Classificatie van Markovbeslissingsketen

Smit, L.

Classificatie van Markov Beslissingsketens
Bachelorscriptie

Laurens Smit

10 augustus 2007

Inhoudsopgave

1 Inleiding **3**

2 Definities **4**

3 Voorbeelden **7**

4 Begrippen en algoritme Markovketen **9**

5 Algoritmes voor Markov Beslissingsketens **17**

6 $\mathcal{NP}$-volledigheid van unichain/multichain **28**

7 Decompositie van Markov Beslissingsketens **33**

1 Inleiding

Als afsluiting van mijn bacheloropleiding wiskunde heb ik onder begeleiding van Lodewijk Kallenberg een scriptie geschreven over de classificatie van Markov Beslissingsketens.
Allereerst zal ik in deze scriptie behandelen wat Markov Beslissingsketens zijn en op welke manieren deze te classificeren zijn. In de derde paragraaf staan voorbeelden van deze classificatie. Daarna worden algoritmes behandeld voor de classificatie en wordt van een algoritme de $\mathcal{NP}$-volledigheid bewezen. Tenslotte wordt kort decompositie van Markov Beslissingsketens besproken.

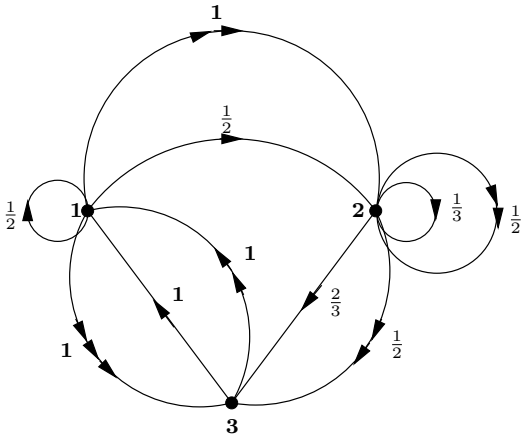
De probleemstellingen bij een gegeven Markov Beslissingsketen die behandeld worden in deze scriptie zijn ook als volgt:

1. (a) Hoe bepaal je of het model communicerend of niet communicerend is?
   (b) Als het niet communicerend is, hoe bepaal je of het zwak communicerend is?
   (c) Bestaan er polynomiale algoritmes voor deze classificatieproblemen?
   (d) Als je geen polynomiale algoritmes kunt vinden, zijn deze problemen dan $\mathcal{NP}$-volledig?
2. (a) Hoe bepaal je of het model irreducibel, unichain of multichain is?
   (b) Bestaan er polynomiale algoritmes voor deze classificatie?
   (c) Als je geen polynomiale algoritmes kunt vinden, zijn deze problemen dan $\mathcal{NP}$-volledig?

2 Definities

Een Markov Beslissingsketen is een soort van opstapeling van gewone Markovketens. In elke toestand $i \in E$, met E de eindige toestandsruimte kies je uit een eindige actieverzameling $A(i)$ een actie a met de daarbij behorende overgangskansen $p_{ij}(a)$ met $i, j \in E$. Verder wordt bij Markov Beslissingsketens gebruik gemaakt van een opbrengst $r_i(a)$ voor elke actie a in toestand i. Deze opbrengst is niet van belang voor de rest van deze scriptie en zal ook niet meer genoemd worden.
Wanneer we in iedere toestand $i \in E$ een actie a kiezen dan geeft de samenstelling van acties een strategie, deze noemen we f en de gekozen acties noemen we dan $f(i)$. De Markovketen die zo ontstaat geven we aan met $P(f)$ met overgangskansen $\{P(f)\}_{ij} = p_{ij}(f(i))$ voor alle $i, j \in E$.

In figuur 1 staat een plaatje van een Markov Beslissingketen, met:
$E = \{1, 2, 3\}$, $A(1) = \{1, 2, 3\}$, $A(2) = \{1, 2\}$, $A(3) = \{1, 2\}$, $p_{11}(1) = p_{12}(1) = \frac{1}{2}$, $p_{12}(2) = p_{13}(3) = 1$, $p_{22}(1) = \frac{1}{3}$, $p_{23}(1) = \frac{2}{3}$, $p_{22}(2) = p_{23}(2) = \frac{1}{2}$, $p_{31}(1) = p_{31}(2) = 1$. De overige overgangskansen zijn 0.



Figuur 1: voorbeeld van Markov Beslissingsketen

Een enkele pijl geeft de eerste actie aan, een dubbele de tweede, enzovoort. De overgangskansen $p_{ij}(a)$ staan bij de pijlen. Bij verdere figuren zal de bovenstaande legenda niet worden weergegeven. Alles is uit de figuur te halen.

Er zijn enkele manieren om Markov Beslissingsketens onder te verdelen. Allereerst kun je dit doen door te kijken naar de mate van communiceren tussen punten. Hierbij ontstaan 3 klassen:

- *communicerend*
  Een Markov Beslissingsketen heet communicerend als er voor iedere $i, j \in$

E een strategie f_1 is zodat in de Markovketen behorende bij deze strategie j bereikbaar is vanuit i en andersom. Dat wil zeggen: $\{P(f_1)^t\}_{ij} > 0$ voor zekere t en $\{P(f_1)^s\}_{ji} > 0$ voor zekere s, voor iedere $i, j \in E$.

- *zwak communicerend*
  Een Markov Beslissingsketen heet zwak communicerend als er een onderverdeling $E = E_1 \cup E_2$ is met $E_1 \cap E_2 = \emptyset$ en E_1 een gesloten communicerende klasse onder een bepaalde strategie en E_2 een (mogelijk lege) verzameling transiënte toestanden onder alle strategieën. E_1 heet gesloten als $p_{ij}(a) = 0$, $\forall i \in E_1$, $\forall j \notin E_1$ en $\forall a \in A(i)$.

- *niet-communicerend*
  Een Markov Beslissingsketen heet niet-communicerend als hij niet communicerend is. Ofwel: $\exists\ i, j \in E$ zodat i niet bereikbaar is vanuit j voor elke strategie f.

Verder kun je de onderverdeling maken in verschil in ergodische structuur. De ketens behoren nu tot een (of meerdere) van de volgende drie klassen:

- *irreducibel*
  Een Markov Beslissingsketen heet irreducibel (ook wel: volledig ergodisch) als deze communicerend is voor elke strategie. Dit houdt dus in dat er voor iedere strategie één recurrente klasse is en geen transiënte toestanden.

- *unichain*
  Een Markov Beslissingsketen heet unichain als hij voor iedere strategie één recurrente klasse heeft met daarbuiten een (mogelijk lege) verzameling van transiënte toestanden.

- *multichain*
  Een Markov Beslissingsketen heet multichain als er een strategie is zodanig dat de bijbehorende Markovketen minimaal 2 ergodische klassen heeft.

Opmerkingen:

Het is eenvoudig in te zien dat:
communicerend$\Rightarrow$*zwak communicerend*: neem de verzameling transiënte toestanden leeg.
irreducibel$\Rightarrow$*unichain*: neem de verzameling transiënte toestanden leeg.

Onderstaande tabel geeft aan waar een Markov Beslissingsketen onder kan vallen, en waaronder niet. Als een geval niet mogelijk is staat in welke stelling dit terug te vinden is en als een geval wel mogelijk is staat in welke figuur een voorbeeld staat.

	Communicerend (zw.-communi.)	Zw. communicerend, niet communicerend	Niet-communi niet zw. communi.
Irreducibel	altijd=stelling 1 figuur 2	niet mogelijk stelling 1	niet mogelijk stelling 1
Unichain, niet irreduc.	mogelijk figuur 3	mogelijk figuur 5	niet mogelijk stelling 2
Multichain	mogelijk figuur 4	mogelijk figuur 6	mogelijk figuur 7

Stelling 1 *irreducibel⇒communicerend*

Bewijs
Irreducibel is communicerend voor elke strategie. (volgt direct uit de definitie)

Gevolg Het geval irreducibel, zwak communicerend en niet commmunicerend is niet mogelijk en het geval irreducibel, niet-communicerend en niet zwak communicerend ook niet.

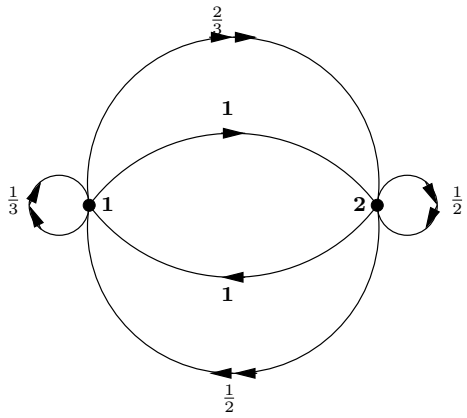
Stelling 2 *Unichain, niet irruducibel⇒zwak communicerend.*

Bewijs
Als een Markov Beslissingsketen unichain, maar niet irreducibel is, dan is er onder elke strategie een recurrente klasse R_f behorende bij strategie f en mogelijk een aantal transiënte toestanden. Er is bij minstens een strategie f zeker één transiënte toestand, want anders is de keten irreducibel.
Twee recurrente klassen R_f en R_g kunnen niet disjunct zijn, immers als R_f en R_g disjunct zijn, dan is er ook een strategie die deze twee klassen beide bevat en deze strategie heeft dan twee recurrente klassen.
R_f en R_g hebben dus minstens een gemeenschappelijke toestand, waarmee elke toestand uit beide communiceren, dus alle toestanden uit $R_f \cup R_g$ communiceren met elkaar.
$C = \cup_f R_f$ is dus een communicerende verzameling. Deze klasse is bovendien gesloten: als hij niet gesloten is dan is er minstens één R_f niet gesloten, want er is dan een overgang van $i \in R_f$ naar $j \notin R_f$: tegenspraak.
Deze verzameling C is nu E_1 uit de definitie van zwak communicerend en de overige toestand(en) zijn E_2. Dus *unichain, niet irreducibel⇒zwak communicerend.*

Gevolg
Het geval unichain, niet irreducibel, niet-communicerend en niet zwak communicerend is niet mogelijk.

