Analisis pencarian rute dalam permainan komputer 2D berbasis grid dengan algoritma Pencarian Melebar dan *Branch and bound*

Routing analysis for 2D grids computer game using Breadth-first Search (BFS) and Branch and Bound (B&B) algorithm

Irfan.Hanif

Program Studi Informatika, Sekolah Tinggi Elektro Informatika,
Institut Teknologi Bandung
Jl. Ganesha no.10 , Bandung.

If15049@students.if.itb.ac.id

Abstrak

Pencarian rute dalam sebuah permainan komputer 2D berbasis grid, contohnya pada berbagai game strategi yang menggunakan matriks berelemen grid dalam menggambarkan petanya, haruslah mengutamakan optimasi dalam algoritmanya agar proses pencarian yang dilakukan program tidak sampai mengganggu kenyamanan pengguna permainan.

Dalam makalah ini, akan menganalisis bagaimana pemakaian algoritma *Branch and Bound (B&B)* dapat melakukan pencarian rute yang seringkali lebih optimal dibandingkan pencarian rute dengan algoritma *BFS (Breadth-first Search)* dalam sebuah permainan tetapi memiliki kelemahan.

Kata kunci: B&B, BFS, permainan 2D.

Abstract

Routefinding in 2D grids computer games, such as Real-time strategy games, where using matriks of grids in map drawing, should be concern in algoritm`s optimation in purpose this program doesn`t make any annoying process for users.

This paper analize how applying Branch and Bound (B&B) algoritm to routefinding problems are usually more optimize then using BFS (Breadth-first Search) in 2D grids computer games but have some weakness.

Keyword: Branch and Bound, Breadth-first Search, 2D computer games.

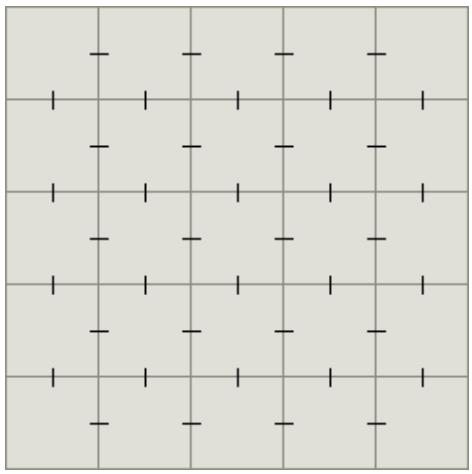
1. Pendahuluan

Permainan komputer berbasis grid ialah permainan komputer yang memiliki peta berupa kumpulan ubin, atau dapat disebut juga merupakan sebuah matriks berelemen grid yang digambarkan seperti ubin, dimana setiap ubin berhubungan dengan ubin tetangganya yaitu tetangga di sebelah utara,timur,selatan dan barat. Selain itu, perlu diketahui bahwa dalam permainan ini tidak mengharuskan semua ubin dapat dilewati oleh setiap objek, contoh permainan seperti ini dapat dilihat pada gambar 1 yang terdapat di halaman ke-2.

Dalam permainan seperti ini, perpindahan suatu objek sangatlah sering terjadi yang artinya menambah kerja seorang programmernya. Jika dalam permainan tersebut objek hanya dapat dijalankan dengan jarak hanya satu petak setiap langkahnya, programmer dari permainan tersebut dapat dengan mudah memprogramnya karena hanya diperlukan kode yang sederhana saja, tetapi jika dalam permainan tersebut suatu objek dapat berjalan dengan hanya mengenali posisi awal dan posisi tujuan, maka diperlukan algoritma yang lebih rumit dari sekedar mengatur satu langkah gerak objek tersebut, algoritma dalam pencarian rute inilah yang saya angkat dalam makalah ini.

Algoritma pencarian rute yang dipakai dalam permainan ini sebenarnya masih menggunakan algoritma pencarian rute yang umum, tetapi karena tiap – tiap ubin dalam permainan ini dapat dianggap sebagai simpul dari suatu graph (seperti yang terlihat pada gambar.2) dan jumlah simpul tersebut sangat banyak serta jarak antara simpul – simpul tetangganya yang sama, maka permasalahan pencarian rute pada permainan ini dapat dianggap kasus khusus dari permasalahan pencarian rute.

Gambar 1. Contoh permainan komputer berbasis grid



Gambar 2. Penggambaran ubin-ubin dalam permainan berbasis grid yang diasosiasikan sebagai suatu graf dengan masing-masing ubin sebagai simpul.

