Penerapan Algoritma *Branch and Bound* dalam Permainan Penempatan Barang oleh Robot

Daniel Riyanto / 13518075
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13518075@std.stei.itb.ac.id

***Abstrak*—Algoritma *Branch and Bound* merupakan suatu pendekatan yang sangat banyak digunakan di dalam kehidupan sehari-hari. Algoritma *Branch and Bound* bisa membantu kita dalam membuat robot yang dapat memproses keefisien dalam bekerja.**

***Kata Kunci*—permainan, *branch and bround*, *cost*, simpul, jarak**

I. PENDAHULUAN

Pada zaman sekarang ini, ilmu pengetahuan dan teknologi sudah berkembang semakin pesat khususnya dalam bidang elektronika dan IT. Hal itulah yang menuntut setiap orang untuk lebih siap dalam menghdapai persaingan khususnya dalam dunia kerja. Perusahaan-perusahaan besar saat ini saling berkompetisi dalam hal berinovasi untuk meningkatkan produktivitas perusahaan. Banyak perusahaan memanfaatkan teknologi yang memiliki kecepatan, akurasi, dan keandalan yang tinggi serta mudah dalam hal pengoperasiannya sebagai alat untuk menunjang produktivitas mereka salah satunya robot.

Robot adalah salah satu teknologi yang paling mutakhir pada era sekarang ini. Banyak lembaga ataupun instansi yang berlomba-lomba mengadakan kompetisi robot baik dalam skala regional, nasional maupun internasional. Tujuan dari diadakan kompetisinya ini tidak lain adalah untuk mendidik generasi-generasi muda khususnya Indonesia agar tidak tertinggal dalam hal riset ilmu pengetahuan dan teknologi. Robot pada dasarnya diciptakan untuk membantu menyelesaikan pekerjaan manusia yang rumit dan memerlukan banyak tenaga.

Beberapa algoritma memiliki teknik-teknik tertentu agar dapat berjalan seoptimal mungkin. Salah satu algoritma yang cukup banyak digunakan adalah Algoritma *Branch and Bound*. Algoritma *Branch and Bound* memiliki prinsip utama yaitu mengambil nilai paling optimal (minimal atau maksimal) dari suatu persoalan. Salah satu penggunaan Algoritma *Branch and Bound* adalah membuat tingkah laku dari *bot* atau *computer player* dalam menentukan langkah-langkah selanjutnya.

Penulisan makalah ini terinspirasi dari salah satu kompetisi robot internasional WRO. Penulis ingin membuat permainan penempatan barang dengan robot yang menggunakan Algoritma *Branch and Bound*.

II. LANDASAN TEORI

A. Algoritma Branch and Bound

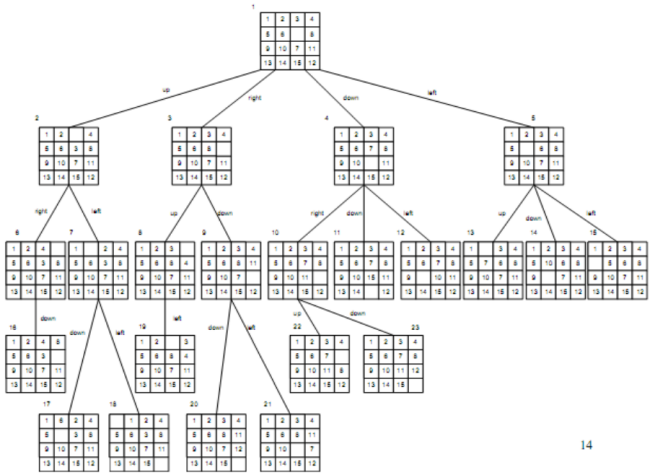
Algoritma *Branch and Bound* biasanya digunakan untuk persoalan optmisasi yaitu untuk meminimalkan atau memaksimalkan suatu fungsi objektif yang tidak melanggar batasan (*constraints*) persoalan. Algoritma *Branch and Bound* menggunakan kombinasi dari algoritma BFS dan *Least Cost Search*. Setiap simpul dalam Algoritma *Branch and Bound* diberi sebuah nilai *cost* yang nilainya ditentukan berdasarkan taksiran lintasan termurah ke simpul tujuan yang melalui simpul tersebut. Simpul berikutnya yang akan di-*expand* dipilih yang nilai *cost*-nya paling kecil (*least cost search*) agar dapat meminimalisir langkah-langkah iterasi.