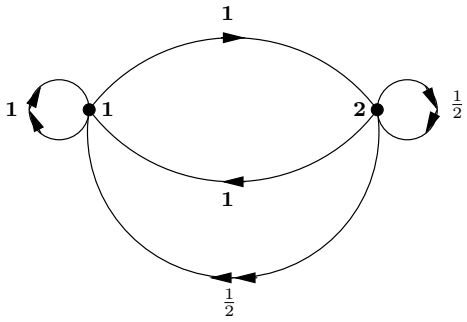
3 Voorbeelden

In deze sectie staan voorbeelden van elke mogelijke klasse:



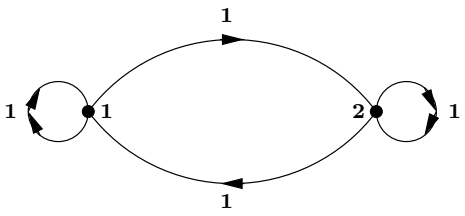
Figuur 2: irreducibel (dus ook communicerend)

In figuur 2 is duidelijk dat je vanuit toestand 1 naar toestand 2 kan komen en andersom, onder alle strategieën. Er is één ergodische klasse en er zijn geen transiënte toestanden. Deze Markov Beslissingsketen is irreducibel en dus ook communicerend.



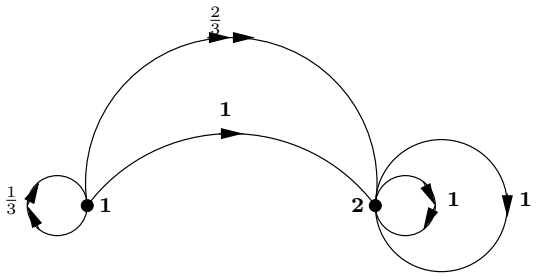
Figuur 3: communicerend, unichain (niet irreducibel)

In figuur 3 is te zien dat je vanuit toestand 1 kan komen in toestand 2 en andersom: dit kan via de strategie die in beide toestanden actie 1 kiest. Echter met een strategie die in toestand 1 actie 2 neemt is het niet mogelijk om vanuit toestand 1 naar toestand 2 te komen. Toestand 1 is dan een irreducibele (ergodische) klasse en toestand 2 is transiënt voor iedere strategie, dus er is sprake van een unichain Markov Beslissingsketen.



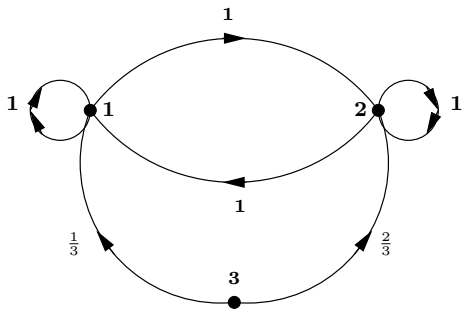
Figuur 4: communicerend, multichain

Figuur 4 geeft een communicerende, multichain Markov Beslissingsketen, immers: nemen we in beide toestanden actie 1, dan is de keten irreducibel (dus communicerend) en nemen we in beide toestanden actie 2 dan geeft dit een keten met twee ergodische klassen.



Figuur 5: zwak communicerend en niet communicerend, unichain (niet irreducibel)

In figuur 5 is te zien dat E_2 (toestand 1) transiënt is voor elke strategie. E_1 (toestand 2) is gesloten en communicerend. Er is dus sprak van een zwak communicerende Markov Beslissingsketen. Omdat E_2 niet leeg is, is deze keten bovendien niet communicerend. Verder geldt dat deze keten unichain is: er is een ergodische klasse en een toestand die transiënt is onder iedere strategie.



Figuur 6: zwak communicerend (niet-communicerend), multichain

De Markov Beslissingsketen uit figuur 6 is zwak communicerend: $E_1 = \{1, 2\}$ is een gesloten communicerende klasse onder de strategie $f(1) = f(2) = f(3) = 1$ en $E_2 = \{3\}$ is een verzameling transiënte toestanden onder iedere strategie. Bovendien is deze keten niet-communicerend: vanuit toestand 1 of 2 kan toestand 3 onder geen enkele strategie bereikt worden. Verder is dit een multichaine Markov Beslissingsketen: er is een strategie zodat de bijbehorende keten twee ergodische klassen heeft: namelijk door in toestand 1 en 2 actie 2 te kiezen.

Figuur 7: niet-communicerend (en niet zwak communicerend), multichain

De Markov Beslissingsketen in figuur 7 heeft twee niet met elkaar communicerende toestanden. Bovendien zijn beide toestanden recurrent voor elke strategie, dus er is sprake van een multichain Markov Beslissingsketen.

4 Begrippen en algoritme Markovketen

Er zijn enkele algoritmes bekend om te bepalen waartoe een Markov Beslissingsketen behoort. In deze sectie wordt eerst een algoritme gegeven dat aangeeft hoe van een Markovketen bepaald kan worden wat de ergodische klassen en transiënte toestanden zijn.
Bij een Markovketen P hoort de gerichte graaf $G(P)$, die als punten de toestanden van de Markovketen heeft en (i, j) is een pijl in $G(P)$ als $p_{ij} > 0$. Een verzameling knooppunten heet streng samenhangend als er een pad tussen ieder tweetal knooppunten is in beide richtingen. Een maximale streng samenhangende verzameling heet een streng samenhangende component. We bespreken eerst een ander algoritme om de streng samenhangende componenten

van een Markovketen te bepalen. Dit algoritme is bedacht door Rao Kosaraju in 1978.[4]

Algoritme 1 *Bepaling van streng samenhangende componenten (algoritme van Kosaraju)*

1. Voer voorwaarts zoeken (DFS) uit op G en nummer de punten in de volgorde van afhandeling.
2. Construeer $G_1 = (V, A_1)$ met $(i, j) \in A_1 \Leftrightarrow (j, i) \in A$. Je draait dus elke pijl uit G om.
3. Voer nu DFS uit op G_1, waarbij begonnen wordt met het hoogst genummerde knooppunt uit de nummering gedaan in stap 1. Als niet elk punt bereikt wordt, wordt een nieuwe boom gestart in het nog niet bereikte knooppunt met het hoogste nummer.
4. Elke losse boom in de uiteindelijke verzameling van bomen is een streng samenhangende component van G.

Stelling 3 *Algoritme 1 is correct en heeft complexiteit* $\mathcal{O}(N^2)$.

Bewijs
Voor de correctheid van dit algoritme moeten we het volgende bewijzen:
Knooppunt v en knooppunt w in dezelfde streng samenhangende component$\Leftrightarrow$ v en w zitten in dezelfde boom na DFS van G_1.
$\Rightarrow$
Omdat v en w in dezelfde s.s.c. zitten is er een pad van v naar w en vice versa. wanneer we met DFS van G_1 starten in een knooppunt x (de wortel) en dan v of w bereiken, dan is er van daar uit een pad naar w respectievelijk v. Dit pad zal dan ook bij deze boom gaan horen, dus beide knooppunten eindigen in dezelfde boom.
$\Leftarrow$
v en w zitten in dezelfde boom na DFS in G_1, met x als wortel (x is mogelijk v of w). We gaan laten zien dat v en x communiceren en op dezelfde manier ook w en x, en dus v en w.
Als v in dezelfde boom zit als x, is v een afstammeling van x. Dit betekent dat er een pad is van x naar v in G_1 en dus een pad van v naar x in G.
Omdat x een hoger nummer heeft dan v (dit is zo omdat je volgens het algoritme begint met DFS in het punt met de hoogste nummer), is v eerder afgehandeld dan x. Dit betekent dat de recursieve actie in v plaatsvindt voor die van x.
Er zijn nu 2 mogelijkheden:

1. v wordt voor x bezocht.
   Als v voor x wordt bezocht, houdt dit in dat er ook DFS van x naar v

moet zijn geweest, want v is eerder afgehandeld dan x, dus er moet een pad terug zijn. Maar dat betekent dat er een kring ontstaat in de boom, en dit gebeurt nooit bij DFS, dus zo'n pad kan niet ontstaan. v kan dus niet voor x bezocht worden.

2. x wordt voor v bezocht.
   Dit houdt in dat er een pad van x naar v is.

Er is dus zowel een pad van v naar x en een van x naar v. Dus x en v communiceren. Zo ook x en w en dus ook v en w. Beide kanten op klopt de stelling.

De complexiteit van dit algoritme is $\mathcal{O}(N^2)$, met N het aantal knooppunten, want DFS heeft complexiteit $\mathcal{O}(N^2)$[3]. In elk punt van de graaf worden alle pijlen vanuit dit punt één keer bekeken. Dit aantal zijn er maximaal N, dus de orde van dit algoritme is van orde $\mathcal{O}(N^2)$. QED

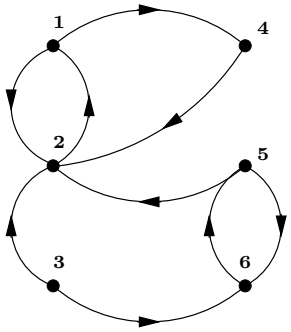
Algoritme 2 *Classificatie van Markovketens*

1. Bepaal de streng samenhangende componenten van $G(P)$, en noem deze $C_1, C_2 \dots C_k$. Dit kan gedaan worden met behulp van Kosaraju's algoritme.
2. Neem $m = 0$ en $T = \emptyset$
3. Voor $i = 1$ tot k:
   als C_i gesloten is, neem $m = m + 1$ en $R_m = C_i$
   anders $T = T \cup C_i$.
   (We kijken of C_i gesloten is door voor elk punt uit deze samenhangende component te kijken of deze een pijl heeft die naar een punt buiten deze component gaat.)

