Penyelesaian Permasalahan Labirin dengan Algoritma Branch and Bound

Aplikasi Algoritma Branch and Bound

Hughie Alghaniyyu Emiliano / 13519217
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
Email : 13519217@std.stei.itb.ac.id

***Abstrak*—Labirin merupakan suatu sistem jalur yang dibentuk menjadi sebuah jalur yang rumit dan memiliki jalan buntu. Pemecah permasalahan labirin harus mencari penyelesaian dari labirin tersebut yaitu mencari jalur yang tepat dari suatu titik yang menjadi tempat untuk masuk ke dalam labirin tersebut menuju titik yang lain yang menjadi tempat untuk keluar dari labirin tersebut. Dalam makalah ini, akan dibahas penggunaan aplikasi dari Algoritma Branch and Bound untuk menyelesaikan permasalahan labirin.**

Kata kunci—Labirin; Algoritma Branch and Bound;

I. PENDAHULUAN

Labirin merupakan sebuah teka-teki yang berhubungan dengan jalur. Labirin merupakan sebuah kumpulan jalur yang dibentuk secara rumit, berliku-liku, dan memiliki jalan buntu yang menjadi pengecoh dalam pencarian jalan keluar dari labirin tersebut, serta memiliki tempat masuk dan tempat keluar. Labirin seringkali dijadikan suatu permasalahan untuk mengasah ketelitian, meningkatkan kinerja otak, dan mempelajari navigasi dan tata ruang. Untuk menyelesaikan permasalahan suatu labirin, seseorang perlu untuk mencari sebuah solusi dari permasalahan labirin tersebut yaitu suatu jalur yang tepat sehingga dapat menghubungkan antara tempat masuk dan tempat keluar.

Permasalahan labirin dapat diselesaikan dengan berbagai jenis cara dan algoritma yang dapat mempermudah seseorang untuk menyelesaikan atau mencari solusi dari permasalahan tersebut. Beberapa algoritma yang dapat membantu menyelesaikan permasalahan labirin yaitu Algoritma Backtracking, BFS, DFS, Branch and Bound, Greedy, dll. Namun, pada makalah ini, penulis akan membahas penyelesaian permasalahan labirin dengan Algoritma Branch and Bound.

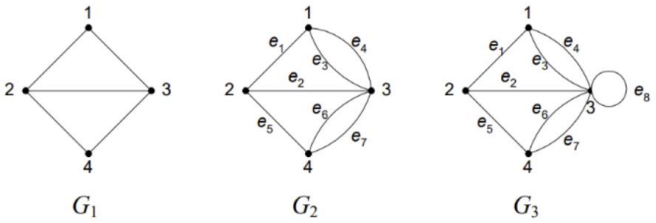
II. LANDASAN TEORI

A. *Graf*

Graf merupakan sebuah cara untuk merepresentasikan objekobjek diskrit dan hubungan antara objek-objek tersebut.

$$G = (V, E)$$

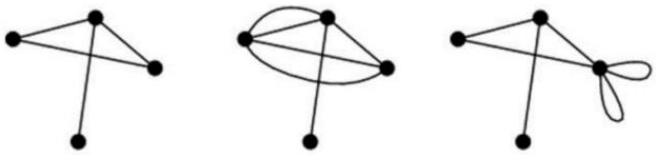
*Keterangan
G = Graf
V = Himpunan tidak kosong dari simpul-simpul
E = Himpunan sisi yang menghubungan sepasang simpul



Gambar 1. Contoh Graf
(Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf)

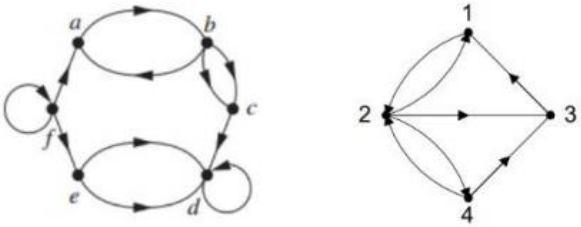
Berdasarkan orientasi arah pada sisi, graf dibedakan atas dua jenis :

1. Graf tak-berarah yaitu graf yang sisinya tidak mempunyai orientasi arah.



Gambar 2. Graf Tak-Berarah
(Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020- 2021/Graf-2020-Bagian1.pdf)

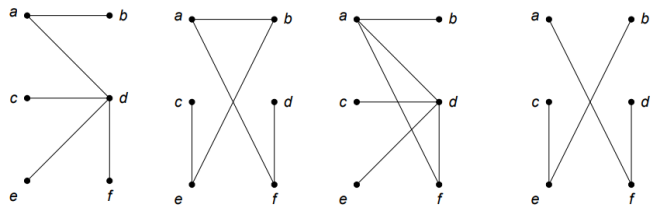
2. Graf berarah yaitu graf yang setiap sisinya diberikan orientasi arah.



Gambar 3. Graf Berarah
(Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf)

B. Pohon

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit.



Gambar 4. Contoh Pohon
(Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf)

Teorema. Misalkan *G = (V, E)* adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini ekivalen :

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan memiliki $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuatnya hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan.

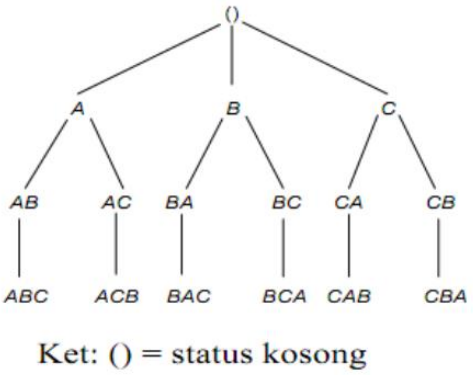
C. Traversal Graf

Traversal Graf merupakan sebuah metode mengunjungi simpul yang ada di dalam suatu graf. Algoritma traversal graf digunakan untuk mengunjungi simpul-simpul dengan cara yang sistematik. Salah satu cara tersebut yaitu pencarian dengan sistem melebar yang digunakan oleh Algoritma *Breadth First Search* (BFS). Dalam implementasinya, traversal graf menjadi algoritma pencari solusi dari sebuah graf terhubung yang merepresentasikan suatu persoalan.