Algoritma *Branch and Bound* hampir sama dengan Algoritma *Backtracking* di mana persamaannya adalah pencarian solusinya dengan pembentukan pohon ruang status dan ‘membunuh’ simpul yang tidak mengarah ke solusi. Perbedaannya adalah Algoritma *Backtracking* bukan untuk masalah optimisasi sedangkan Algoritma *Branch and Bound* untuk masalah optimisasi dan pembangkitan simpul untuk Algoritma *Backtracking* adalah menggunakan DFS sedangkan Algoritma *Branch and Bound* menggunakan beberapa ‘aturan’ tertentu yang umumnya itu *Best First Rule*.

Algoritma *Branch and Bound* menerapkan “pemangkasan” pada jalur yang dianggap tidak lagi mengarah pada solusi. Ciri-ciri pemangkasan secara umum:

- Nilai simpul tidak lebih baik dari nilai sejauh ini.
- Simpul yang tidak merepresentasikan solusi yang *feasible* karena ada batasan yang dilanggar.
- Solusi yang *feasible* pada simpul hanya terdiri atas satu titik yang artinya tidak ada pilihan lain dengan membandingkan nilai fungsi objektif dengan solusi yang terbaik saat ini dan mengambil yang terbaik.

Banyak persoalan yang menggunakan Algoritma *Branch and Bound* di antaranya adalah *The N-Queens Problem*, Permainan 15-Puzzle, *Travelling Salesperson Problem* (TSP) dan lain-lain.



Gambar 1. Penggunaan Algoritma *Branch and Bround* untuk Permainan 15-Puzzle

B. World Robotic Olympiad (WRO) Foundation Inc.

World Robot Olympiad (WRO) Foundation Inc. adalah organisasi *non-profit* yang didirikan untuk mendukung *World Robot Olympiad (WRO) Association* dan upayanya untuk mendorong sebanyak mungkin kaum muda untuk menaruh minat dalam bidang sains, teknologi, teknik dan matematika (STEM) melalui kompetisi dan kegiatan pendidikan robot di Amerika Serikat dan seluruh dunia.

Gambar 2. Logo WRO
Sumber: wro-association.org

Asosiasi WRO didirikan pada tahun 2004 untuk memberikan kesempatan bagi kaum muda dari seluruh dunia untuk memperluas wawasan mereka dan mendapatkan keterampilan STEM yang berharga melalui kompetisi robot. Pada tahun pertama, 12 negara ambil bagian dalam kompetisi final internasional perdana yang diadakan di Singapura.

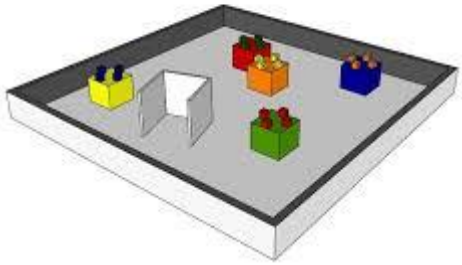
Pada tahun 2018, Dewan Direksi Asosiasi WRO memutuskan untuk membuat yayasan 501C3 yang berbasis di Amerika Serikat. WRO *Foundation* akan memunginkan Asosiasi WRO untuk membangun kesuksesan masa lalu mereka dengan membolehkan lebih banyak pemuda dari AS dan seluruh dunia untuk mengalami pengalaman yang mengubah hidup dari kompetisi robotik internasional.

Larry Rzepka adalah *Vice President of Philantrophy* dan *Chief Operations Officer* (COO)untuk WRO *Foundation*. Beliau memiliki lebih dari 35 tahun pengalaman nirlaba, penceranaan strategis dan pengembanagan sumber daya untuk organisasi mulai dari *South Carolina's Governors School for Science and Mathematics Foundation* hingga *National Science Teachers Association* dan yang terbaru adalah *National Defense University Association*. Beliau memiliki gelar BA dalam *Public Relations/ Journalism* dari Univeristas Madonna di Michigan.