Nu kunnen we kijken waartoe de Markovketen P behoort:

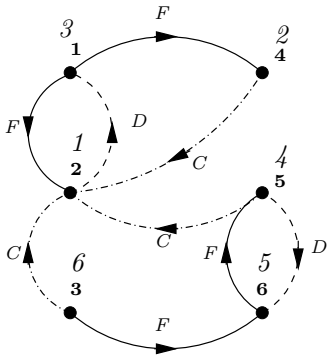
- Als $m = 1$ en $T = \emptyset$ dan is de Markovketen irreducibel.
- Als $m = 1$ dan is de Markovketen unichain.
- Als $m \geq 2$ dan is de Markovketen multichain.

Onderstaand voorbeeld laat duidelijk zien hoe het algoritme werkt. Stel we hebben de graaf uit figuur 8:



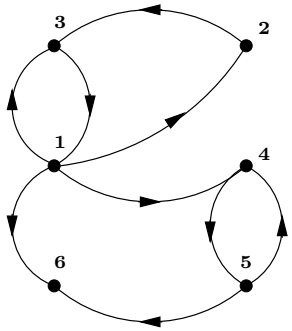
Figuur 8: Graaf G

- **stap 1**
  Met behulp van DFS krijgen we de graaf uit figuur 9:



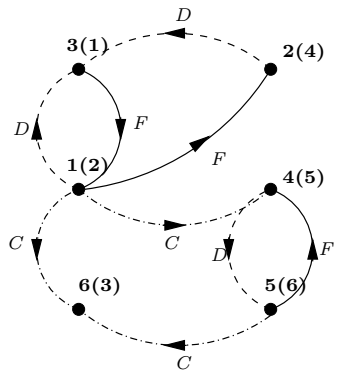
Figuur 9: Graaf G na DFS

Met F worden pijlen uit de boom aangegeven, met C cross-pijlen en met D achterwaartse pijlen, en met I voorwaartse pijlen die niet in de boom zitten. Met de grote nummers wordt de volgorde van afhandeling aangegeven. Vervolgens construeren we de graaf G_1 met als nummering van knooppunten de volgorde van afhandeling bij DFS (zie figuur 10):



Figuur 10: Graaf G_1

Hier voeren we DFS op uit en krijgen dan de boom uit figuur 11:



Figuur 11: Graaf G_1 na DFS

We zien dat de bomen (de F-pijlen) in deze figuur $C_1 = \{1, 2, 4\}$, $C_2 = \{3\}$, $C_3 = \{5, 6\}$ zijn (oorspronkelijke nummering staat tussen haakjes in figuur).
Dit zijn de streng samenhangende componenten van G.

- **stap 2** $m = 0$ en $T = \emptyset$
- **stap 3**
  - *i=1* C_1 is gesloten, want er zijn geen pijlen van $\{1, 2, 4\}$ naar $\{3, 5, 6\}$. Dus $m = 1$ en $R_1 = C_1$.
  - *i=2* C_2 is niet gesloten want er is een pijl van 3 naar 2 (en een van 3 naar 6). Dus $T = C_2$.
  - *i=3* C_3 is niet gesloten want er is een pijl van 5 naar 2. Dus $T = C_2 \cup C_3$.

We hebben nu dus: $m = 1$, $R_m = C_1$, $T = C_2 \cup C_3$. Dit houdt in dat deze Markovketen niet irreducibel en unichain is, met als recurrente klasse $\{1, 2, 4\}$ en als transiente toestanden $\{3, 5, 6\}$.

Stelling 4 *Algoritme 2 is correct en heeft complexiteit $\mathcal{O}(N^2)$.*

De Markovketen is irreducibel als er maar één streng samenhangende component is: alle toestanden communiceren dan met elkaar. De Markovketen kan niet irreducibel zijn als er een niet gesloten s.s.c. is. Er is dan zeker een s.s.c. transiënt, anders zouden de streng samenhangende componenten ook met elkaar samenhangen. Als er meerdere s.s.c. gesloten zijn, zijn er meerdere recurrente klasse en is de keten dus multichain. Dit wordt allemaal bijgehouden met behulp van m, R_m en T: als T leeg is zijn er geen transiënte toestanden en als m groter of gelijk aan 2 is zijn er minstens 2 recurrente klassen. Zo kan eenvoudig bepaald worden waar een Markovketen bijhoort.

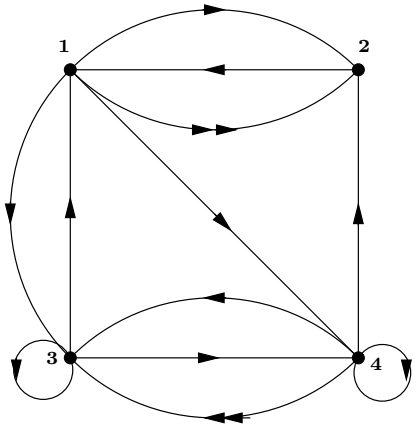
We bekijken de complexiteit van dit algoritme. We noemen: $N = \#E$, het aantal knooppunten in G. Het bepalen van de streng samenhangende componenten via Kosaraju's algoritme heeft $\mathcal{O}(N^2)$.
In het algoritme hoeft verder weinig gedaan te worden: elk punt uit de samenhangende componenten wordt nog een keer beschouwd en gekeken of deze een pijl heeft naar een punt buiten deze component. Dit heeft dus ook weer complexiteit $\mathcal{O}(N^2)$ dus het algoritme heeft in totaal een complexiteit van $\mathcal{O}(N^2)$.
QED

Om volgende algoritmes wat makkelijker te kunnen opschrijven introduceren twee begrippen. Bij een Markov Beslissingsketen G horen de volgende twee gerichte grafen: G^1 en G^2, beide met E als knooppuntenverzameling. We definieren G^1 en G^2:

G^1 heeft een pijl van i naar j als in de Markov Beslissingsketen G er voor elke actie $a \in A(\mathrm{i})$ een positieve kans is om van i naar j te gaan in één stap. Dit houdt dus in dat G^1 een graaf met alleen maar knooppunten en geen pijlen kan zijn, als er voor elk knooppunt i geen knooppunt j is zodat elke actie uit $A(\mathrm{i})$ een positieve kans heeft om in een stap in dit knooppunt te komen. Lussen laten we weg.

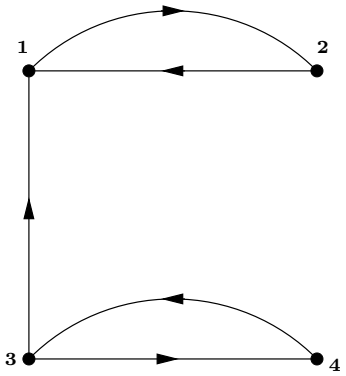
G^2 heeft een pijl van i naar j als in de Markov Beslissingsketen G er voor minstens één actie $a \in A(\mathrm{i})$ een positieve kans is om van i naar j te gaan in één stap. Wederom laten we lussen weg.

In figuur 12 staat een voorbeeld van een Markov Beslissingsketen G, waarbij een pijl tussen i en j aangeeft dat tussen deze twee punten een positieve kans is om van i naar j te gaan in één stap met een bepaalde actie.



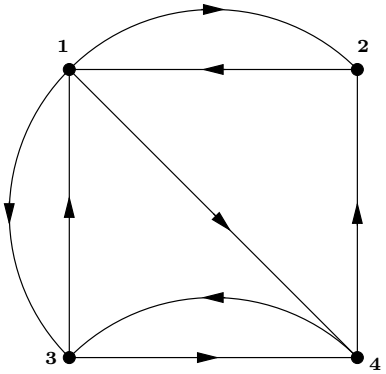
Figuur 12: Markov Beslissingsketen G

Hierbij hoort de volgende gerichte graaf G^1:



Figuur 13: Graaf G^1

En de volgende gerichte graaf G^2:



Figuur 14: Graaf G^2

Let op het verschil in notatie: bij de Markov Beslissingsketen G geeft een enkele pijl actie 1 aan, terwijl bij de grafen G^1 en G^2 een pijl alleen de richting aangeeft.

We bekijken de complexiteit van deze constructies. We nemen $T = \sum_{i \in E} \#A(i)$, ofwel T is het totaal aantal acties in G. We moeten alle acties bekijken en deze acties kunnen naar elk van de N knooppunten gaan, dus de orde voor het bepalen van G^1 en G^2 is $\mathcal{O}(T \cdot N)$.

Verder introduceren we de gecondenseerde graaf G^1_c. G^1_c heeft een samengesteld knooppunt voor elke streng samenhangende component van G^1. Neem i en j knooppunten van G^1_c en C_i en C_j de samenhangende componenten uit G^1 waaruit deze voortkomen. Er is een pijl van i naar j in G^1_c als elke Markovketen van de Markov Beslissingsketen een positieve overgangskans heeft voor een knooppunt uit C_i naar een uit C_j. Dit houdt eigenlijk in dat er een knooppunt in C_j moet zijn waarvoor je onder elke actie in één stap naar een willekeurig knooppunt uit C_j kan komen. De complexiteit hiervan is $\mathcal{O}(T \cdot N)$.

Je kunt deze constructie blijven herhalen (dus ook de gecondenseerde graaf van de gecondenseerde graaf bepalen, enz.) totdat er geen veranderingen meer ontstaan bij condensatie. Deze graaf noemen we $(G^1_c)^*$ en elke s.s.c. uit deze graaf bestaat uit een enkel knooppunt.

5 Algoritmes voor Markov Beslissingsketens

In deze paragraaf bespreken we algoritmes die bepalen waar een bepaalde Markov Beslissingsketen toe behoort. Door alle algoritmes samen te voegen krijg je een groot algoritme dat direct test in welke klasse een keten zit.