Dalam proses pencarian solusi, graf direpresentasikan dengan dua pendekatan yaitu graf statis dan graf dinamis. Graf statis merupakan graf yang sudah terbentuk sebelum proses pencarian dimulai. Pada graf statis, graf direpresentasikan sebagai struktur data. Graf dinamis merupakan graf yang terbentuk saat proses pencarian sedang dilakukan. Pada graf dinamis, graf direpresentasikan sebagai suatu struktur data.

D. Pohon Ruang Status (Space State Tree)

Pohon ruang status merupakan pengorganisasian semua kemungkinan dari terbentuknya pohon solusi dari suatu permasalahan menjadi suatu himpunan pohon solusi. Setiap simpul dalam pohon ruang status berasosiasi dengan pemanggilan rekursif.

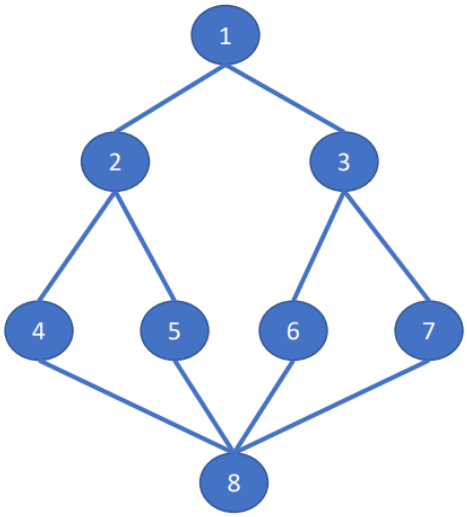


Gambar 5. Pohon Ruang Status Permutasi ABC
(Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag2.pdf)

E. Algoritma Breadth First Search (BFS)

Breadth First Search atau pencarian melebar adalah algoritma yang mengunjungi semua simpul secara preorder yaitu mengunjungi suatu simpul dilanjutkan dengan mengunjungi semua simpul tetangga dengan simpul tersebut terlebih dahulu. Kemudian, dilanjutkan dengan mengunjungi simpul-simpul yang bertetangga dengan simpul-simpul yang telah dikunjungi.

Algoritma ini memanfaatkan struktur data queue atau antrian untuk menyimpan simpul yang telah dikunjungi. Setiap simpul yang telah dikunjungi masuk ke dalam antrian hanya satu kali dan digunakan sebagai acuan untuk mengunjungi simpul-simpul yang bertetangga dengan simpul tersebut.



Gambar 6. Contoh Pencarian dengan Algoritma BFS
(Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf)

Secara umum cara kerja BFS adalah sebagai berikut.

1. Memasukkan simpul pertama ke dalam antrian

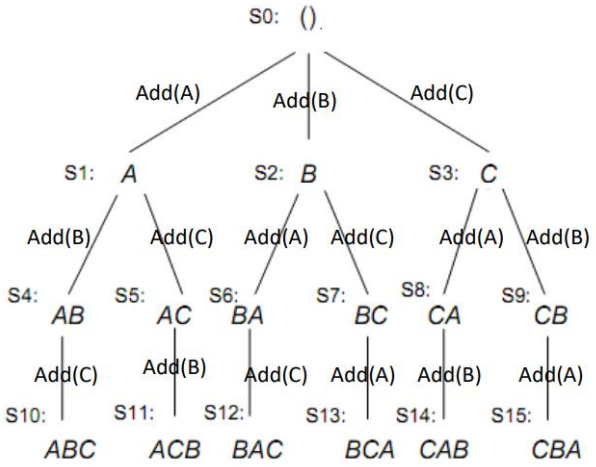
2. Ambil head dari antrian dan cek apabila simpul tersebut adalah solusi
3. Apabila simpul merupakan solusi, maka selesai
4. Apabila simpul bukan merupakan solusi, maka masukkan semua simpul yang bertetangga ke dalam antrian.
5. Lakukan sampai antrian kosong, apabila antrian kosong dan belum menemukan solusi, maka kesimpulannya adalah solusi tidak ada.

Iterasi	V	Q	dikunjungi							
			1	2	3	4	5	6	7	8
Inisialisasi	1	{1}	T	F	F	F	F	F	F	F
Iterasi 1	1	{2,3}	T	T	T	F	F	F	F	F
Iterasi 2	2	{3,4,5}	T	T	T	T	T	F	F	F
Iterasi 3	3	{4,5,6,7}	T	T	T	T	T	T	T	F
Iterasi 4	4	{5,6,7,8}	T	T	T	T	T	T	T	T
Iterasi 5	5	{6,7,8}	T	T	T	T	T	T	T	T
Iterasi 6	6	{7,8}	T	T	T	T	T	T	T	T
Iterasi 7	7	{8}	T	T	T	T	T	T	T	T
Iterasi 8	8	{}	T	T	T	T	T	T	T	T

Urutan simpul2 yang dikunjungi: 1, 2, 3, 4, 5, 6, 7, 8

Gambar 7. Ilustrasi Pencarian Jalur dari Simpul 1 Menuju Simpul 8 dengan Algoritma BFS
(Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf)

Pada implementasi pohon ruang status, BFS memerlukan pembangkitan status. Pembangkitan status baru dilakukan dengan cara mengaplikasikan operator kepada status (simpul) pada jalur yang diinginkan. Jalur dari simpul akar sampai ke simpul tujuan berisi rangkaian operator yang menghasilkan solusi dari permasalahan.



Gambar 8. Algoritma BFS untuk Permutasi ABC
(Sumber : https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag2.pdf)

F. Algoritma Branch and Bound

Algoritma Branch and Bound merupakan sebuah algoritma yang digunakan untuk menyelesaikan suatu permasalahan optimasi, yaitu meminimalkan atau memaksimalkan suatu fungsi objektif yang tidak melanggar suatu syarat dari permasalahan tersebut. Penggunaan Algoritma Branch and Bound merupakan penggabungan dari Algoritma BFS + *least cost search*. Pada Algoritma BFS, simpul-simpul yang akan diekspansi dipilih berdasarkan urutan pembangkitannya (FIFO) sedangkan pada Algoritma Branch and Bound, setiap simpul memiliki nilai *cost*, nilai taksiran jalur dari simpul tersebut menuju simpul tujuan, yang digunakan untuk menjadi nilai acuan sebagai simpul yang mana yang akan dipilih.

