Aplikasi Algoritma Backtrace pada Simulasi Pemecahan Labirin oleh Robot Micromouse

Dimas Faidh Muzaki - 13520156
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): dimasfaid@gmail.com

***Abstract*— Micromouse merupakan sebuah kontes robot kecil cerdas yang mampu memecahkan sebuah labirin berukuran 16x16. Ada banyak macam pencapaian yang dapat digunakan oleh robot untuk menyelesaikan tugasnya. Salah satunya adalah dengan algoritma backtacking. Pada makalah ini akan dibahas pembuatan simulasi labirin dan robot pada perlombaan micromouse. Robot akan menggunakan algoritma Backtracking dalam melaksanakan tugasnya. Hasilnya, algoritma berhasil diimplementasikan dan dapat diterapkan pada robot *micromouse* sungguhan.**

Keywords—robot; backtracking; labirin; micromouse;

I. PENDAHULUAN

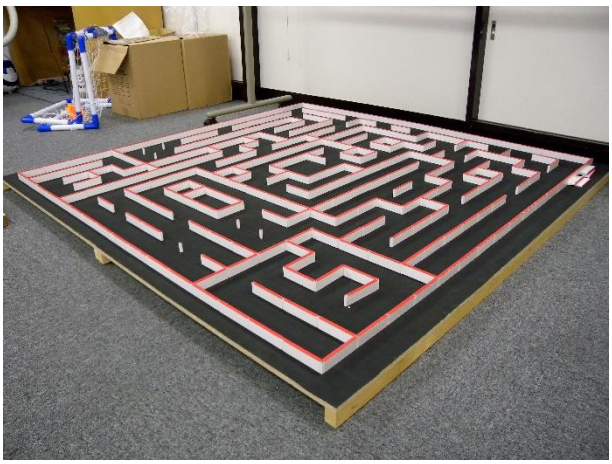


Fig. 1. Contoh labirin micromouse

Micromouse merupakan sebuah jenis perlombaan robot kecil yang populer sejak tahun 1970an. Inti dari perlombaan ini adalah mencipatakan sebah robot kecil dengan panjang dan lebar maksimum 25 x 25 cm yang dapat menelusuri labirin dan menemukan lintasan ke lokasi spesifik pada labirin. Robot tersebut harus fully autonomous atau bergerak dengan sendirinya saat perlombaan tanpa dikontrol oleh peserta lomba.

Arena lomba yang digunakan adalah sebuah labirin berukuran 16 x 16. Tiap petak labirin berukuran 18 x 18 cm yang dibatasi dengan tembok setinggi 5 cm. Tembok-tembok tersebut terbuat dari kayu dengan ketebalan 1.2 cm dan (pada umumnya) dicat warna putih. Lantai dari labirin berwarna pada umumnya berwarna hitam.

[1]Titik awal lintasan robot adalah pada salah satu sudut pada labirin. Titik awal tersebut harus dikelilingi oleh tiga tembok. Sementara itu, titik akhir atau goal yang harus dicapai adalah sebuah ruangan berbentuk persegi yang terdiri dari 4 buah petak. Goal hanya memiliki satu pintu masuk dan umumnya berada di tengah labirin.

