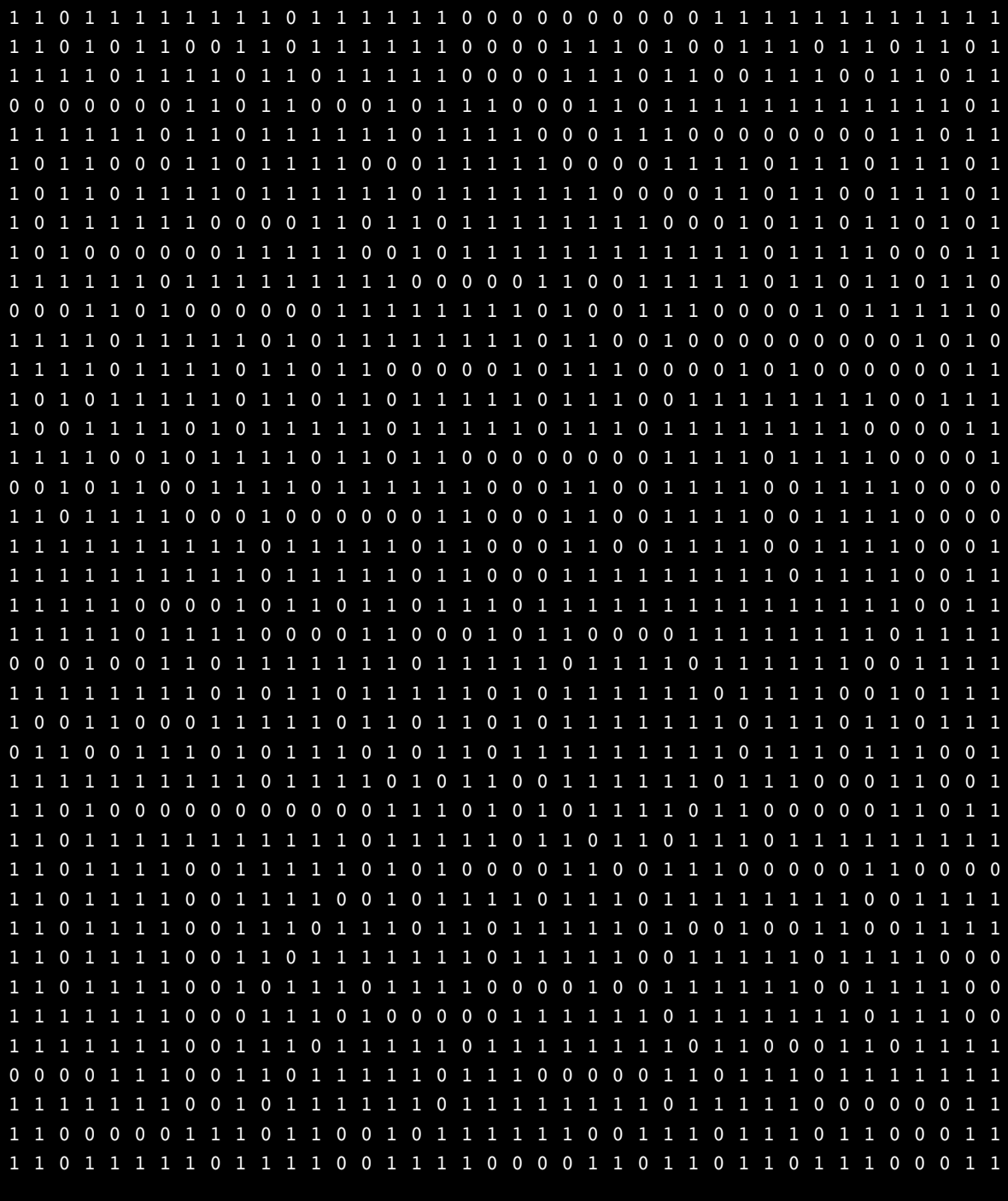
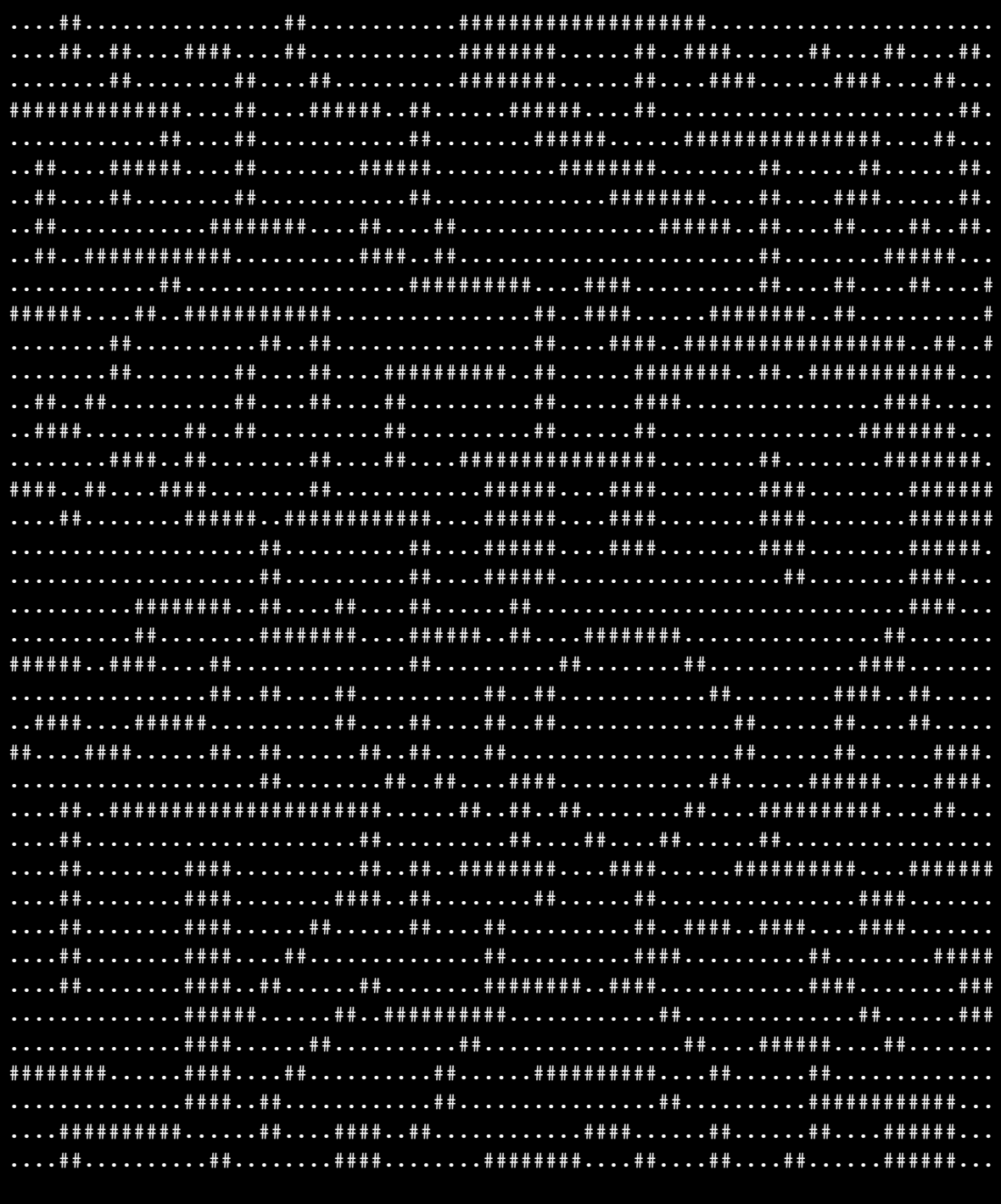
Reitinhaku liikkuvan vihollisen luokse pääsemiseen