Algoritme 3 *Communicerend*

1. Construeer de graaf G^2
2. Bepaal de streng samenhangende componenten van G^2,
   noem deze $C_1, C_2, \ldots C_k$(Kosajaru)
3. Als $k = 1$: de Markov Beslissingsketen is communicerend
   Als $k \geq 2$: de Markov Beslissingsketen is niet communicerend

Stelling 5 *Algoritme 3 is correct en heeft complexiteit $\mathcal{O}(T \cdot N)$.*

Bewijs
Als twee knooppunten i en j in eenzelfde s.s.c. van G^2 zitten, betekent dit dat er een pad in G^2 is van i naar j en vice versa. Dit houdt in dat er een strategie is die een pad bevat van i naar j (volgt uit de constructie van G^2). Als er één s.s.c. is dan is er dus zo'n strategie voor alle i en j en is de Markov Beslissingsketen communicerend. Als er meerdere s.s.c. zijn dan zijn er een i en j waarvoor er niet zo'n pad is en is de keten niet communicerend.

De complexiteit van dit algoritme wordt bepaald door het maximum te nemen van het construeren van G^2 en het bepalen van de s.s.c.: dit is het maximum van $\mathcal{O}(T \cdot N)$ en $\mathcal{O}(N^2)$. Dit maximum is $\mathcal{O}(T \cdot N^2)$, want $T > N$: het totaal aantal acties is groter dan het aantal knooppunten, omdat je in elk knooppunt minstens één actie kan kiezen. QED

Algoritme 4 *Irreducibel*

Als uit het vorige algoritme is gebleken dat de keten niet communicerend is, dan hoeft dit algoritme niet uitgevoerd te worden, immers irreducibel⇒ communicerend, dus niet communicerend⇒niet irreducibel.

1. Construeer de graaf G^1 en neem $D = G^1$
2. Bepaal de streng samenhangende componenten van D,
   noem deze $C_1, C_2, \ldots C_k$ (Kosajaru)

3. Ga als alle componenten uit één knooppunt bestaan naar 4. Anders, construeer G_c^1, neem $D = G_c^1$ en ga naar 2

4. Als $k = 1$: de Markov Beslissingsketen is irreducibel
   Als $k \geq 2$: de Markov Beslissingsketen is niet irreducibel

Stelling 6 *Algoritme 4 is correct en heeft complexiteit* $\mathcal{O}(T \cdot N^2)$

Bewijs
Stel $(G_c^1)^*$ bestaat uit één knooppunt. Dan volgt direct uit de definitie van $(G_c^1)^*$ dat de Markov Beslissingsketen irreducibel is. Immers elk punt in een s.s.c. communiceert met elkaar onder elke strategie.
Stel $(G_c^1)^*$ bestaat uit twee of meer knooppunten. Dan is er een knooppunt i in $(G_c^1)^*$ zonder inkomende pijl, anders zouden minstens één tweetal knooppunten toch communiceren. Dus kan er in de knooppunten uit $j \neq i$ een actie worden zodat dat de overgangskansen naar een knooppunt uit i 0 zijn. Dus is de keten niet irreducibel.

De complexiteit van de stappen 1-3 één keer doorlopen is $\mathcal{O}(T \cdot N)$ (zie complexiteit vorig algoritme). Elke gecondenseerde graaf heeft minimaal één knooppunt minder, want er worden er minimaal twee samengevoegd tot één. Het condenseren kan dus maximaal N keer gebeuren, dus heeft het gehele algoritme complexiteit $\mathcal{O}(T \cdot N^2)$.

Algoritme 5 *Zwak communicerend*

Als uit het eerste algoritme is gebleken dat de Markovketen communicerend is, hoeft het volgende algoritme niet te worden uitgevoerd, immers communicerend $\Rightarrow$*zwak communicerend.*

1. Construeer de graaf G^2

2. Bepaal de streng samenhangende componenten van G^2,
   noem deze $C_1, C_2, \ldots C_k$(Kosajaru)

3. (a) Neem $m = 0$ en $T = \emptyset$
   (b) Doe, voor $i = 1$ tot k:
   als C_i is gesloten: $m = m + 1$
   anders: $T = T \cup C_i$

4. Als $m \geq 2$: de Markov Beslissingsketen is niet zwak communicerend KLAAR
   Als $m = 1$: als $T = \emptyset$: de Markov Beslissingsketen is communicerend, en dus zwak communicerend KLAAR
   anders: nog geen uitsluitsel, ga naar stap 5

5. (a) Neem $c_i = 1$ voor $i \notin T$ en $c_i = 0$ voor $i \in T$
   (b) $S = \emptyset$
   (c) Doe, voor elke $i \in T$:
   als $\sum_j p_{ij}(a)c_j > 0$ voor elke $a \in A(i)$, dan: $c_i = 1$ en $S = S \cup \{i\}$
   (d) Als $S = \emptyset$: de Markov Beslissingsketen is niet zwak communicerend. KLAAR
   Anders: $T = T \backslash S$ en ga naar 5e
   (e) Als $T = \emptyset$: de Markov Beslissingsketen is zwak communicerend. KLAAR
   Anders: ga naar 5b

Opmerking: te zien is dat het algoritme voor communiceren ook geïntegreerd is in dit algoritme. Toch staat het algoritme voor communiceren apart, omdat dan soms het algoritme voor zwak communiceren niet uitgevoerd hoeft te worden.

Stelling 7 *Algoritme 5 is correct en heeft complexiteit* $\mathcal{O}(T \cdot N^2)$

Bewijs
We weten dat zwak communicerend inhoudt dat voor elke toestand geldt dat deze transiënt is onder elke strategie of behoort tot een gesloten, communicerende klasse. Bovendien is er maar één gesloten klasse.
Het eerste deel van het algoritme (1-4) is bekend: als er meer dan één recurrente klasse is, is de keten niet zwak communicerend en als er maar één recurrente klasse is en geen transiënte toestanden ($T = \emptyset$), is de keten communicerend.

Wanneer er echter maar één recurrente klasse is en wél transiënte toestanden onder een strategie, moet nog gekeken worden of deze toestanden ook transiënt zijn onder iedere strategie. Dit gebeurt in stap 5: in de verzameling S worden toestanden gestopt die zeker transiënt zijn onder iedere strategie, in T zitten de toestanden waarvan we het nog niet zeker weten. Als $c_i = 1$ dan behoort de toestand of bij de recurrente klasse, of is hij transiënt onder iedere strategie.
In stap 5c kijken we of een toestand $i \in T$ onder elke strategie een positieve overgangskans heeft naar een transiënte toestand onder elke strategie of naar de recurrente klasse. Als dit zo is, is hij zelf ook transiënt en stoppen we hem in S, en wordt c_i gelijk aan 1.
In 5d kijken we of er transiënte toestanden zijn ontstaan: als dit niet zo is, zijn er alleen toestanden die transiënt zijn onder sommige strategieën en is de keten niet zwak communicerend. Als dit niet zo is halen we de transiënte toestanden onder elke strategie uit T (dit is er minstens één) en kijken we of T nog elementen bevat. Is dit niet zo, dan is elke toestand die in T zat transiënt onder elke strategie en is de keten wel zwak communicerend. Anders gaan we de elementen nog verder onderzoeken.

Stap 1-4 worden maar één keer uitgevoerd, hoe dan ook, dus is hiervan de complexiteit $\mathcal{O}(T \cdot N)$ (het maken van G^2). In stap 5 vindt een handeling plaats voor elk van de knooppunten uit T en voor elke actie. Dit is dus ook $\mathcal{O}(T \cdot N)$. Omdat T na elke keer het doorlopen van 5 strict klneiner wordt, wordt dit maximaal N keer gedaan, en is de complexiteit van dit algoritme dus: $\mathcal{O}(T \cdot N^2)$. QED

Wat we nu gezien hebben is dat we al bijna het volledige onderscheid in klassen kunnen maken; alleen voor het onderscheid tussen unichain en multichain is nog geen algoritme beschreven. De gegeven algoritmes hebben polynomiale complexiteit, dus om te komen tot dit punt (alleen nog geen onderscheid tussen unichain en multichain) heeft ook polynomiale complexiteit.

We kunnen de algoritmes en hun werking als volgt in een tabel weergeven:

	Communicerend (zw.-communi.)	Zw. communicerend, niet communicerend	Niet-communi niet zw. communi.
Irreducibel	A B	niet mogelijk	niet mogelijk
Unichain, niet irreduc.	A b	a C	niet mogelijk
Multichain	A b	a C	a c

Legenda:

Algoritme	**positief**	**negatief**
Communicerend	A	a
Irreducibel	B	b
Zwak Communicerend	C	c

Te zien is dat bij een positief antwoord van het communicerend-algoritme (A) je alleen nog maar het irreducibel-algoritme (B,b) hoeft uit te voeren, en bij een negatief antwoord (a), slechts het zwak communicerend-algoritme (C,c).

Een algoritme met polynomiale complexiteit voor het onderscheid tussen unichain en multichain is niet bekend. Er is wel een polynomiaal algoritme dat echter geen uitsluitsel geeft in alle gevallen.

Voor dit algoritme introduceren we het volgende begrip: de uitgraad van een punt i in een graaf. Dit is het aantal pijlen dat i als beginpunt heeft. Met uitgraad 1 bedoelen we dus dat er één uitgaande pijl is van i naar een willekeurig j.

