Introduction to heuristics and meta-heuristics in Python

Notes of class

Iván Andrés Trujillo Abella

Facultad de ingeniería

1 Some concepts about graph theory

a graph $G = (V, E)$ composed of two set the set of vertex V and the set of edges E. the set V could be person, items or another entities while E is the set that linked them.

for instance: $V = \{(A, B, C)\}$ and $E = \{(A, B), (A, C), (B, C)\}$. generally the graphical representation is by points and lines, where vertex and points and lines that connected the vertices represented edges.

in some cases could be exist loops inside graphs for instance: $V = \{(A, B, C, D\}$ and $E = \{(A, A), (A, B), (C, D)\}$ note that so far we had been dispensing of the loop (A, A).

2 Algorithms and search

- algorithm hill climbing
- search A*
- minmax
- alfa and beta algorithm

rubik cube.

peeling: peladura fading: desvanecimiento.

mainstream: circunvent: braid:

amusement: noughts; scrambled: spotlight: shelves: halted: advertisements: rear: trasero, craze: plummeted: shortage: sued: appealed: overturned: amended: disclosed: upheld: concealed: inward: screw: advertised: flipped: scrambled: takes up: lament; awful: silinesss

3 Algorithms in blind

expaction function and stop function:

Depth first search (DFS): it is an algorithm, but not guaranteed the optimum route.

stack structure:

hash set:

3.1 DFS algorithm

depth first search how works here the data structure ?
important here longitute of path and branching factor:

3.1.1 LIFO structure

LIFO (last in first out), the stack is a example, the last element added is the first removed. Are useful to implement recursion.

```
stack = ['A','B','C']
stack.append('D') # Add to the tail 'D'
stack.pop() # remove the last element namely 'D'
```

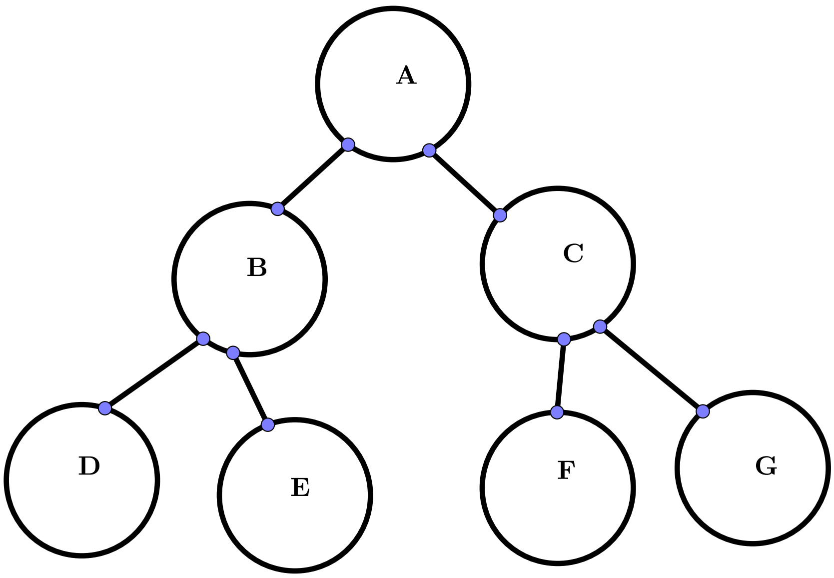
3.2 BFS algorithm

3.2.1 FIFO structure

FIFO (first in first out), the queue is a example, the first element added is the first element removed. imagine good or services, line in a restaurant.

```
queue = ['A','B','C','D']
queue.pop(0) # Remove the first element 'A'
queue.append('Z') # Add the 'A' to the tail.
```

4 Applying BFS and DFS to a particular tree



4.1 Dijkstra algorithm

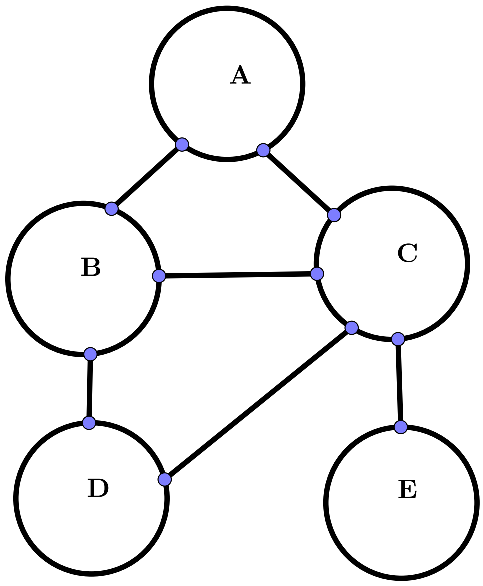
4.2 Asymtotic analysis of DFS and BFS

Memory and time;
what is branching factor? if is variable then uses the mean.
how traverse a tree is the question:
traversing is visit each node of graph:

4.3 question

whats mean greedy algorithm what means recursive algorithm
whats is queue?

5 how represent graphs in python



The last figure is a undirected graph; we have some representations to this graph, by dictionaries or by adjacency matrix.

```
# defining a graph
graph = {
'A':[B,C]
'B':[A,C]
'C':[D,E]
'E':[C]
}
```

5.1 Adjacency matrix

Result that we can also represent in a adjacency matrix in the following sense.

	A	B	C	D	E
A	0	1	1	0	0
B	1	0	1	1	0
C	1	1	0	1	1
D	0	1	1	0	0
E	0	0	1	0	0

Note that the matrix dimension is defined by number of vertex, $|V| * |V|$.

5.2 From matrix to graph

A matrix could be represented by a graph if the matrix is symmetric, $x_{i,j} \in \{0, 1\}$ where 1 if exist a edge among the vertex i and j in otherwise 0, However we can uses $k \in \mathbb{N}$ where k count the number of edges that link two vertex.

5.3 Incidence matrix

This matrix have the dimension $|V| * |E|$ and represent with x_{ij} if the in directed graphs (only one direction is allowed) the representation is similar. The algorithms rely on in data structures,

5.4 Konigsbiger bridge problems

resolver by euler an analytical solution an a brute force solution.

$V = (A, B, C, D)$ and $E = \{(A, B), (A, B), (A, C), (A, C), (B, D), (A, D), (C, D)\}$ note here that appear two times two different edges.

```
graph = {
'A':[B,B,C,C,D]
'B':[A,A,D]
'C':[A,A,D]
'D':[A,B,C]
}
```

in adjacency matrix born the concept of weights,thus:

	A	B	C	D
A	0	2	2	1
B	2	0	0	1
C	2	0	0	1
D	1	1	1	0

6 Maze-solving algorithms

perfect mazes are three from stand point of graph theory.

6.1 random mouse algorithm

6.2 wall follower

6.3 pledge algorithm

6.4 Hill climbing algorithm

Stuck in a local maximum.

6.5 branch and bound algorithm

7 A* search algorithm

8 tabu search

8.1 Grey wolf optimizer

9 Neural networks MIT

9.1 Deep Learning

9.2 What is backpropagation

9.3 softmax

9.4 Recurrent neural network

9.5 LSTM network

Is important the idea behind threshold!

9.5.1 Representation of weights and counter map

follow the gradient...

9.5.2 Theorem of gradient

9.5.3 Whay is the steepest way??

9.6 When gradient descent dont work

discontinuous function:
gradient require a continuous space:

10 Knasck problem