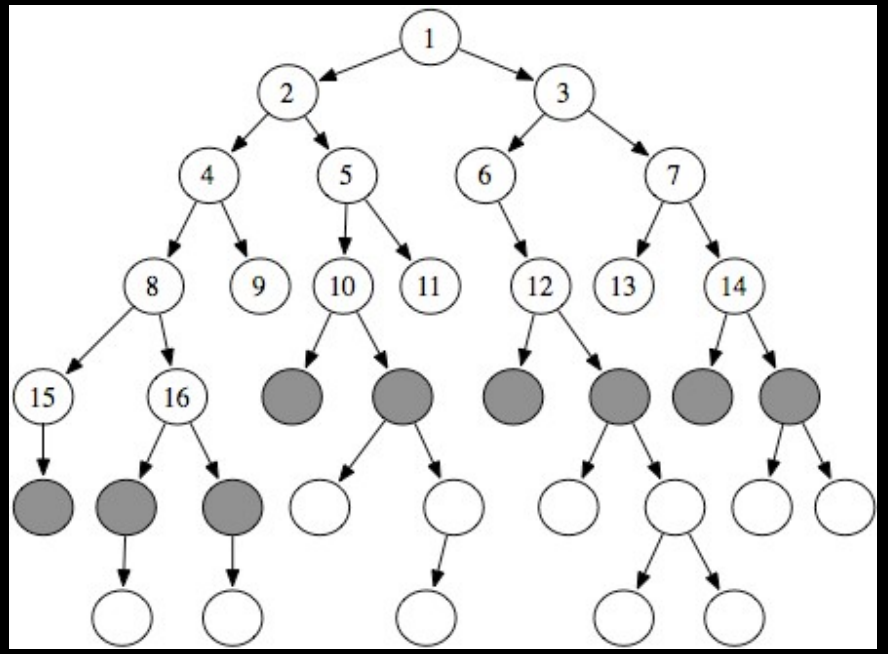
Ohjelman kulku

- Lue luola tiedostosta boolean-taulukkoon
- Generoi nodet (kartan kuljettavat ruudut) ja aseta niille naapurussuhteet
- Käytä A*-hakua jahdattavan hahmon kulkeman reitin määrittämiseen
- Etsi jahtaajan reitti jahdattavan luo, jos sellainen on olemassa

Käytetyt algoritmit: leveyssuuntainen haku

- Lisää alkunode jonoon
- Niin kauan kuin jono ei ole tyhjä
  - Hae jonon ensimmäinen node
  - Tarkista onko kyseessä maali
  - Lisää kaikki naapurit jonoon

- Aikavaativuus $O(b^m)$
- Tilavaativuus $O(b^m)$



Käytetyt algoritmit: IDA*

- “simuloi syvyyssuuntaista hakua” käymällä rekursiivisesti läpi kaikki korkeintaan tietylle syvyydelle vievät puut
- Mikäli maalinodea ei löydy, sallitaan syvemmälle eteneminen

- Aikavaativuus $O(b^m)$
- Tilavaativuus O(bm)

Algoritmeihin tehdyt muutokset: leveyssuuntainen haku

- “maaliksi” annetaan taulukko nodeja
  - Taulukon ensimmäinen alkio on jahdattavan sijainti alussa, toinen alkio sijainti yhden vuoron jälkeen jne
- Jokainen node tietää, monenko askelen päässä alkusolmusta se on
- Yhden askelen jälkeen maalisolmu on maalitaulukon toinen node, toisen askeleen jälkeen kolmas jne.