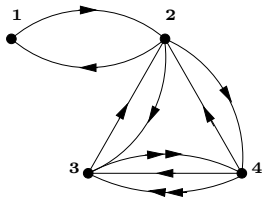
Stel de Markov Beslissingsketen is niet irreducibel. Wanneer we graaf G^1 hebben geconstrueerd en deze vervolgens condenseren tot $(G_c^1)^*$, bestaat deze uit meer dan één knooppunt (anders is de keten irreducibel). Er kan dan een knooppunt

i met uitgraad 1 zijn, zeg naar j. Je kunt dus onder elke strategie van een punt uit i naar een punt uit j (i en j zijn mogelijk samengestelde knooppunten en kunnen dus uit meerdere knooppunten bestaan). Er kan geen pad van j naar i zijn, anders zouden i en j nog samengevoegd kunnen worden en was $(G_c^1)^*$ nog niet bereikt.
We gaan nu i uit $(G_c^1)^*$ met uitgraad 1 naar j samenvoegen met j. Hierdoor komen er dus een aantal transiënte toestanden bij een mogelijk recurrente klasse. Als de keten unichain is, is er hooguit één recurrente klasse en door transiënte toestanden hieraan toe te voegen blijft deze klasse recurrent, alleen zijn er transiënte toestanden in deze klasse.
Als de keten multichain is, zal samenvoegen niet leiden tot vermenging van twee recurrente klassen: i is transiënt naar j (die zelf mogelijk ook transiënt is).

Na deze samenvoeging kan de graaf weer gecondenseerd worden, door weer nieuwe pijlen te maken, zoals eerder beschreven. Daarna kunnen er weer nieuwe punten met uitgraad 1 ontstaan zijn en worden die weer samengevoegd. Zo kunnen we doorgaan totdat er geen veranderingen meer optreden, deze graaf noemen we $(G^1)^+$.

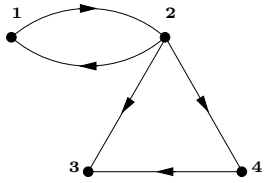
Hieronder een voorbeeld van hoe punten op deze manier worden samengevoegd.

Voorbeeld samenvoegen
Neem de volgende Markov Beslissingsketen (de overgangskansen zijn weggelaten, een pijl geeft aan dat er een strict positieve overganskans is):



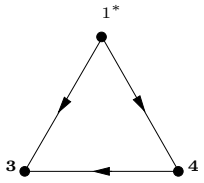
Figuur 15: Markov Beslissingsketen

Hierbij hoort deze graaf G^1:



Figuur 16: G^1

$(G_c^1)^*$ is dan, met $1^* = \{1, 2\}$:



Figuur 17: $(G_c^1)^*$

Nu kunnen we de punten 3 en 4 samenvoegen (4 heeft maar 1 uitgaande pijl) en maken dan weer pijlen zoals bij een gecondenseerde graaf: $(1^* = \{1, 2\}, 2^* = \{3, 4\})$

Figuur 18: G na condensatie en samenvoeging

Deze graaf is gelijk de nieuwe $(G_c^1)^*$. Samenvoegen (1^* heeft maar 1 uitgaande pijl) geeft als $(G^1)^+$ een enkel punt.

Uit het aantal knooppunten in $(G^1)^+$ kan deels worden afgeleid waartoe een Markov Beslissingsketen hoort. Het algoritme voor unichain\multichain is dan ook als volgt:

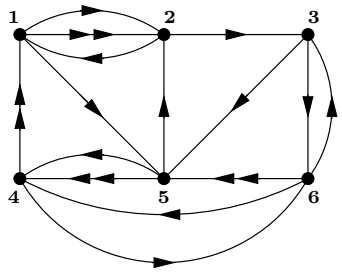
Algoritme 6 *Unichain\multichain*

1. Construeer G^1 op de bekende manier
2. Construeer uit G^1 de (meerdere malen) gecondenseerde graaf $(G_c^1)^*$
3. (a) Voeg knooppunten met uitgraad 1 samen met het punt waar ze naar toewijzen
   (b) Voeg nieuwe pijlen toe, zoals bij het condenseren van een graaf
4. (a) Noem de graaf die nu is ontstaan $(G^1)^+$
   (b) Als $(G^1)^+ = G^1$: ga naar stap 5
   anders: neem $G^1 := (G^1)^+$ ga naar stap 2
5. Bepaal de streng samenhangende componenten $C_1, C_2, \ldots, C_k$ van $(G^1)^+$

6. Als $k = 1$: de Markov Beslissingsketen is unichain
   Als $k = 2$: de Markov Beslissingsketen is multichain
   Als $k \geq 3$: de Markov Beslissingsketen kan nog zowel unichain als multichain zijn

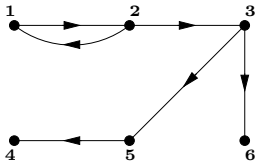
Voordat de correctheid en de complexiteit van dit algoritme zullen worden aangetoond, zijn hier voorbeelden van een unichain en een multichain Markov Beslissingsketen en hoe deze herkend kunnen worden.

Voorbeeld unichain
Bekijk de volgende Markov Beslissingsketen:



Figuur 19: Markov Beslissingsketen

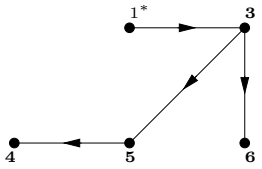
Na stap 1 van het algoritme:



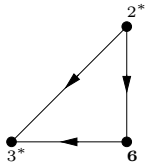
Figuur 20: G^1

Nu kunnen we samenvoegingen uitvoeren in de graaf (stap 3): 1^* heeft maar één uitgaande pijl en wordt samengevoegd met 3 tot 2^*. 5 heeft ook maar één uitgaande pijl en wordt samengevoegd met 4 tot 3^*. De pijlen zijn in deze figuur zijn gemaakt zoals in 3b beschreven.

Na stap 2 van het algoritme hebben we $(G_c^1)^*$ verkregen, met $1^* = \{1, 2\}$

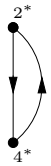


Figuur 21: $(G_c^1)^*$



Figuur 22: $(G^1)^+$, ofwel $(G_c^1)^*$ na samenvoeging

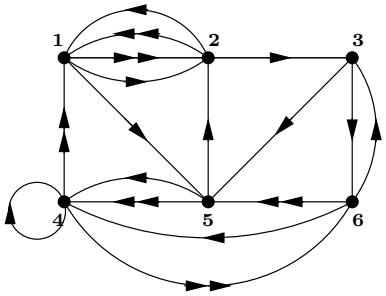
We keren weer terug bij stap 2, want $(G^1)^+ \neq G^1$. Deze graaf is gelijk $(G_c^1)^*$, dus we voegen gelijk weer knooppunten samen: 6 heeft één uitgaande pijl, naar 3^*, dus worden deze twee samengevoegd tot 4^*. Nu krijgen we deze graaf:



Figuur 23: $(G^1)^+$, na 2e keer samenvoeging

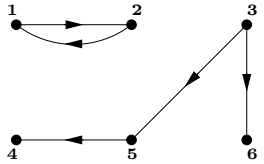
Deze graaf kan nog gecondenseerd worden tot één punt, dus de uiteindelijke graaf $(G^1)+$ bestaat uit één streng samenhangende keten, dus $k = 1$ en de keten is unichain.

Voorbeeld multichain
Beschouw nu weer de vorige Markov Beslissingsketen met een paar kleine aanpassingen:



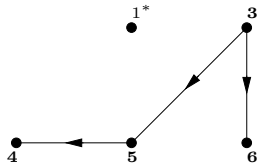
Figuur 24: Markov Beslissingsketen

Na stap 1 algoritme:



Figuur 25: G^1

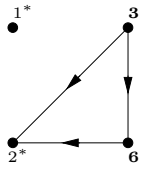
Na stap 2 algoritme hebben we $(G_c^1)^*$ verkregen, met $1^* = \{1, 2\}$



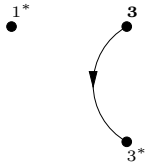
Figuur 26: $(G_c^1)^*$

Nu kunnen we samenvoegingen uitvoeren in de graaf (stap 3): 5 heeft maar één uitgaande pijl en wordt samengevoegd met 4 tot 2^*. De pijlen zijn in deze figuur zijn gemaakt zoals in 3b beschreven.

We keren weer terug bij stap 2, want $(G^1)^+ \neq G^1$. Deze graaf is gelijk $(G_c^1)^*$, dus we voegen gelijk weer knooppunten samen: 6 heeft één uitgaande pijl, naar 2^*, dus worden deze 2 samengevoegd tot 3^*. Nu krijgen we deze graaf:



Figuur 27: $(G^1)^+$, ofwel $(G^1_c)^*$ na samenvoeging



Figuur 28: $(G^1)^+$, na 2e keer samenvoeging

Deze graaf kan niet gecondenseerd worden, maar de knooppunten 3 en 3^* wel samengevoegd. Zo ontstaat de uiteindelijke graaf $(G^1)^+$, die uit 2 knooppunten bestaat, dus de Markov Beslissingsketen is multichain.

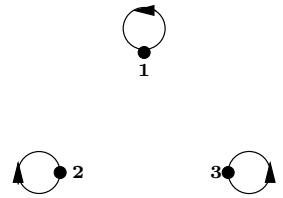
Nu de werking duidelijk is gemaakt, kan het algoritme bewezen worden.

Stelling 8 *Algoritme 6 is correct en heeft complexiteit* $\mathcal{O}(T \cdot N^2)$

Bewijs We laten zien dat als $k = 1$ er geen sprake kan zijn van multichain, als $k = 2$ dat er geen sprake kan zijn van unichain en voor $k \geq 3$ geven we een voorbeeld van zowel unichain als multichain.

- Stel $k = 1$ en de keten is niet unichain. Dan zijn er onder een bepaalde strategie minstens twee recurrente klassen. Tijdens het algoritme worden echter alleen transiënte toestanden voor elke strategie samengevoegd met een recurrente klasse. Ook twee klassen die communiceren onder elke strategie worden samengevoegd, bij condensatie. Twee klassen die onder een bepaalde strategie niet communiceren zullen dus nooit samengevoegd worden op deze manier en kan de keten bij $k = 1$ alleen maar unichain zijn.
- Stel $k = 2$ en dat de uiteindelijke graaf uit twee samengestelde knooppunten bestaat, i^* en j^*. Dan is er dus in deze uiteindelijke graaf geen pijl van i^* naar j^* of andersom. Dan is er een strategie f^1 waarbij i^* gesloten is en een strategie f^2 waarbij j^* gesloten is. Wanneer we deze twee strategieën combineren, krijgen we een strategie f^3 met beide samengestelde knooppunten gesloten. De keten is dus multichain.

