FACULTEIT INGENIEURSWETENSCHAPPEN
EN ARCHITECTUUR

Gevorderde algoritmen

Bert De Saffel

Master in de Industriële Wetenschappen: Informatica Academiejaar 2018–2019

UNIVERSITEIT
GENT

Inhoudsopgave

Deel I

Theorie

Hoofdstuk 1

Inleiding

Het vak gevorderde algoritmen behandelt vier luiken:

1. **Gegevensstructuren**. Dit onderdeel behandelt meer efficiënte zoekbomen zoals Rood-zwarte bomen, Splay trees, B-trees en meerdimensionale gegevensstructuren zoals Quadtrees en k-d trees. Er wordt ook bekeken hoe zeer grote datastructuren (die niet vollledig in het geheugen passen) behandelt worden.

2. **Grafen**. Een uitbreiding op de grafentheorie. Onderwerpen zoals stroomnetwerken, _ToDo: aanvullen wanneer stof gezien is

3. **Strings**. Dit hoofdstuk gaat dieper in op stringfuncties. Een aantal onderwerpen zijn efficiënte zoekmethoden, de theorie achter reguliere expressies en hun grammatica en het samenvatten van teksten.

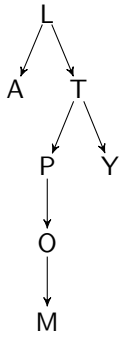
4. **Complexe problemen**. Dit onderdeel behandelt de NP-problemen. Wat zijn ze? Hoe kunnen we NP-problemen transformeren naar een eenvoudiger probleem? Hoe kan men benaderde oplossingen gebruiken?

Hoofdstuk 2

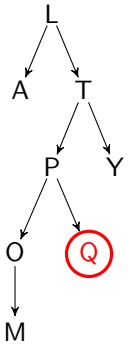
Efficiënte zoekbomen

2.1 Herhaling binaire zoekbomen

Een binaire boom is een belangrijke boomstructuur waarin drie operaties belangrijk zijn: *zoeken, toevoegen en verwijderen*. Deze drie operaties worden kort herhaald op volgende binaire boom:

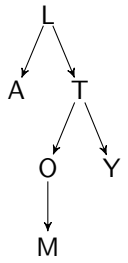


- Zoeken naar een knoop komt overeen met binair zoeken. Stel dat we knoop P zoeken dan weten we dat dit groter is dan L dus gaan we naar rechts, en kleiner is dan T dus gaan we naar links.
- Toevoegen van een knoop komt overeen met zoeken naar die knoop, en dan die knoop op die positie toe te voegen. Stel dat we een knoop Q willen toevoegen, dan zal het binair zoeken leiden dat deze knoop in het rechterkind van P moet komen.



- Verwijderen van een knoop onderscheidt drie gevallen:

    1. *De knoop heeft geen kinderen*: dan kan de knoop eenvoudig verwijderd worden. Indien knoop Q verwijderdt wordt krijgen we terug de originele boom.
    2. *De knoop heeft één kind*: In dit geval moet de ouder die naar de te verwijderen knoop wijst, nu wijzen naar het enige kind van de te verwijderen knoop. Indien we knoop P verwijderen, moet het linkerkind van knoop T nu wijzen naar knoop O.



    3. *De knoop heeft twee kinderen*: In dit geval

Een binaire boom is niet altijd efficiënt. Indien de sleutels ingelezen worden in volgorde, zal de binaire boom een gelinkte lijst voorstellen, waardoor de operaties $O(n)$ worden i.p.v. $O(\lg n)$. Dit probleem werd opgelost door bomen die zichzelf zo evenwichtig mogelijk trachten te houden. Er zijn drie soorten van deze bomen:

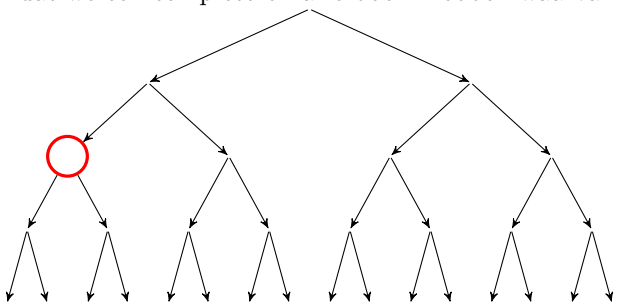
1. **Rood-zwarte bomen**. Dit soort bomen tracht elke operatie steeds efficiënt te maken door ervoor te zorgen dat de structuur van de boom nagenoeg perfect blijft. Dit is de meest robuste methode.

2. **Splay trees**. Deze soort wordt de vorm van de boom meermaals aangepast, zodat de structuur van de boom nooit slecht is. Bij deze bomen is het gemiddelde geval steeds efficiënt (geamortiseerd), maar een individuele operatie kan soms slecht uitvallen.

3. **Randomized search trees** (treap). Deze boomstructuur zorgt ervoor dat de boom zo willekeurig mogelijk blijft, ongeacht de volgorde van toevoegen of verwijderen. Dit zorgt ervoor dat de verwachtingswaarde van elke operatie $O(\lg n)$ is.

2.2 Rood-zwarte bomen

Stel dat we een complete binaire boom hebben waarvan we de rode knoop willen verwijderen:

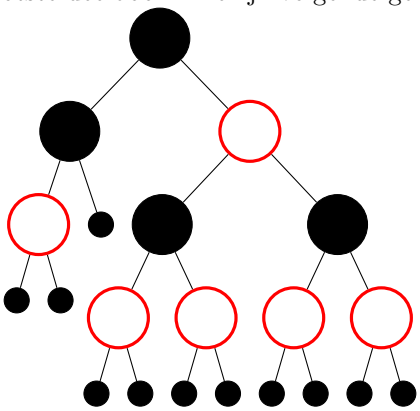


_ToDo: doesnt make sense

Formeel kan een rood-zwarte zoekboom als volg gedefinieerd worden:

- Het is een binaire zoekboom.
- De knopen zijn rood of zwart (deze extra informatie kan bijgehouden worden met slechts één bit, aangezien er slechts twee kleuren zijn).
- Een rode knoop heeft *nooit* een rood kind.
- Elke virtuele knoop is zwart.
- De zwarte hoogte z van een knoop k is hetzelfde voor elk pad naar een virtuele knoop startend vanaf k, met k niet meegerekend. De zwarte hoogte is het aantal zwarte knopen op dit pad.

Door deze voorwaarden is het gegarandeerd dat de kleinste deelboom minstens half zo diep is als de grootste deelboom. Bekijk volgende geldige rood-zwarte zoekboom:



De zwarte hoogte van de wortel is 2, want voor elk pad naar een virtuele knoop is het aantal zwarte knopen 2. Een deelboom met wortel w en zwarte hoogte z zal minstens $2^z - 1$ inwendige knopen bevatten.