dapat dipindahkan. Namun, buah tersebut dapat disingkarkan dari papan jika buah tersebut 'ditangkap'. Definisi ditangkap disini adalah ketika buah tersebut dikelilingi oleh semua buah lawan dan buah yang dikelilingi tersebut bertetangga dengan semua buah yang mengelilinginya (menempel tepat di sampingnya).



Ilustrasi penangkapan buah
Sumber gambar :
http://en.wikipedia.org/wiki/File:Go_capturing.png

Pemain dapat melakukan *pass* atau tidak melakukan gilirannya. Permainan berakhir saat kedua pemain

melakukan *pass*. Pada saat kedua pemain tersebut setuju untuk tidak melanjutkan lagi permainan, pada saat itulah permainan berakhir dan dilanjutkan sesi penentuan area penguasaan. Pemain dengan penguasaan area yang lebih tinggi lah yang memenangkan permainan. Perlu dicatat bahwa penguasaan permainan disini adalah wilayah yang dikelilingi oleh buah-buah pemain.

Permainan Go ini merupakan sebuah tantangan yang sulit dalam lingkungan intelegensia buatan untuk Computer Go dan lebih sulit untuk dipecahkan persoalan pencarian solusi terbaiknya dibandingkan jenis permainan catur lainnya pada umumnya. Hal tersebut karena pada Go, kemungkinannya sangatlah banyak dan lebih banyak kepada hal yang bersifat heuristik sebagai pembatas pencarian solusinya seiring dengan meningkatnya kualitas pemain yang bermain dan meningkatnya jumlah buah yang terdapat di papan. Oleh karena itu, menjadi sebuah hal yang menarik untuk mengangkat topik ini menjadi bahan analisis.

II. TEORI ALGORITMA BRANCH & BOUND

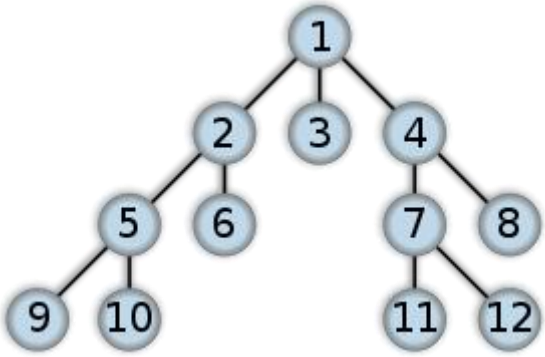
Sebelum mengenal lebih jauh mengenai algoritma Branch & Bound, terlebih dahulu akan dibahas mengenai algoritma yang mendasarinya yaitu *Breadth First Search* (BFS).

2.1. Breadth First Search (BFS)

2.1.1. Definisi

Metode pencarian BFS adalah sebuah strategi pencarian sebuah solusi pada sebuah graf pohon dimana pencarian dilakukan ke seluruh anaknya dari sebuah node dengan terlebih dahulu menghidupkan anak-anaknya tersebut. Metode tersebut diulangi terus sampai ada sebuah node yang ditelusuri merupakan sebuah solusi.

Metode ini mengunjungi semua node yang merupakan tetangga dari sebuah akar dan kemudian untuk setiap node yang telah dikunjungi, kunjungi kembali seluruh node yang merupakan tetangga dari node tersebut tetapi belum dikunjungi dan begitu seterusnya sampai ketemu pada solusi.



Ilustrasi penelusuran BFS

Sumber Gambar :
http://en.wikipedia.org/wiki/File:Breadth-first-tree.svg

2.2. *Branch & Bound (B&B)*

Algoritma *Branch & Bound* ini merupakan metode pencarian di dalam ruang solusi secara sistematis. Ruang solusi diorganisasikan ke dalam pohon ruang status. Pembentukan pohon ruang status pada algoritma *B&B* dilakukan secara dinamis dengan skema BFS. Untuk mempercepat pencarian ke simpul solusi, maka setiap simpul diberi sebuah nilai ongkos (*cost*). Simpul berikutnya yang akan diekspansi tidak lagi berdasrkan urutan pembangkitannya, tetapi simpul yang memiliki ongkos yang minimalis (paling kecil / paling besar tergantung algoritma pembatasnya) di antara simpul-simpul yang hidup. Nilai ongkos pada setiap simpul i menyatakan taksiran ongkos minimalis lintasan dari simpul i ke simpul solusi (*goal node*).