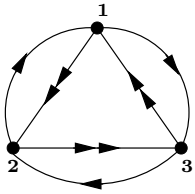
- Een voorbeeld van een multichaine keten met $k = 3$:



Figuur 29: multichain met $k = 3$

De gecondenseerde, samengevoegde graaf $(G^1)^+$ van deze Markov Beslissingsketen bestaat uit drie loss knooppunten, dus duidelijk is dat $k = 3$ (er zijn uiteindelijk drie streng samenhangende componenten van $(G^1)^+$) en multichain kan. Dit kan ook voor meerdere knooppunten, er ontstaan dan meerdere streng samenhangende componenten dus $k > 3$ en multichain kan ook.

Een voorbeeld van een unichaine keten met $k = 3$:



Figuur 30: unichain met $k = 3$

Als we uit deze Markov Beslissingsketen G^1 construeren, krijgen we gelijk een graaf met drie knooppunten zonder pijlen, en dit is gelijk $(G^1)^+$.
Dat deze graaf unichain is, is gemakkelijk in te zien: wanneer je in elk knooppunt actie 1 kiest krijg je een communicerende Markovketen. Hetzelfde geldt voor overal actie 2 kiezen. Als je in twee knooppunten actie 1 kiest en in de andere actie 2, dan krijg je één recurrent knooppunt en twee transiënte, dus is de Markov Beslissingsketen unichain.
Ook deze Markov Beslissingsketen is weer uit te breiden naar meerdere knooppunten, dus het geldt voor $k \geq 3$.

De complexiteit van algoritme 6: G^1, G^1_c en het bepalen van de streng samenhangende componenten gebeurt zoals bekend in $\mathcal{O}(T{\cdot}N)$. Dit hoeft weer hooguit N keer gedaan te worden, omdat elke nieuw gecondonseerde of samengevoegde graaf strikt kleiner is dan de vorige graaf. Dus de complexiteit is $\mathcal{O}(T \cdot N^2)$. QED

6 $\mathcal{NP}$-volledigheid van unichain/multichain

We hebben nu een algoritme van polynomiale complexiteit om onderscheid te maken tussen unichain en multichain. Echter dit algoritme geeft niet in alle gevallen uitsluitsel. In deze sectie zal aangetoond worden dat een algoritme dat wel onderscheid maakt tussen unichain en multichain $\mathcal{NP}$-volledig ($\mathcal{NPC}$) is.

Om dit te doen stellen we het unichain/multichain-probleem op als herkenningsprobleem (*ja-nee-probleem*). Vervolgens laten we voor dit probleem zien dat dit in $\mathcal{NP}$ zit en dat het 3SAT probleem, waarvan al bekend is dat het in $\mathcal{NPC}$ zit, polynomiaal gereduceerd kan worden tot het unichain/multichain-probleem. Dan is het unichain/multichain-probleem ook $\mathcal{NP}$-volledig. We definiëren hiervoor eerst zowel het unichain/multichain-probleem als het 3SAT probleem precies.

Het unichain/multichain-probleem

Gegeven: Een Markov Beslissingsketen met n toestanden en bij elke toestand maximaal m acties.

Probleem: Is er een strategie f waaronder de bijbehorende Markovketen twee ergodische klassen heeft? Zo ja, dan is de Markov Beslissingsketen multichain. Als er niet zo'n ja-instantie is, dan is de Markov Beslissingsketen unichain.

Het 3SAT probleem

Gegeven:

- n boolse variabelen, $x_1, x_2, \ldots x_n$ (dus $x_i \in \{0, 1\}$) en hun ontkenningen $\bar{x}_i := 1 - x_i$. Deze variabelen noemen we ook wel letters.
- Clusters C_m, de disjuncties van drie letters, bijvoorbeeld $x_1 \cup x_2 \cup \bar{x}_4$. Clusters zijn eigenlijk samenvoegingen van letters, ook wel woorden.
- Een zin $\mathcal{C}$, de doorsnede van woorden: $\mathcal{C}$=$C_1 \cap C_2 \cap \ldots \cap C_m$.

Een woord is waar als er minstens één van de drie letters in dit woord waar (dus de waarde 1 heeft) is. De zin is waar als alle woorden in deze zin waar zijn.

Probleem: gegeven een zin $\mathcal{C}$, is er een toekenning van de letters in deze zin zodat de zin waar is?

Voorbeeld
We bekijken deze zin:
$\mathcal{C}$=$(x_1 \cup \bar{x}_4 \cup x_5) \cap (\bar{x}_2 \cup x_3 \cup \bar{x}_4) \cap (\bar{x}_1 \cup x_2 \cup x_5) \cap (\bar{x}_2 \cup x_3 \cup x_4) \cap (\bar{x}_1 \cup x_3 \cup \bar{x}_5) \cap (x_2 \cup \bar{x}_3 \cup \bar{x}_4)$
We nemen de volgende toekenning van boolse variabelen: $x_1 = 1$, $x_2 = 0$, $x_3 = 1$, $x_4 = 0$ en $x_5 = 1$. Wanneer we deze invullen zien we dat de zin waar is.

Voorbeeld
We bekijken ook deze zin:
$\mathcal{C}$=$(x_1 \cup x_2 \cup x_3) \cap (x_1 \cup \bar{x}_2 \cup \bar{x}_3) \cap (\bar{x}_1 \cup x_2 \cup \bar{x}_3) \cap (\bar{x}_1 \cup \bar{x}_2 \cup x_3) \cap (x_1 \cup x_2 \cup \bar{x}_3) \cap (x_1 \cup \bar{x}_2 \cup x_3) \cap (\bar{x}_1 \cup x_2 \cup x_3) \cap (\bar{x}_1 \cup \bar{x}_2 \cup \bar{x}_3)$
We zien dat x_1, x_2 en x_3 verwisselbaar zijn en dat ook x_1 en zijn ontkenning dezelfde status hebben. We kunnen dus z.v.v.a. aannemen dat $x_1 = 1$. Dan zien we dat x_2 of $\bar{x}_3$ waar is en $\bar{x}_2$ of x_3 waar ook. Dus of $x_2 = x_3 = 1$ of $x_2 = x_3 = 0$. Als $x_2 = x_3 = 1$ dan is $(\bar{x}_1 \cup \bar{x}_2 \cup \bar{x}_3)$ niet waar, dus dat kan niet. Als $x_2 = x_3 = 0$ dan is $(\bar{x}_1 \cup x_2 \cup x_3)$ niet waar, dus dat kan ook niet.
Er is voor dit probleem dus geen verdeling van boolse variabelen zodat de zin waar is.

Voor het 3SAT-probleem is geen polynomiaal algoritme bekend, maar het is wel mogelijk om in polynomiale tijd te checken of een oplossing voldoet.

Stelling 9 *Het unichain/multichain-probleem is $\mathcal{NP}$-volledig.*

Bewijs
We laten eerst zien dat dit probleem in $\mathcal{NP}$ zit, daarna dat 3SAT gereduceerd kan worden tot unichain/multichain.
Na gaan of een bepaalde Markovketen uit meerdere klassen bestaat, kan in polynomiale tijd (zie algoritme 2). Controleren of een strategie een ja-instantie is kan in polynomiale tijd, dus het probleem zit in $\mathcal{NP}$.

Nu gaan we bewijzen dat het 3SAT probleem gereduceerd kan worden tot unichain/multichain.
We nemen hiervoor een 3SAT-instantie met n boolse variabelen en m clusters. Hieruit construeren we de volgende Markov Beslissingsketen met deze toestanden:

- toestand a en toestand b
- toestanden s_i, s'_i, t_i en f_i met $i \in \{1, 2, \ldots, n\}$
- toestanden c_j met $j \in \{1, 2, \ldots, m\}$

Bij de toestanden horen de volgende acties en overgangskansen:

toestand a: $A(a) = \{1\}$ met overgangskansen $p_{as_i}(1) = \frac{1}{n+m}$ voor alle $i \in \{1, 2, \ldots, n\}$ en $p_{ac_j}(1) = \frac{1}{n+m}$ voor alle $j \in \{1, 2, \ldots, m\}$

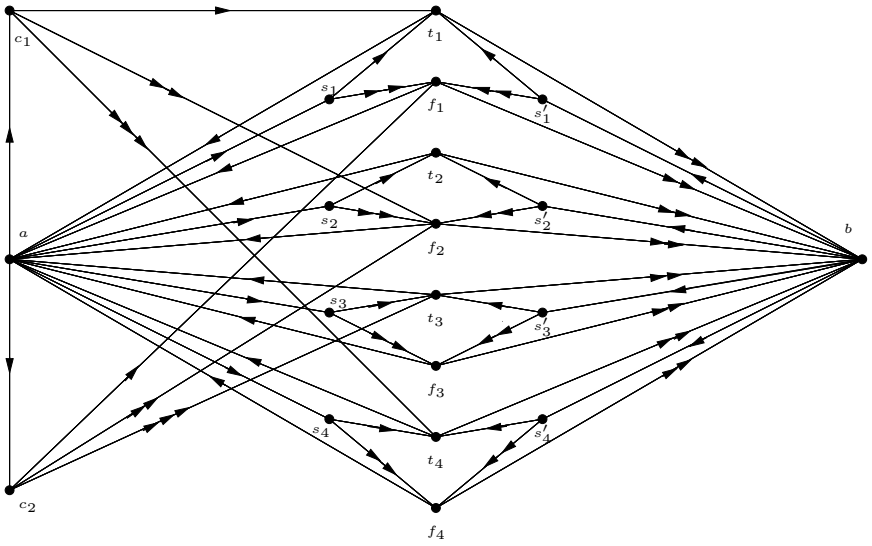
toestand b: $A(b) = \{1\}$ met overgangskansen $p_{bs'_i}(1) = \frac{1}{n}$ voor alle $i \in \{1, 2, \ldots, n\}$

toestanden s_i en s'_i: $A(s_i) = A(s'_i) = \{1, 2\}$ met overgangskansen $p_{s_i t_i}(1) = p_{s_i f_i}(2) = p_{s'_i t_i}(1) = p_{s'_i f_i}(2) = 1$ voor alle $i \in \{1, 2, \ldots, n\}$.

toestanden t_i en f_i: $A(t_i) = A(f_i) = \{1, 2\}$ met overgangskansen $p_{t_i a}(1) = p_{t_i b}(2) = p_{f_i a}(1) = p_{f_i b}(2) = 1$ voor alle $i \in \{1, 2, \ldots, n\}$.

toestanden c_j: $A(c_j) = \{1, 2, 3\}$ met de volgende overgangen:
als het k-de element uit cluster C_j van de 3SAT-instantie van de vorm x_i is, dan is de overgang onder actie k naar t_i.
als het k-de element uit cluster C_j van de 3SAT-instantie van de vorm $\bar{x}_i$ is, dan is de overgang onder actie k naar f_i.

