Penyelesaian Permainan “Pacman” yang disederhanakan dengan Algoritma Backtracking

Anis Istiqomah
NIM 13505116
Program Studi Tekik Informatika, Institut Teknologi Bandung
e-mail: if15116@students.if.itb.ac.id

ABSTRAK

Pacman adalah *video game* terpopuler sepanjang masa. Game ini dimainkan di lebih dari 100.000 arkade (mesin game berbasis koin) pada tahun pertama rilisnya dan menghasilkan pendapatan sebesar satu miliar dolar A.S. Ide pembuatan game ini sebetulnya sangat sederhana. Karena melihat sepotong pizza yang hilang dalam loyangnya, programmer Toru Iwatani terpikir untuk mengembangkan game pacman yang karakternya mengacu pada pizza di loyangnya.

Pada perkembangannya, budaya pacman, yang selanjutnya dikenal sebagai wabah "Pacmania", semakin menjalar. Pasalnya ini adalah game pertama yang menggunakan tokoh sentral. Setelah itu, karakter pacman muncul di mana-mana. Mulai dari acara televisi, mainan, bungkus makanan, t-shirt, sampai lagu Pacman Fever ciptaan Jerry Buckner dan Gary Garcia pun jadi hit.

Masalah dalam permainan pacman dapat diselesaika dengan berbagai cara. Salah satunya adalah dengan algoritma backtracking. Algoritma ini dipakai agar pacman berhasil memakan semua makanan dalam game.

Kata kunci: Pacman, algoritma, backtracking

1. PENDAHULUAN

Permainan ini mengharuskan pemain (pacman) berjalan di dalam sejenis *maze* atau labirin. Di sepanjang labirin tersebut terdapat semacam makanan yang dapat diambil pacman agar pemain memperoleh nilai. Dalam labirin tersebut juga ada makhluk lain selain pacman, yaitu hantu. Hantu ini merupakan musuh dari pacman. Ada lebih dari satu hantu dalam labirin yang akan mengganggu pergerakan pacman. Hantu-hantu ini berjalan secara random. Jika pacman bertabrakan dengan salah satu hantu ini, maka nilai pemain akan berkurang. Selain makanan yang akan menambah nilai pacman, terdapat bonus makanan yang jika diambil maka pacman akan memperoleh kekuatan lebih. Jika pacman memakan makanan ini, maka semua hantu akan berubah warna yang menunjukkan bahwa hantu-hantu ini dapat dimakan pacman. Pacman juga akan mendapat nilai jika memakan makanan bonus ini atau jika memakan hantu. Jika makanan di sepajang labirin telah habis, permainan akan berlanjut ke level berikutnya. Biasanya ada beberapa level dalam game ini.

Seiring dengan berjalannya waktu, *game* dapat dibuat semakin bagus dengan gambar, warna, dan suara yang menarik. Begitu pula dengan pacman, banyak orang membuat permainan ini dengan tampilan yang berbeda-beda dan lebih menarik daripada ketika pertama kali dibuat.

1.1.Deskripsi Masalah

Dalam permainan ini, pemain harus berjalan melalui labirin dan mengambil makanan di sepanjang labirin tersebut. Selain itu, pemain juga harus menghindari hantu agar tidak bertabrakan. Pemain harus berhati-hati agar tidak terjebak di jalan buntu atau bertabrakan dengan hantu.

Dalam hal ini, permainan akan disederhanakan. Tidak ada hantu atau musuh yang berkeliaran mengganggu pacman. Hal yang harus dilakukan adalah memakan semua makanan yang ada di sepanjang labirin. Permainan selesai jika semua makanan habis.