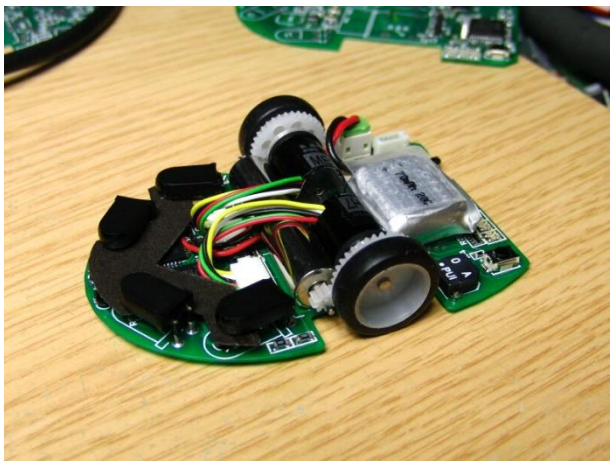


Fig. 2. “Egg Torte” sebuah robot micromouse

Robot yang ditandingkan pada umumnya terdiri dari 3 komponen utama, yaitu sensor, motor, dan microcontroller. Sensor digunakan untuk mendeteksi keberadaan tembok disekitar robot. Hal itu dapat dicapai dengan sensor berjenis infrared, proximity, ataupun ultrasonic sensor. Motor menjadi penggerak roda untuk memungkinkan robot berpindah. Sementara microcontroller berperan sebagai otak dari robot yang mengatur pergerakan robot menggunakan algoritma yang sudah ditentukan. Dengan ketiga komponen tersebut, robot micromouse diharapkan untuk dapat melakukan beberapa pergerakan penting seperti maju, belok ke kiri dan kanan, dan berputar balik.

Pada dasarnya, robot maupun peserta tidak mengetahui konfigurasi labirin yang harus dipecahkan. Oleh sebab itu, pemecahan labirin dilakukan bersamaan dengan pemetaan labirin oleh robot ke dalam penyimpanannya.

II. Landasan Teori

A. *Backtracking*

Algoritma Backtracking merupakan sebuah teknik-algoritma pemecahan masalah dengan pendekatan pembangunan solusi secara *incremental*. Algoritma ini merupakan perbaikan dari *exhautive search* [2]. Pada *exhaustive search*, semua kemungkinan solusi dieksplorasi dan dievaluasi satu per satu. Sedangkan pada algoritma *backtracking*, pilihan yang dieksplorasi hanya pilihan yang mengarah ke solusi. Sementara itu, pilihan yang gagal memenuhi *constraint* permasalahan akan dieliminasi (*pruning)*. Algoritma ini pertama kali diperkenalkan oleh D.H. Lehmer pada tahun 1950. Ada tiga tipe permasalahan yang dapat dipecahkan oleh algoritma *backtracking*, yaitu masalah keputusan (decision problem – mencari solusi yang mungkin), masalah optimisasi (mencari solusi terbaik), dan masalah enumerasi (mencari semua solusi yang mungkin).

Dalam algoritma *backtracing* terdapat tiga properti umum, yaitu:

- Solusi persoalan yang dinyatakan sebagai vector n-*tuple* X = (x1,x2,x3,...,xn) dengan xi anggota Si
- Fungsi pembangkit, yaitu predikat untuk membangkitkan nilai untuk xk yang merupakan komponen vector solusi
- Fungsi pembatas, yaitu predikat yang memberikan nilai kebenaran atas suatu vector jika mengaarah ke solusi. Pembangkitan nilai akan dilanjutkan hanya jika fungsi pembatas memberikan nilai benar.

Semua kemungkinan solusi dari suatu permasalahan dapat disebut sebagai ruang solusi. Ruang solusi pada umumnya dapat diorganisasikan ke dalam bentuk pohon berakar. Simpul pada pohon menyatakan status pencarian solusi permasalahan sedangkan sisi pohon menyatakan nilai/keputusan yang dilakukan untuk mencapai status tersebut. Oleh sebab itu, lintasan dari akar menuju daun akan menyatakan sebuah solusi. Pengorganisasian pohon ruang solusi diacu sebagai pohon ruang status. Terdapat tiga jenis simpul pada pohon yang dibangun, yaitu:

- Simpul ekspan, simpul yang sedang diperluas.
- Simpul hidup, simpul yang sudah dibangkitkan.
- Simpul mati, simpul yang tidak mengarah ke solusi sehingga “dimatikan”

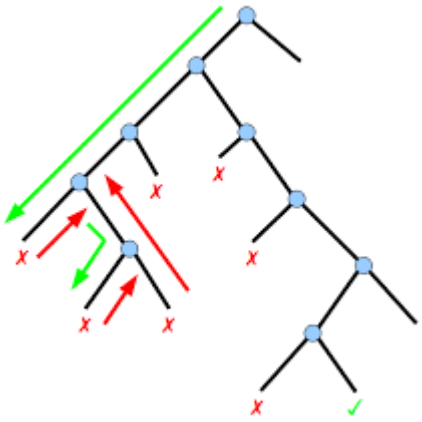


Fig. 3. Skema pencarian pada algoritma backtracking

Pada dasarnya, pembangkitan simpul pada algoritma ini mengikuti aturan *Depth First Search* (DFS). Ekspansi simpul berimplikasi pada penambahan Panjang lintasan. Simpul yang tidak mengarah ke solusi akan dimatikan oleh fungsi pembatas dan kembali ke simpul orang tua dari simpul yang telah dimatikan. Ketika itu juga, kita telah melaksanakan *pruning.*