Voorbeeld
Stel we hebben de volgende 3SAT instantie ($n = 4, m = 2$):
$\mathcal{C}$=$(x_1 \cup \bar{x}_2 \cup x_4) \cap (\bar{x}_1 \cup \bar{x}_2 \cup x_3)$. Dan hoort hierbij de volgende Markov Beslissingsketen: (de overgangskansen zijn allemaal 1, behalve vanuit a, dan zijn ze $\frac{1}{6}$ en vanuit b, dan zijn ze $\frac{1}{4}$)



Figuur 31: Markov Beslissingsketen geconstrueerd uit 3SAT

We gaan nu laten zien dat we een ja-instantie van 3SAT hebben dan en slechts dan als we een ja-instantie van unichain/multichain hebben.

Stel we hebben een ja-instantie van 3SAT. Dan is er dus een toekenning van boolse variabelen zodat de zin waar is. We kiezen dan de volgende strategie in de bijbehorende Markovketen:

- In toestand c_j kiezen we de actie die behoort bij (een van) de boolse variabele uit C_j die waar is. Als deze de vorm x_k had, is de overgang dus naar t_k en als deze de vorm $\bar{x}_k$ heeft naar f_k (zo gedefinieerd).

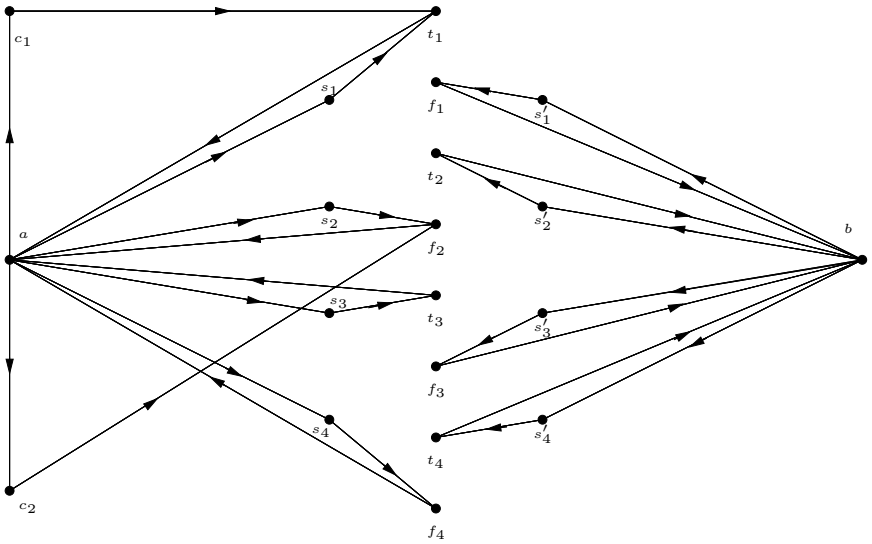
- In elke toestand s_i kiezen we actie 1 als x_i waar is (dus naar t_i) en actie 2 als x_i niet waar is (dus naar f_i).
- In elke toestand s_i' kiezen we actie 1 als x_i niet waar is (dus naar t_i) en actie 2 als x_i waar is (dus naar f_i).
- In elke toestand t_i kiezen we actie 1 als x_i waar is (dus naar a) en actie 2 als x_i niet waar is (dus naar b).
- In elke toestand f_i kiezen we actie 1 als x_i niet waar is (dus naar a) en actie 2 als x_i waar is (dus naar b).

Stel dat we beginnen in toestand a. Als de volgende toestand s_i is gaan we, als x_i waar is, naar toestand t_i en dan weer naar a. Als x_i niet waar is, gaan we vanuit s_i naar toestand f_i en dan naar a. En tenslotte als we vanuit toestand a naar c_j gaan, dan, als C_j waar is door x_k gaan we naar t_k en dan naar a en als C_j waar is door $\bar{x}_k$ gaan we naar f_k en dan naar a.
We zien dat a recurrent is en dat we nooit in toestand b komen als we in a starten.
Stel dat we beginnen in toestand b. Vanuit b gaan we naar toestand s_i'. Als x_i waar is gaan we vanuit s_i' naar f_i en dan vanuit daar weer naar b. Als x_i niet waar is gaan we vanuit s_i' naar t_i en dan weer naar b. Dus ook b is recurrent en hoort bij een andere ergodische klasse dan a.
Er is dus een strategie als er een ja-instantie van 3SAT is, zodat er twee ergodische klassen zijn. Dan is de Markov Beslissingsketen multichain en is dit ook een ja-instantie van unichain/multichain.

Bij het vorige voorbeeld is er een ja-instantie van 3SAT, bijvoorbeeld: $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 0$. De Markovketen die geconstrueerd kan worden door de hierboven beschreven strategie te kiezen staat in figuur 32 (het is duidelijk dat deze multichain is).

Stel we hebben een ja-instantie van unichain/multichain. Dan is er een strategie waaronder er minstens twee ergodische klassen zijn. Omdat we na elke drie overgangen weer in $\{a, b\}$ zitten (welke strategie ook gekozen worden) moeten a en b wel in twee verschillende ergodische klassen zitten en kunnen er ook maar maximaal twee ergodische klassen zijn.
De strategie die voor twee ergodische klassen zorgt heeft deze vorm:
Als vanuit s_i naar t_i gegaan wordt, wordt in t_i actie 1 gekozen naar a, omdat b niet bereikbaar is vanuit a. Vanuit s_i' ga je dan naar f_i, omdat a niet bereikbaar is vanuit b, dus naar t_i gaan kan niet.
Andersom geldt hetzelfde: als je vanuit s_i naar f_i gaat, wordt in f_i actie 1 gekozen om weer terug te gaan naar a. Vanuit s_i' ga je dan naar t_i.

We doen nu het volgende: als in s_i actie 1 is gekozen (dus naar t_i), dan geven we x_i de waarde 1, en als in s_i actie 2 gekozen is (dus naar f_i), dan geven we x_i de waarde 0.



Figuur 32: Multichaine Markovketen geconstrueerd uit ja-instantie van 3SAT

Stel nu dat de gekozen actie uit c_j naar t_i gaat. Dan zit x_i in C_j. Omdat t_j dan bereikbaar is vanuit a zal t_i weer teruggaan naar a en dus zal ook in s_i actie 1 naar t_j gekozen zijn. En volgens de alinea hierboven is x_i waar, dus C_j is waar. Stel dat de gekozen actie vanuit c_j naar f_i gaat. Dan zit $\bar{x}_i$ in C_j. Omdat f_j dan bereikbaar is vanuit a zal in f_i de actie gekozen zijn die weer teruggaat naar a en dus zal ook in s_i actie 2 naar t_j gekozen zijn. En dan blijkt weer uit de aanname hierboven dat x_i de waarde 0 heeft en dus $\bar{x}_i$ waar is, dus C_j waar. Zo is elke C_j waar.

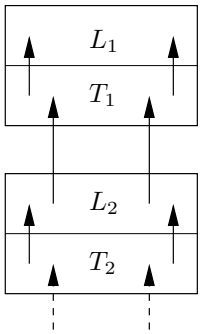
Dus als we een ja-instantie van unichain/multichain hebben, hebben we een ja-instantie van 3SAT.
Ja-instanties van unichain/multichain komen dus overeen met ja-instanties van 3SAT en dus is 3SAT te reduceren tot unichain/multichain.

Unichain/multichain is dus $\mathcal{NP}$-volledig. QED

7 Decompositie van Markov Beslissingsketens

De toestanden van een Markov Beslissingsketen kunnen ook ingedeeld worden door onderscheid te maken in de bereikbaarheid van deze toestanden. We maken hiervoor niveau's van toestanden gebaseerd op de bereikbaarheid (decompositie).
Het eerste niveau L_1 bestaat uit de toestanden die behoren bij gesloten, streng samenhangende deelverzamelingen van de Markov Beslissingsketen. Dit zijn dus de gesloten streng samenhangende componenten van G^2.
De overige toestanden $E_1 := E \backslash L_1$ gaan we nu onderverdelen in transiënte toestanden onder elke strategie en toestanden waarvan absorptie in L_1 kan worden vermeden. Toestanden die onder elke strategie worden geabsorbeerd plaatsen we in de transiënte verzameling T_1 en dan wordt $E_1 := E \backslash \{L_1 \cup T_1\}$.

