

## Testausdokumentti

- \*Mitä on testattu, miten tämä tehtiin
- \*Minkälaisilla syötteillä testaus tehtiin (vertailupainotteisissa töissä tärkätä)
- \*Miten testit voidaan toistaa
- \*Ohjelman toiminnan empiirisen testauksen tulosten esittäminen graafisessa muodossa.
- \*Testaus on ideaalitapauksessa suoritettava ohjelma. Tällöin testi on helposti toistettavissa, mikä helpottaa toteutuksen tekoa jo varhaisessa vaiheessa. On erittäin suositeltavaa käyttää testaukseen JUnitia.

## Testit

- \* Itse toteutettujen tietorakenteiden (tira-paketti) kaikki julkiset metodit on yksikkötestattu, lukuunottamatta Pari-luokkaa.
- \* AStar-haulla on muutama testi (haku-paketti): haku ylipäänsä toimii ja eri laskimet reiteille toimivat.
- \* App-luokassa (käynnistysluokka) on kattavasti debug-metodeja, joilla olen toteutusvaiheessa varmistanut luokkien toimivuutta.
- \* Useissa luokissa on lisäksi mukana debug-metodeja, jotka keräävät ja tulostavat/palauttavat tietoa luokkien toiminnasta.
- \* App-luokan testikäyttöliittymällä voi testata reitinhakua sekä pysäkkiverkossa että satunnaisgeneroidussa verkossa. Reittihaku suoritetaan aina kymmenen kertaa, ja suorituksen jälkeen tulostetaan saatu reitti ja keskiarvo kuluneesta ajasta, pienin aika sekä suurin aika.
- \* Gui-luokan testikäyttöliittymässä voi luoda uuden satunnaisen verkon sekä suorittaa hakuja pysäkki- ja satunnaisessa verkossa. Haun tulokset piirretään.

## Testien tuloksia: suorituskykytestausta

Kuvissa kuljettu reitti on punaisella, käydyt solmut vaaleanpunaisella, alkusolmu keltaisella ja loppusolmu vihreällä.

