Polunhakualgoritmit ja -järjestelmät – referaatti

Rodion “rodde” Efremov

16. syyskuuta 2014

Lyhimpien polkujen haku painotetuissa tai painottamattomissa verkoissa on perustavanlaatuinen ongelma, joka ei ole tärkeä vain itsessään, vaan sillä on sovelluksia joukossa algoritmeja, joissa lyhimmän polun haku on tarvittava alioperaatio; es on $O((m+n)\log n)$. Parannus edelliseen on ilmestynyt vuonna 1987 Fibonacci-keon myötä, jonka minimin poisto käy tasoitetussa (engl. *amortized*) ajassa $O(\log n)$, ja muut sisältöä muokkaavat operaatiot tasoitetussa tai eksaktissa vakioajassa, jolloin paremmaksi Dijkstran algoritmin aikavaativuudeksi on tullut $O(m+n\log n)$.im. Edmonds-Karp algoritmi hakee aina lyhimmät painottamattomat polut verkossa (engl. *residual graph*) ratkaistaakseen maksimivuo-ongelman (engl. *maximum flow problem*). Tietenkin, varsinainen sovellus on reitin optimointi erilaisten mittojen mukaan: toisinaan halutaan päästää pisteestaä A pisteeseen B mahdollisimman pienessä ajassa (lentokone), toisinaan mahdollisimman ekologisesti (polkupyörä), jne.

Vuonna 1959 Edsger W. Dijkstra esitti polynomisessa ajassa toimivan algoritmin [**?**], joka, saatuaan lähtösolmun s, laskee lyhimpien polkujen puun solmusta s kaikkiin muihin saavutettaviin solmuihin painotetuissa verkoissa. Kuten monet muut, Dijkstran algoritmi vaatii prioriteettijonorakenteen (engl. *priority queue*, *heap*), mikä on vain yksi optimointiulottuvuus. Helpoin, tehokkaaksi kutsuttu prioriteettijonorakenne on binäärikeko, joka toteuttaa kaikki sisältöä muokkaavat operaatiot ajassa $O(\log n)$, jolloin Dijkstran algoritmin aikavaativuus

Mitä tulee muihin nopeutustekniikoihin, Hart et al. esittivät $A*$-polunhakualgoritmin [**?**], joka ei lajittele solmut pelkän g-arvon mukaan (solmun u g-arvo on toistaiseksi paras etäisyys lähtösolmusta u:hun), vaan käyttää f-arvoa, jolle $f = g+h$, missä h on solmun optimistinen (eli aliarvioiva) etäisyys

maalisolmuun. Intuitio tämän järjestelyn takana on se, että $A*$ "tietää" mihin suuntaan kannattaa lähteä päästääkseen maalisolmuun, ainakin paremmin kuin Dijkstran algoritmi, joka etenee "kaikkiin suuntiin".

Mikäli kysytään, mikä on tehokkain tapa laskea lyhin polku solmujen s, v välillä, niin todennäköisesti päädytään aikaan $\Theta(N)$, missä N on lyhimmän s, v-polun solmujen määrä. Tällainen tehokkuus kuitenkin vaatii ns. "edeltäjämatriisin' (engl. *predecessor matrix*), jonka voi laskea ajaamalla kaikkien parien algoritmin (engl. *all-pairs shortest paths*). Niistä kaksi tunnetuinta ja helpommin toteuttavaa ovat FLOYD-WARSHALL- ja "Johnsonin" algoritmit, jotka käyvät ajassa $O(V^3)$ ja $O(V^2 \log V + VE)$ vastaavasti, jolloin on selvää, ettei niitäkään pysty ottaa käyttöön, kun verkkona on esimerkiksi kokonaisen valtion tieverkosto, jossa solmuja on reilusti yli 10000.