III. Pembahasan

A. *Simulasi Robot dan Labirin*

Pada dasarnya, perlombaan *micromouse* merupakan sebuah ajang untuk menyelesasikan *Rat in a Maze problem.* Pada permasalahan ini, kita diberikan sebuah labirin berukuran N*N dengan titik asal koordinat berada pada pojok kiri atas. Pada umumnya, pemecahan labirin dengan backtracing akan memanfaatkan konsep rekursi. Namun, hal itu tidak akan cocok pada *micromouse.* Hal ini disebabkan oleh proses *backtracing* oleh robot memerlukan ia untuk berjalan kembali ke status sebelumnya tidak seperti proses rekursi yang otomatis oleh thread. Selain itu, robot micromouse juga memiliki constraint seperti hanya dapat bergerak maju dan berputar arah. Maka, terdapat *effort* lebih yang harus dilakukan robot yang tidak perlu dilakukan oleh rekursi.

Berdasarkan pertimbangan kurangnya sumber daya dan waktu yang memadai, penulis memutuskan implementasi dan pembahasan bukan dilakukan langsung pada sebuah robot *micromouse*, melainkan akan dibuat sebuah simulasi yang merepresentasikan robot dan labirin. Simulasi akan dibuat menggunakan sebuah program berparadigma pemrograman berbasis objek dengan bahasa Java.

Program memiliki sebuah kelas dengan nama Mouse yang akan merepresentasikan robot. Interface yang dibuat untuk kelas tersebut akan mengikuti batasan yang dimiliki robot. Hal ini termasuk pergerakan robot yang hanya bergerak maju dan berputar arah.

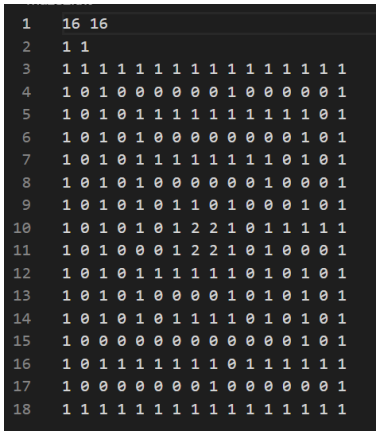


Fig. 4. Contoh konfigurasi labirin

Labirin akan direpresentasikan menjadi sebuah kelas bernama Maze. Labirin dapat berukuran N*M serta memiliki informasi mengenai koordinat serta orientasi arah robot pada saat memulai. Konfigurasi labirin dapat diatur menggunakan sebuah text file dengan format baris pertama adalah dimensi, baris kedua adalah titik koordinat awal, lalu diikuti oleh labirin dengan 1 adalah tembok, 0 adalah jalan, dan 2 adalah finish.

Labirin beserta robot dapat divisualisasikan melalui *command line interface* (CLI). Sebagai contoh, lihat gambar berikut:

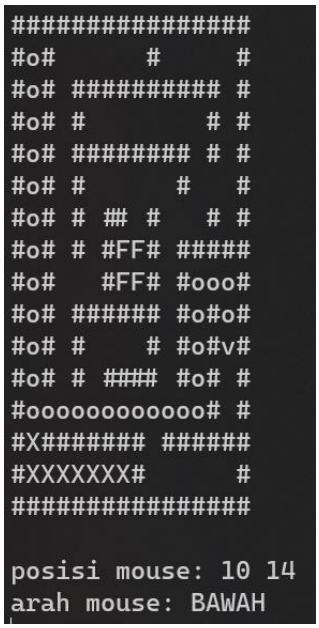


Fig. 5. Contoh visualisasi

Pada gambar di atas dapat dilihat terdapat beberapa karakter yang memiliki makna masing masing seperti:

- #, tembok labirin
- ‘ ‘, jalan kosong
- o, jalan yang telah dilalui
- X, jalan “mati” yang merujuk ke jalan buntu
- F, finish
- v, robot micromouse. Adapun karakter lain yang menunjukkan arah orientasi robot seperti:
    - ^, robot mengarah ke atas
    - v, robot mengarah ke bawah
    - >, robot mengarah ke kanan,
    - <, robot mengarah ke kiri

Dengan pengetahuan tersebut beserta visualisasi labirin pada suatu status, kita dapat memetakannya ke dalam pohon status. Kita dapat melihat bahwa, titik awal robot berada merupakan akar dari pohon. Jalan dengan karakter X merupakan simpul mati karena tidak mengarah ke finish. Simpul ekspan adala karakter dari robot sesuai dengan arahnya. Lalu simpul hidup dapat digambarkan dengan jalan kosong yang bertetangga dengan jalan yang telah dilalui.

B. Pemecahan Labirin.

1) Mapping Properti Algoritma Backtracking

Berikut adalah mapping permasalahan pemecahan labirin ke dalam properti Algoritma Backtracking:

a) Solusi persoalan pada pemecahan labirin adalah

$$S = \{(x1,y1),(x2,y2),\ldots,(xn,yn)\}$$

Dengan pasangan x dan y adalah koordinat dari petak pada labirin yang dilalui.

b) Fungsi pembangkit pada persoalan ini akan memiliki prioritas seperti memilih jalan yang kosong dengan urutan pemilihan lurus, belok kiri, dan belok kanan. Urutan ini akan digunakan pada saat robot menemukan sebuah persimpangan.

c) Fungsi pembatas akan membatasi bahwa simpul yang dibangkitkan harus menuju ke solusi dan tidak membuat sirkuit. Oleh sebab itu, fungsi pembatas akan memastikan bahwa saat tidak ada lagi jalan yang bisa dilalui ataupun jalan mengarah ke jalan yang sudah dilalui sebelumnya, robot akan memasuki fase backtrace.

2) Konsep Umum

Konsep umum penerapan algoritma Backtracking dalam pemecahan labirin adalah sebagai berikut:

a) Membuat sebuah matriks untuk merepresentasikan labirin. Nantinya matriks tersebut akan menyimpan status dari suatu blok (tembok, jalan, jalan visited, dead).

b) Melakukan pengecekan kondisional sebagai berikut:

- Melihat apakah di depan merupakan jalan kosong atau finish, maju bila benar.
- Jika salah, lihat apakah sebelah kiri merupakan jalan kosong atau finish, putar ke kiri bila benar.
- Jika salah lihat apakah sebelah kiri merupakan jalan kosong atau finish, putar ke kanan apabila benar.

Proses ini merupakan proses pembangkitan karena robot akan berpindah ke block baru dengan cara maju. Block sebelumnya akan ditandai sebagai visited pada matriks.

Selanjutnya, robot akan mengulangi poin b apabila salah satu kondisional terpenuhi.

c) Apabila kondisional b tidak ada yang terpenuhi maka secara implisit ini memiliki arti bahwa robot perlu melakukan backtrace dan lintasan yang sebelumnya dilalui tidak menuju ke solusi. Untuk itu, robot akan memilih untuk berputar balik apabila di belakang robot merupakan blok visited. Jika ternyata di belakangnya adalah dead maka robot akan melihat sekitar mencari path kosong (dengan poin b) ataupun blok visited dan menuju kesana. Blok yang dilewati selama proses backtracking akan dicap sebagai blok dead pada matriks.

d) Pencarian ini akan terus dilakukan sampai robot berada pada blok finish.

3) Implementasi

Dalam pelaksanaannya, algoritma pemecahan akan dilakukan secara traversal. Di dalamnya, algoritma akan memiliki perulangan *while* untuk meneruskan pencarian sampai robot berada di titik finish. Di dalam perulangan *while* terdapat kondisional bercabang untuk menyesuaikan tindakan robot dengan orientasi posisi serta keadaan sekitar. Secara sederhana, berikut merupakan diagram alur dari algoritma backtracing.