Algoritmeihin tehdyt muutokset: IDA*

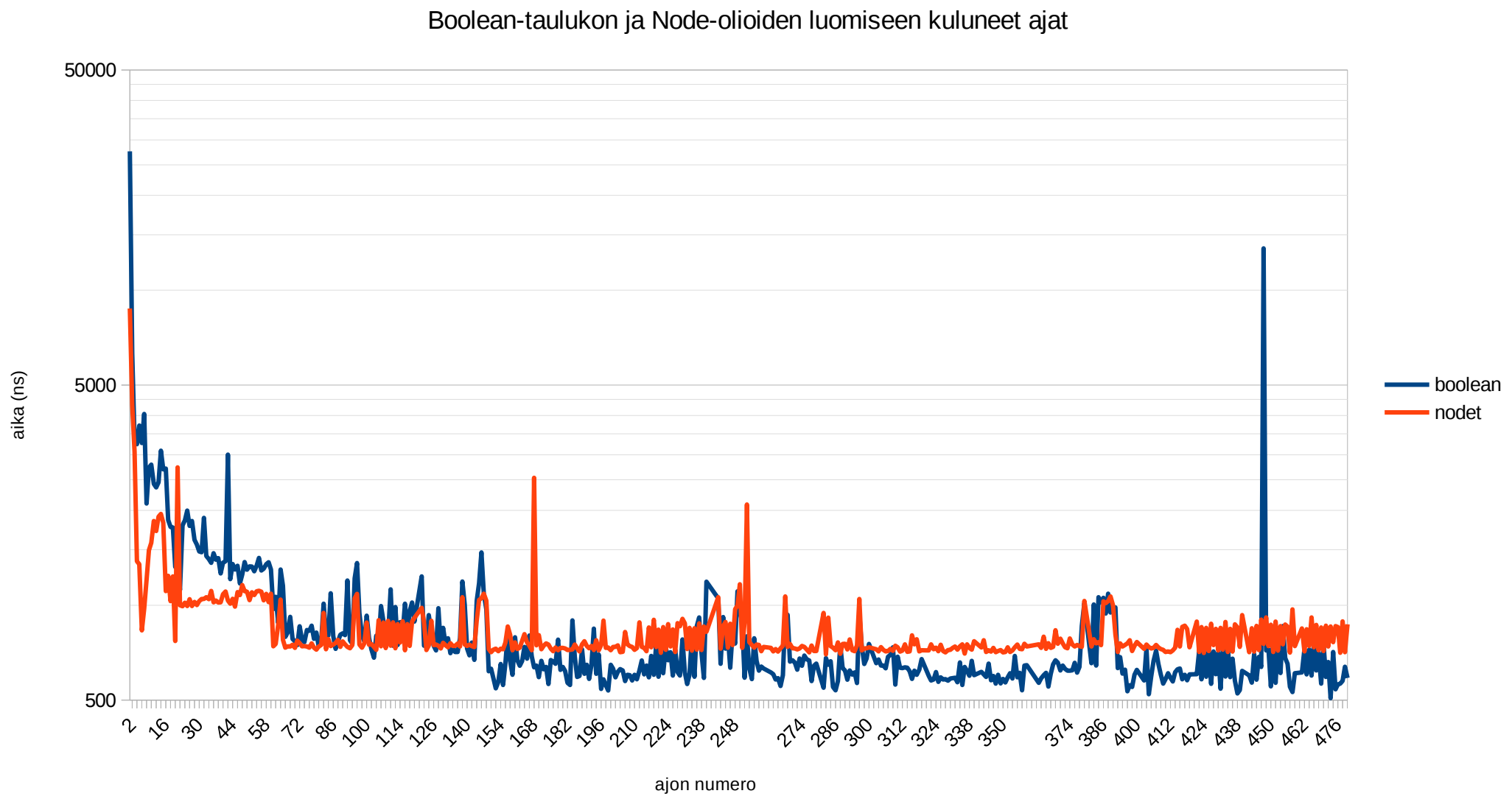
- Maalinode poimitaan jokaiselle syvyydelle erikseen
- Muistin säästäminen ei relevanttia ja nodejen käyminen uudelleen hyödytöntä
  - ➔ pidetään kirjaa matkasta nodeen parhaalla löydetyllä tavalla eikä tutkita nodeja, joihin on nyt tultu pidempää reittiä