1.2.Dasar Teori

Algoritma runut-balik (*backtracking*) adalah algoritma yang berbasis pada DFS untuk mencari solusi persoalan secara lebih mangkus. Runut-balik, yang merupakan perbaikan dari algoritma *brute-force*, secara sistematis mencari solusi persoalan di antara semua kemungkinan solusi yang ada. Algoritma ini tidak memeriksa semua kemungkinan solusi yang ada, hanya pencarian yang mengarah ke solusi saja yang selalu dipertimbangkan. Semua kemungkinan solusi dari persoalan disebut ruang solusi. Untuk memfasilitasi pencarian ini, maka ruang solusi diorganisasikan dalam struktur pohon. Lintasan dari akar ke daun menyatakan solusi yang mungkin. Seluruh lintasan dari akar ke daun membentuk ruang solusi.

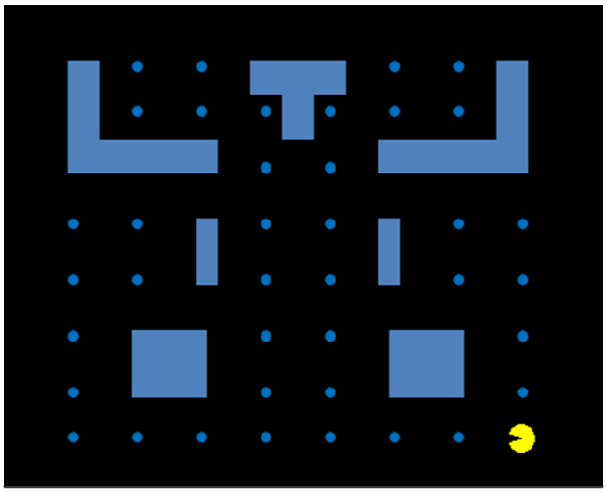
Untuk menerapkan algoritma runut-balik pada pencarian solusi, hanya akan ditinjau pencarian solusi pada pohon ruang status yang dibangun secara dinamis :

1. Solusi dicari dengan membentuk lintasan dari akar ke daun dengan aturan pembentukan pohon yang dipakai mengikuti metode DFS. Simpul-simpul yang sudah dilahirkan dinamakan simpul hidup. Simpul hidup yang sedang diperluas dinamakan simpul-E (*Expand-node*).
2. Jika lintasan yang sedang dibentuk tidak mengarah ke solusi, maka simpul-E menjadi simpul mati (*dead node*) dengan fungsi pembatas (*bounding functon*).
3. Jika pembentukan lintasan berakhir dengan simpul mati, maka proses pencarian diteruskan dengan membangkitkan simpul anak yang lain. Bilai tidak ada lagi simpul anak yang dapat dibangkitkan, maka pencarian solusi dilanjtkan dengan melakukan runut balik ke simpul hidup terdekat.
4. pencarian dihentikan bila kita telah menemukan solusi (solusi ditemukan) atau tidak ada lagi simpul hidup untuk runut-balik (solusi tidak ditemukan).
   Saat ini algoritma runut-balik banyak diterapkan untuk program *games* (seperti permainan *tic-tac-toe*, menemukan jalan keluar dalam sebuah labirin, catur, dll) dan masalah-masalah pada bidang kecerdasan buatan (*artificial intelligence*). Algoritma ini akan dipakai dalam penyelesaian permainan pacman.

2. METODE

Untuk menyelesaikan masalah ini, digunakan algoritma backtracking. Pacman akan berjalan di dalam labirin. Jika menemui jalan buntu atau jalan yang akan dilalui sudah tidak memiliki makanan, maka pacman akan melakukan backtracking ke langkah sebelumnya sampai ditemukan jalan yang belum dilalui. Hal ini terus dilakukan sampai pacman memakan semua makanan yang ada di sepanjang labirin.

