MÄÄRITTELYDOKUMENTTI **8.5.2013**
Fifteen Puzzle – tietorakenteiden harjoitustyö

@Tero Mäntylä

Käytetyt algoritmit ja tietorakenteet

Tarkoituksena on käyttää 15-Puzzlen ratkaisuun algoritmia, joka löytää lyhimmän mahdollisen siirtosarjan, jolla palapelin palat saadaan oikeille paikoilleen. Tähän tarkoitukseen sopiva tietorakenne on iteratiivisesti syventyvä A* eli IDA*. Haasteena tulee olemaan sopivan heuristiikan löytäminen, jolla arvioidaan 'etäisyyttä' valmiiseen palapeliin. Luultavasti joudun kokeilemaan useampaa kuin yhtä heuristiikkaa. Ensimmäinen mieleen tuleva vaihtoehto on laskea kuinka monta palaa on oikealla paikalla. Toinen ja varmaan toimivampi vaihtoehto voisi olla arvioida Manhattan-etäisyyttä palan oikeaan paikkaan ja summata kaikkien palojen etäisyys yhteen.
IDA*:n heikkoutena on iteraatio joka aiheuttaa samojen laskutoimitusten suorittamisen useasti. Tämä voidaan välttää muistamalla missä kohtaa laskennassa oltiin kun se katkaistiin ja jatkamalla siitä seuraavassa iteraatiossa eteenpäin. Näiden katkaisukohtien määrä kasvaa hyvin nopeasti erittäin suureksi. Tällaisen lähestymistavan käyttö toimii vain jos kaikkein lupaavimmat laskennan kohdat muistetaan ja jatketaan laskentaa niistä. Kuinka tämän voi toteuttaa on vielä aika hämärän peitossa ja voin olla ihan harhateillä. Tässä prioriteettijonosta voisi olla hyötyä.

Erilaisten palakombinaatioiden määrä on varsin suuri (16! eli n. 2*10^13). Kaikki näistä yhdistelmistä eivät ole kuitenkaan ratkaistavissa olevia. Itseasiassa vain puolet näistä kombinaatioista ovat ratkaistavissa. Toiselle puolelle ratkaisun saaminen on mahdotonta. Miksi näin? Hyvää perustelua en vielä osaa antaa. Wikipedian sivuilta löytää hieman selitystä aiheeseen. Keskiviikkona tekemäni löytö liittyy osittain tähän ratkeavuuteen.

Ensitöikseni tiistain haastattelun jälkeen aloin luomaan algoritmia palojen sekoittamiseen. Tein metodit jotka suorittavat palojen siirtelyä ja metodin joka tekee tuhat satunnaista siirtoa. Kun katselin tuloksia näistä sekoituksista aloin ihmettelemään sekoituksen toimivuutta. Tyhjä paikka ei näyttänyt ilmestyvän joihinkin kohtiin palapeliä ollenkaan. Hetken mietittyäni hoksasin miksi näin kävi. Koska käyttämiäni satunnaisia siirtoja on parillinen määrä niin tyhjä paikka voi olla vain parillisen etäisyyden päässä lähtöpaikastaan. Jouduin tekemään muutoksen, jossa sekoittaja arpoo ensin käytetäänkö tuhatta siirtoa vai tuhatta ja yhtä siirtoa.

Syöte ja palautus

Ratkaisija-algoritmi tulee saamaan syötteenään kaksiulotteisen taulukon, jossa on jokin arvottu permutaatio. Palautuksena annetaan siirtosarja, jolla palapeli ratkeaa. Tavoitteena on että ohjelma palauttaisi lyhimmän siirtosarjan ratkaisulle, mutta tämä saattaa osoittautua liian työlääksi toteuttaa kurssin puitteissa. Vaihtoehtoisesti helpoille ja mahdollisesti keskivaikeille palautettaisiin lyhin siirtosarja. Vaikeille tapauksille jouduttaisiin tyytymään suhteellisen hyvään ratkaisuun, mutta jäätäisiin jokunen siirto lyhimmästä mahdollisesta.

Aika- ja tilavaativuudet

IDA*:n aikavaativuutta hallitsee eksponentiaalisesti laajeneva syvyyssuuntainen hakupuu. Vaativuus on varmasti parempi kuin O(4^n), jossa n on tehtyjen siirtojen määrä. Kantaluku on varmasti pienempi kuin 4, koska jos palapelin aukko on keskellä niin haarautuminen voi tapahtua neljään (laatta voidaan siirtää neljästä eri suunnasta tyhjään paikkaan). Kuitenkin enemmistö ruuduista on reunaruutuja, joissa haarautuminen voi tapahtua vain kolmeen ja nurkkaruuduissa vain kahteen. Kun vielä poistetaan edellisen siirron kumoamismahdollisuus eli laatan siirtely edestakaisin päästään kantalukuun joka on jotain välillä 2 – 3. Arviointiheuristiikalla pystytään tätä kantalukua vielä pienentämään.

Hankalasti ratkaistavissa permutaatioissa tarvittavien siirtojen määrä suuri. Vaikeimpia ratkaistavia permutaatioita on 17 kappaletta, niiden järjestämiseen tarvitaan minimissään 80 siirtoa. Tästä voidaan arvioida että ratkaisuun tarvittava aika on suuruusluokkaa 2^80. Ei hyvältä näytä):
Jotta koko hommasta tulisi yhtään mitään niin tarvitaan jotain tehokasta heuristiikka katkoa hakupuusta 'väärään' suuntaan eteneviä haaroja. Tämä arviointi olisi hyvä tapahtua vähintäänkin lineaarisesti palapelin kokoon nähden. Tai oikeammin sanottuna ihan hemmetin nopeasti, koska se tehdään jokaisessa haarautumisessa.

IDA* -algoritmin tilavaativuus on pieni. Se ei muista rekursiossa edellisistä tasoista mitään, joten vaativuusluokka tulee rekursion syvyydestä, joka on suhteessa lyhimmän ratkaisuun johtavan siirtosarjan pituuteen. Eli jos n on ratkaisuun tarvittavien siirtojen määrä niin tilavaativuus IDA* on O(n).

Lähteet

- https://en.wikipedia.org/wiki/15_puzzle
- http://heuristicswiki.wikispaces.com/IDA*
- http://en.wikipedia.org/wiki/IDA*

@ Tero Mäntylä 8.5.2013