Gambar 3. Larry Rzepka
Sumber: wro-association.org

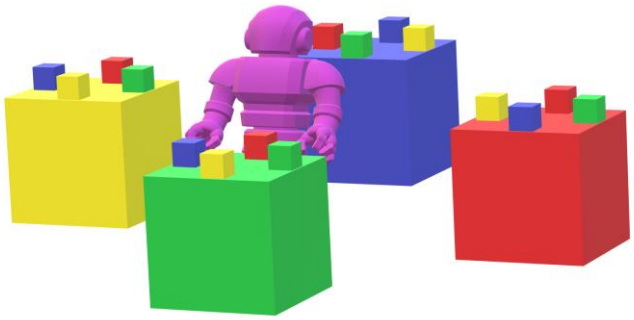
III. PENJELASAN PERSOALAN

Pada Bab I sudah dituliskan bahwa penulisan jurnal ini terinspirasi dari salah satu kompetisi WRO. Nama kompetisi WRO adalah SMART Greenhouse yang akan diadakan pada tahun ini (atau 2020). Tantangan dari kompetisi ini adalah robot diharuskan mendapatkan poin sebanyak mungkin dalam mengurus perkebunan di rumah kaca.



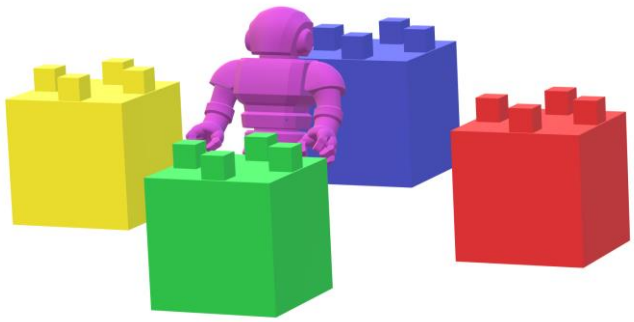
Gambar 4. SMART Greenhouse
Sumber; wro-association.org

Penulis mendapatkan ide untuk membuat persoalan *Branch and Bound* untuk persoalan penempatan barang oleh robot seperti gambar di bawah ini.



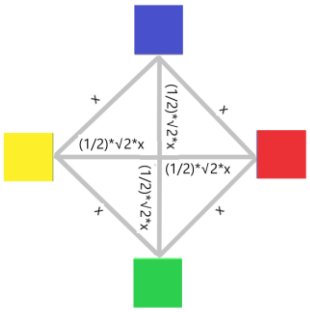
Gambar 5. Persoalan Penempatan Barang

Goal dari persoalan ini adalah semua kubik kecil berada di atas kubik besar yang warnanya sama dengan waktu pemindahan kubik-kubik kecil ini paling kecil atau optimal. Peraturannya adalah robot hanya bisa memegang dua barang di kedua tangannya dan setiap kubik besar hanya bisa menampung paling banyak empat kubik kecil di atasnya.



Gambar 6. *Goal* Persoalan

Jarak antara setiap kubik besar dipaparkan pada gambar di bawah ini. Variabel x di sini adalah jarak antar kubik terdekat. Pada gambar di bawah ini, bisa dilihat bahwa lokasi kubik-kubik besar ini rapi sehingga jarak dari satu kubik ke kubik yang lain itu sama untuk setiap kubik.



Gambar 7. Jarak antar kubik besar

IV. PELAKSANAAN PERSOALAN

Pada Bab II sudah disebutkan bahwa Algoritma *Branch and Bound* perlu menentukan cara hitungan *cost* untuk setiap simpul. Sebelum menentukan *cost*, penulis memikirkan adanya tiga asumsi untuk memudahkan penentuan *cost* ini yaitu sebagai berikut.

1. Robot dapat memanjangkan dan memendekkan tangan dengan sangat cepat dan hanya bisa mengambil kubik kecil jika berada sangat dekat dengan kubik besarnya.
2. Robot dapat mencengkram atau melepaskan benda dengan sangat cepat.
3. Robot dapat berotasi atau berputar badan dengan sangat cepat.

Dengan asumsi-asumsi inilah maka bisa ditentukan *cost* dengan rumus sebagai berikut.