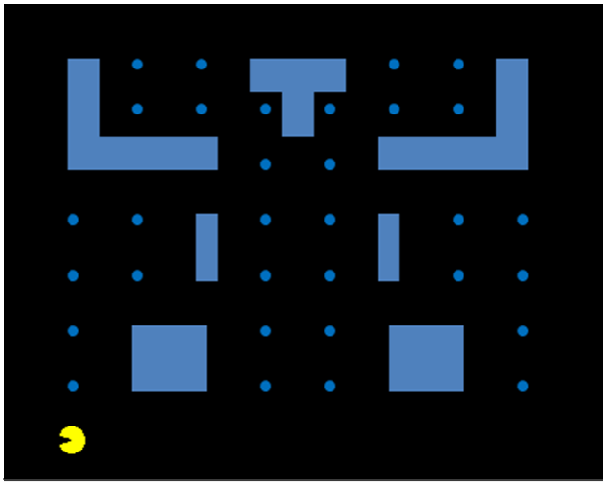
Di bawah ini adalah contoh permasalahan dalam permainan pacman yang disederhanakan



Gambar (1) Contoh persoalan Game Pacman

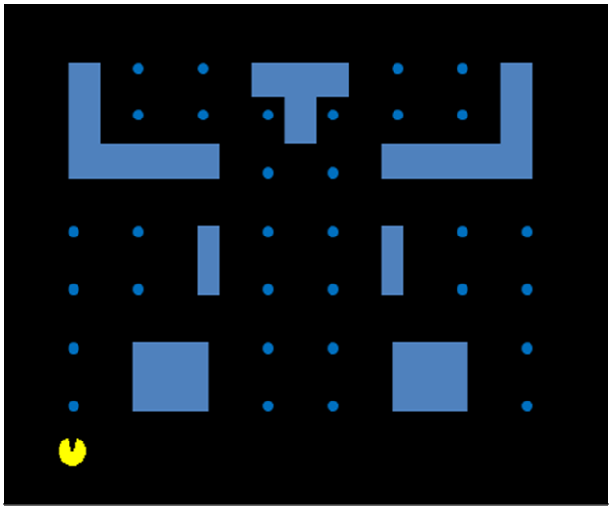
Pacman harus berjalan melalui labirin dan memakan semua makanan (titik-titik). Jumlah titik yang harus dirmakan akan diketahui untuk mempermudah penyelesaian.

Penyelesaian :



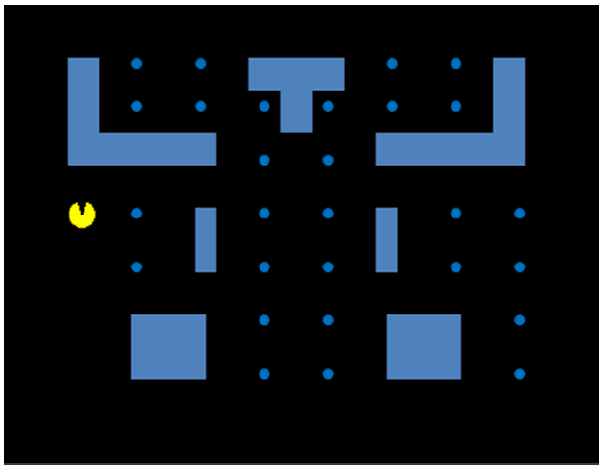
Gambar (2) Pacman mulai berjalan

Pacman akan berjalan lurus sampai ujung dan terhalang tembok.



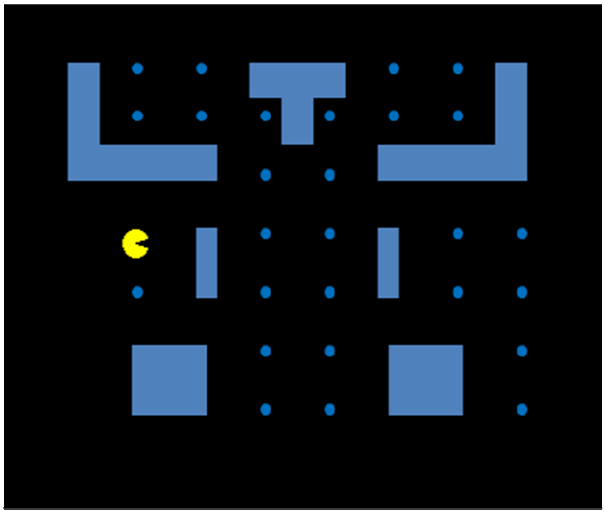
Gambar (3) Pacman berganti arah

Pacman akan mencari arah lain yang dapat dilalui. Untuk kasus ini pacman akan memilih arah ke atas atau sisi sebelah kanan pacman karena arah itu yang paling mungkin untuk dilewati.



