Polunhakualgoritmit ja -järjestelmät - aihealueen lyhyt kuvaus

Rodion “rodde” Efremov

11. syyskuuta 2014

Vuonna 1959 Dijkstra esitti polynomisessa ajassa toimivan algoritmin, jolla voi etsiä lyhimmän polun kahden mielivaltaisen solmun välillä painotetuissa verkoissa. Algoritmi kasvattaa lyhimpien polkujen puun lähtösolmusta aina siihen asti kunnes puu kohtaa maalisolmun. Kesti noin yhdeksän vuotta kunnes ongelma sai uuden ratkaisun: $A*$ käyttää optimistisen (etäisyyttä aliarvioivan) heuristiikkafunktion, jonka myöten algoritmi “tietää”, mihin suuntaan kannattaa lähteä päästääkseen maalisolmuun, ja, täten, tarjoaa paremman tehokkuuden kuin Dijkstran algoritmi. $A*$:n heikkoutena suhteessa Dijkstran algoritmiin on kuitenkin juuri heuristiikkafunktion tarve, jota, ongelmasta riippuen, ei aina pystytä tyydyttämään. Lisäksi, $A*$ sai kilpailevan algoritmin: ironisesti, juuri Dijkstran algoritmin *kaskisuuntaisesta* (engl. *bidirectional*) versiosta, joka kasvattaa kaksi lyhimpien polkujen puuta molemmasta päätesolmusta kunnes kaksi puuta kohtaavat. Ellei oteta kantaa käytettävään prioriteettijonoon, yksisuuntainen haku tekee $\sum_{i=0}^{N} d^i$ verran työtä siinä missä kaksisuuntainen tekee vaan $2\sum_{i=0}^{N/2} d_i$, missä lyhimmän polun kaarien määrä on N ja d on verkon solmujen keskiarvoinen aste. Kaksisuuntaisuus on kannattava nopeutus aina kun heuristiikkafunktiota ei ole mahdollista johtaa, eli Dijkstran algoritmi ja myöt leveyssuuntainen haku hyötyvät ko. järjestelystä. Kaksisuuntainen $A*$ ei kuitenkaan välttämättä aina tekee vähemmän laskentatyötä kuin yksisuuntainen versionsa, sillä on mahdollista että kaksisuuntaisen version kaksi hakupalloa ohittavat esteen verkossa eri puolilta, näin tehden enemmän työtä kuin yksisuuntainen $A*$.