c(i) = ongkos minimal dari simpul status i ke status tujuan

Dengan kata lain, nilai c(i) menyatakan batas bawah dari ongkos pencarian solusi dari status i. Ongkos ini dihitung dengan sebuah fungsi pembatas yang membatasi agar pembangkitan simpul tidak mengarah ke simpul solusi.

2.1.2. Prinsip Pencarian Solusi

Pada metode ini, semua anak yang dibangkitkan dari sebuah simpul-E berada pada aras yang sama. Algoritma BFS menggunakan antrian untuk menyimpan simpul-simpuul yang baru dibangkitkan. Simpul-simpul yang baru dibangkitkan tersebut diletakkan di bagian belakang sebuah antrian. Simpul berikutnya yang akan diekspansi adalah simpul yang berada pada bagian kepala antrian. Prinsip antrian pada BFS ini sama dengan skema *FIFO* (*First In First Out*). Dengan skema FIFO ini, simpul hidup dimasukkan ke dalam antrian, simpul berikutnya yang akan menjadi simpul E adalah simpul yang pertama masuk ke dalam antrian.

Pada algoritma *B&B*, pencarian ke simpul solusi dapat dipercepat dengan memilih simpul hidup berdasarkan nilai ongkos (*cost*). Setiap simpul hidup diasosiasikan dengan sebuah ongkos yang menyatakan nilai batas (*bound*). Untuk setiap simpul X, nilai batas ini dapat berupa :

- jumlah simpul dalam upapohon X yang perlu dibangkitkan sebelum simpul solusi ditemukan, atau
- Panjang lintasan dari simpul X ke simpul solusi terdekat (di dalam upapohon X)

Untuk memilih simpul hidup yang akan menjadi simpul E, simpul-simpul hidup diurut berdasarkan nilai batasnya pada antrian tersebut(dibuat *priority queue*). Strategi memilih simpul E seperti ini dinamakan strategi

pencarian bedasarkan biaya terkecil (*least cost search*).

Bentuk fungsi heuristik yang menjadi fungsi untuk menghitung taksiran nilai *cost* secara umum adalah sebagai berikut :

$$c(i) = f(i) + g(i)$$

Keterangan :

c(i) = ongkos untuk simpul i

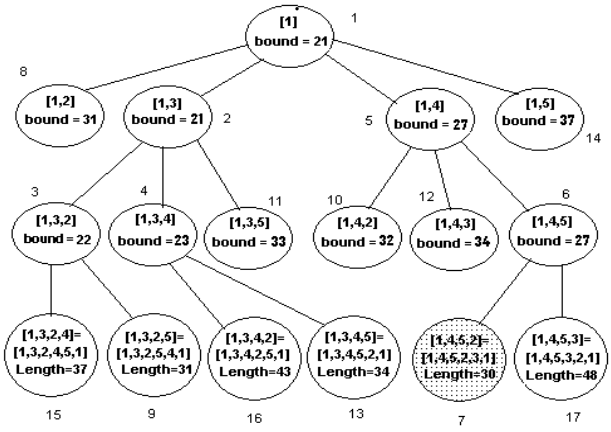
f(i) = ongkos mencapai simpul i dari akar

g(i) = ongkos mencapai simpul tujuan dari simpul i

Nilai c digunakan untuk mengurutkan pencarian. Simpul berikutnya yang dipilih untuk diekspansi adalah simpul yang memiliki nilai c yang minimum.

