Algoritmen en Datastructuren 2: Doorlopen van Grafen

Youri Coppens
Youri.Coppens@vub.be

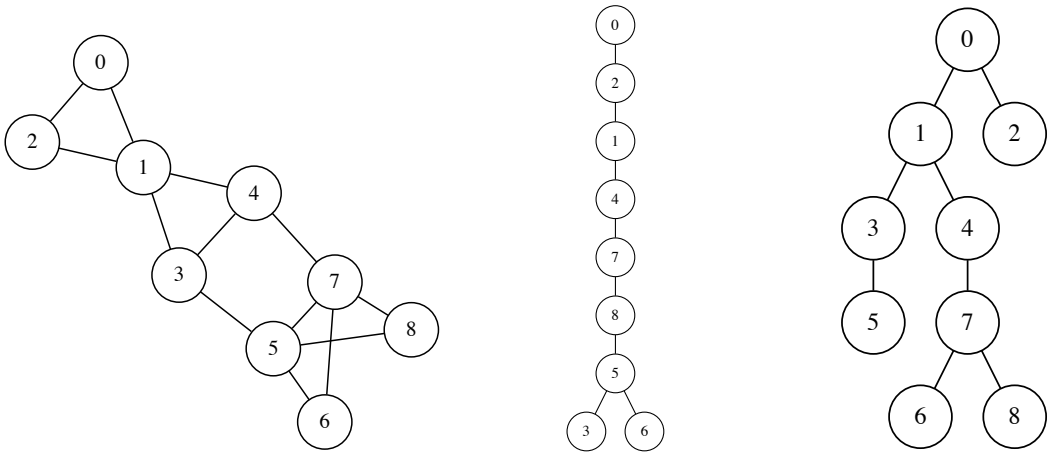
Inleiding

Het doel van dit WPO is niet om graaftraversals aan te passen, maar ze wel correct te leren gebruiken. Al de code voor het doorlopen van grafen vind je terug in `a-d/graph-traversing`. Hier vind je implementaties van Depth-first Traversal (DFT) en Breadth-first Traversal (BFT) voor de verschillende types grafen. De traversals zijn hogereorde procedures die inwendig gebruik maken van de graaf-procedures `for-each-node` en `for-each-edge`. Je kan procedures meegeven die op een bepaald "moment" tijdens de traversal uitgevoerd worden om zo bepaalde zaken te berekenen voor grafen. Daarnaast is het ook mogelijk om een lijst mee te geven als optioneel argument die de knopen bevat van waaruit de traversal dient te starten. Enkele basisvoorbeelden van het gebruik van DFT en BFT vind je in de bestanden `dft-characterisations.rkt` en `bft-characterisations.rkt`. Deze traversals zullen ook vaak de fundamenten vormen van de algoritmes die je in Hoofdstuk 11 en 12 zult zien.

De relevante bestanden voor dit WPO zijn: `dft-unweighted.rkt, dft-labeled.rkt, bft.rkt en bft-labeled.rkt`. Merk op dat de verscheidene implementaties slechts verschillen in subtiele zaken in de bodies (bijv. het aantal argumenten in de hulpprocedures). De implementaties van BFT en DFT gebruiken eveneens een programmeerconcept dat jullie nog niet bekend is, nl. *continuaties*. Om hier wat feeling voor te krijgen, is het sterk aangeraden om het bijgeleverde bestand `intermezzo-continuations.rkt` door te nemen. Op die manier kan je de exacte implementatie van DFT en BFT beter begrijpen.

1 Ouder-kind-relaties in DFT- & BFT-bossen

Zowel DFT als BFT genereren tijdens het doorlopen van de graaf een boom of een bos (**DFT-bos** of **BFT-bos**). Een voorbeeld hiervan is gegeven in Figuur 1. Het is duidelijk dat een traversal van DFT of BFT over dezelfde graaf verschillende bomen kan opleveren.



(a) De `kite` graaf (b) DFT-boom (c) BFT-boom

Figuur 1: Voorbeeld van een DFT- en BFT-boom door de respectievelijke traversal te starten vanuit knoop 0.

Vanuit het standpunt van eenders welke knoop in een DFT- of BFT-boom kunnen we een **reeks van voorouders** opsommen. Deze reeks reikt van de huidige knoop tot aan de **wortel** van de DFT- of BFT-boom. Alleen de wortel zelf heeft geen reeks van voorouders. Bijvoorbeeld:

- Knoop 7 in de DFT-boom in Figuur 1 heeft als voorouders $\{4, 1, 2, 0\}$
- Knoop 7 in de BFT-boom in Figuur 1 heeft als voorouders $\{4, 1, 0\}$

Uit dit voorbeeld kan je afleiden dat deze reeksen van voorouders in een DFT- of BFT-boom overeenstemmen met mogelijke **paden** in de graaf tussen knoop 7 en knoop 0 (de wortel).

Maak nu gebruik van de DFT- en BFT-implementaties om **tijdens het doorlopen** van de graaf voor elke knoop zijn reeks van voorouders af te beelden. Je moet hiervoor de functies `dft` en `bft` niet aanpassen. Je kan dit doen door enkel gebruik te maken van de procedures `root-discovered`, `node-discovered`, etc. tijdens een oproep naar `dft` of `bft`.

Je kan je oplossing testen op de grafen `connected`, `three-cc` en `kite`. Deze werden reeds geïmporteerd in de file `Vraag1-Tests.rkt`, meegeleverd met deze opgave.

1. Doe dit voor **DFT**. Gebruik tijdens de traversal een lijst om voorouders bij te houden.

2. Doe dit voor **BFT**. Een lijst zal nu niet meer werken, waarom niet? Gebruik een node-indexed vector om tijdens de traversal voorouders bij te houden.

2 DFT & BFT als routeplanner

Met een gelabelde graaf kunnen we op een eenvoudige manier een wegennetwerk voorstellen. In de file `Vraag2-Tests.rkt` hebben we een gelabelde graaf geïmporteerd die het Belgische wegennetwerk voorstelt (`highways-belgium`). Een visualisatie van de graaf vind je in de cursustekst (figuur 10.13 op pagina 84). In dit wegennetwerk houden we geen rekening met de afstand van de wegen. We drukken de lengte van een traject tussen twee steden uit als **het aantal tussenliggende steden**.

We zullen nu deze graaf gebruiken om een eenvoudige routeplanner te maken. Gegeven twee knopen uit de graaf (`start` en `goal`), schrijf een functie die alle tussenliggende steden teruggeeft. Je kan hiervoor een hulpfunctie schrijven die alsook de namen van de wegen teruggeeft, bijvoorbeeld `Gent --[E40]--> Aalst --[E40]--> Brussel`.

1. Gegeven dat de graaf `highways-belgium` een wegennetwerk voorstelt, welke graafrepresentatie lijkt jou het beste om deze graf voor te stellen: Adjacency List of Adjacency Matrix? Waarom?

2. Plan de route door gebruik te maken van DFT.

3. Plan de route door gebruik te maken van BFT.

4. Welke traversal (DFT of BFT) lijkt jou het meest geschikt voor deze toepassing als routeplanner? Waarom?