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

c(i) = ongkos untuk simpul i

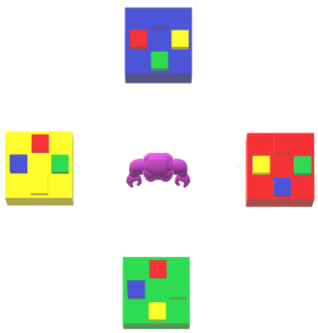
f(i) = ongkos mencapai simpul i dari akar (didapatkan dari panjang jalannya robot)

g(i) = ongkos mencapai simpul tujuan dari simpul i (didapatkan dari jumlah kubik kecil yang belum berada pada posisi yang sesuai)

Dari simpul-simpul yang di-*expand* ada tiga jenis simpul berdasarkan kegiatan robot yaitu sebagai berikut.

1. Simpul ambil, di mana robot berjalan dan mengambil satu kubik kecil pada saat simpul ini.
2. Simpul ambil dan taruh, di mana robot berjalan, mengambil kubik kecil baru dan menukar dengan kubik kecil yang lama.
3. Simpul taruh, di mana robot berjalan dan menaruh kubik kecil yang dipegangnya.

Dengan adanya asumsi-asumsi, cara penentuan *cost*, dan pembedaan tiga jenis simpul maka sudah bisa memulai tahap-tahap pengerjaan persoalan ini.

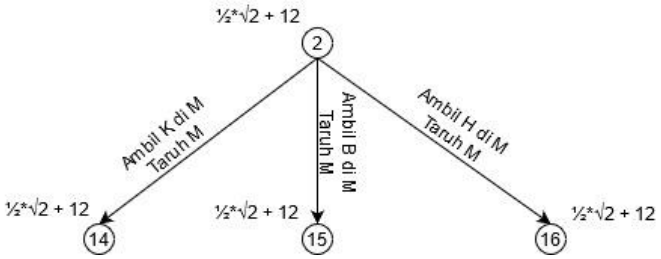


Gambar 8. Gambar Lokasi-Lokasi Kubik dari Atas Sebelum

Gambar di atas untuk memudahkan penjelasan penentuan *cost* untuk simpul akar. f(i) masih bernilai 0 karena robot sama sekali belum berpindah tempat atau berjalan. g(i) bernilai 12 karena ada 12 kubik kecil yang belum berada pada tempat yang sesuai. Jadi *cost* untuk simpul akar ini adalah 12. Kemudian simpul akar di-*expand* dan dihidupkanlah 12 simpul karena ada

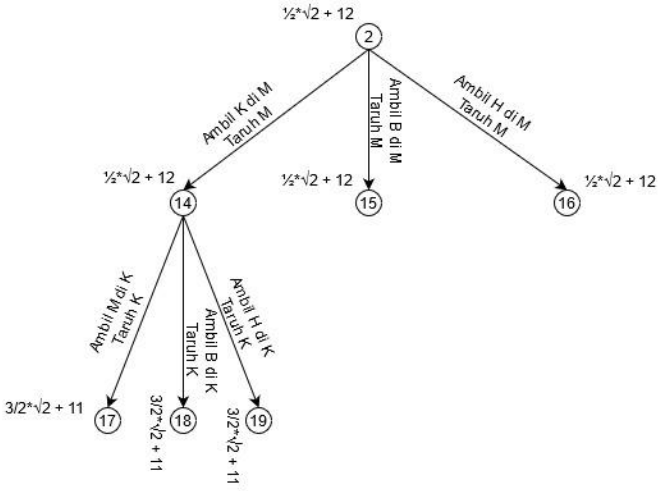
12 kubik kecil yang belum berada pada tempatnya. Setiap simpul yang dihidupkan ini memiliki *cost* yang sama karena simpul-simpul ini tergolong simpul awal di mana robot hanya berjalan dan mengambil satu kubik kecil saja. *Cost* untuk setiap simpul yang baru dihidupkan adalah ½*√2 + 12 karena robot berjalan dengan jarak ½*√2 m dan masih ada 12 kubik kecil yang belum berada pada tempatnya. Gambar untuk penghidupan simpul awal ini bisa dilihat pada Gambar 9. Teks "ambil M di K" itu artinya mengambil kucik kecil yang berwarna merah di kubik besar kuning. Teks ini disingkat supaya gambarnya lebih teratur dan mudah dilihat.