Algoritma B&B:

1. Masukkan simpul akar ke dalam antrian Q. Jika simpul akar adalah simpul solusi (*goal node*), maka solusi telah ditemukan. Stop.
2. Jika Q kosong, tidak ada solusi. Stop.
3. Jika Q tidak kosong, pilih dari antrian Q simpul i yang mempunyai c(i) paling kecil. Jika terdapat beberapa simpul i yang memenuhi, pilih satu secara sembarang.
4. Jika simpul i adalah simpul solusi, berarti solusi sudah ditemukan, stop. Jika simpul i bukan simpul solusi, maka bangkitkan semua anak-anaknya. Jnika i tidak mempunyai anak, kembali ke langkah 2.
5. Untuk setiap anak j dari simpul i, hitung c(j) dan masukkan semua anak-anak tersebut ke dalam antrian Q.
6. Kembali ke langkah 2.



Ilustrasi Branch & Bound

Sumber Gambar :

http://www.academic.marist.edu/~jzbv/algorithms/Branch%20and%20Bound_files/image004.gif

III. PENERAPAN B&B PADA COMPUTER GO

3.1. Kompleksitas Computer Go

Pencarian solusi pada permainan Go merupakan tantangan yang sulit dalam lingkungan intelegensia buatan. Kompleksitasnya yang sangat banyak karena ukuran papan yang berukuran besar (19 x 19) dan dinamika peraturan dalam permainannya inilah yang membuat persoalan ini menjadi jauh lebih rumit daripada persoalan pencarian solusi dari *chess computer*.

Pola yang dipakai pada pencarian solusi *Computer Go* ini juga lebih sulit karena semakin meningkatnya dinamika permainan dan tingkat permainan menjadikan permainan ini lebih bersifat intuitif dari setiap langkah yang diambil. Para profesional di bidang ini menunjukkan hal itu bahwa dalam tahap profesional, kalkulasi untuk menghitung setiap langkah yang tepat menjadi lebih intuitif berdasarkan pengalaman yang ada. Oleh karena komputer tidak bisa bertindak secara intuitif, maka pendekatan heuristik lah yang dapat dipakai dalam permasalahan ini.

Computer Go yang pertama didasarkan pada *pattern recognition* yang mengestimasi setiap teritori langkah dari setiap kemungkinan solusi yang diambil.

Oleh karena kesulitan dalam mencari langkah terbaiknya ini lah, bahkan banyak pemain profesional yang kuat yang dapat mengalahkan Computer Go ini bahkan dengan memberikan *handicap* yang cukup besar sekitar 25-30 buah.

Sekarang ini, pengembangan pada Computer Go baru dapat menunjukkan performa yang mumpuni setingkat pemain profesional sebatas papan 9x9 saja.

Hal-hal yang menjadi kendala kompleksitas pada Computer Go ini adalah sebagai berikut :

- Ukuran papan

  Ukuran papan (19x19, 361 perpotongan) menjadi kendala akan banyaknya kombinasi kemungkinan yang mungkin terjadi pada papan tersebut. Oleh karena itu banyaknya komputasi yang dijalankan menjadikan kendala tersendiri dalam pencarian solusi pada *Computer Go*.
- Banyak sekali kemungkinan langkah

  Membandingkan dengan catur, langkah-langkah pada Go tidak terlalu terbatas pada peraturan permainan. Sebagai perbandingan, langkah pertama pada catur mempunyai 20 kemungkinan. Pada Go, terdapat 55 langkah termasuk pada simetrisitas yang terdapat pada papan. Nilai langkah ini terus bertambah eksponensial seiring berkurangnya simetrisitas pada susunan buah di papan dan kemudian pada akhirnya 361 titik pada papan harus dievaluasi satu per satu.
- Sifat dari permainan itu sendiri

  Pada permainan *board game* umumnya, jumlah buah akan berkurang terus seiring semakin permainan mendekati *ending*. Namun, pada setiap langkah Go terjadi hal

yang sebaliknya. Semakin permainan mendekati akhir, kompleksitas dan kemungkinan terus bertambah sampai pada titik dimana batas area menjadi benar-benar jelas.