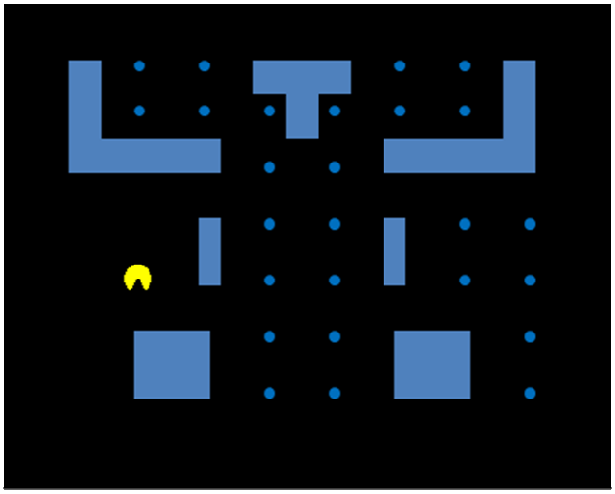
Gambar (4) Pacman terhalang tembok

Pacman akan kembali terhalang oleh tembok.



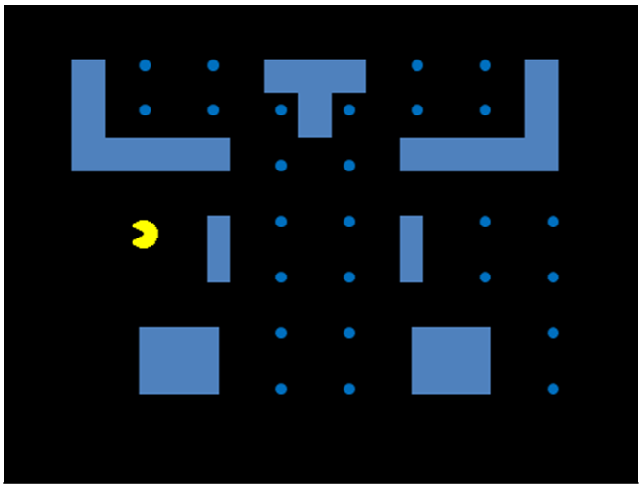
Gambar (5) Pacman berganti arah

Pacman memilih arah lain dan terhalang tembok lagi.



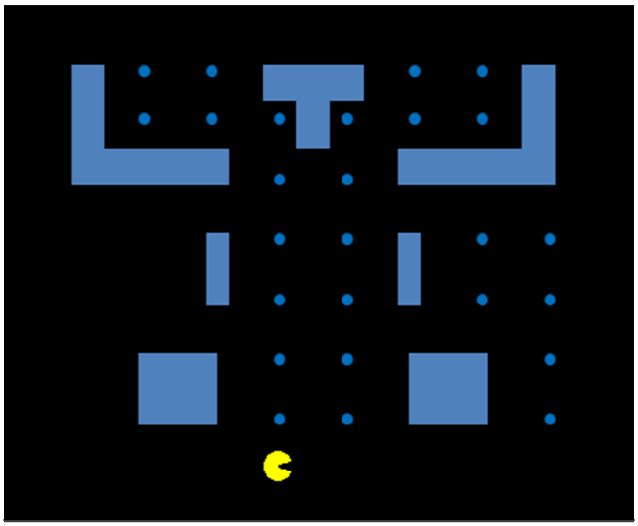
Gambar (6) Pacman memilih arah lain

Pacman berganti arah lagi. Sampai disini, pacman menghadapi tembok lagi. Pilihan arah disini adalah ke kiri atau atas.



