**METODE PENCARIAN *UNIFORM COST SEARCH* (UCS)**

**LAPORAN KARYA TULIS ILMIAH**

diajukan untuk memenuhi salah satu tugas mata kuliah kecerdasan buatan



**oleh**

Asep Kurnia (55201120036)

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER DAN INFORMATIKA**

**UNIVERSITAS NURTANIO BANDUNG**

**BANDUNG**

**2022**

**BAB I**

**LANDASAN TEORI**

1. ***Search***

Menurut Luger (2005), *search* adalah sebuah Teknik menyelesaikan masalah (*problem solving*) yang mengembangkan sebuah ruang permasalahan secara sistemik dalam sebuah proses. Terdapat 4 kriteria untuk menentukan performa sebuah metode percarian, yaitu *Completeness, Time Complexity, Space Complexity*, dan *Optimality*. Metode Pencarian dibagi menjadi dua strategi, yaitu *uniformed search* dan *informed search*. *Uniform search* merupakan suatu strategi pencarian tanpa ada informasi mengenai *cost* (bobot) atau informasi tertentu, sedangkan *Informed search* merupakan suatu strategi pencarian yang membutuhkan informasi mengenai *cost* (bobot) atau informasi tertentu.

1. ***Uniform Search***

*Uniform Search* juga sering disebut dengan istilah *Blind Search*. Istilah tersebut menggambarkan bahwa Teknik pencarian ini tidak memiliki informasi tambahan mengenai kondisi diluar dari yang disediakan oleh definisi masalah. Yang dilakukan oleh algoritma ini adalah melakukan *generate* dari *successor* dan membedakan *goal state* dari *non-goal state*. Pencarian dilakukan berdasarkan pada urutan mana saja node yang hendak di expand. Jika solusi sudah ketemu, maka pencarian akan dihentikan.

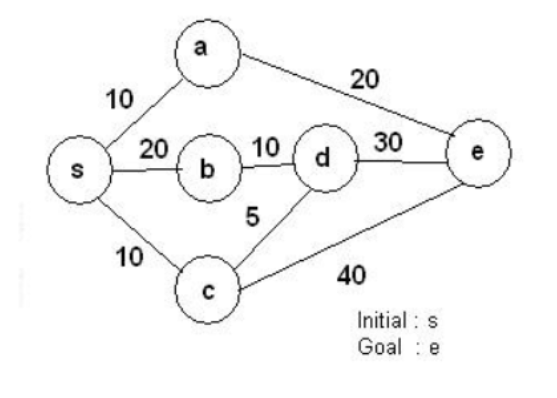
1. ***Uniform Cost Search***

*Uniform Cost Search* adalah algoritma *Search Tree* (graph) yang digunakan untuk menyelesaikan beberapa persoalan. Algoritma ini memulai pencarian dari *root node*, kemudian dilanjutkan ke *node-node* selanjutnya. Dimana *node* tersebut dipilih yang memiliki harga/nilai (*cost*) terkecil dari *root node*. Algoritma ini merupakan modifikasi dari *Bread First Search* (BFS).

Dalam implementasi algoritma ini, melibatkan semua *node* yang berhubungan dengan *root node*, dan meletakannya dalam *priority queue* untuk mencapai *node* tujuan. Dimana *node-node* yang dipilih merupakan node yang berharga/bernilai kecil.

**BAB II  
CONTOH PENERAPAN**

**METODE *UNIFORM COST SEARCH* (UCS)**

****

Pada permasalahan diatas telah ditentukan jarak antara *node*. Maka *pada Uniform Cost Search* (UCS) akan membuka node yang memiliki nilai (*cost*) antar *node* yang terendah. Pada gambar diatas jika kita buka :

a = 10

b = 20

c = 10

Karena nilai a dan c sama, maka kita bebas memilih *node* mana yang akan dipilih terlebih dahulu. Seandainya kita membuka *node* c, maka kita teruskan pencariannya, jika kita buka :

d = 10 + 5 = 15

e = 10 + 40 = 50 (mencapai *goal*, namun nilai *cost* dirasa masih terlalu besar)

Maka kita buka *node* d, lalu akan didapat:

E = 10 + 5 + 30 = 45 (nilai pada pencarian ini pun dirasa masih terlalu besar)

Maka dari itu kita buka *node* yang kecil satunya lagi yaitu *node* a. Setelah kita buka *node* a akan didapat :

e = 10 + 20 = 30 (didapatkan goal dengan solusi terbaik)

Dari kasus diatas dapat kita lihat, ada banyak cara untuk mendapatkan solusi. Namun dari berbagai macam penyelesaian kasus, kita dapat mencari solusi yang paling optimal dan inilah keunggulan dari metode pencarian menggunakan *Uniform Cost Search* (UCS)