Teknik-teknik pada Computer Chess yang tidak dapat diaplikasikan pada Computer Go :

- Fungsi evaluasi

  Fungsi penghitung langkah yang akan diambil pada catur sangat berbeda dengan Go dalam hal intelegensia buatan. Pada catur, pengevaluasian setiap langkahnya dapat dihitung berdasarkan nilai poin setiap buah yang tersisa di papan, baik itu dari segi posisi maupun nilai materil. Namun, pada Go, hal tersebut tidak bisa diterapkan secara efektif karena evaluasi posisi pada Go tergantung pada analisis yang sangat kompleks apakah sebuah grup buah adalah grup hidup atau bukan, buah yang mana yang menghubungkan ke buah yang lain, peletakan posisi yang secara heuristik dapat meningkatkan kualitas posisi, maupun peletakan langkah yang menyerang posisi lawan yang lemah.
- Permasalahan kombinatorial

  Terkadang, beberapa persoalan kombinatorial yang sulit dalam beberapa kasus ini dapat dikonversi menjadi permasalahan dalam Go(masalah NP-Complete). Masalah NP-Complete ini cenderung lebih mudah bagi manusia untuk memecahkannya daripada komputer. Oleh karena itu, gagasan bahwa permasalahan NP-Complete dikonversikan pada permasalahan Go ini tidak membantu dalam menjelaskan keunggulan manusia dibanding Computer Go.
- Akhir Permainan

  Pada catur, akhir permainan dikalkulasi dengan database ending yang ada dan dicocokkan. Pada Computer Go, perhitungan pada tahap akhir permainan sulit karena walaupun Computer Go dapat mengkalkulasikan akhir permainan pada sebuah lokal area tertentu secara tepat, belum tentu hal tersebut tepat secara keseluruhan papan. Persoalan hidup dan mati buah dalam papan tersebut juga menjadi persoalan sendiri dalam akhir permainan. Pada akhir permainan Go sendiri, area area yang tadinya terpisah pun saling menyambung menjadi sebuah kesatuan permainan yang utuh membuat keadaan alami yang terbentuk dalam permainan Go yang dinamik. Hal ini dapat membuat situasi sangat kompleks seperti terjadinya *Triple KO*, *Quadruple KO*.

Oleh karena itu, manusia masih lebih baik dalam permainan Go ini karena tidak seperti Catur yang dapat berpindah buahnya maupun reversi yang mengubah state buahnya, lebih mudah bagi manusia untuk membaca rentetan langkah-langkah yang hal tersebut irelevan bagi sebuah komputer.

3.2. Pencarian nilai Ongkos B&B

Pada persoalan intelegensia buatan dalam Computer Go ini, nilai *cost* yang dihitung berdasarkan nilai heuristik yang ada. Nilai heuristik tersebut dapat dihitung berdasarkan fungsi yang menghitung nilai berdasarkan kondisi kemungkinan yang dapat mempengaruhi ‘nilai’ posisi dari sebuah kondisi pada papan.

Pertama-tama cari dahulu setiap kemungkinan posisi yang dapat di telusuri awalnya. Lalu hitunglah nilai *cost* dari setiap kemungkinan langkah yang ada tersebut berdasarkan fungsi heuristik yang dibentuk. Nilai fungsi heuristik dalam Computer Go dapat berdasarkan evaluasi keseluruhan yang terjadi dalam papan. Hal tersebut dapat berupa evaluasi penilaian secara heuristik tentang area kekuasaan yang diperoleh jika memilih langkah tersebut, keterhubungan antara buah yang satu dengan yang lainnya, dan nilai-nilai heuristik lainnya yang mengambil kontribusi pada nilai tertentu pada fungsi heuristik tersebut. Nilai cost yang terhitung kemudian diurutkan berdasarkan nilai yang terbesar. Nilai cost terbesar yang kemudian akan dilanjutkan dengan mengevaluasi kemungkinan langkah berikutnya dengan mengambil langkah terbaik sementara tersebut.

