# Algoritmen en Datastructuren 2

## Project Dynamisch Programmeren

Maximaal palindromische paden in gerichte grafen

Ruben Wambacq

Universiteit Gent

28 Oktober, 2018

# Inhoud

# Algoritme in pseudocode

Het algoritme werkt met 2 functies, waarvan de eerste (maxPalindroom) de tweede (evalueer) aanstuurt.

Ook is er een globale variabele (rooster), een 2D-array met sequenties van indices waarin wordt bijgehouden wat het langste palindroom is dat begint bij het rijnummer en eindigt bij het kolomnummer. Stel dat bv het langste palindroom dat gaat van index i tot index j in de graaf gelijk is aan de nodes met indices [i, a, b, c, j], dan wordt die lijst opgeslagen in rooster[i][j].

```
maxPalindroom () {
        volgendeIteratie = [(int start, int einde)]
        volgendeIteratie.add(([0..n], [0..n]))
        while (volgende niet leeg){
                temp = [(int start, int einde)]
                for each x in volgendeIteratie {
                        temp.add(evalueer(x))
                }
                volgendeIteratie = temp
        }
}
```

```
[(int, int)] evalueer ((int start, int einde)) {
        If ( start == einde ){
                rooster[start][einde] = start
        } else if ( nog geen palindroom in rooster[start][einde] && eind is buur van start ){
                rooster[start][einde] = [start, einde]
        } else {
                if ( label start == label einde ) {
                        check voor alle buren van start a en alle voorgangers van einde b:
                                if( rooster[a][b] == palindroom &&
                                        lengte rooster[a][b] + 2 > lengte rooster[start][einde]){
                                        rooster[start][einde] = start + rooster[a][b] + einde
                                }
                }
        }
        return [(start, alle buren einde)]
}
```

## Theoretische vragen

1. **Bewijs dat jouw algoritme voor acyclische grafen correct is.**
   Doordat elk mogelijk pad binnen de graaf volledig wordt afgelopen, zullen alle mogelijke combinaties van begin- en eindnodes (gevonden door het volgen van de gerichte bogen) zullen worden nagekeken in het algoritme, waardoor elk mogelijk "woord" in graaf zal worden gecheckt.

   Daardoor wordt het bewijs gereduceerd tot het bewijzen dat, indien een woord door het algoritme wordt beoordeeld en het een palindroom is, het zal worden opgeslagen (in de 2D-array).

Dit kan gemakkelijk worden bewezen door het beschouwen van een recursieve definitie van een palindroom:
Een woord is een palindroom indien het teken aan het begin en het einde van het woord gelijk is en ook het woord zonder deze twee tekens (indien dat er is) voldoet aan deze definitie van een palindroom.

Het algoritme dat ik gebruik zal nakijken voor elke combinatie van begin- en eindnodes of deze twee gelijk zijn en dan ook of het tussenliggende woord een palindroom is. Deze tweede voorwaarde wordt op dynamische wijze nagekeken en uit de 2D-array gehaald, waardoor dit niet telkens opnieuw zal moeten worden berekend.

Doordat de woorden in volgorde van lengte worden nagekeken en in het rooster worden gestopt (dus eerst alle woorden van lengte 1, dan de woorden van lengte 2, 3, enz.) zal bij het beschouwen van een woord met lengte > n steeds al in het rooster ingevuld staan of het woord van lengte n een palindroom is.

2. **Bepaal en bewijs de complexiteit van jouw algoritme voor acyclische grafen.**
    a. **In functie van het aantal toppen n**
    De ondergrens van de uitvoeringstijd zal op n liggen, dit is namelijk het geval waar elke node verschillend is, dan zal in de evalueer functie niet worden verder gegaan bij elke node die wordt nagekeken, waardoor enkel de strings van lengte 1 worden ingevuld in het rooster (dat is dus de hoofddiagonaal), wat neerkomt op n stappen.

    De bovengrens zal worden gevonden in een graaf met de maximale hoeveelheid links tussen toppen. Daar hebben we n checks voor strings van lengte 1, (n-1) checks voor strings van lengte 2, dit gaat zo verder tot 1 check voor een string van lengte n, namelijk de string die alle nodes bevat.