Suorituskykytestaus

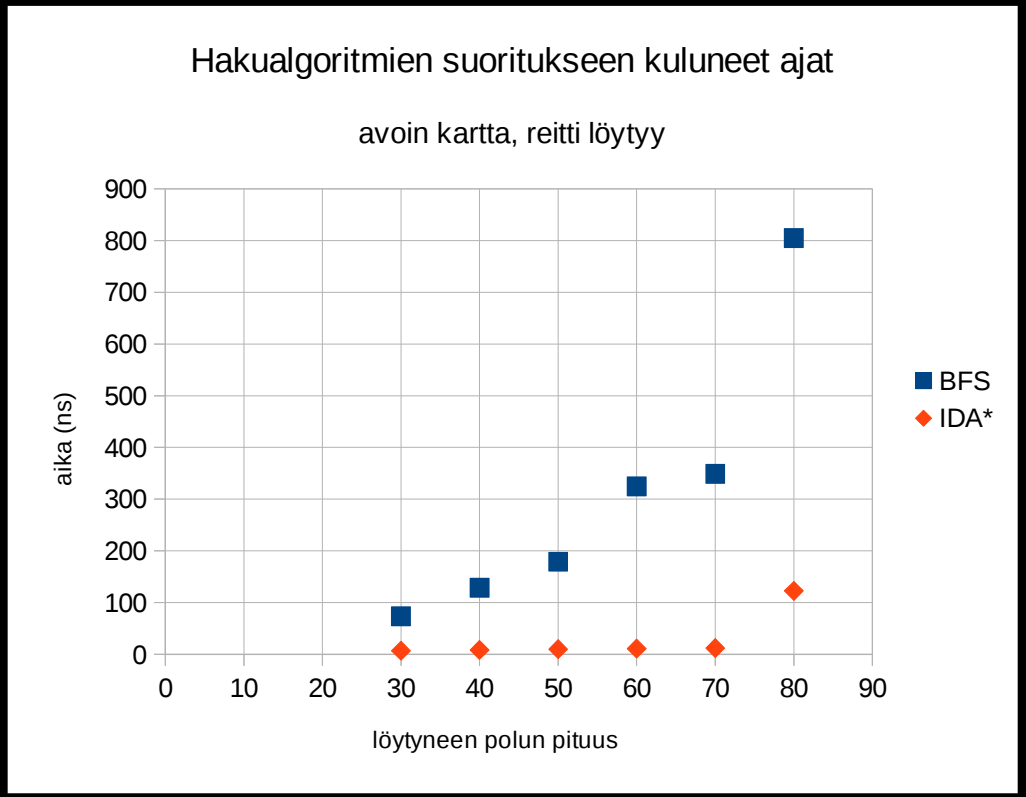
- Erilaiset kartat
  - Avoin vs. labyrinttimainen
- Eri kokoiset kartat

