Toteutusdokumentti

Labyrintingeneroija ja –ratkoja

Tietorakenteet ja algoritmit harjoitustyö

Juri Kuronen

Kesä 2014

Sisältö

1. Ohjelman yleisrakenne

Labyrintti-luokka on ohjelman runko. Labyrintti-luokka sisältää *korkeus x leveys* -kokoisen byte-arrayn, johon on tallennettu tieto labyrintin soluista lähtevistä kaarista. Luokka sisältää monia metodeja tiedon lukemiseen ja päivittämiseen tästä byte-arraysta. Lisäksi Labyrintti-luokan vastuulla on käynnistää Labyrintti-luokkaan liitetyt generointi- ja ratkaisualgoritmit oikein.

LabyrinthGenerator-luokka on labyrintin generoijien yliluokka. Jokainen aliluokka toteuttaa generateLabyrinth()-metodin. Labyrintin generoijat täyttävät (eli generoivat satunnaisesti) Labyrintti-luokan byte-arrayn. Labyrintin generoijia ovat Kruskalin algoritmi, Primin algoritmi ja Rekursiivinen peruuttava haku.

LabyrinthSolver-luokka on labyrintin ratkojien yliluokka. Jokainen aliluokka toteuttaa solveLabyrinth()-metodin. Labyrintin ratkojia ovat ”Oikean käden sääntö”-algoritmi, leveys- ja syvyyssuuntainenhaku ja A*-hakualgoritmi. Yliluokan avulla voi hakea ratkaisun jälkeen tietoja ratkaisusta. Yliluokka sisältää mm. maaliin vievän reitin hakemiseen aliluokkien tekemästä työstä.

Toteutetut tietorakenteet ovat omina luokkina, paitsi Kruskalin algoritmin käyttämä joukkoalkio-aputietorakenne sekä LabyrinthSolver-luokan käyttämä TreeNode-aputietorakenne reitinhakuun. Kaikille tietotyypeille on toteutettu lista, pino sekä jono. Prioriteettikeko on toteutettu A*-hakualgoritmia varten.

Ohjelmaa käytetään graafisen käyttöliittymän avulla, joka avautuu ohjelman käynnistyessä. Graafisen käyttöliittymällä voi asettaa labyrintin koon sekä valita generoivan- että ratkovan algoritmin.

Suorituskykytestaukset tapahtuvat RunTimeTesting-luokan kautta.

2. Saavutetut aika- ja tilavaativuudet

Tässä on kasattu taulukkoon generointi ja –ratkonta-algoritmien käyttämiä keskimääräisiä aikavaativuuksia. Kaikilla ko'oilla suoritettu algoritmi n = 50 kertaa ja laskettu keskiarvo.

Generoivan algoritmin alla on keskimääräinen generointiaika. Ratkovien algoritmien tieto on muodossa ”keskimäärinen ratkomisaika” / ”keskimääräisesti vierailtujen solujen määrä”.

10x10			
Ratkoja \ generoija	Prim's algorithm 0,780 ms	Kruskal's algorithm 0,942 ms	Recursive backtracker 0,383 ms
Randomized DFS	0,288 ms / 59	0,551 ms / 59	0,015 ms / 66
Randomized BFS	0,385 ms / 97	0,156 ms / 82	0,019 ms / 65
A* search	0,388 ms / 52	0,173 ms / 53	0,215 ms / 59
Wall follower	0,068 ms / 59	0,032 ms / 63	0,008 ms / 67
Wall follower (optim.)	0,096 ms / 59	0,447 ms / 63	0,004 ms / 67

25x25			
Ratkoja \ generoija	Prim's algorithm 3,775 ms	Kruskal's algorithm 1,798 ms	Recursive backtracker 1,004 ms
Randomized DFS	0,132 ms / 330	0,193 ms / 370	0,134 ms / 387
Randomized BFS	0,175 ms / 621	0,098 ms / 520	0,131 ms / 377
A* search	0,297 ms / 261	0,132 ms / 315	0,074 ms / 348
Wall follower	0,031 ms / 326	0,027 ms / 334	0,026 ms / 403
Wall follower (optim.)	0,025 ms / 326	0,024 ms / 334	0,022 ms / 403

... täytetään 50x50, 100x100, 250x250, ..., myöhemmin.

500x500			
Ratkoja \ generoija	Prim's algorithm 102,081 ms	Kruskal's algorithm 563,115 ms	Recursive backtracker 84,359 ms
Randomized DFS	26,000 ms / 137 280	24,735 ms / 129 970	28,838 ms / 143 094
Randomized BFS	59,414 ms / 249 980	49,216 ms / 212 192	25,770 ms / 126 167
A* search	18,246 ms / 51 553	55,319 ms / 177 678	27,043 ms / 124 468
Wall follower	10,401 ms / 125 184	11,053 ms / 126 385	10,564 ms / 146 789
Wall follower (optim.)	9,818 ms / 125 184	9,904 ms / 126 385	9,753 ms / 146 789

Myöhemmin analyysiä miten aikavaativuudet saavutettiin.