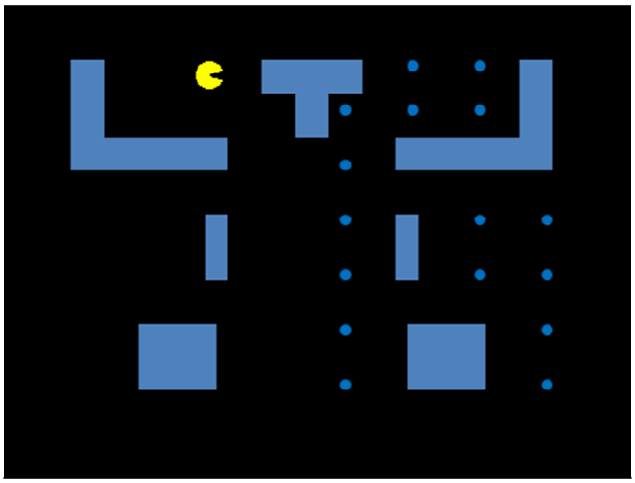
Gambar (7) Pacman memulai backtracking pertama

Pacman melakukan backtracking. Dengan algoritma backtracking, pacman akan berbalik ke arah sebelumnya dan berjalan sampai menemukan arah lain yang belum dan dapat dilalui.



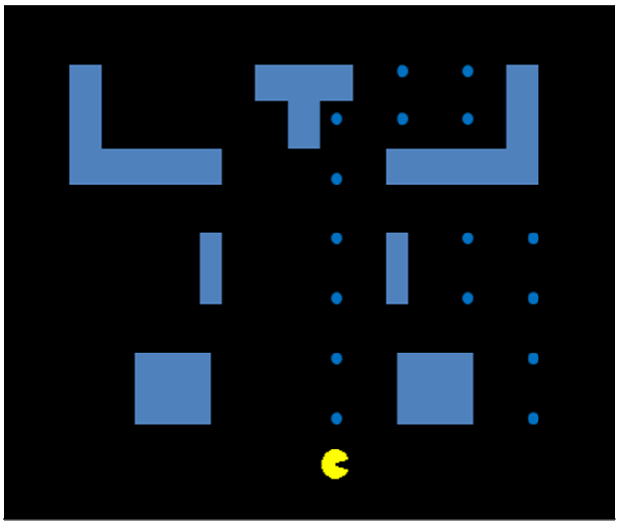
Gambar (8) Akhir dari backtracking pertama

Pacman selesai melakukan backtracking karena sudah menemukan arah lain yang belum dilewati.



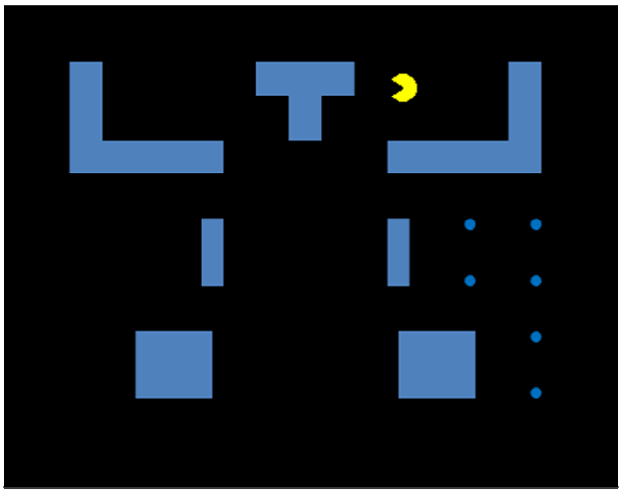
Gambar (10) Pacman melanjutkan perjalanan

Pacman melewati arah yang dipilih tersebut sampai ujung. Pacman harus melakukan backtracking lagi untuk melanjutkan perjalanan.