Algoritma Branch and Bound menerapkan suatu fungsi pembatas pada algoritmanya. Fungsi tersebut digunakan untuk memangkas jalur-jalur yang dianggap tidak akan mengarah pada solusi.

III. Metode Penyelesaian Masalah

Strategi untuk menyelesaikan permasalahan labirin dengan Algoritma Branch and Bound adalah sebagai berikut. Sebuah permasalahan labirin yang ingin diselesaikan akan dipetakan menjadi titik-titik berukuran 1 petak yang menandakan apakah titik tersebut merupakan sebuah jalan atau tembok (pembatas). Dalam implementasi kode, jalan dapat diilustrasikan dengan garis bawah (_) dan tembok/pembatas dapat diilustrasikan dengan simbol tagar (#). Setelah itu, setiap titik tersebut direpresentasikan dengan graf terhubung yang menghubungkan setiap jalan yang dapat dilalui sehingga simpul graf tersebut merupakan petak jalan dari labirin tersebut dan sisi dari graf tersebut merupakan jalan yang menghubungkan antarpetak dari petak jalan labirin tersebut.

Operasi yang dapat dilakukan oleh algoritma tersebut merupakan gerak ke atas, kanan, bawah, dan kiri, selain jalur yang menuju simpul sebelumnya atau jalur yang mengarah ke tembok. Jadi, semua gerakan yang dipilih yang dioperasikan pada suatu simpul selalu bergerak maju dan menjauh dari pintu masuk. Nilai *cost* yang digunakan yaitu jarak dari simpul akar menuju simpul tersebut, dalam kasus ini setiap simpul berjarak 1 petak. Fungsi pembatas yang digunakan yaitu petak jalan, selain pintu keluar, harus selalu memiliki pilihan jalan selanjutnya (tidak boleh buntu) sehingga jika terdapat jalur yang berakhir pada jalan buntu, jalur tersebut akan dipangkas dari pohon solusi. Prioritas pemilihan simpul yang akan diekspan yaitu berdasarkan implementasi gerak sebelumnya, yaitu atas, kanan, bawah, lalu kiri.

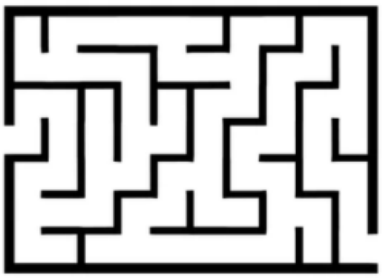
Secara umum, metode penyelesaiannya adalah sebagai berikut.

1. Pilih petak jalan yang merupakan pintu masuk sebagai simpul akar dan petak jalan yang merupakan pintu keluar sebagai simpul tujuan.
2. Lakukan gerak ke atas, kanan, bawah, dan kiri, kecuali jika mengarah ke tembok, pada simpul akar.
3. Hitung nilai cost dari simpul yang dihasilkan dan terapkan fungsi pembatas.

4. Pilih salah satu dari simpul hidup yang tersisa, kemudian lakukan semua gerakan (kecuali gerak yang mengarahkan pada simpul sebelumnya atau mengarah pada tembok) pada simpul tersebut.
5. Hitung nilai cost dari simpul yang dihasilkan dan terapkan fungsi pembatas.
6. Jika simpul-simpul yang dihasilkan bukanlah simpul tujuan (pintu keluar), ulangi kembali langkah 4. Jika gerakan menghasilkan simpul tujuan, pencarian diselesaikan.

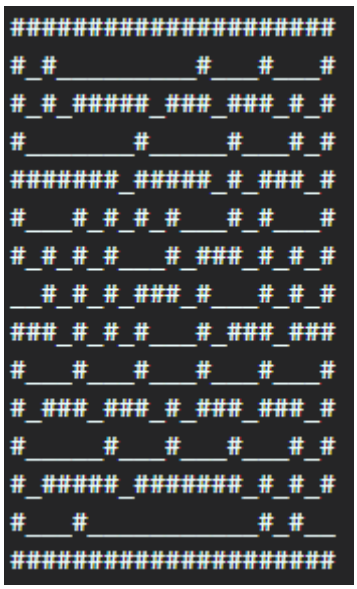
IV. Studi Kasus

Pada bab ini, akan dibahas implementasi Algoritma Branch and Bound pada penyelesaian permasalahan labirin berikut.

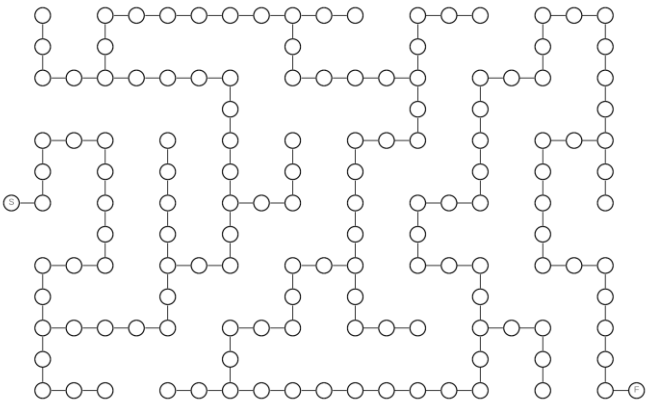


Gambar 9. Permasalahan Labirin
(Sumber : https://www.shutterstock.com/image-vector/illustration-simple-labyrinth-maze-conundrum-kids-1135158989)

Sebelum mengimplementasikan algoritma branch and bound, labirin tersebut akan dipetakan menjadi petak-petak yang berukuran 1 petak dan jalan dari labirin tersebut akan direpresentasikan dengan graf.

