Tietorakenteet ja algritmit harjoitustyö

A* ja Dijkstran algoritmin vertailuun tarkoitettu ohjelma

Testausdokumentti

Juhani Heliö

Testaus

Ohjelmassa on automaattiset Junit -testit tietorakenteet -paketin luokille. Tämän lisäksi ohjelmaa on testattu testisyötteillä, jotka ovat antaneet esim seuraavanlaiset verkot tuloksena:

A*:

```
=========          = : tyhjä ruutu
===ooooo=          o : opensetissä oleva solmu
==oscccco          o : opensetissä oleva solmu
==oxcccco          c : closedsetissä oleva solmu eli läpikäyty solmu
==oxxooo=          x : reitti
===oxccco          s ja w : start ja win eli alku ja maali
===oxccco          0 : este
===oxxxw=
====ooo==

sxxxcccco
cc0xxxcco
000ccxco=
==occxcco
===ocxcco
===ocxcco
===ocxoo=
====oxxw=
=====oo==

sccccccco
xccccccco
xocooocco
xxccccccco
oxoooccco
oxxxxccco
=ocoxooo=
==ooxxxw=
====ooo==
```

Dijkstran algoritmi:

```
ccccccccc
ccccccccc
cccsxxxxc
ccccccxc
ccccccxc
ccccccxc
ccccccxo
cccccccw=
occcccco==
```

```
sxxxxxxxc
cc0ccccxc
000ccccxc
cccccccxc
cccccccxc
cccccccxc
cccccccxo
ccccccow=
occccco=

sxxxxxxxc
cccccccxc
cccccccxc
cccccccxc
cccccccxc
cccccccxc
cccccccxc
ccccccwo
cccccco=
```

Näistä testeistä huomataan, että algoritmit käyvät verkot läpi oikein ja toimivat muutenkin normaalisti. Testisyötteet olivat:

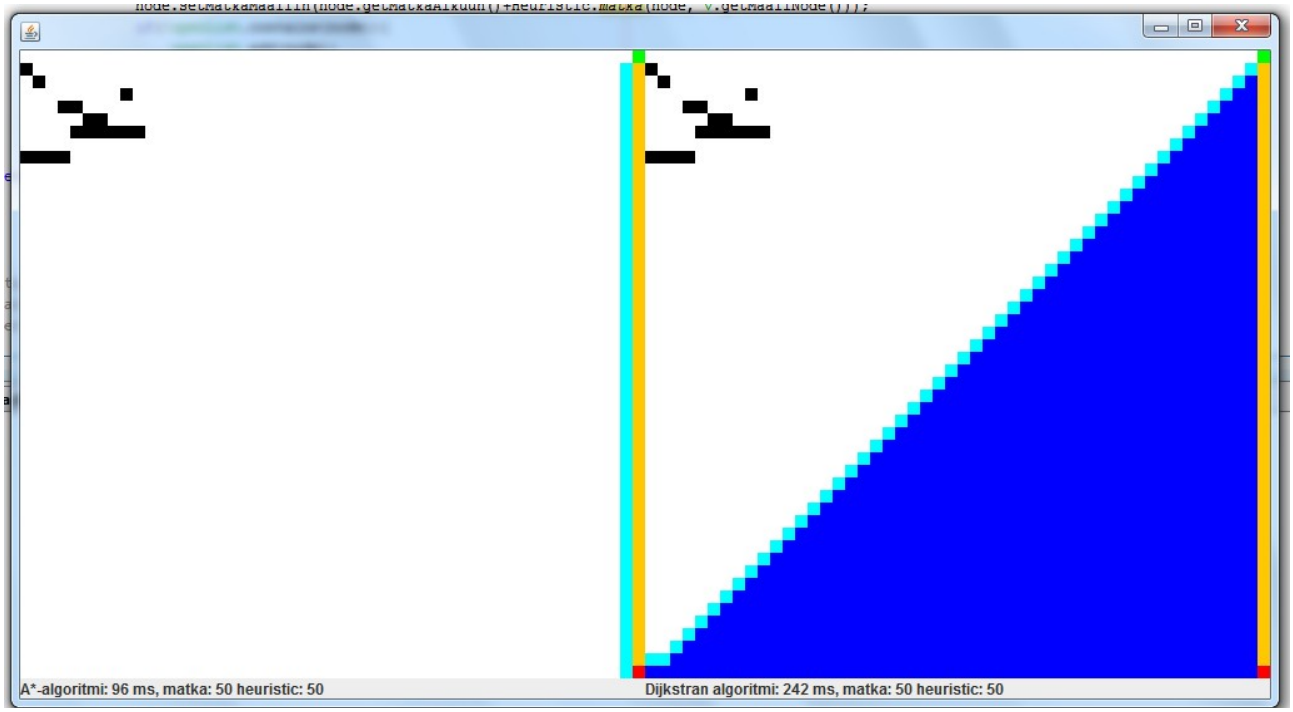
```
    private static int[][] verkko=new int[9][9];
    private static int[][] verkko2=new int[9][9];


        Verkko v1=new Verkko(7, 7, 2, 3, verkko, true);
        verkko2[2][1]=1;  //esteitä
        verkko2[2][2]=1;
        verkko2[1][2]=1;
        verkko2[2][0]=1;
        Verkko v2=new Verkko(7,7,0,0,verkko2, true);
        Verkko v3=new Verkko(7,7,0,0,verkko, true);
        Verkko v4=new Verkko(7, 7, 2, 3, verkko, false);
        verkko2[2][1]=1;  //esteitä
        verkko2[2][2]=1;
        verkko2[1][2]=1;
        verkko2[2][0]=1;
        Verkko v5=new Verkko(7,7,0,0,verkko2, false);
        Verkko v6=new Verkko(7,7,0,0,verkko, false);

        Astar a1=new Astar(v4);
        Astar a2=new Astar(v5);
        Astar a3=new Astar(v6);

        a1.piirra();
        a2.piirra();
        a3.piirra();

        System.out.println("----------------------------");
```

```
        Dijkstra d1=new Dijkstra(v1);
        Dijkstra d2=new Dijkstra(v2);
        Dijkstra d3=new Dijkstra(v3);

        d1.piirra();
        d2.piirra();
        d3.piirra();
```