Kemudian, simpul ke-2 diekspan menghidupkan tiga buah simpul tengah karena sebelumnya robot mengambil kubik kecil berwarna merah dan di atas kubik besar berwarna merah ada tiga kubik kecil yang seharusnya tidak berada di atas kubik itu. Tidak mungkin robot berjalan ke kubik besar lain selain warna merah karena akan memperlambat kinerja pemindahan barang. *Cost* untuk tiga simpul yang baru dihidupkan bernilai (½*√2 + 1) + 11 karena robot sudah berjalan sebesar x meter sehingga f(i)-nya bernilai ½*√2 dari jarak berjalan sebelumnya ditambahkan dengan 1 dari jarak berjalan sekarang sehingga hasilnya adalah ½*√2 + 1. Pada kondisi sekarang, jumlah kubik kecil yang belum berada pada posisinya yang sesuai adalah 11 sehingga g(i)-nya bernilai 11. Gambar di bawah ini menunjukkan hasil penghidupan tiga simpul.



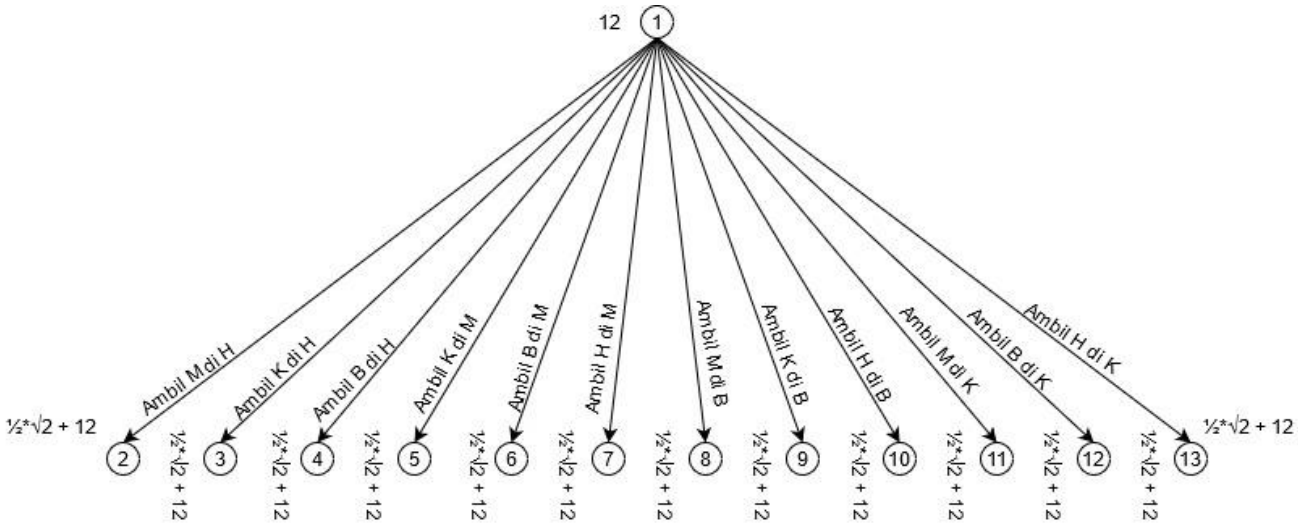
Gambar 10. Penghidupan tiga simpul baru

Kemudian dilanjutkan pengekspanan simpul 14 walaupun semua simpul itu sama *cost*-nya agar lebih cepat menemukan *goal*-nya karena *level* simpul 14 lebih dalam daripada yang lain dan simpul terdepan dari hasil ekspan simpul 2. Dari hasil ekspan simpul 14 ini dihidupkanlah tiga simpul baru lagi dengan masing-masing *cost*-nya sebesar (½*√2 + 1 + √2) + 10 atau 3/2*√2 + 11 karena robot sekarang berjalan sebesar √2 x meter sehingga f(i)-nya bernilai ½*√2 + 1 + √2. Pada ketiga simpul ini, jumlah kubik kecil yang belum berada pada posisi yang sesuai adalah 10 sehingga g(i)-nya bernilai 10. Gambar di bawah ini menunjukkan graf hasil ekspan simpul 14.