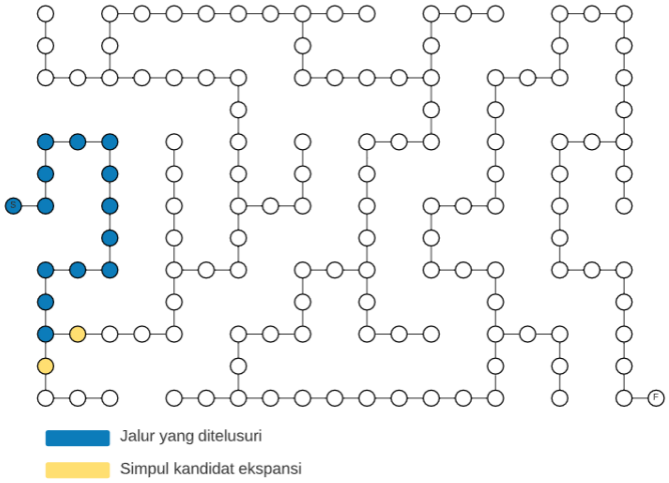


Gambar 10. Pemetaan Labirin dalam Implementasi Kode
(Sumber : Dokumen Penulis)



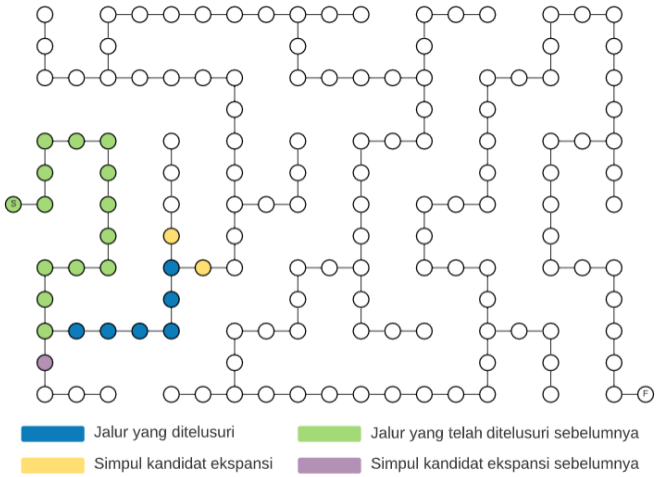
Gambar 11. Pemetaan Labirin dalam Representasi Graf
(Sumber : Dokumen Penulis)

Dari graf tersebut, dipilih simpul S sebagai simpul akar dari pohon Algoritma Branch and Bound dan simpul F sebagai simpul tujuan dari pohon Algoritma Branch and Bound. Karena dalam operasi terdapat *constraint* yaitu gerakan yang dipilih hanyalah gerakan yang mengarah maju, tidak mungkin menabrak tembok ataupun mundur, percabangan penentuan keputusan pada pohon Algoritma Branch and Bound hanya akan terjadi ketika terdapat persimpangan pada jalan atau terdapat suatu simpul yang minimal 3 sisi pada representasi graf. Oleh karena itu, simpul-simpul tersebut dapat ditelusuri hingga menemukan simpul yang memiliki lebih dari 2 sisi.



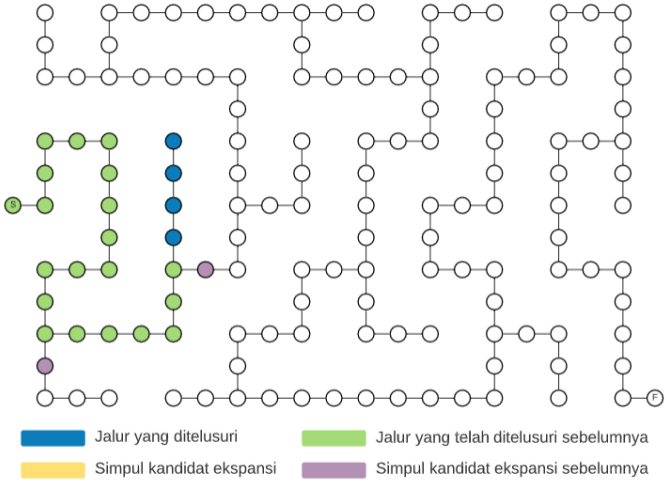
Gambar 12.a Penelusuran Labirin Tahap 1
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.a. Pada penelusuran tersebut, didapatkan cost 13 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, yaitu atas, kanan, bawah, lalu kiri, simpul kanan akan diekspansi selanjutnya.



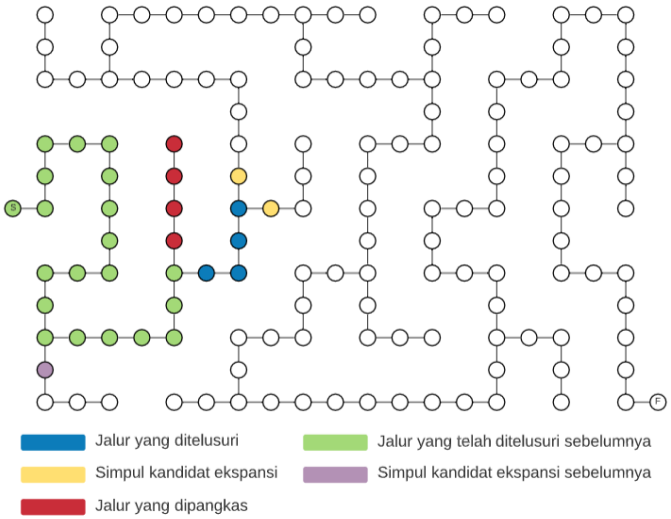
Gambar 12.b Penelusuran Labirin Tahap 2
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.b. Pada penelusuran tersebut, didapatkan cost 19 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul atas akan diekspansi selanjutnya.



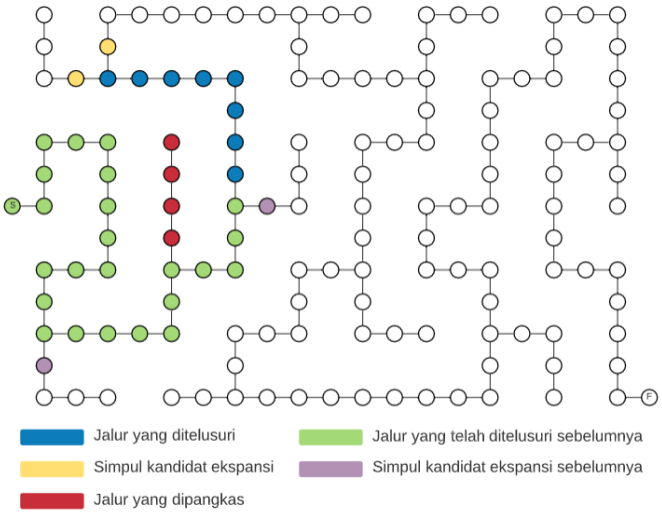
Gambar 12.c Penelusuran Labirin Tahap 3
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.c. Pada penelusuran tersebut, didapatkan cost 23 langkah. Namun, pada tahap ini, jalur tidak memenuhi fungsi pembatas sehingga jalur pada tahap ini perlu dipangkas. Kemudian, penelusuran dilanjutkan melalui penelusuran labirin tahap sebelumnya, yaitu tahap 2, dengan simpul ekspansi selanjutnya, yaitu simpul kanan.



