

Generator Labiryntu

Marcin Data

Kraków 17.01.2017

1. Zawartość projektu :

Projekt składa się z trzech plików napisanych w języku Python:

- main.py
- graph.py
- sdj.py

W pliku main.py znajduje część kodu odpowiedzialna za utworzenie instancji generatora a także wygenerowania labiryntu przy pomocy każdego z trzech algorytmów.

W pliku sdj.py znajduje się implementacja zbiorów rozłącznych przy pomocy słownika, którą zaczerpnięto z podręcznika Cormana.

W pliku graph.py znajduje się klasa generatora oparta na grafie z macierzą sąsiedztwa. Metody zawarte w klasie Lab_Graph to przede wszystkim konstruktor przyjmujący na wejściu wymiary labiryntu, metoda __str__, showHash, showDollar w celu wypisania wygenerowanego labiryntu (odpowiednio przy pomocy kwadratów, # i \$) oraz metody KruskalGenerate(), PrimGenerate(), AldousBroderGenerate().

2. Opis najważniejszych metod

a) KruskalGenerate()

Metoda KruskalGenerate wykorzystuje przy tworzeniu labiryntu nieco zmieniony algorytm Kruskalla służący do znajdowania minimalnego drzewa rozpinającego. Algorytm generujący działa w sposób następujący :

V - zbiór wierzchołków

M - macierz sąsiedztwa

DJ - zbiór wszystkich wierzchołków w postaci zbiorów rozłącznych

s = losowy z V startowy

**while wszystkie zbiory w DJ nie są jednym
zbiorem**

oznacz V jako odwiedzony

**jeśli s nie jest w tym samym zbiorze DJ co któryś z
sąsiadujących wierzchołków e z V:**

połącz s z e w DJ oznacz (s,e) w M

while s jest odwiedzony

s = losowy z V

b) PrimGenerate()

Metoda PrimGenerate wykorzystuje przy tworzeniu labiryntu zmodyfikowany algorytm Prima służący znajdowaniu minimalnego drzewa rozpinającego. Algorytm generujący działa w następujący sposób:

V - zbiór wierzchołków

M - macierz sąsiedztwa

Q - kolejka priorytetowa

s = losowy z V startowy

**wrzucić do Q krawędź (s,s) z maksymalnym możliwym
priorytetem**

oznacz s jako gotowy

while Q nie jest pusta:

if wierzchołek e z V będący sąsiadem s nie jest gotowy:

wrzucić do Q krawędź (s,e) z losowym priorytetem

oznacz e jako gotowy

Zdejmij krawędź (l , i) o najmniejszej wadze

if i nie jest częścią labiryntu:

oznacz krawędź(l,i) w M

oznacz wierzchołek i jako gotowy

s = i

c) AldousBroderGenerate()

Metoda AldousBroderGenerate wykorzystuje przy tworzeniu labiryntu zmodyfikowany algorytm Aldous-Broder'a. Algorytm generujący działa w następujący sposób:

V - zbiór wierzchołków

M - macierz sąsiedztwa

s = losowy wierzchołek z V

oznacz s jako odwiedzony

while jakiś wierzchołek jest nie odwiedzony:

wylosuj wierzchołek e z V będący sąsiadem s

przenieść się do e

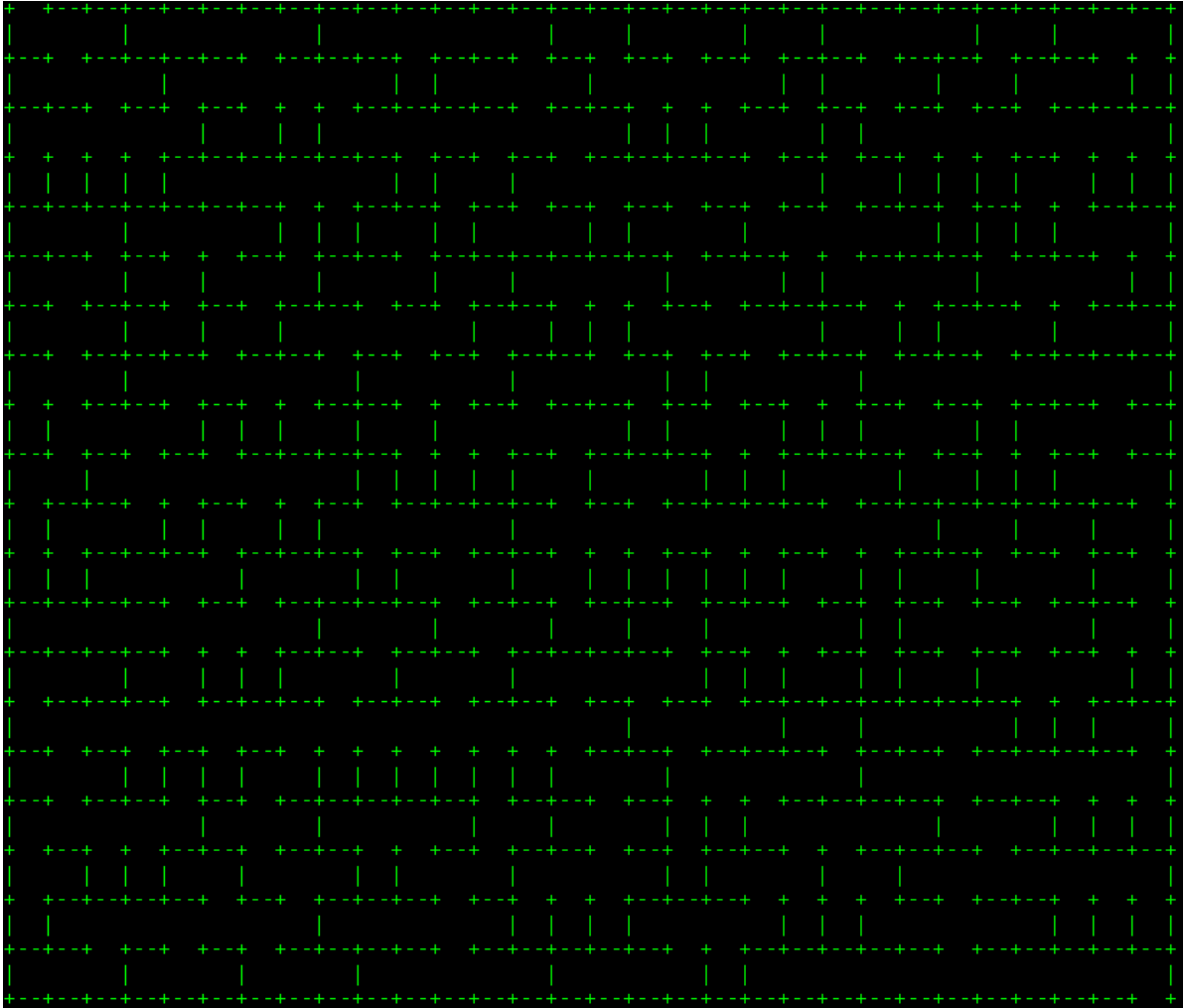
jeśli e nie jest oznaczony:

oznacz e jako odwiedzony

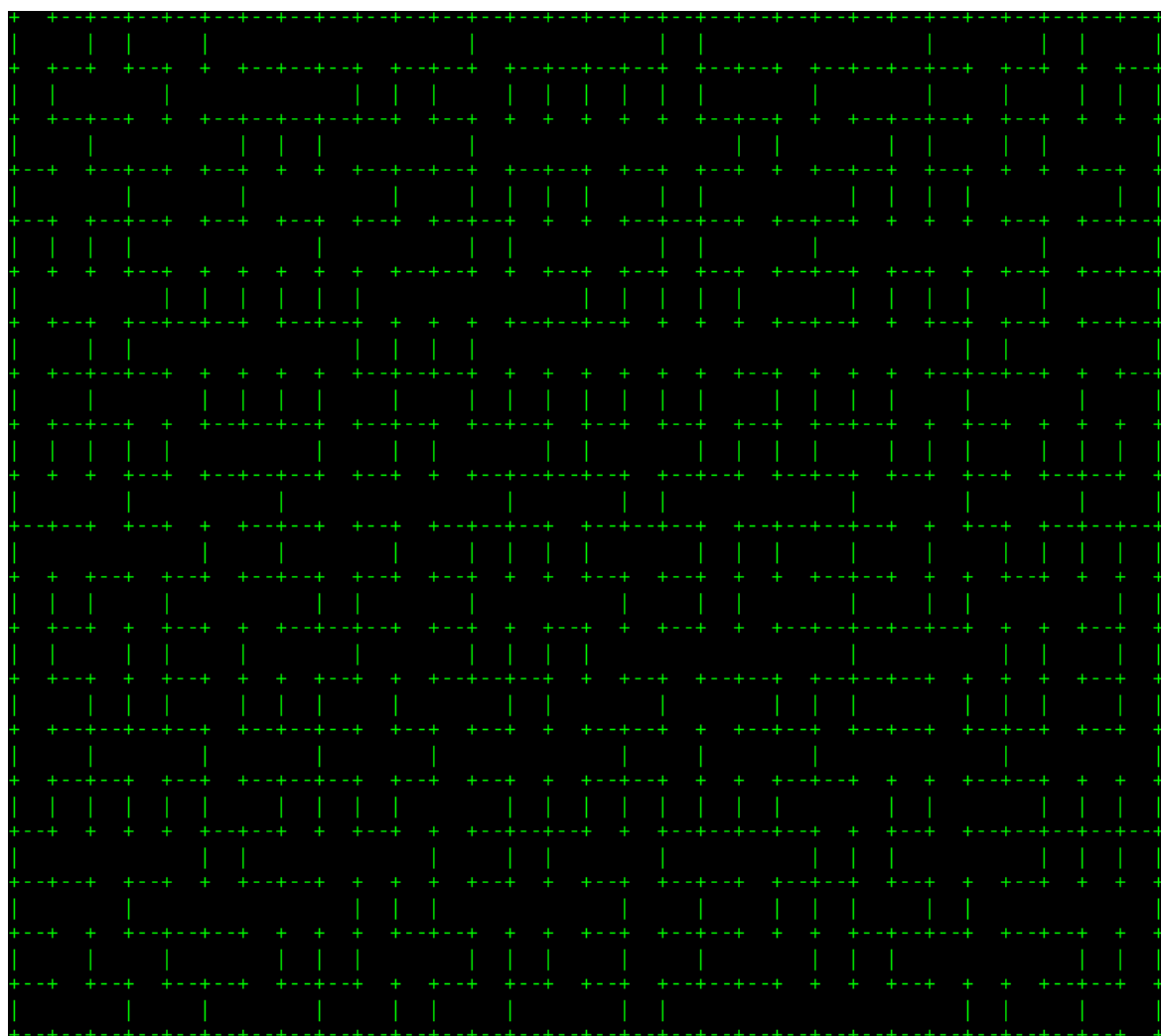
oznacz krawędź (s,e) w M

3. Prezentacja przykładowych
wyników:

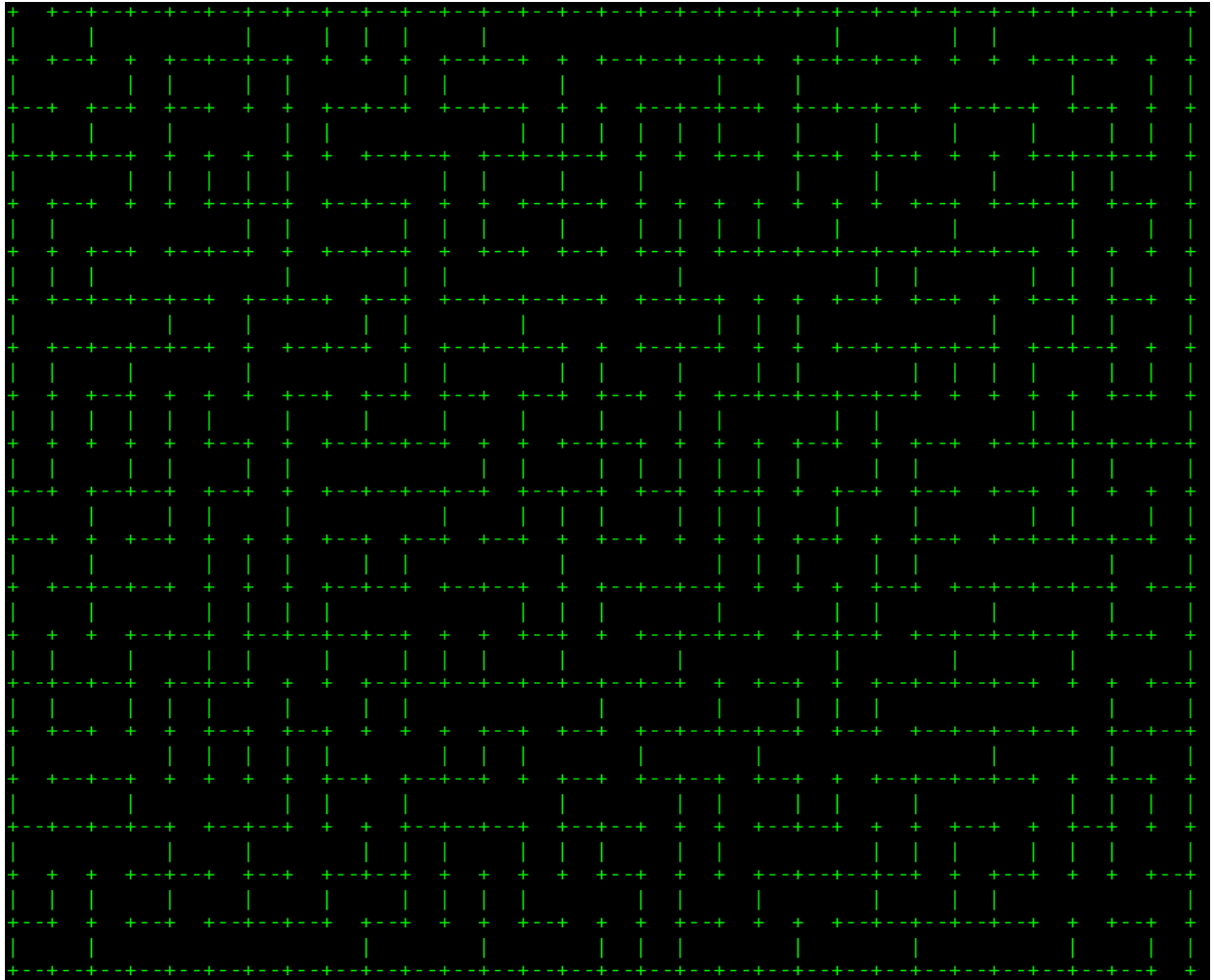
a) Labirynt 20x30 wygenerowany przy pomocy metody KruskalGenerate:



b) Labirynt 20x30 wygenerowany przy pomocy metody PrimGenerate:



c) Labirynt 20x30 wygenerowany przy pomocy metody
AldousBroderGenerate:



Porównując szybkość działania zastosowanych algorytmów,
najszybciej wykonującym się jest metoda KruskalGenerate, a
najwolniejszym jest AldousBroderGenerate.