Tietorakennevertailun testiraportti

Marko Haanranta

15. kesäkuuta 2013

Projektissa toteutin binäärikeon, binomikeon, fibonaccin keon ja AVL-puun. Lisäksi käytin vertailussa apunani priority queue:ta. Ja suoritin yksikkö- ja suorituskykytestausta.

1 Yksikkötestaus

Jotta tietorakenteita oli mahdollista vertailla keskenään niin ensin oli varmistuttava, että toteuttamani tietorakenteet toimivat oikein. Tätä varten tein kattavat yksikkötestit kaikille toteuttamilleni tietorakenteille. Lause- ja haaraumakattavuuden varmistamiseen käytin cobertura työkalua.

Binomikeko, fibonaccin keko ja AVL-puun solmuilla on useita eri parametreja joten ne oli järkevää toteuttaa olioina. Näille solmu luokille ei tehty omia yksikkötestejä, koska niissä ei ole muita metodeja kun gettereitä ja settereitä joiden testaamista ei katsota tarpeelliseksi. Coberturaa käyttäen voi kuitenkin havaita, että näidenkin luokkien testikattavuus on lähes 100%

2 Suorituskykytestaus

Tätä varten testasin delete ja insert metodien nopeuksia suurilla syötteillä. Pienimmän alkion haku, kun ei poisteta mitään on kaikilla rakenteilla sen verran nopea operaatio etten katsonut sen testaamisen olevan tarpeellista.

2.1 Tilavaativuudet

Tilavaativuus kaikkien toteutettujen tietorakenteiden osalta on O(1), koska mikään toiminnallisuutta toteuttava metodi ei käytä kuin vakiotilaisia apumuuttujia.

2.2 Aikavaativuudet

Teoreettiset aikavaativuudet poikkeavat hieman toisistaan. Kaikkien toteutettujen tietorakenteiden poisto operaation teoreettinen aikavaativuus on O(logn). Fibonaccin keon lisäys operaation teoreettinen aikavaativuus on O(1)(amortized).

Javan valmiin priority queue-tietorakenteen lisäys ja poisto operaatioille luvataan O(logn) aikavaativuus, joten näin voin kätevästi vertailla toimivatko toteuttamini tietorakenteet niin nopeasti kuin niiden teoreettisesti tulisi toimia.

2.3 Testisyötteet

Testeissä

2.4 Testitulokset