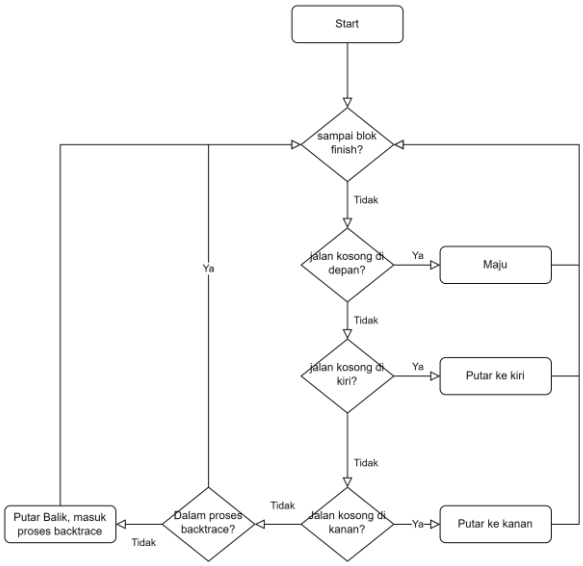


Fig. 6. Diagram alur

C. Uji Implementasi

Pada makalah ini, kita akan menggunakan contoh labirin dengan konfigurasi yang sudah disebutkan sebelumnya. Berikut adalah contoh visualisasi awal dari labirin.

Fig. 7. Iterasi pertama

Gambar di atas merupakan visualisasi dari iterasi pertama pemecahan labirin. Dapat dilihat bahwa robot mulai dari posisi kiri atas dengan orientasi arah ke bawah. Beradasarkan algoritma yang telah diimplementasikan, robot akan bergerak maju. Selama pergerakan itu, blok yang dilalui robot akan ditandai dengan karakter o yang akan merepresentasikan bahwa blok tersebut sudah dikunjungi oleh robot.

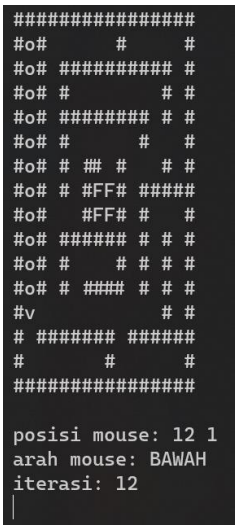


Fig. 8. Iterasi ke-12

Pada iterasi ke-12, robot akan dihadapkan kepada sebuah persimpangan. Pada posisi ini robot memiliki 2 pilihan/simpul hidup dihadapannya, yaitu lurus atau berbelok kiri. Sesuai dengan urutan prioritas yang sudah kita atur sebelumnya, robot akan memilih untuk bergerak maju ke posisi (13,1).

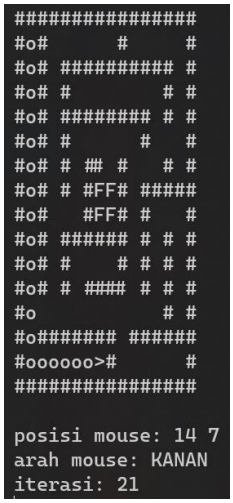


Fig. 9. Iterasi ke-21

Pada iterasi ke-21 robot tidak memiliki jalan yang dapat dilewati di depan maupun di samping kanan dan kiri robot. Oleh sebab itu robot akan memulai proses backtracking. Tidak hanya oleh tembok, proses backtracking juga dapat dimulai apabila jalan yang tersedia merupakan blok yang sudah dikunjungi. Pada proses backtracking robot akan memiliki prioritas sebagai berikut:

TABLE I. TABEL BACKTRACE

Keadaan	Tindakan
Tembok atau visited Tembok atau visited ⬆ Tembok atau visited visited	Putar Balik
visited visited ⬆ visited Dead	maju-kiri-kanan (prioritas sesuai urutan).

Maka dari itu, robot yang kita miliki akan bergerak menjadi sampai iterasi ke-30 sebagai berikut.

Fig. 10. Iterasi ke-30

Pada posisi ini, robot sudah tidak lagi dalam fase backtrace karena menemukan simpul hidup di sebelah kanannya. Robot akan memilih untuk berbelok ke kanan.

