Toteutusdokumentti

1. Ohjelman yleisrakenne

Olen pyrkinyt jakamaan ohjelman kahteen erilliseen osaan: logiikkaan ja sitä ohjailevaan käyttöliittymään. Tärkein komponentti ohjelmassa on siis Logiikka-luokka, sillä sen alta löytyvät kaikki algoritmien käyttämiseen tarvittavat toiminnot.

2. Saavutetut aika- ja tilavaativuudet

Bellman-Ford

```
algoritmi
   ISS()
   for(kaarten lkm)
      for(solmujen lkm)
         loysaaNaapurit()
```

Koska metodin ISS() aikavaativuus on O(solmujen lkm) ja metodin loysaaNaapurit() aikavaativuus on toteutetussa algoritmissa O(1), saamme koko algoritmin aikavaativuudeksi O(solmujen lkm + kaarten lkm * solmujen lkm), mikä oli alkuperäinen tavoitteemmekin.

Dijkstra ja A*
Koska toteutetussa algoritmissa A*:n käyttämän heuristiikkafunktion aikavaativuus on O(1), pitäisi Dijkstran ja A*:n aikavaativuus olla sama. Pseudokoodina:

```
algoritmi
   ISS()
   for(solmujen lkm)
      valitseParasSolmu()
      loysaaNaapurit()
```

Yllä ovat algoritmit aikavaativuuden kannalta yksinkertaistettuna. Metodi valitseParasSolmu() käyttää apunaan minimikeko-tietorakennetta ja on täten aikavaativuudeltaan O(log solmujen lkm). loysaaNaapurit() taas käy läpi kaikki kartan kaaret ja on täten aikavaativuudeltaan O(kaarten lkm). Koko algoritmin aikavaativuus on tällöin O(solmujen lkm * (log solmujen lkm + kaarten lkm)), kuten oli tarkoituskin.