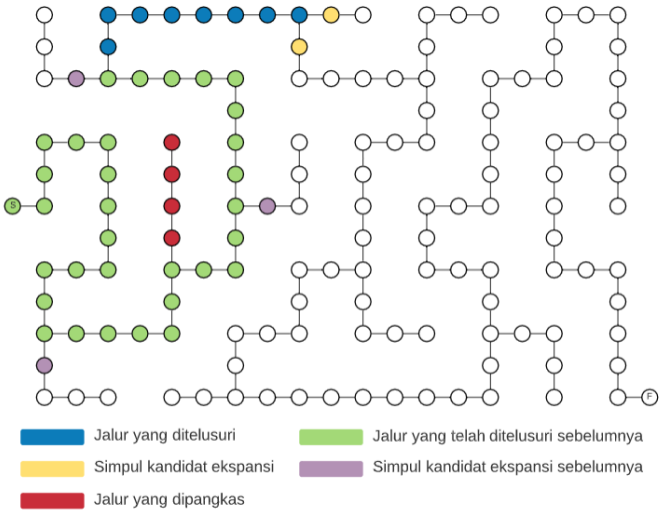
Gambar 12.d Penelusuran Labirin Tahap 4
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.d. Pada penelusuran tersebut, didapatkan cost 23 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul atas akan diekspansi selanjutnya.



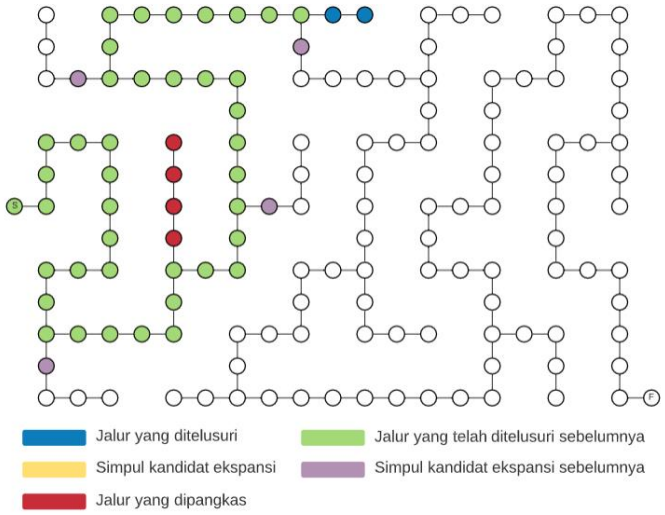
Gambar 12.e Penelusuran Labirin Tahap 5
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.e. Pada penelusuran tersebut, didapatkan cost 31 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul atas akan diekspansi selanjutnya.



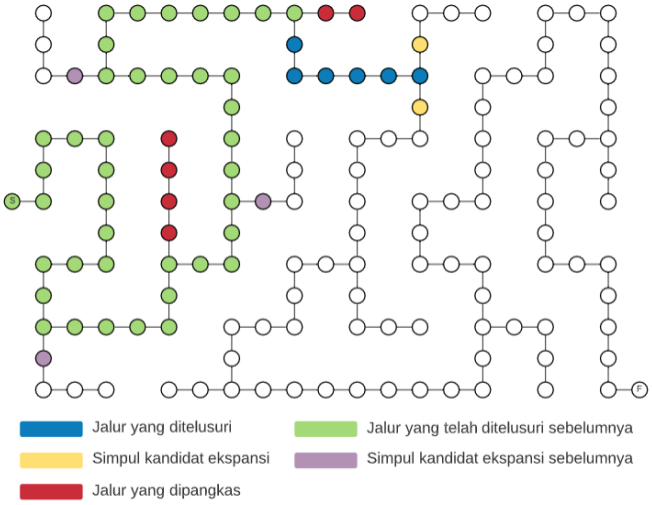
Gambar 12.f Penelusuran Labirin Tahap 6
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.f. Pada penelusuran tersebut, didapatkan cost 39 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul kanan akan diekspansi selanjutnya.



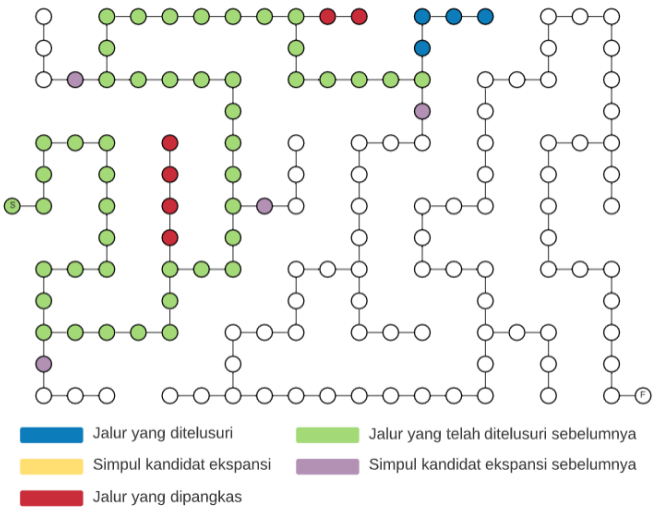
Gambar 12.g Penelusuran Labirin Tahap 7
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.g. Pada penelusuran tersebut, didapatkan cost 41 langkah. Namun, pada tahap ini, jalur tidak memenuhi fungsi pembatas sehingga jalur pada tahap ini perlu dipangkas. Kemudian, penelusuran dilanjutkan melalui penelusuran labirin tahap sebelumnya, yaitu tahap 6, dengan simpul ekspansi selanjutnya, yaitu simpul bawah.