Fig. 11. Iterasi ke-91

Pada iterasi ke-95, sisi depan robot merupakan blok visited. Oleh sebab itu, robot akan memutuskan untuk putar balik. Hal ini dilakukan untuk mencegah terjadinya robot bergerak berputar putar pada sebuah sirkuit

Fig. 12. Iterasi ke-111

Pada situasi ini robot akan melakukan bakctrace untuk sampai akhirnya sampai ke blok finish.

Fig. 13. Pencarian selesai

D. Implementasi Pada Robot

Algoritma yang diterapkan pada simulasi seharusnya sudah dapat digunakan oleh robot *micromouse* asli. Namun, masih ada beberapa hal yang perlu diperhatikan. Sebagai contoh, perpindahan robot pada simulasi dapat dengan mudah kita implementasikan menggunakan matriks. Padahal, pada robot asli kita perlu memikirkan tentang panjang lintasan, sudut putaran, gesekan dengan lantai, dan lain-lain. Selain itu, objek Mouse pada program simulasi memiliki atribut yang menyimpan konfigurasi labirin. Seharusnya, robot tidak memiliki informasi seperti itu. Robot perlu menelusuri sambil menggambar denah peta labirin sendiri.

E. Improvisasi

Dengan algoritma yang sudah ada, kita hanya mencari satu kemungkinan lintasan dari titik awal menuju tujuan. Pengembangan lebih lanjut dapat dilakukan untuk backtrace lebih mendalam demi menemukan kemungkinan solusi lain yang mungkin lebih efektif.

Gerakan yang harus dilakukan oleh robot dapat disimpan ke dalam sebuah stack. Dengan begitu, robot tidak perlu melakukan komputasi pada saat melakukan telusur balik dan dalam mengeksekusi lintasan paling pendek. Dengan stack, robot dapat mengimplementasikan beberapa trik untuk mempercepat pergerakan. Sebagai contoh jika pada stack terdapat beberapa jalan lurus berurutan, maka robot dapat meningkatkan kecepetan untuk melewatinya. Contoh lain, apabila robot menemukan lintasan zigzag dan ukuran robot cukup kecil, robot dapat melaluinya hanya dengan berputar ke derajat yang benar dan bergerak lurus dengan cepat.

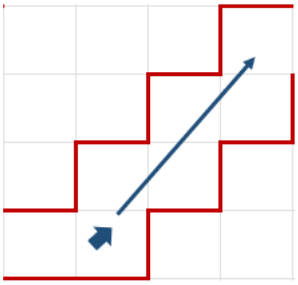


Fig. 14. Lintasan ZigZag

IV. KESIMPULAN DAN SARAN

Berdasarkan simulasi yang dilakukan, algoritma Backtracking dapat diterapkan pada robot *micromouse* untuk menemukan lintasan solusi labirin. Algoritma ini cukup mudah untuk dipahami. Namun, masih ada yang dapat dikembangkan lebih lanjut dalam hal penemuan semua kemungkinan lintasan yang ada atau setidaknya lintasan terbaik. Selain itu, simulasi belum sepenuhnya mengikuti peraturan yang terdapat pada perlombaan *micromouse*. Penelitian lebih lanjut mungkin diperlukan untuk memperbaiki kekurangan yang ada pada penelitian ini.

V. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa atas berkah dan rahmatnya, penulis dapat menulis makalah ini dapat diselesaikan sebenar-benarnya. Terima kasih kepada seluruh keluarga, teman, kerabat, dan pasangan penulis atas support yang diberikan selama menjalani perkuliahan. Terima kasih kepada jajaran dosen mata kuliah IF2211 Strategi Algoritma terutama kepada Dr. Ir. Rinaldi Munir, M.T. atas ilmu yang diberikan. Tak lupa asisten-asisten mata kuliah IF2211 yang telah membantu keberjalanan kuliah selama semester ini.

LINK REPOSITORI GITHUB

https://github.com/maspaitujaki/MicromouseSimulationBacktrace

REFERENSI

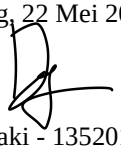
[1] Misra, R., & Adler, R. 2018. Micromouse Competition Rules. 1, 2-4

[2] Munir, Rinaldi. 2009. Diktat Kuliah IF2211 Strategi Algoritmia. Departemen Teknik Informatika IT

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, 22 Mei 2022



Dimas Faidh Muzaki - 13520156