Dit proces herhaalt zich als $E_1 \neq \emptyset$: eerst construeren we een nieuwe Markov Beslissingsketen met als toestandsverzameling E_1 en als acties $a \in A(i)$ met $\sum_{j \notin \{L_1 \cup T_1\}} p_{ij}(a) = 1$, dus acties die met positieve kans naar $L_1 \cup T_1$ gaan worden buiten beschouwing gelaten.
Zo construeren we L_2 en T_2 en gaan net zo lang door totdat $E_i = \emptyset$ is en elke toestand bij een L- of T-verzameling hoort. $L_i \cup T_i$ is het i-de niveau.
Grafisch ziet dit er als volgt uit:



Figuur 33: Niveau 1 en 2

Bij de decompositie van de toestandsruimte hoort het volgende algoritme:

Algoritme 7 *Decompositie van de toestandsruimte*

1. (a) Neem $m = 0$ en $E_m = E$
   (b) Construeer G^2 op de gebruikelijke manier van de Markov Beslissingsketen met als toestandsruimte E_m

(c) Doe $m = m + 1$

(d) Bepaal L_m, de verzameling van streng samenhangende componenten van G^2

(e) Doe $E_m = E_m \backslash L_m$

2. (a) Neem $c_i = 1$ als $i \in L_m$ en $c_i = 0$ als $i \in E_m$. Verder, neem $T_m = \emptyset$

(b) Neem $S = \emptyset$

(c) Doe, voor elke $i \in E_m$:
als $\sum_{j \in E_{m-1}} p_{ij}(a) c_j > 0$ voor elke $a \in A(i)$: $c_i = 1$ en $S = S \cup \{i\}$

(d) Als $S = \emptyset$: ga naar stap 3
anders: $T_m = T_m \cup S$, $E_m = E_m \backslash S$ en ga naar stap 2e

(e) Als $E_m = \emptyset$: stop
anders: ga naar stap 2b

3. (a) Doe, voor alle $i \in E_m$:
doe voor alle $a \in A(i)$ als $\sum_{j \in \{L_1 \cup T_1\}} p_{ij}(a) > 0$: $A(i) = A(i) \backslash \{a\}$

(b) ga naar stap 1b

Stelling 10 *Algoritme 7 is correct en heeft complexiteit* $\mathcal{O}(T \cdot N^2)$

Bewijs
Dit algoritme volgt precies wat is uitgelegd voor het algoritme. In stap 1 wordt L_m bepaald door de streng samenhangende componenten van G^2 te nemen. In stap 2 worden de transiënte toestanden bepaald op een manier analoog aan het bepalen van de transiënte toestanden bij algoritme 5 (uitleg staat daar). In stap 3 worden de acties behorend bij de nieuwe Markov Beslissingsketen (met als knooppunten E_m) bepaald, dus acties met een positieve overgangskans naar een punt buiten E_m weggelaten.

Het algoritme bestaat uit hooguit N iteraties: m wordt in elke iteratie één opgehoogd en in elke iteratie wordt L_m bepaald en deze kan niet leeg zijn (er is altijd minstens één streng samenhangende component). Dus L_m wordt maximaal N keer bepaald.
We beschouwen de complexiteit van één iteratie door van iedere stap de complexiteit te bepalen:

stap 1 Het bepalen van G^2 is, zoals bekend van $\mathcal{O}(T \cdot N)$. Het bepalen van de streng samenhangende componenten van $\mathcal{O}(N^2)$, dus stap 1 heeft complexiteit $\mathcal{O}(T \cdot N)$, want $T \geq N$. (onderdeel a, c en e: $\mathcal{O}(1)$)

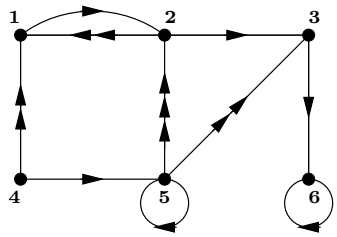
stap 2 In onderdeel a loop je elk knooppunt af, dus $\mathcal{O}(N)$, in onderdeel c loop je van elk knooppunt elke actie af, dus $\mathcal{O}(T \cdot N)$. Tenslotte in onderdeel d loop je weer elk knoopppunt af dus $\mathcal{O}(N)$. Stap 2 heeft dus complexiteit

$\mathcal{O}(T \cdot N)$ (onderdeel b en e: $\mathcal{O}(1)$)
NB: Je kunt deze stap apart wel vaker doorlopen, maar dan doorloop je het algoritme als geheel een keer minder. Dus dat maakt voor de complexiteit niet uit.

stap 3 In onderdeel a loop je weer in elk knooppunt elke actie af, dus stap 3 heeft complexiteit $\mathcal{O}(T \cdot N)$. (onderdeel b: $\mathcal{O}(1)$)

Één iteratie heeft complexiteit $\mathcal{O}(T \cdot N)$ en er zijn maximaal N iteraties, dus de complexiteit van het algoritme is $\mathcal{O}(T \cdot N^2)$. QED

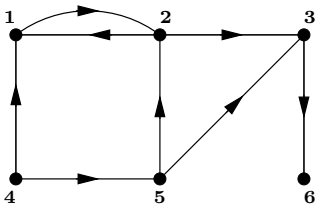
Voorbeeld
Neem de volgende Markov Beslissingsketen:



Figuur 34: Markov Beslissingsketen

We volgen het algoritme stap voor stap.

stap 1: G^2 is als volgt:



Figuur 35: G^2

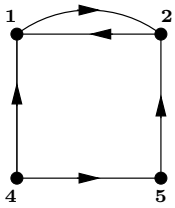
De streng samenhangende componenten zijn $C_1 = \{1, 2\}$, $C_2 = \{3\}$, $C_3 = \{4\}$, $C_4 = \{5\}$ en $C_5 = \{6\}$. Alleen $C_5 = \{6\}$ is gesloten, dus $L_1 = \{6\}$ en $E_1 = \{1, 2, 3, 4, 5\}$.

stap 2: We krijgen $c_6 = 1$ en $c_i = 0$ voor $i \in \{1, 2, 3, 4, 5\}$ en $T_1 = S = \emptyset$. In stap 2c zien we dat $\sum_{j \in E_{m-1}} p_{ij}(a) c_j > 0$ voor $i = 3$, dus $c_3 = 1$ en

$S = \{3\}$. In stap 2d zien we dan dat $T_1 = \{3\}$ en $E_1 = \{1, 2, 4, 5\}$ en in stap 2e gaan we weer naar 2b. Nu $S = \emptyset$ en er is in stap 2c geen $i \in E_1$ waarvoor geldt dat $\sum_{j \in E_{m-1}} p_{ij}(a)c_j > 0$. In stap 2d gaan we naar stap 3.

stap 3: We krijgen hier dat $A(2) = \{2\}$, want $p_{23}(1) > 0$ en $A(5) = \{1, 3\}$, want $p_{53}(2) > 0$. We gaan weer naar stap 1b.

stap 1: G^2 is nu weer als volgt, met als toestandsverzameling $E_1 = \{1, 2, 4, 5\}$:



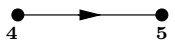
Figuur 36: G^2

De streng samenhangende componenten zijn hier: $C_1 = \{1, 2\}$, $C_2 = \{4\}$, $C_3 = \{5\}$. Alleen $C_1 = \{1, 2\}$ is gesloten, dus $L_2 = \{1, 2\}$ en $E_2 = \{4, 5\}$.

stap 2: We krijgen $c_1 = 1$, $c_2 = 1$, $c_4 = 0$, $c_5 = 0$ en $T_2 = S = \emptyset$. In stap 2c zien we dat $\sum_{j \in E_{m-1}} p_{ij}(a)c_j > 0$ voor geen enkele $i \in E_2$. In stap 2d gaan we dus gelijk naar stap 3.

stap 3: We krijgen hier dat $A(4) = \{1\}$, want $p_{41}(2) > 0$ en $A(5) = \{1\}$, want $p_{52}(3) > 0$. We gaan weer naar stap 1b.

stap 1: G^2 is nu weer als volgt, met als toestandsverzameling $E_2 = \{4, 5\}$:

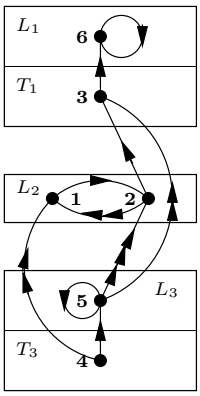


Figuur 37: G^2

De streng samenhangende componenten zijn hier: $C_1 = \{4\}$ en $C_2 = \{5\}$. Alleen $C_2 = \{5\}$ is gesloten, dus $L_3 = \{5\}$ en $E_3 = \{4\}$.

stap 2: We krijgen hier dat $c_5 = 1$, $c_4 = 0$ en $T_3 = S = \emptyset$. In stap 2c zien we dat $\sum_{j \in E_{m-1}} p_{ij}(a)c_j > 0$ voor $i = 4$, dus $c_4 = 1$, $S = \{4\}$ en in stap 2d dat $T_3 = \{4\}$ en $E_3 = \emptyset$. In stap 2e zijn we klaar met het algoritme.

We hebben nu de hele Markov Beslissingsketen onder verdeeld in niveau's en dit ziet er als volgt uit:



Figuur 38: Markov Beslissingsketen met aangegeven niveau's

Duidelijk te zien is dat je vanuit de toestanden in T_m altijd naar een niveau hoger gaat en dat dit vanuit een toestand in L_m onder bepaalde stragieën ontweken kan worden.

Referenties

[1] L.C.M. Kallenberg, *Classification problems in MDPS*, *Markov Processes and Controlled Markov Chains* Kluwer, Boston. **151-165** (2002)

[2] John N. Tsitsiklis, *NP-Hardness of Checking the Unichain Condition in Average Cost MDPs*, *Operations Research Letters*, Vol. 35, No. 3 **319-323**, 2007

[3] L.C.M. Kallenberg, *Dictaat Besliskunde A*

[4] Kosaraju, lcm.csa.iisc.ernet.in/dsa/node171.html

[5] P. van Oostrum, *Handleiding Latex*, 1996