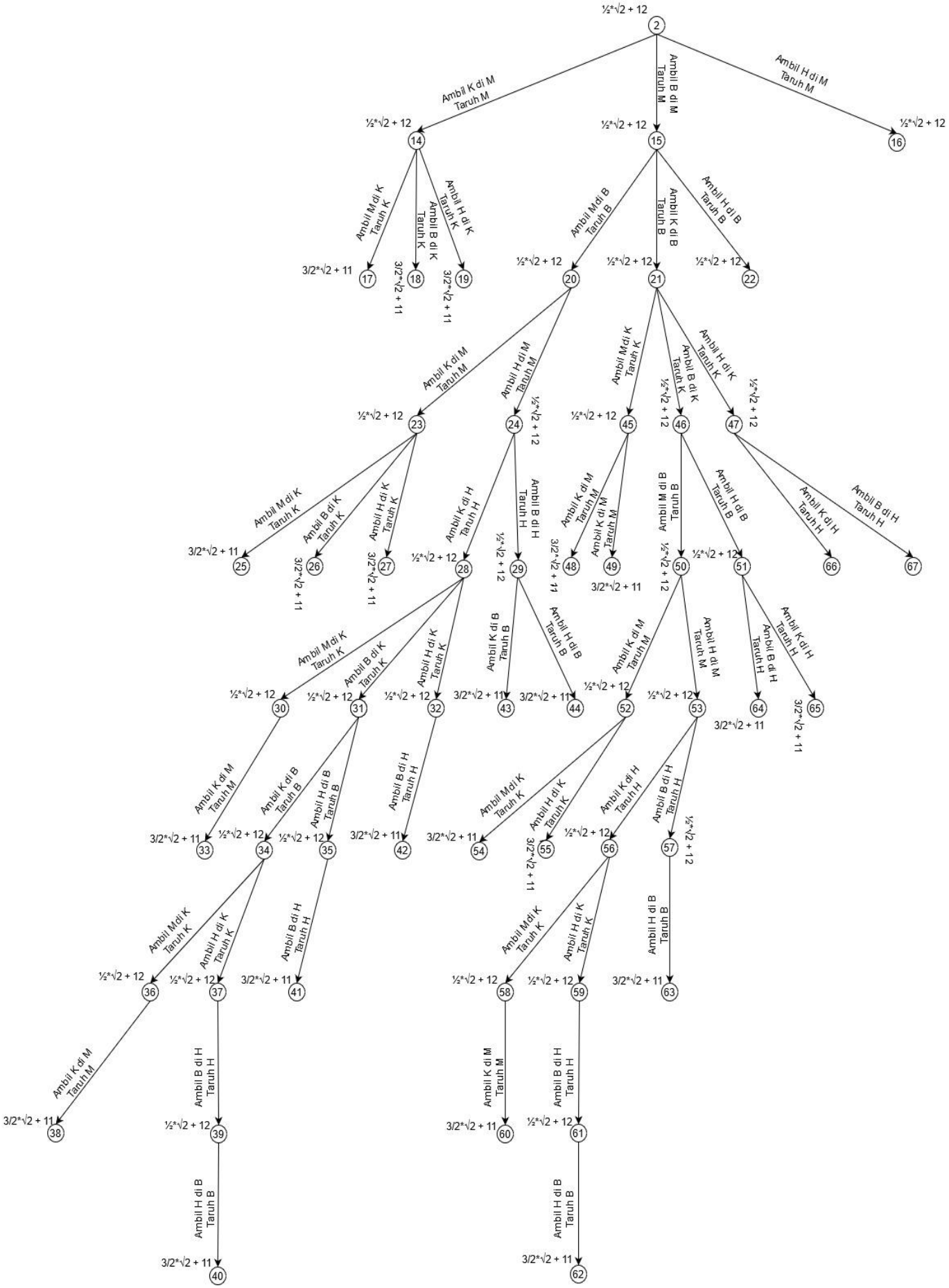


Gambar 11. Penghidpuan tiga simpul baru kedua

Tiga simpul yang baru dihidupkan ini memiliki *cost* yang lebih besar dari lainnya sehingga lanjut ke pengekspanan simpul 15. Mulai dari Langkah pengekspanan ini tidak akan dijelaskan tahap per tahapnya lagi karena mirip-mirip saja dengan langkah-langkah ekspan simpul sebelumnya. Gambar 12 menunjukkan hasil dari pengekspanan beberapa simpul yang dilakukan tetapi belum selesai dan tersisa masih banyak iterasinya untuk mencapai satu *goal*.