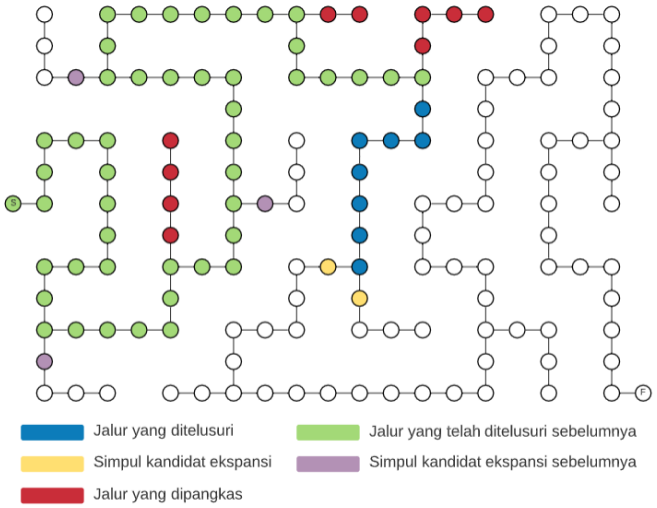
Gambar 12.h Penelusuran Labirin Tahap 8
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.h. Pada penelusuran tersebut, didapatkan cost 45 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul atas akan diekspansi selanjutnya.



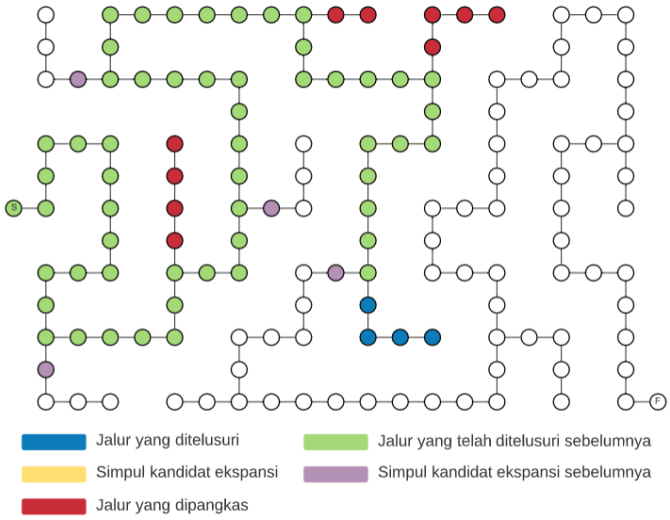
Gambar 12.i Penelusuran Labirin Tahap 9
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.i. Pada penelusuran tersebut, didapatkan cost 49 langkah. Namun, pada tahap ini, jalur tidak memenuhi fungsi pembatas sehingga jalur pada tahap ini perlu dipangkas. Kemudian, penelusuran dilanjutkan melalui penelusuran labirin tahap sebelumnya, yaitu tahap 8, dengan simpul ekspansi selanjutnya, yaitu simpul bawah.



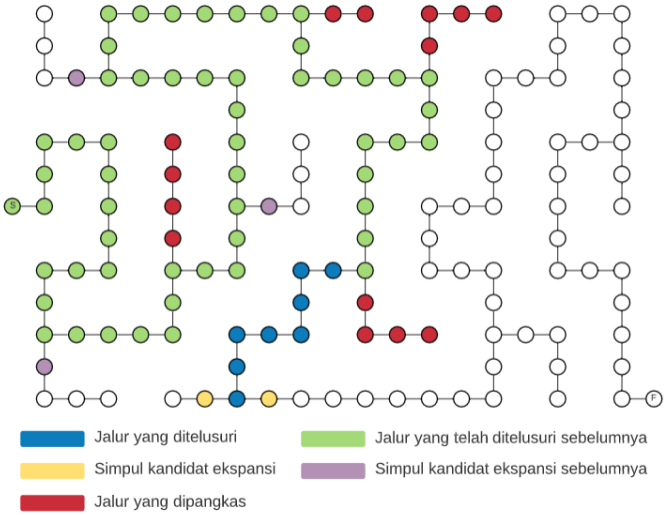
Gambar 12.j Penelusuran Labirin Tahap 10
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.j. Pada penelusuran tersebut, didapatkan cost 53 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul bawah akan diekspansi selanjutnya.



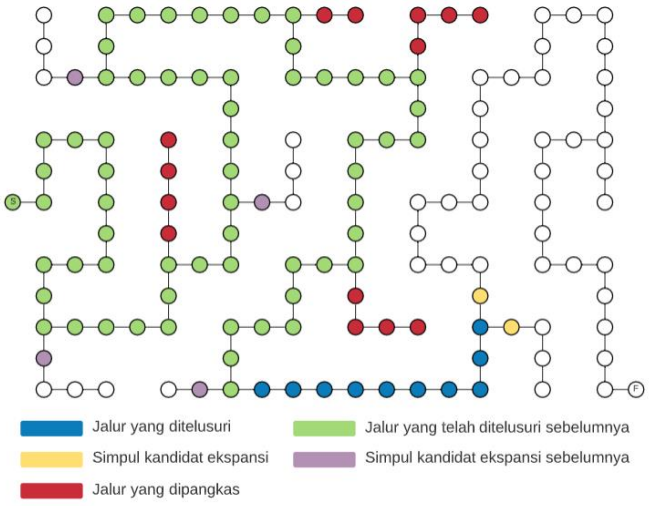
Gambar 12.k Penelusuran Labirin Tahap 11
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.k. Pada penelusuran tersebut, didapatkan cost 57 langkah. Namun, pada tahap ini, jalur tidak memenuhi fungsi pembatas sehingga jalur pada tahap ini perlu dipangkas. Kemudian, penelusuran dilanjutkan melalui penelusuran labirin tahap sebelumnya, yaitu tahap 10, dengan simpul ekspansi selanjutnya, yaitu simpul kiri.