Algoritma pencarian rute yang dipakai pada umumnya sebenarnya sangatlah beraneka ragam, algoritma terbaru dan paling populer adalah algoritma A* yang dikemukakan oleh Hart, Nilsson, dan Raphael di tahun 1968.Disamping itu, masih ada algoritma Djikstra, *Right-hand Rule*, dan *Deep-first Search* yang semuanya dapat diterapkan dalam permasalahan pencarian rute dalam permainan berbasis grid seperti ini, tetapi dalam makalah ini kami akan fokuskan kepada metode Pencarian Melebar *(Breadth-first Search* atau *BFS)* dan metode *Branch and Bound (B&B)* yang menurut saya relatif sederhana tetapi cukup optimal.

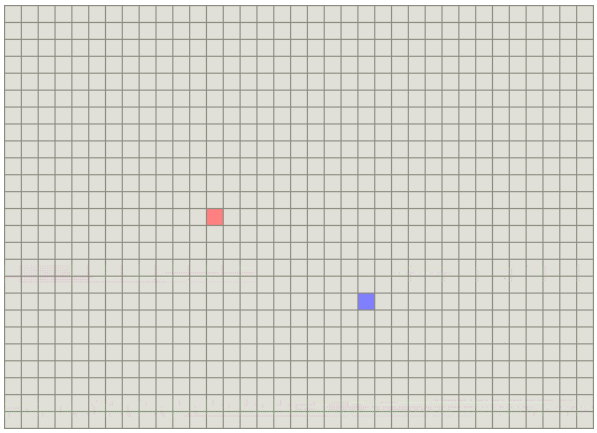
Makalah ini menjelaskan cara kerja kedua algoritma pencarian rute dalam permainan (*B&B* dan *BFS)*, serta membandingkan optimasi dari kedua algoritma tersebut yang kemudian diharapkan dapat membantu kita untuk menentukan mana algoritma yang lebih optimal.

2. Metode

Sekarang saya akan langsung membandingkan kinerja dari kedua algoritma yang akan dibandingkan lewat dua buah kasus yang unik, kasus yang pertama adalah kasus dimana diantara posisi awal dan tujuan semua ubin/grid dapat dilalui objek, dan kasus kedua yaitu ketika posisi awal dan posisi tujuan terdapat halangan yang tidak dapat dilalui objek.

2.1. Kasus 1 – pencarian rute tanpa penghalang

Misalkan gambar dibawah ini mewakilkan setiap grid yang ada dalam permainan :



Gambar 3. Kasus 1 pencarian rute

Dengan ubin merah sebagai posisi awal (status awal), dan ubin biru sebagai posisi tujuan (status tujuan).

2.1.a. dengan mengunakan algoritma *BFS*

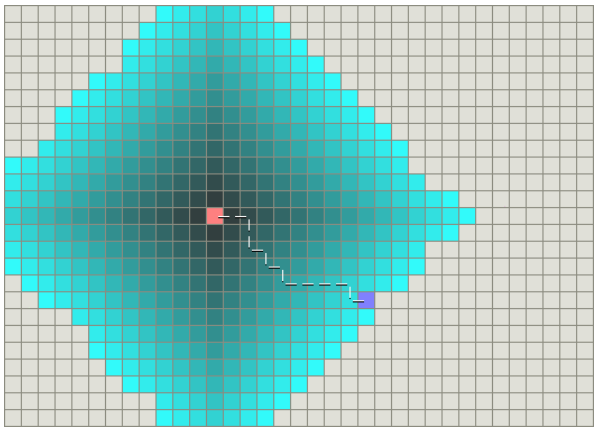
Pencarian solusi dengan metode *BFS* dapat digambarkan sebagai berikut:

1. Masukkan simpul akar (dalam kasus ini adalah posisi awal objek/ubin merah) ke dalam antrian Q.

Jika simpul akar adalah simpul tujuan, maka solusi ditemukan. Stop.

2. Jika Q kosong, tidak ada solusi. Stop.
3. Ambil simpul dari kepala antrian, bangkitkan semua anak-anaknya yang tidak lain merupakan simpul/ubin yang bertetanggaan. Jika v tidak mempunyai anak lagi , kembali ke langkah 2. Tempatkan semua anak dari v di belakang antrian sesuai dengan urutan kelahirannya.
4. Jika suatu simpul anak dari v adalah simpul solusi, maka solusi telah ditemuakn, kalau tidak kembali lagi ke langkah 2.