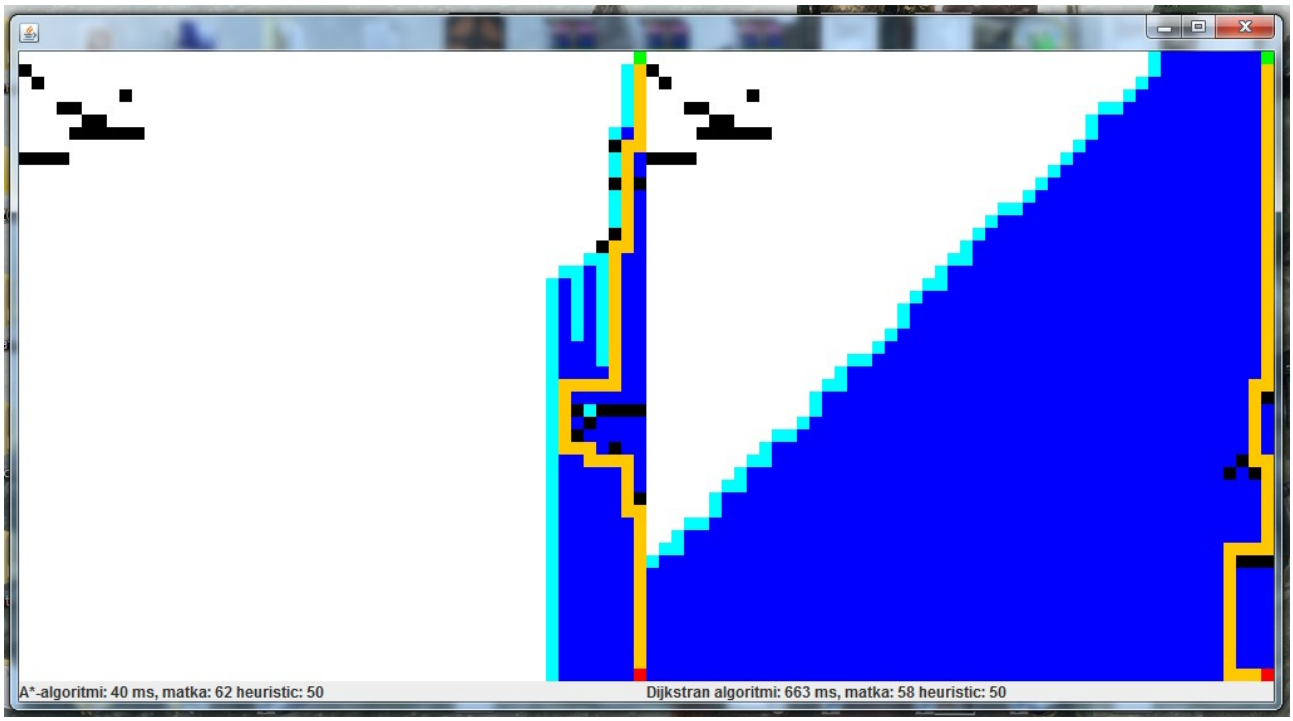
Tämän lisäksi ohjelmaa on testattu myös isommilla verkoilla:

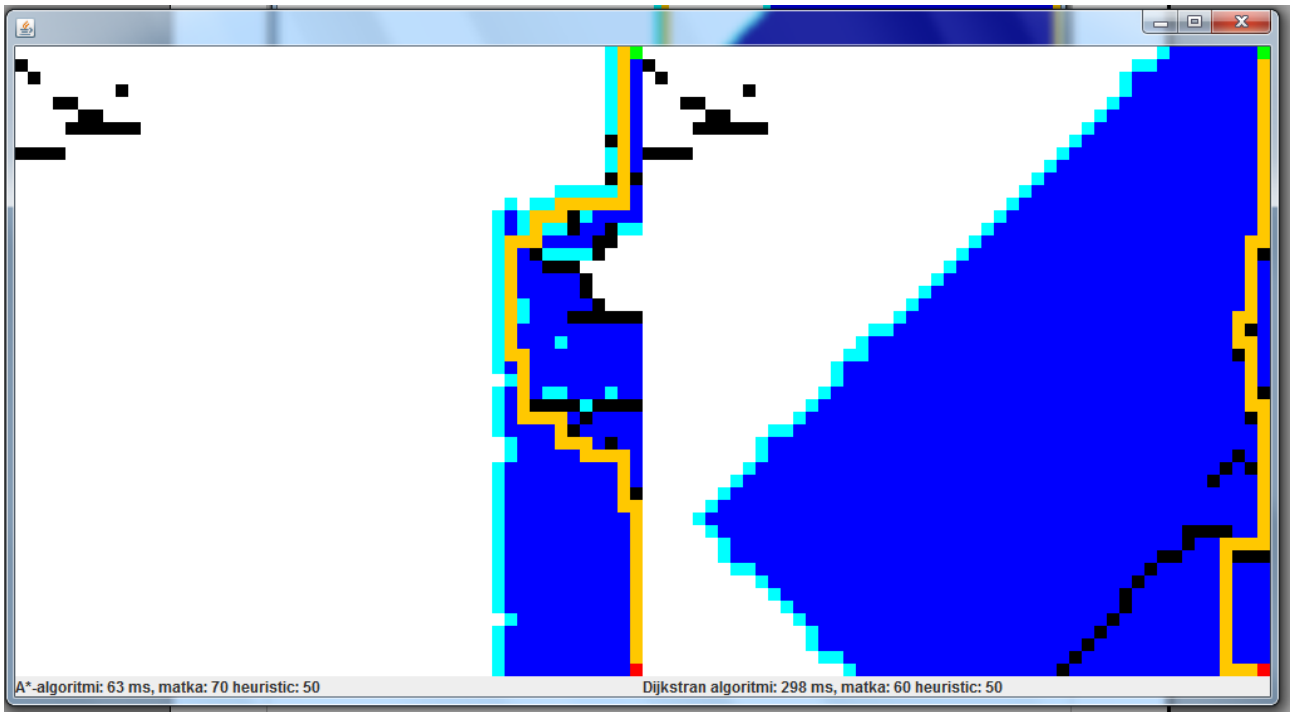
```
private int[][] verkko=new int[50][50];
    private int[][] verkko2=new int[50][50];

        verkko[6][4]=1;  //esteitä
        verkko[6][5]=1;
        verkko[6][6]=1;
        verkko[6][7]=1;
        verkko[6][8]=1;
        verkko[6][9]=1;
        verkko[8][0]=1;
        verkko[8][1]=1;
        verkko[8][2]=1;
        verkko[8][3]=1;
        verkko[5][5]=1;
        verkko[5][6]=1;
        verkko[4][3]=1;
        verkko[4][4]=1;
        verkko[2][1]=1;
        verkko[1][0]=1;
        verkko[3][8]=1;

        verkko2[6][4]=1;  //esteitä
        verkko2[6][5]=1;
        verkko2[6][6]=1;
        verkko2[6][7]=1;
        verkko2[6][8]=1;
        verkko2[6][9]=1;
        verkko2[8][0]=1;
        verkko2[8][1]=1;
        verkko2[8][2]=1;
        verkko2[8][3]=1;
        verkko2[5][5]=1;
        verkko2[5][6]=1;
        verkko2[4][3]=1;
        verkko2[4][4]=1;
        verkko2[2][1]=1;
        verkko2[1][0]=1;
        verkko2[3][8]=1;

        astar=new Astar(new Verkko(0,49,49,49,verkko,false));
        dijkstra=new Dijkstra(new Verkko(0,49,49,49,verkko2,true));
```

Tämä tuottaa seuraavanlaiset kuvat tuloksena



Tästä voidaan vielä muuttaa verkkoja lennossa ja kokeilla erialsia verkkoja, jotta nähdään että algoritmit toimivat.




A*-algoritmi: 63 ms, matka: 70 heuristic: 50
Dijkstran algoritmi: 298 ms, matka: 60 heuristic: 50