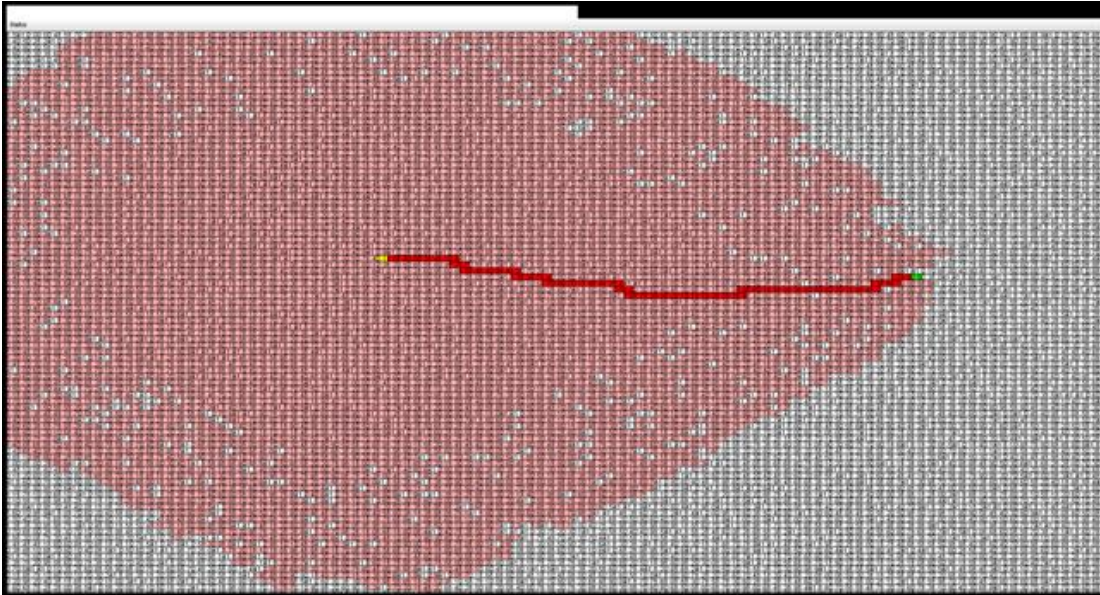
### 1. AStar-haku ja heuristiikka

#### Toistaminen:

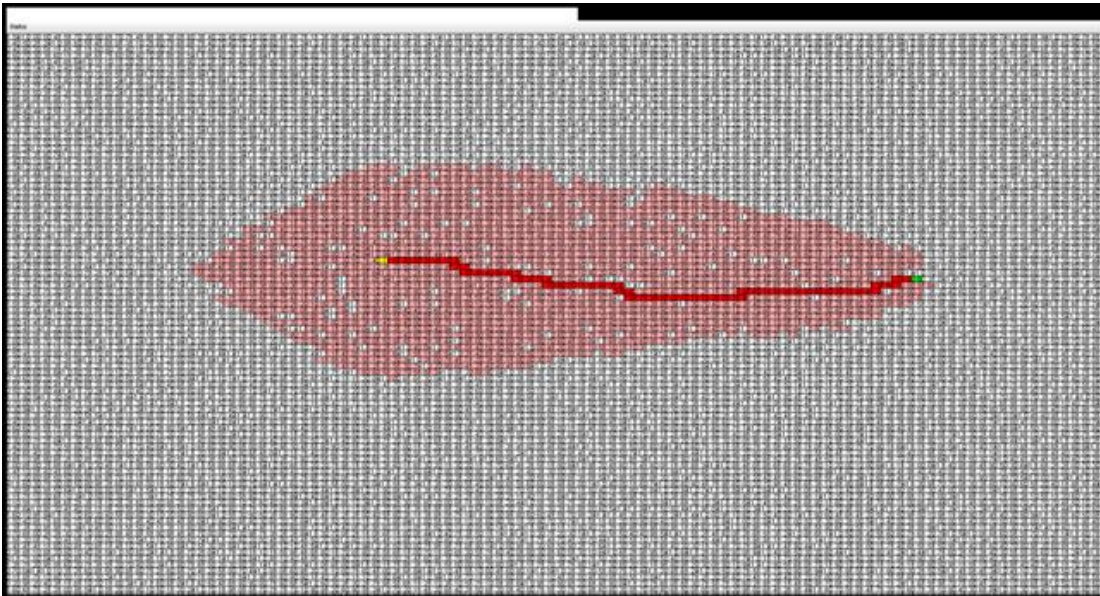
Asetetaan aluksi käyttöön uusi satunnainen verkko syötteellä 300:300:1:2:2:1. Valitaan sitten laskimeksi BFS, suoritetaan haku ja sitten valitaan heuristinen laskin ja suoritetaan haku uudestaan.

Solmujen käsittelyjärjestyksen määrää niiden hyvyys  $g(n)$ : reitin kustannus  $f(\text{alku}, n)$  ja arvioitu jäljellä oleva kustannus  $h(n)$ .

Heuristisen funktion  $h(n)$  arvo vaikuttaa haun toimintaan. Jos  $h(n)=0$ , saadaan leveyssuuntainen haku:



Sama haku suoritettuna siten, että  $h(n) = |n.x - \text{maali}.x| + |n.y - \text{maali}.y|$ :



Solmuja ei ole siis käsitelty niin paljoa turhaan.

Mitä paremmin  $h(n)$  vastaa oikeaa jäljellä olevaa kustannusta  $f(n, \text{maali})$ , sitä tehokkaammin haku toimii. Tätä voi kokeilla valitsemalla verkon siten, että minimiPaino on hyvin suuri. Jos valitaan kerroinPaino siten, että syntyvät verkon painot ovat arvoltaan lähellä yhtä, haku toimii tehokkaasti:

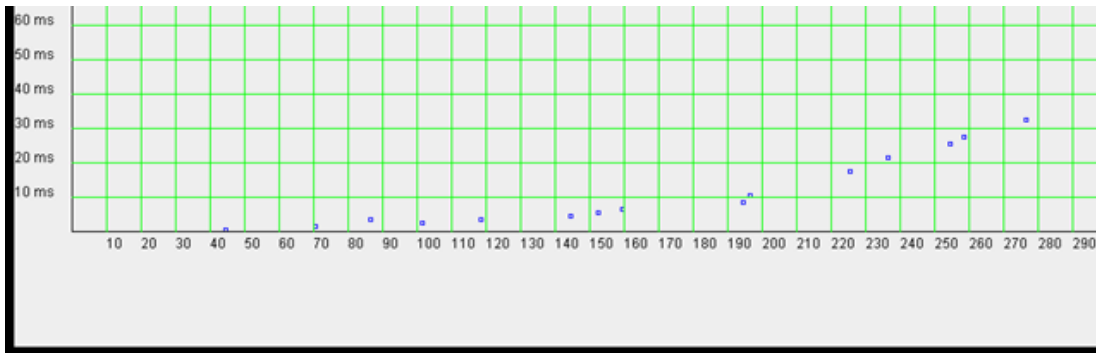


300:300:1:2:2:0 luotu verkko. Suoritetaan näillä testejä-painikkeen mukaiset testit.