Disebabkan adanya 4 tetangga di setiap ubin, maka karena tiap ubin dapat dianalogikan sebagai sebuah simpul, maka disetiap simpul permasalahan ini mempunyai anak sebanyak 4 buah simpul sehingga banyaknya simpul yang perlu dihidupkan hingga ditemukan solusi untuk jarak antara simpul awal dengan simpul solusi sebesar n jika dihitung secara kasar adalah $1 + 4^1 + 4^2 + 4^3 + ... + 4^n$. Tetapi dikarenakan dalam menyusun pohon *BFS*, tidak diperkenankan membangkitkan status yang sama lebih dari satu kali, maka dengan menyimpan semua simpul yang sudah kita bangkitkan dalam sebuah senarai dan setiap kali simpul akan dikembangkan, periksa kedalam senarai apakah simpul anaknya ada yang sama dengan simpul di dalam senarai, sehingga status yang sama tidak kita bangkitkan berulang kali. Maka simpul yang dibangkitkan akan terlihat pada gambar 4 dibawah ini :



Gambar 4. Kasus 1 pencarian rute- diselesaikan dengan metode *BFS*

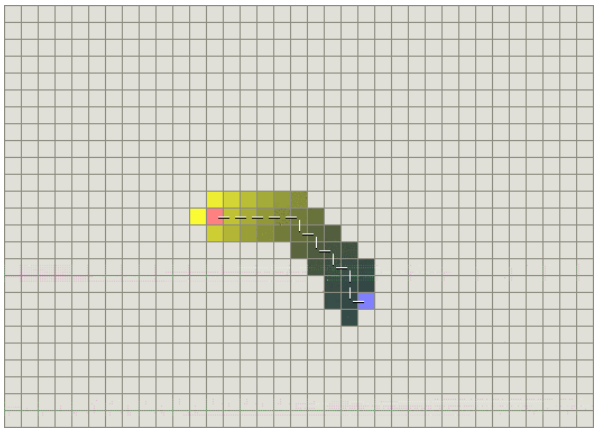
Simpul yang dibangkitkan ditunjukkan dengan ubin berwarna pada gambar diatas, dengan kata lain jumlah maksimum simpul yang dibangkitkan untuk jarak antara simpul awal dengan simpul solusi sebesar n adalah sebesar luas daerah pencarian rute tersebut yaitu :

((2*n + 1).(2*n +1)/2) + ½. Yang berarti lebih kecil dari nilai maksimum *BFS* pada umumnya. Jadi kompleksitas waktu *BFS* yang diterapkan pada permasalahan pencarian rute dalam permainan jika tidak ada halangan antara posisi awal dengan posisi tujuan adalah $\mathbf{O}(4n^2) = \mathbf{O}(n^2)$.

2.1.b. dengan mengunakan algoritma *B&B*

Pencarian solusi dengan metode *Branch and Bound (B&B)*, yang pertama kali diperkenalkan oleh H. Land and A. G. Doig di tahun 1960 sebenarnya tidak banyak berbeda dengan *BFS* murni terutama dalam hal pembentukan pohon solusinya, Akan tetapi dalam algoritma *(B&B)* untuk mempercepat pencarian ke simpul solusi, setiap simpul diberi sebuah nilai ongkos, kemudian urutan pembangkitan simpul solusinya ditentukan dari simpul yang memiliki ongkos yang paling kecil diantara simpul – simpul hidup lainnya. Untuk kasus kali ini, taksiran ongkos ditentukan lewat jarak yang dihitung lewat rumus dx + dy, sehingga simpul yang dipilih untuk dibangkitkan anak-anaknya terlebih dahulu adalah simpul yang memiliki koordinat terdekat dengan simpul tujuan.

Sehingga untuk kasus pertama tadi, jika menggunakan algoritma *B&B* simpul – simpul yang dibangkitkan adalah sebagai berikut :



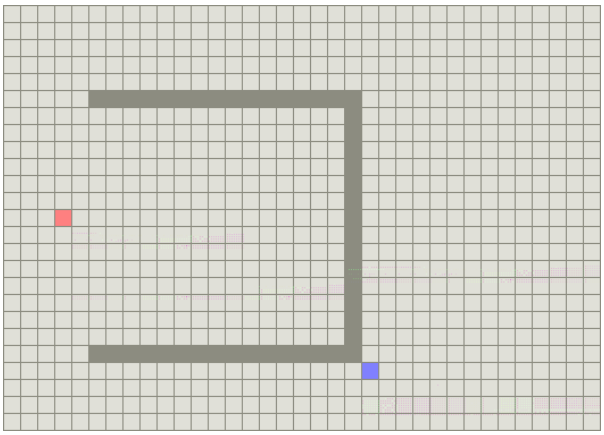
Gambar 5. Kasus 1 pencarian rute- diselesaikan dengan metode *B&B*