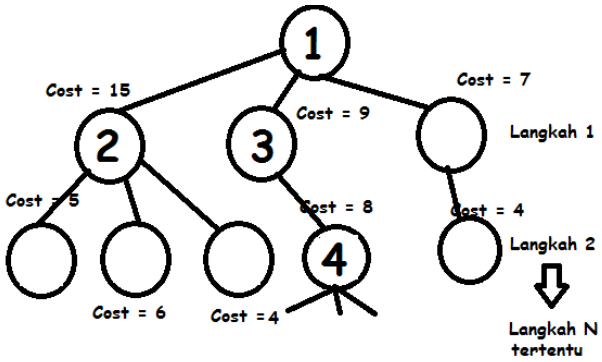
Rincian fungsi heuristik tersebut menjadi demikian :

$$c(i) = f(1) + f(2) + ... + f(n)$$

Keterangan :

c(i) adalah nilai cost yang dihitung

f(1)..f(n) menandakan fungsi-fungsi heuristik yang diperhitungkan untuk mendapatkan cost c(i) tersebut



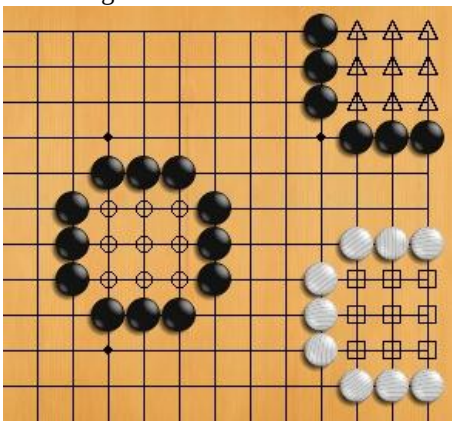
Ilustrasi urutan penelusuran B&B
Sumber : dibuat sendiri

Penghitungan nilai ongkos dapat dilakukan seperti ilustrasi di atas. Pertama-tama simpul akar akan menghidupkan seluruh anak-anaknya. Kemudian anak-anak yang dihidupkan tersebut dimasukkan ke dalam antrian dengan skala prioritas yang paling besar yang akan jadi kepalanya. Lalu, lakukan perhitungan seluruh nilai cost dari anak-anaknya berdasarkan fungsi heuristik yang ada. Dari situ, carilah nilai maksimum dari langkah tersebut (langkah 1 alias langkah yang akan dijalankan). Dari situ, nilai cost yang maksimum dari anak-anak yang sudah dihidupkan tersebut(kepala dari antrian) ditelusuri kembali seluruh kemungkinan langkah selanjutnya dari simpul tersebut dengan menghidupkan seluruh anak-anak dari simpul tersebut dan masukkan ke dalam antrian. Lalu lakukan langkah yang sama seperti yang sebelumnya dengan menghitung cost seluruh anak-anaknya. Lakukan kalkulasi begitu terus berikutnya sampai langkah ke –N dimana N adalah batasan pengkalkulasian akan dilakukan kalkulasi hingga berapa langkah ke depan. Misalkan N adalah 10, artinya Go Computer akan melakukan kalkulasi langkah yang akan diambilnya sampai nilai posisi di langkah ke 10. Penghidupan simpul-simpul juga tidak usah dihidupkan semua kemungkinan yang ada, melainkan kemungkinan rasional yang bernilai secara heuristik.

Fungsi-fungsi heuristik yang dapat harus diperhitungkan dalam Go Computer dalam mencari cost ini antara lain sebagai berikut :

- Nilai Teritori / Kekuasaan

  Setiap langkah yang diambil mengkalkulasian teritori yang didapat dari pengambilan keputusan langkah tersebut. Jika nilai teritori yang dikuasai bertambah dari hasil langkah tersebut, maka nilai batasan ongkos akan ditambahkan. Hal tersebut akan menambahkan persentase kemungkinan langkah tersebut diambil. Hal tersebut karena jelas semakin besar teritori yang terambil, maka akan semakin besar pula kemungkinan kemenangan.