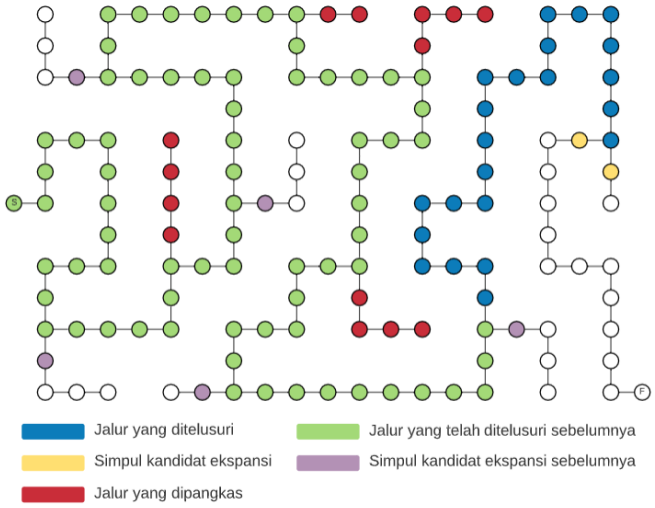
Gambar 12.l Penelusuran Labirin Tahap 12
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.l. Pada penelusuran tersebut, didapatkan cost 61 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul kanan akan diekspansi selanjutnya.



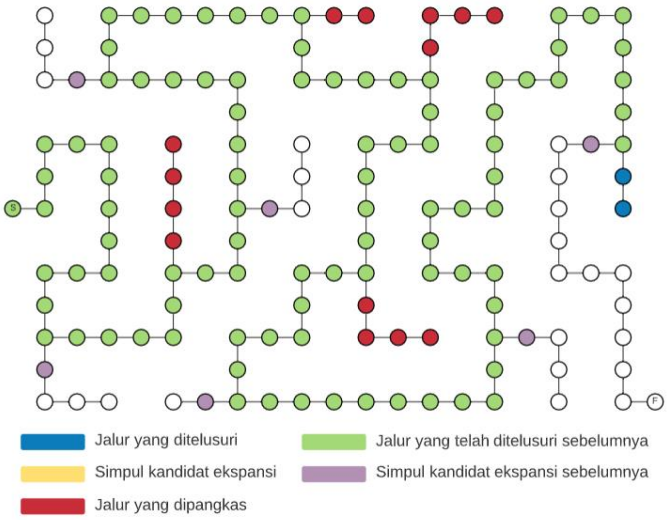
Gambar 12.m Penelusuran Labirin Tahap 13
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.m. Pada penelusuran tersebut, didapatkan cost 71 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul atas akan diekspansi selanjutnya.



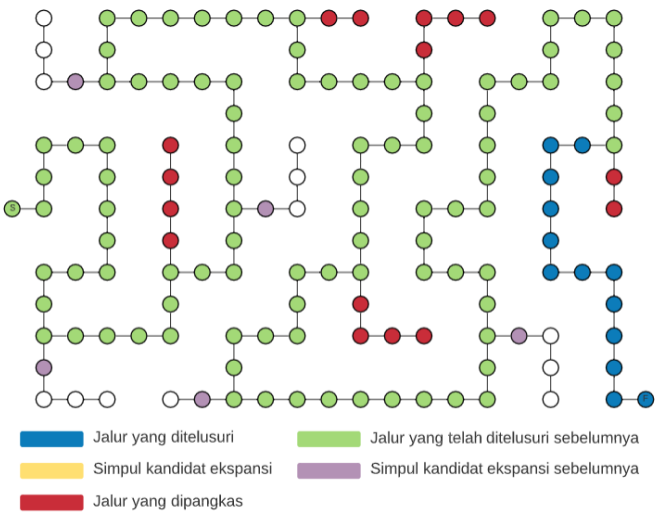
Gambar 12.n Penelusuran Labirin Tahap 14
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.n. Pada penelusuran tersebut, didapatkan cost 93 langkah. Setelah itu, akan ditentukan simpul yang perlu diekspansi selanjutnya. Berdasarkan prioritas, simpul bawah akan diekspansi selanjutnya.



Gambar 12.o Penelusuran Labirin Tahap 15
(Sumber : Dokumen Penulis)

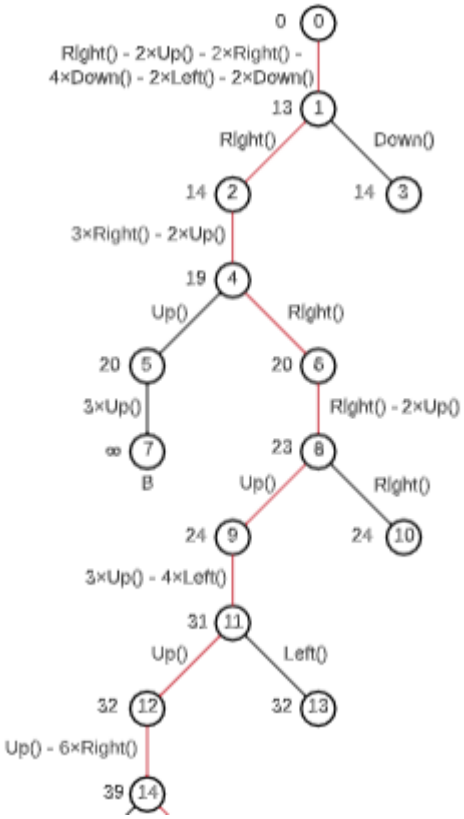
Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.o. Pada penelusuran tersebut, didapatkan cost 95 langkah. Namun, pada tahap ini, jalur tidak memenuhi fungsi pembatas sehingga jalur pada tahap ini perlu dipangkas. Kemudian, penelusuran dilanjutkan melalui penelusuran labirin tahap sebelumnya, yaitu tahap 14, dengan simpul ekspansi selanjutnya, yaitu simpul kiri.