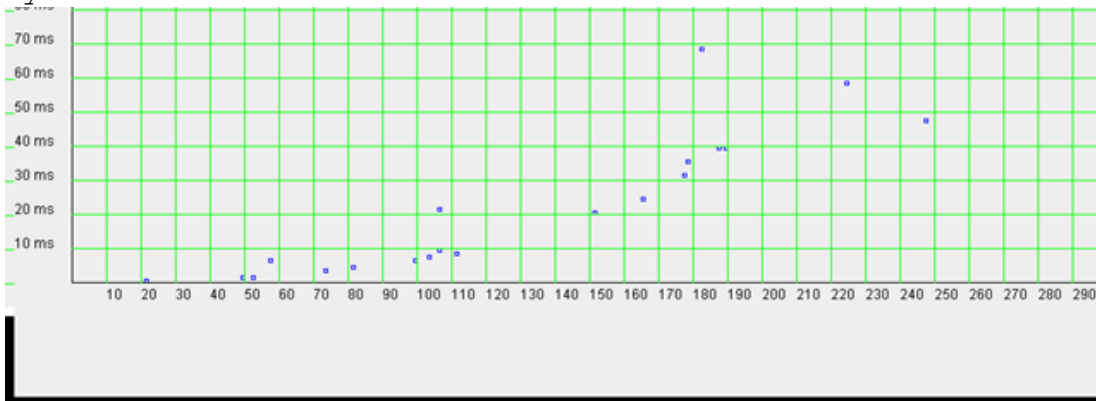
Oletusarvona verkossa voi kulkea vain koordinaattiakselien suuntaisesti. Valinnalla 300:300:1:2:2:0 luotu verkko sallii liikkumisen myös sivuittain.

Minimиаikavaatimus verkossa kulkemiseen on  $O(k*d)$ , jossa  $k$  on naapureiden lukumäärä kullekin solmulle ja  $d$  on reitin pituus. Tavallisessa tilanteessa kaikkien käsiteltyjen solmujen  $n$  naapurit  $k$  lisätään käsittelyjärjestykseen: saadaan aikavaatimus  $O(n*k)$ . Sivuttaisen liikkumisen sallivassa verkossa naapureita on 8, vain akselien suuntaisen liikkumisen sallivassa verkossa 4. Annetulla reitin pituudella  $d$  minimиаikavaatimuksien suhde on siis 2. Jos sivuttaissuuntainen kulkeminen sallitaan, lyhyin mahdollinen reitti on puolet siitä jos kulkeminen estetään. Pahimmillaan sivuttaisen liikkumisen sallivassa verkossa käsitellyksi tulee paljon suurempi määrä solmuja kuin vain akselien suuntaan liikkuen.

Vain akselien suuntaan:

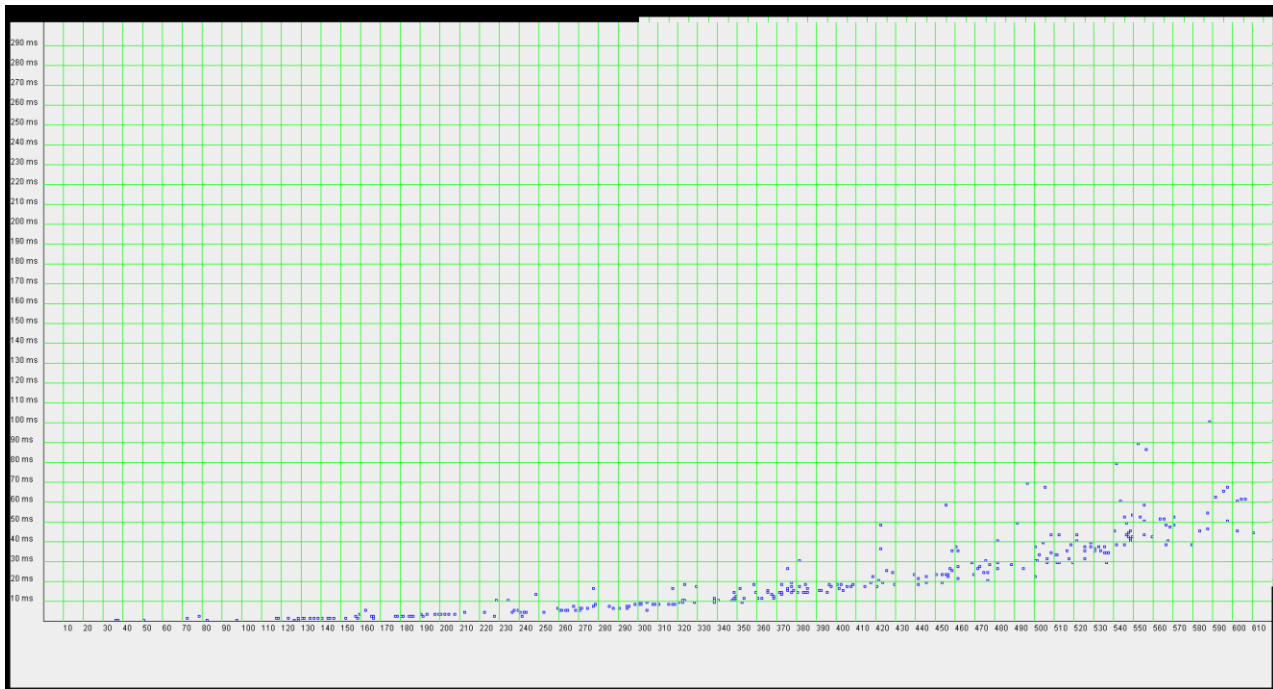


Myös sivusuunnassa:

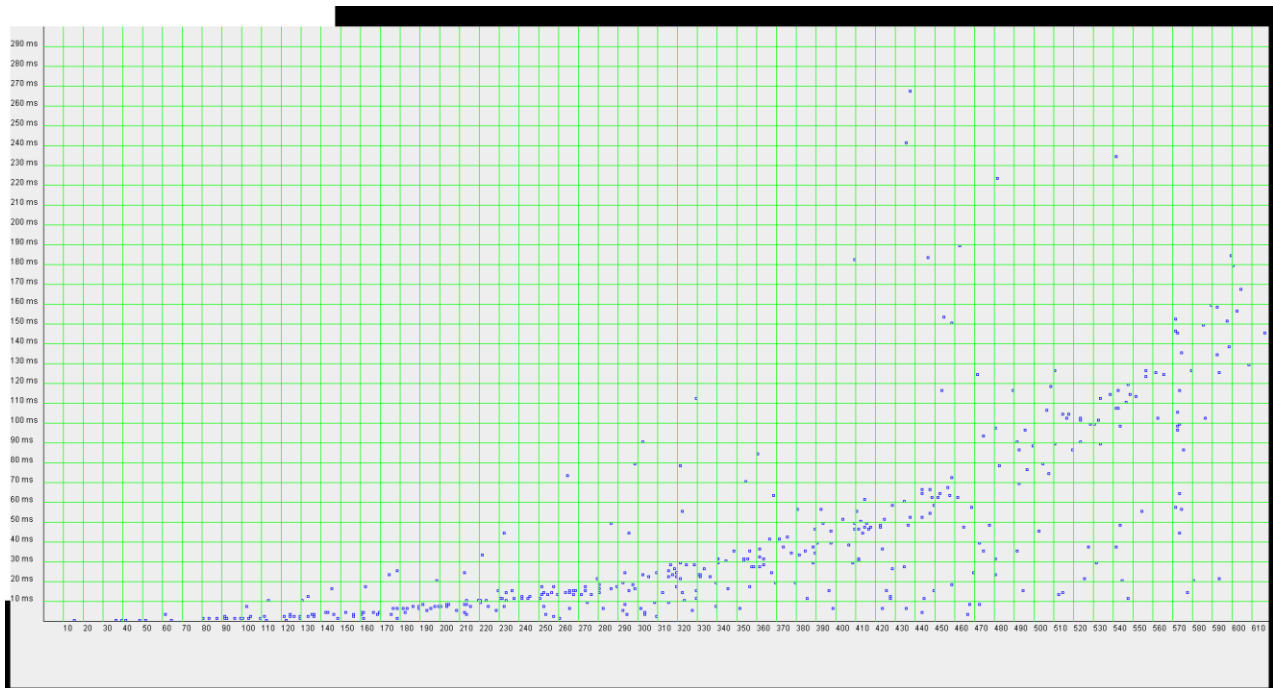


900x900 verkossa 500 näytteellä ero korostuu entisestään. Vain akselien suuntaan:





Pahimmillaan suoritusaika on eksponentiaalinen  $O(k^d)$ , tämä näkyy parhaiten kuvassa, jossa on sallittu liikkuminen myös sivusuunnassa:



### 3. Verkon ja polun koko

Toistaminen: luodaan syötteellä 1000:1000:1:2:2:1 verkko, jonka koko on 1000x1000. Nyt ohjelman piirtämään kuvaajaan ei mahdu kaikki pisteet. Myös kokonaissuoritusaika on melkoinen.

Suurilla verkoilla ja siten myös pitkillä poluilla aikavaatimus kasvaa. Kuvassa 500 hakua 900x900 -verkossa.