Terlihat bahwa simpul yang dibangkitkan algoritma *B&B* jauh lebih sedikit dibandingkan dengan metode *BFS*. Jumlah maksimum simpul yang dibentuk untuk kasus dimana tidak ada penghalang antara posisi awal dan posisi tujuan dapat dihitung karena dengan metode *B&B* setiap level pohon yang terbentuk selalu menghasilkan simpul yang semakin mendekati simpul solusi dibandingkan simpul ayahnya, dan dengan asumsi setiap langkah hanya menghasilkan 2 simpul lainnya yang arahnya tidak menuju simpul tujuan, maka banyaknya maksimum simpul yang dibangkitkan untuk jarak antara simpul awal dengan simpul solusi sebesar n adalah sebanyak (3 * n). Berarti kompleksitas waktu *B&B* yang diterapkan pada permasalahan pencarian rute dalam permainan, jika tidak

ada halangan antara posisi awal dengan posisi tujuan adalah **O**(n).

2.2. Kasus 2 – Pencarian rute dengan penghalang

Untuk kasus 2, dipilih matriks yang beberapa grid diantaranya tidak dapat dilalui, contohnya matriks permainan seperti dibawah ini :

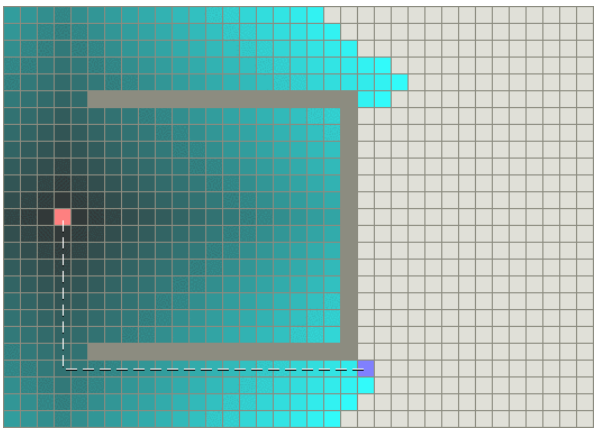


Gambar 6. Kasus 2 pencarian rute

Simpul awal digambarkan dengan simpul merah, dan simpul tujuan digambarkan dengan simpul biru.

2.2.a. dengan mengunakan algoritma *BFS*

Kemudian dengan algoritma BFS yang sama dengan pada kasus 1, maka akan dibangkitkan simpul-simpul yang dapat digambarkan sebagai berikut :



Gambar 7. Kasus 2 pencarian rute – penyelesaian dengan algoritma *BFS*

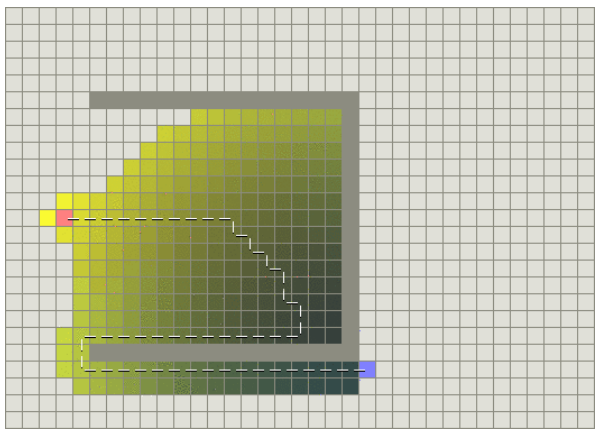
Dapat terlihat bahwa justru dengan adanya halangan, maka simpul yang dibangkitkan akan berkurang. Hal ini disebabkan beberapa simpul yang bertetanggaan dengan penghalang akan dimatikan karena tidak mempunyai tetangga yang simpulnya belum dibangkitkan lagi. Dengan demikian kesimpulan yang dapat diambil dari kasus ini ialah, semakin banyaknya penghalang dalam sebuah permainan justru akan membuat proses pencarian dengan *BFS* semakin optimal. Jadi kompleksitas waktu *BFS* untuk pencarian rute dengan halangan, sama dengan kasus 1 dimana tidak terdapat halangan, yaitu **O**(n^2), dengan n = jarak *shortest path* dari posisi awal ke posisi akhir.