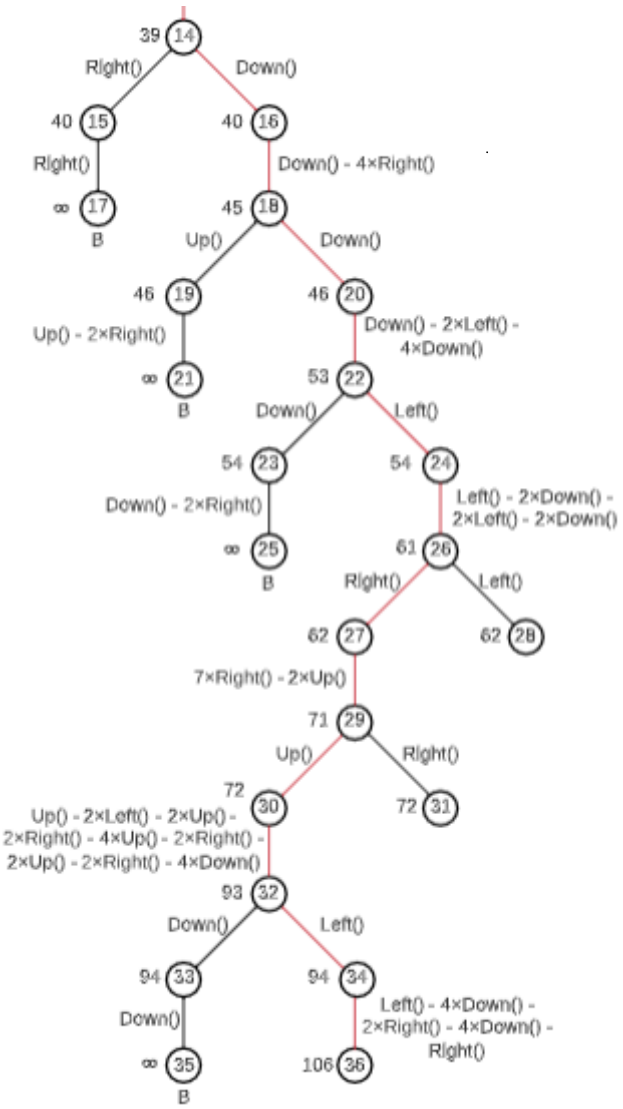
Gambar 12.p Penelusuran Labirin Tahap 16
(Sumber : Dokumen Penulis)

Setelah dilakukan penelusuran, didapat jalur seperti pada Gambar 12.p. Pada penelusuran tersebut, didapatkan cost 106 langkah. Pada tahap ini, jalur berakhir pada pintu keluar sehingga pada tahap ini pohon solusi berhasil mencapai simpul tujuan.

Berdasarkan ke-16 tahap di atas, dihasilkan pohon solusi dari Algoritma Branch and Bound sebagai berikut.



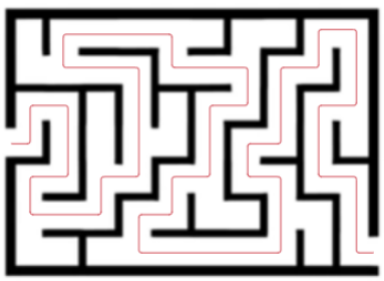
Gambar 13.a. Pohon Solusi Permasalahan Labirin (Bagian 1)
(Sumber : Dokumen Penulis)



Gambar 13.b. Pohon Solusi Permasalahan Labirin (Bagian 2)
(Sumber : Dokumen Penulis)

V. KESIMPULAN

Algoritma Branch and Bound dapat diaplikasikan terhadap permasalahan labirin. Pada studi kasus sebelumnya, Algoritma Branch and Bound berhasil menghasilkan solusi dari permasalahan labirin yang diberikan, yaitu sebagai berikut.



Gambar 14. Solusi Permasalahan Labirin pada Studi Kasus
(Sumber : Modifikasi dari https://www.shutterstock.com/image-vector/illustration-simple-labyrinth-maze-conundrum-kids-1135158989)

VI. UCAPAN TERIMA KASIH

Penulis ingin menyampaikan ucapan syukur kepada Tuhan Yang Maha Esa atas segala rahmat-Nya penulis dapat menyelesaikan makalah ini dengan baik. Penulis pun berterima kasih kepada orang tua penulis yang telah mendukung secara moral dan materil sehingga penulis dapat berkuliah di Institut Teknologi Bandung hingga saat ini. Penulis juga mengucapkan terima kasih kepada dosen mata kuliah Strategi Algoritma, Bapak Dr. Ir. Rinaldi Munir, M.T., yang telah membimbing penulis dalam belajar Strategi Algoritma selama satu semester ini. Terakhir, penulis juga mengucapkan terima kasih kepada teman-teman penulis yang mendukung penulis sehingga pembuatan makalah ini dapat selesai tepat waktu.

REFERENSI

[1] Munir, Rinaldi. 2020. Slide Bahan Kuliah Graf Bagian 1. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf2020-Bagian1.pdf (diakses tanggal 11 Mei 2021)

[2] Munir, Rinaldi. 2020. Slide Bahan Kuliah Pohon Bagian 1. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf (diakses tanggal 11 Mei 2021)

[3] Munir, Rinaldi. 2021. Slide Bahan Kuliah Breadth First Search (BFS) dan Depth First Search (DFS) (Bagian 1). https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag1.pdf (diakses tanggal 11 Mei 2021)

[4] Munir, Rinaldi. 2021. Slide Bahan Kuliah Breadth First Search (BFS) dan Depth First Search (DFS) (Bagian 2). https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/BFS-DFS-2021-Bag2.pdf (diakses tanggal 11 Mei 2021)

[5] Munir, Rinaldi. 2021. Slide Bahan Kuliah Algoritma Runut-Balik (Backtracking) (Bagian 1). https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-backtracking-2021-Bagian1.pdf (diakses tanggal 11 Mei 2021)

[6] Munir, Rinaldi. 2021. Slide Bahan Kuliah Algoritma Branch and Bound (Bagian 1). https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-backtracking-2021-Bagian1.pdf (diakses tanggal 11 Mei 2021)

[7] LUCKY_CAT. Illustration with simple labyrinth, maze conundrum for kids. Baby puzzle with entry and exit. Children riddle game. https://www.shutterstock.com/image-vector/illustration-simple-labyrinth-maze-conundrum-kids-1135158989 (diakses tanggal 11 Mei 2021)

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.

Cirebon, 11 Mei 2021

Hughie Alghaniyyu Emiliano
13519217