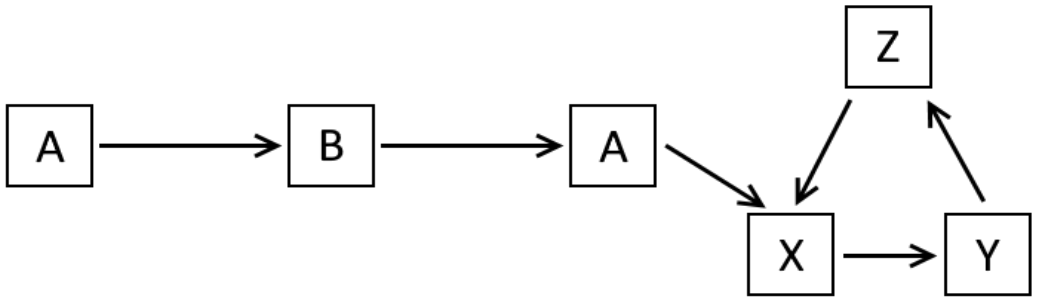
    Dit gaat zorgen voor een complexiteit van
    n + (n-1) + … + 2 + 1 = (n * (n+1))/2
    wat neerkomt op een kwadratische ($O(n^2)$) complexiteit.

b. **In functie van het aantal toppen n en het aantal bogen m**
Voor de complexiteit met de toppen en de bogen is er weinig verschil. Dit zal ook neerkomen op een kwadratische complexiteit, er worden namelijk even veel checks gedaan, maar doordat het in functie van m+n moet worden uitgedrukt, zal het uiteindelijke resultaat minder groot zijn, het wordt namelijk nog gedeeld door een getal tussen 1 en 2 , doordat het aantal bogen tussen 0 en n-1 ligt. Het uiteindelijke resultaat moet dus worden uitgedrukt a.d.h.v. een waarde die tussen n en 2n ligt.

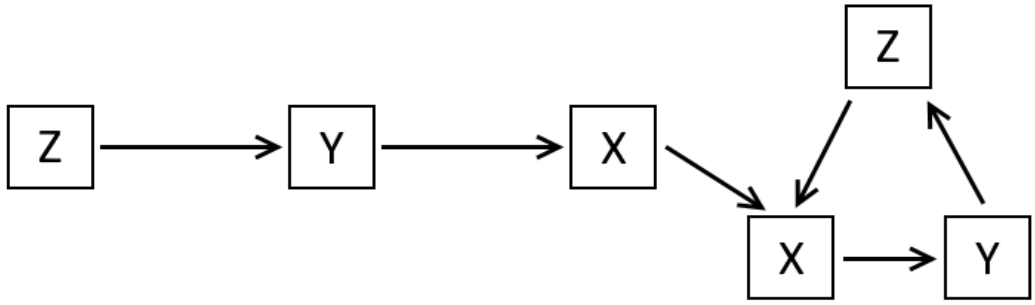
**3. Geef een voorbeeld van de volgende situaties en leg het uit, of bewijs dat het onmogelijk is.**

a. **Een acyclische graaf die palindromisch onbegrensd is**:
Dit zal niet bestaan. Het langst mogelijke palindroom in een acyclische graaf met n toppen is er een van lengte n, indien alle toppen van de graaf deel uitmaken van het langste palindroom. Er zal geen palindroom van lengte groter dan n bestaan in deze graaf. Aangezien dat juist de definitie is van palindromische onbegrensdheid, wil dit dus zeggen dat een acyclische graaf onmogelijk palindromisch onbegrensd kan zijn.

b. **Een gerichte graaf met cykel die palindromisch begrensd is:**
Deze gerichte graaf bevat een cykel, namelijk bij de nodes met de labels X, Y en Z, maar is palindromisch begrensd, aangezien het grootste palindroom in deze boom van lengte 3 is, meer bepaald ABA. Er zal geen groter palindroom gevonden kunnen worden doordat de letters in de cykel in de cykel niet gelijk zijn en dus nooit een palindroom zullen vormen.



c. **Een maximaal palindromisch pad dat een cykel bevat:**
Het maximaal palindromisch pad in deze gerichte graaf met cykel is ZYXXYZ, dit pad bevat de nodes uit de cykel. Maar aangezien de labels van de nodes in deze cykel niet gelijk zijn, zal de graaf palindromisch begrensd zijn en zullen er dus geen langere palindromen kunnen gevonden worden.




d. **Een palindromisch onbegrensde graaf zonder palindromische cykels:**
Dit zal niet bestaan. In 3 a is al bewezen dat een acyclische graaf niet palindromisch onbegrensd kan zijn, dan zou het in dit geval enkel onbegrensd kunnen worden door het toevoegen van een niet-palindromische cykel aan een acyclische graaf. De enige plaatsen waar het invoegen van een niet-palindromische cykel een verschil zou kunnen maken is aan het begin van een palindroom, aan het einde van een palindroom of in het midden van een palindroom. Geen van deze opties zal er voor zorgen dat een palindroom oneindig lang wordt, aangezien de cykel op zich niet palindromisch is.