Selain itu perlu diperhatikan juga bahwa algoritma *BFS* menjamin ditemukkannya *shortest path* karena dalam *BFS* selalu membangkitkan semua simpul anak yang memungkinkan di setiap levelnya tanpa terkecuali, sehingga simpul tujuan akan ditemukan terlebih dilevel paling atas yang memungkinkan.

2.2.b. dengan mengunakan algoritma *B&B*

Masih pada kasus kedua, dimana terdapat penghalang diantara posisi awal dan posisi tujuan, ternyata jika penyelesaian dengan algoritma *BFS* kita bandingkan dengan menggunakan algoritma *B&B*, ternyata kita masih dapat menemukan bahwa algoritma *Branch and Bound* tetap lebih optimal dalam permasalahan ini. Hal ini didasarkan oleh jumlah simpul yang dibangkitkan algoritma ini tidaklah sebanyak yang dibangkitkan jika kita menggunakan algoritma *BFS*.

Banyaknya simpul yang dibangkitkan oleh algoritma *Branch and Bound* dalam kasus kedua ini dapat terlihat pada gambar 8 dibawah ini :



Gambar 8. Kasus 2 pencarian rute – penyelesaian dengan algoritma *B&B*

Permasalahan timbul karena ternyata penyelesaian dengan algoritma ini tidaklah menghasilkan rute yang merupakan rute terpendek. Hal ini disebabkan oleh urutan

simpul yang dibangkitkan oleh algoritma ini bukan ditentukan oleh jaraknya dari simpul awal, melainkan ditentukan lewat sebuah taksiran ongkos ĉ(P) yang sebenarnya hanyalah sebuah intuisi yang kita harapkan bahwa dengan ĉ(P) yang lebih kecil akan lebih mendekatkan kita ke simpul tujuan yang ternyata tidak berlaku apabila terdapat penghalang diantara posisi awal dan posisi tujuan.

Oleh karena itu, sangatlah sulit mencari cara untuk menghitung kompleksitas waktu rata-rata untuk pencarian rute secara *Branch and Bound* ,untuk rute dengan penghalang diantara posisi awal dan posisi tujuan. Tetapi kita dapat menyimpulkan bahwa kompleksitas waktu maksimumnya adalah sama dengan kompleksitas waktu untuk *BFS* murni yaitu $\mathbf{O}(n^2)$, dengan n = jarak *shortest path* dari posisi awal ke posisi akhir.

3. Kesimpulan

Dalam pencarian rute dalam sebuah permainan berbasisi grid, penggunaan algoritma *Branch and Bound (B&B)* jauh lebih optimal dibandingkan penggunaan algoritma *Breadth-first Search (BFS)*, terutama dalam pencarian yang dimana tidak terdapat penghalang diantara posisi awal objek dengan posisi tujuan objek, dalam kasus tersebut, kompleksitas waktu yang dibangun oleh algoritma *B&B* bergerak secara linier **(O(n))**, dibandingkan algoritma *BFS* murni yang jika diterapkan pada permainan ini memiliki kompleksitas waktu $\mathbf{(O(n^2))}$. Hanya saja, algoritma *B&B* tidak menjamin simpul solusi yang didapat dalam pembentukkan pohon solusi *B&B* ditemukan pada level terendah yang memungkinkan, sehingga rute yang didapatkan melalui algoritma *B&B* belum pasti merupakkan rute terpendek yang dapat dicapai, lain halnya jika kita menggunakan algoritma *BFS.*

4. Saran

Dalam pemrograman permainan berbasis grid yang membutuhkan pencarian rute, lebih baik digunakan algoritma *Branch and Bound* dibandingkan algoritma *Breadth-first Search (BFS).* Walaupun begitu, algoritma *BFS* lebih layak dipilih apabila dalam permainan tersebut membutuhkan kecerdasan buatan yang lebih baik dengan keharusan untuk selalu memilih jarak terpendek, terlebih lagi jika didalam permainan berbasis tersebut terdapat banyak penghalang.

5. Daftar Pustaka

[1] Munir, Rinaldi, 2007. Diktan kuliah IF2251 - Strategi Algoritmik. Bandung. Informatika.

[2] http://en.wikipedia.org/wiki/Breadth-first_search/, waktu akses 22 Mei 2007, 20.00 WIB.

[3] http://theory.stanford.edu/~amitp/GameProgramming/, waktu akses 22 Mei 2007, 20.00 WIB.