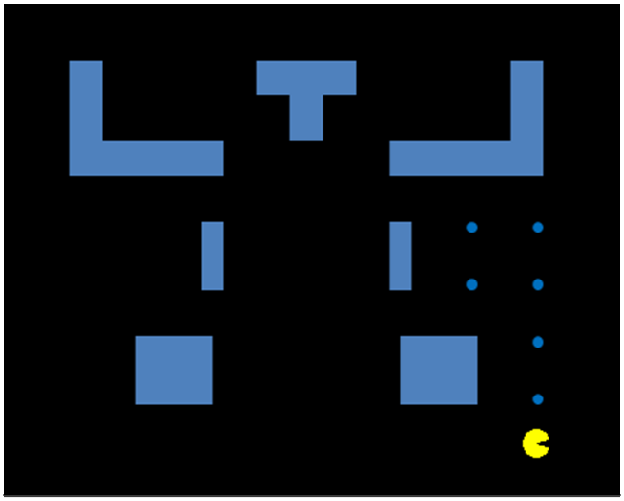
Gambar (11) Pacman melakukan backtrack kedua kalinya

Pacman backtracking dari status sebelumnya dan sampai ke status ini.



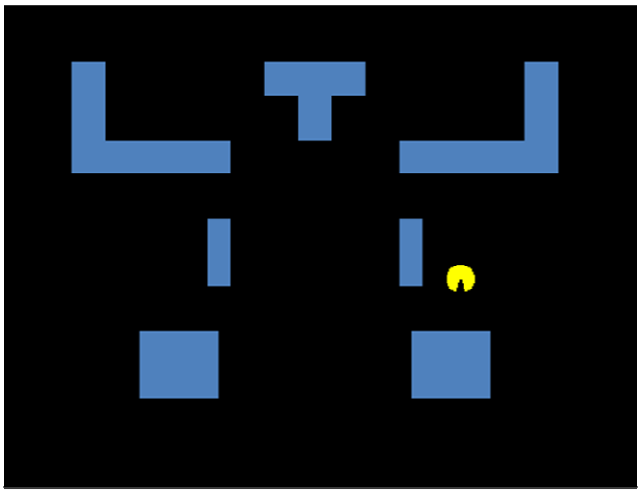
Gambar (12) Pacman melanjutkan perjalanan

Pacman kembali memakan titik-titik dan sampai ke ujung lagi dan harus melakukan backtracking.



Gambar (13) Pacman selesai backtrack ketiga kalinya

Pacman backtracking sampai ke sini dan menemukan titik-titik yang belum dimakan.



Gambar (14) Pacman menyelesaikan permainan

Jumlah titik yang harus dimakan sudah sesuai. Titik-titik sudah habis. Sampai disini permainan selesai.

Pseudo Code

```
Procedure EatDot(input jmlTitik :
integer)

Kamus :
  n : integer

Algoritma :

n = 0
while(n!=jmlTitik)
     while((!ada tembok)&& adaDot) do
             Maju1langkah
             n++
     endwhile
```

```
     do
             LiatKananAtauKiri();
             CekArah++;
     while(
       ((Arah Yang dituju sudah tidak
       ada dot) || (ada tembok)) &&
       (cekArah<2))

     CekArah=0;
     if ((cekArah==2) &&
         (Arah Yang dituju sudah tidak
         ada dot) || ada tembok))
     then
         backtrack ke arah sebelumnya
         sampai menemukan arah yang
         belum dilewati
   endWhile();
```

III. KESIMPULAN

Untuk menyelesaikan permasalahan dalam permainan pacman, yang dalam hal ini adalah semua bagian jalan atau labirin dilewati oleh pemain, dapat digunakan algoritma backtracking. Algoritma akan memungkinkan semua bagian jalan terlewati oleh pemain. Karena, jika menemui jalan buntu pemain akan kembali ke jalan sebelumnya.

Namun, dalam permainan ini ada banyak sekali kemungkinan jalan yang dapat dilewati oleh pacman. Sehingga, ada saat-saat ketika algoritma backtracking memilih jalan memutar yang lebih jauh, padahal terdapat jalan yang lebih dekat.

REFERENSI

[1] Rinaldi Munir, "Strategi Algoritmik", Diktat Kuliah IF2251, 2007

[2] www.detik.com