Gambar 9. Penghidupan simpul-simpul awal



Gambar 12. Graf Hasil Pengekspanan Beberapa Simpul yang Belum Ketemu Simpul *Goal*

Rupanya ada banyak solusi yang ditemukan dari hasil iterasi yang sangat banyak. Langkah-langkah dari salah satu solusinya adalah:

1. Berjalan ke kubik hijau dan ambil kubik merah di atas kubik hijau.
2. Berjalan ke kubik merah, ambil kubik biru di atas kubik merah dan menaruh kubik merah.
3. Berjalan ke kubik biru, ambil kubik merah di atas kubik biru dan menaruh kubik biru.
4. Berjalan ke kubik merah, ambil kubik hijau di atas kubik merah dan menaruh kubik merah.
5. Berjalan ke kubik hijau, ambil kubik kuning di atas kubik hijau dan menaruh kubik hijau.
6. Berjalan ke kubik kuning, ambil kubik biru di atas kubik kuning dan menaruh kubik kuning.
7. Berjalan ke kubik biru, ambil kubik kuning di atas kubik biru dan menaruh kubik biru.
8. Berjalan ke kubik kuning, ambil kubik hijau di atas kubik kuning dan menaruh kubik kuning.
9. Berjalan ke kubik hijau, ambil kubik biru di atas kubik hijau dan menaruh kubik hijau.
10. Berjalan ke kubik biru, ambil kubik hijau di atas kubik biru dan menaruh kubik biru.
11. Berjalan ke kubik hijau dan menaruh kubik hijau.
12. Berjalan ke kubik merah dan ambil kubik kuning di atas kubik merah,
13. Berjalan ke kubik kuning, ambil kubik merah di atas kubik kuning dan menaruh kubik kuning.
14. Berjalan ke kubik merah dan menaruh kubik merah. Program berhenti.

Cost dari simpul *goal* ini adalah 9/2*√2 + 9 karena robot sudah berjalan sebesar 9/2*√2 + 9 x meter dan sudah tidak ada kubik kecil yang tidak sesuai posisinya. *Cost* untuk beberapa solusi lain juga sama besar *cost*-nya.

Analisis dari hasil solusi yang didapatkan ini adalah tidak *worth it* jika memakai Algoritma *Branch and Bround* jika nilai x-nya kecil atau kira-kira bernilai satu atau dua meter karena dengan menggunakan algoritma DFS bisa lebih cepat selesai tanpa memperdulikan masalah optimal karena lebih sedikit waktunya dalam memproses algoritmanya.

V. KESIMPULAN

Algoritma *Branch and Bound* bisa menyelesaikan masalah yang berhubungan dengan optimisasi. Akan tetapi untuk beberapa persoalan atau kasus, algoritma *Branch and Bound* bisa membuat lambat suatu *bot* untuk melakukan sesuatu karena *bot*-nya harus menganalisa dengan sangat lama untuk menghasilkan nilai yang optimal sedangkan kalau robot menggunakan Algoritma DFS atau lainnya kemungkinan robot sudah langsung selesai memproses suatu persoalan.

UCAPAN TERIMA KASIH

Pertama-tama, penulis mengucapkan syukur kepada Tuhan yang Maha Esa karena karunia-Nya, penulis dapat menyelesaikan makalah berjudul "Penerapan Algoritma *Branch and Bround* dalam Permainan Penempatan Barang oleh Robot". Ucapan terima kasih juga diberikan kepada Bapak Dr. Ir. Rinaldi Munir selaku dosen mata kuliah IF2211 Strategi Algoritma untuk K3 yang telah membimbing dan memberikan materi mengenai algoritma penyelesaian suatu permasalahan selama proses pembelajaran mata kuliah Strategi Algoritma.

VIDEO LINK AT YOUTUBE

https://youtu.be/p-k1_eIt9M8

REFERENSI

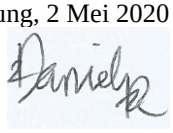
[1] Munir, Rinaldi. Algoritma Greedy (2020). https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2019-2020/Algoritma-Branch-And-Bround-(2020).pdf. Diakses tanggal 1 Mei 2020, pukul 17.03.

[2] https://wro-association.org. Diakses tanggal 1 Mei 2020, pukul 17.36.

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, 2 Mei 2020



Daniel Riyanto
13518075