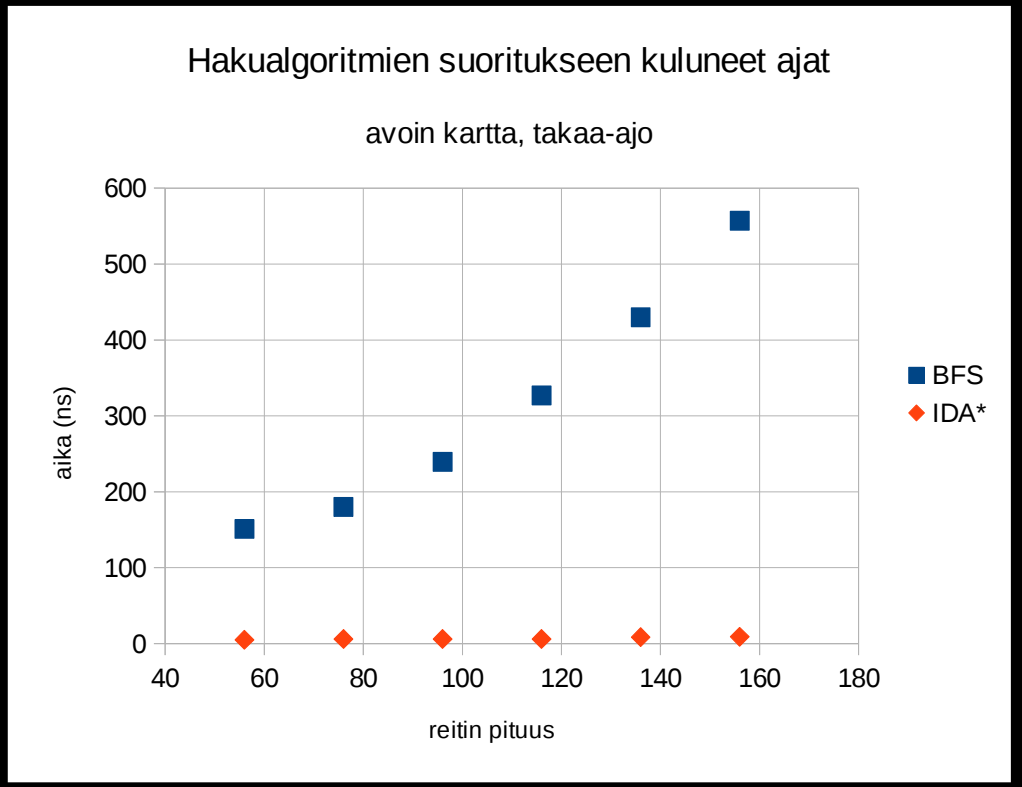
- Mittasin kuhunkin suoritusvaiheeseen kuluvan ajan erikseen