Ilustrasi Area
Sumber Gambar :
http://upload.wikimedia.org/wikipedia

- Nilai Buah

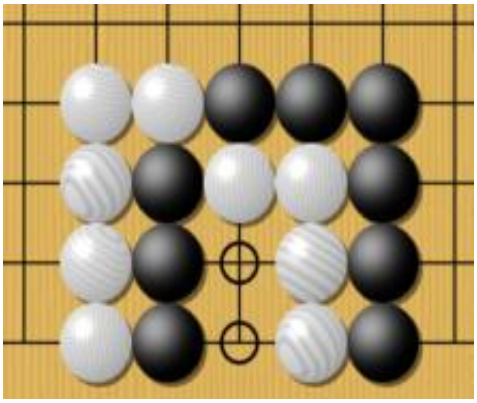
  Peletakkan buah Go itu sendiri pada papan memberikan andil terhadap nilai cost pada Go dengan menghitung nilai evaluasi posisi yang lebih mendalam secara heuristik seperti peletakkan buah yang mempunyai posisi yang baik dalam mempertahankan area yang sudah ada maupun peletakkan buah untuk menyerang area musuh sehingga mempersempit area musuh sehingga tidak harus selalu memperluas area sendiri.

- Mati dan Hidup Buah

  Dalam permainan Go, ada istilah yang merujuk apakah buah-buah pada papan Go adalah buah mati atau buah hidup. Buah tersebut dikatakan buah hidup jika terdapat minimal 2 mata hidup dalam rangkaian buah-buahnya. Hal ini menjadi perhitungan sendiri dalam fungsi cost tersebut karena untuk dapat mempertahankan wilayah yang sudah ada, haruslah terdapat minimal 2 mata hidup dan jika ingin membuat buah lawan mati harus dapat mencegah agar lawan tidak mempunyai 2 mata hidup.

- Taktik-taktik lainnya

  Taktik-taktik yang dapat mempengaruhi terhadap keberjalanan sebuah permainan Go dalam Computer Go ini sangatlah banyak dan lagi bergantung pada nilai heuristik yang ada. Misalnya situasi seki (*mutual life*) pada permainan Go dapat mempengaruhi nilai fungsi heuristik apakah sebuah langkah tersebut dapat membentuk situasi tersebut makin buruk atau tidak.



Ilustrasi Seki
Sumber Gambar :
http://en.wikipedia.org/wiki/File:Goseki.png

IV. KESIMPULAN

Penerapan BFS pada ummnya dan Branch and Bound pada khususnya sangatlah banyak sekali. Dalam hal ini,

penerapan algoritma ini pada Computer Go cukup rasional untuk diterapkan secara logika walaupun dengan sekian banyaknya fungsi heuristik yang harus diimplementasikan untuk menghitung costnya. Namun, hal tersebut masih bersifat subjektif dan masih sulit sekali untuk dibuktikan secara objektif dari segi hasil kalkulasi nilai cost tersebut.

Oleh karena itu, walaupun teknik pencarian B&B belum dapat dibuktikan secara ampuh terhadap pencarian solusi langkah efektif pada Computer Go, namun teknik ini masih dapat dikembangkan dengan pendekatan yang ada dengan meningkatkan kalkulasi terhadap fungsi heuristik yang menjadi tonggak algoritma Branch and Bound ini.

.

REFERENCES

[1] http://www.wired.com/2014/05/the-world-of-computer-go/
diakses tanggal 17 Mei 2014 pukul 22:03 WIB

[2] http://c2.com/cgi/wiki?GameOfGo
diakses tanggal 17 Mei 2014 pukul 22:12 WIB

[3] http://www0.cs.ucl.ac.uk/staff/d.silver/web/Applications_files/mcrave.pdf
diakses tanggal 17 Mei 2014 pukul 22:13 WIB

[4] http://www.chilton-computing.org.uk/acl/literature/reports/p019.htm
diakses tanggal 17 Mei 2014 pukul 22:20 WIB

[5] http://archive.org/stream/byte-magazine-1981-04/1981_04_BYTE_06-04_Future_Computers#page/n101/mode/2up
diakses tanggal 17 Mei 2014 pukul 23:01 WIB

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 17 Mei 2014

Tony
13512018