Suorituskykytestaus

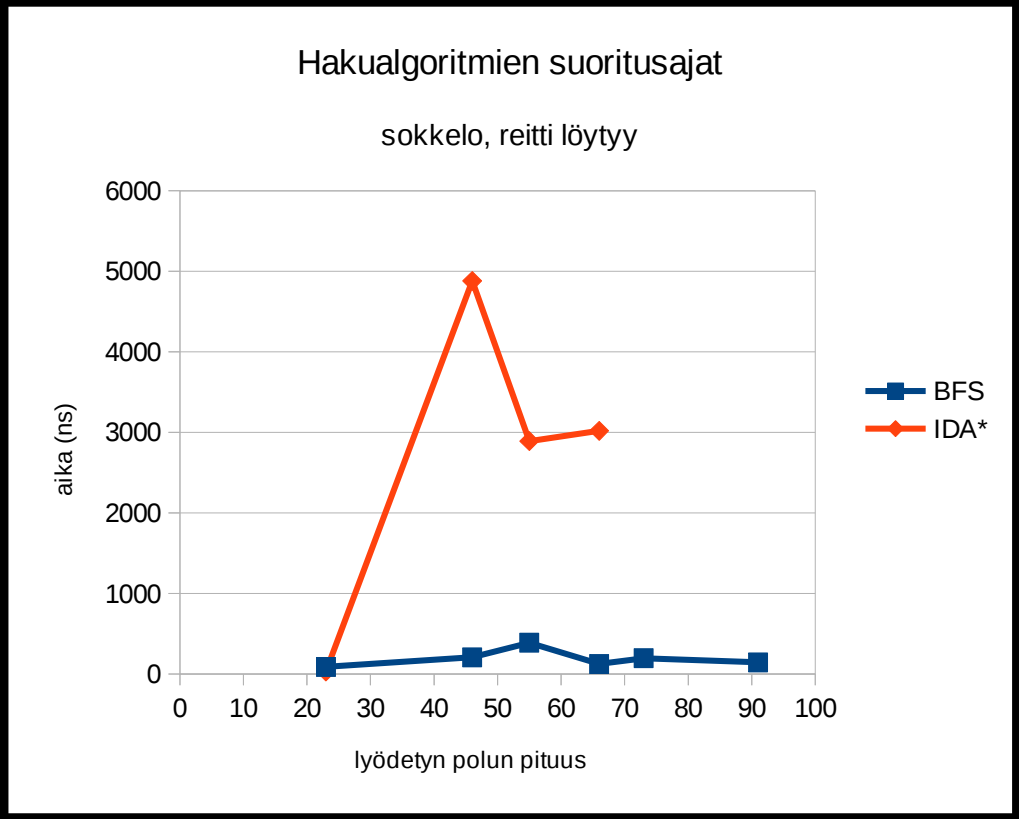


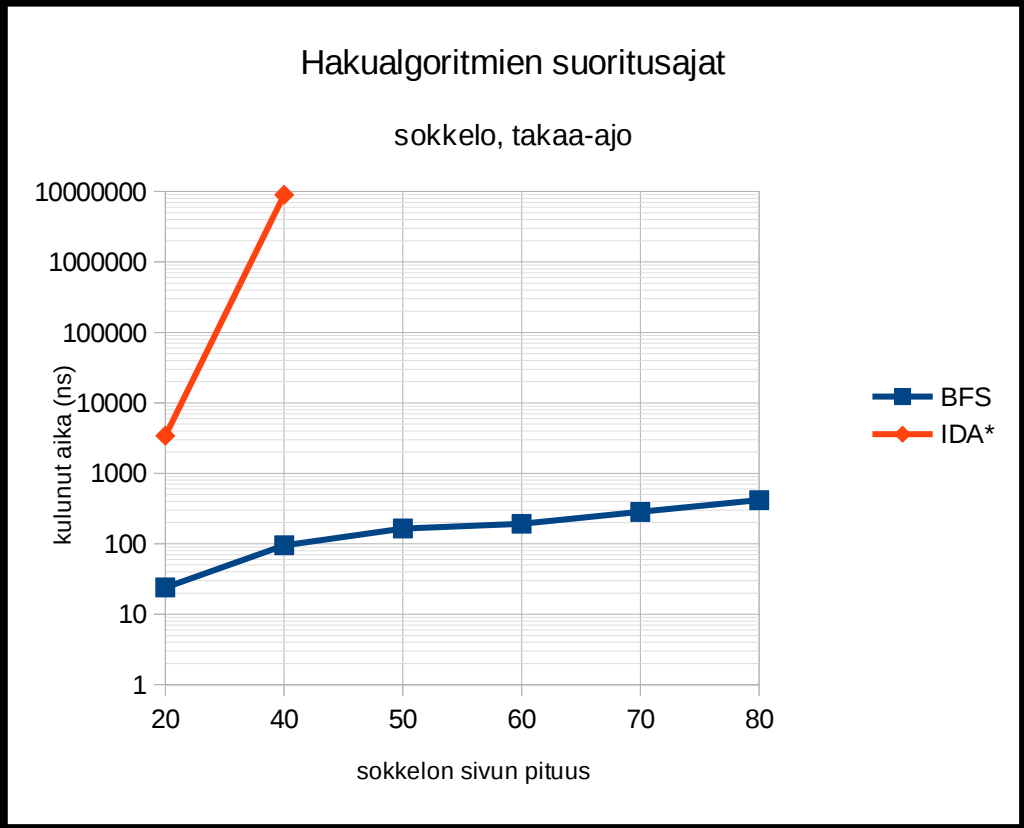
Suorituskykytestaus





Suorituskykytestaus





Huomaa logaritminen asteikko

Päätelmät

- Avoimella kartalla IDA* on hyödyllinen
- Kapeissa käytävissä puolestaan BFS toimii paremmin
-